ІНФАРМАТЫКА

У.М. Котаў
А.І. Лапо
Ю.А. Быкадораў
А.М. Вайцеховіч

8

ІНФАРМАТЫКА

Вучэбны дапаможнік для 8 класа
ўстаноў агульнай сярэдняй адукацыі
з беларускай мовай навучання

Дапушчана
Міністэрствам адукацыі
Рэспублікі Беларусь

Мінск «Народная асвета» 2018

УДК 004(075.3=161.3)
ББК 32.81я721
И74

Аўтары:
У. М. Котаў, А. І. Лапо, Ю. А. Быкадораў, А. М. Вайцеховіч

Пераклад з рускай мовы *К. І. Даніленка*

Рэцэнзенты:
кафедра сучасных тэхналогій праграміравання факультэта матэматыкі і інфарматыкі ўстановы адукацыі «Гродзенскі дзяржаўны ўніверсітэт імя Янкі Купалы» (кандыдат педагагічных навук, дацэнт *Н. П. Макарава*); настаўнік інфарматыкі вышэйшай кваліфікацыйнай катэгорыі дзяржаўнай установы адукацыі «Сярэдняя школа № 4 г. Дзяржынска» *С. Р. Пузіноўская*

ISBN 978-985-03-2984-4

ЗМЕСТ

Г л а в а 3. Асновы алгарытмізацыі і праграміравання

Г л а в а 4. Тэхналогія апрацоўкі тэкставых дакументаў

Ад аўтараў

Дарагія васьмікласнікі! Вы трымаеце ў руках вучэбны дапаможнік па інфарматыцы.

Даследаванні, якія праводзяцца ў гэтай навуцы, запатрабаваны як у прыродазнаўча-матэматычных, так і ў сацыяльна-гуманітарных галінах ведаў.

Мы, аўтары вучэбнага дапаможніка, паспрабавалі зрабіць так, каб ён дапамагаў вам засвоіць новыя веды і паглыбіць тыя, якія ўжо маюцца. Спадзяёмся, што ў далейшым вы зможаце выкарыстаць атрыманыя ўменні для рашэння практычных задач з розных прадметных абласцей.

Матэрыял кожнага параграфа ў дадзеным вучэбным дапаможніку падзелены на дзве калонкі. Колер фону вызначае прызначэнне размешчанай на ім інфармацыі:

— асноўны матэрыял, абавязковы для вывучэння;

— прыклады, якія ілюструюць асноўны матэрыял;

— азначэнні асноўных паняццяў;

— гістарычныя звесткі, інфармацыя пра вучоных, якія ўнеслі ўклад у развіццё інфарматыкі, і іншыя цікавыя факты.

Умоўныя абазначэнні, якія выкарыстоўваюцца ў вучэбным дапаможніку:

— пытанні для праверкі ведаў;

— раздзел «Практыкаванні» змяшчае заданні, пры выкананні якіх выкарыстоўваецца камп'ютар;

— раздзел «Практыкаванні» змяшчае заданні для выканання ў сшытку;

— раздзел «Практыкаванні» змяшчае заданні, пры выкананні якіх можа быць выкарыстана інфармацыя, змешчаная ў электронным адукацыйным рэсурсе на Нацыянальным адукацыйным партале (http://e-vedy.adu.by). У такіх заданнях вам будзе прапанавана адкрыць або загрузіць файл. Акрамя спасылкі, вы можаце выкарыстаць матрычны QR-код:

* — заданне або прыклад для цікаўных.

Імя файла для спампоўвання з электроннага адукацыйнага рэсурса змяшчае нумар параграфа і нумар практыкавання пасля гэтага параграфа. Напрыклад, імя файла upr3_1 азначае, што файл належыць да першага практыкавання пасля трэцяга параграфа. Таксама на партале змешчаны файлы з праграмамі, разгледжанымі ў прыкладах. Такія файлы маюць імя Program13_5.pas (праграма для прыкладу 13.5).

У гэтым вучэбным годзе вы зможаце навучыцца апрацоўваць гук і відэа, ствараць аніміраваныя фільмы. Таксама вы працягнеце вывучаць алгарытмы і праграміраванне і ўдасканаліце свае навыкі па стварэнні і афармленні тэкставага дакумента. Вам давядзецца засвоіць новыя інструменты ў вядомых праграмах і пазнаёміцца з новымі праграмамі. У вучэбным дапаможніку шмат рознага ілюстрацыйнага матэрыялу. Экранныя копіі прызначаны для першапачатковага азнаямлення з інтэрфейсамі праграм, для паказу размяшчэння асобных элементаў. Падрабязна разгледзець усе структурныя элементы акна праграмы, якая выкарыстоўваецца, можна на экране камп'ютара.

Да дадзенага вучэбнага дапаможніка распрацаваны электронны дадатак, змешчаны на Нацыянальным адукацыйным партале.

Жадаем вам поспехаў у вывучэнні інфарматыкі!

Глава 1
ТЭХНАЛОГІЯ АПРАЦОЎКІ АЎДЫЯ- І ВІДЭАІНФАРМАЦЫІ

§ 1. Запіс аўдыя- і відэаінфармацыі

1.1. Аўдыя- і відэафайлы

Аўдыяінфармацыя (гуказапіс) — гукавая інфармацыя, запісаная якім-небудзь чынам, прыдатным для ўзнаўлення.

Гукавыя ваганні паветра (гукавыя сігналы) маюць форму, якую называюць **аналагавай** (прыклад 1.1).

Раней аудыяінфармацыя ў аналагавай форме запісвалася ў студыях, а **праслухоўвалася** з дапамогай фанографаў, грамафонаў, патэфонаў, магнітафонаў і электрапрайгравальнікаў.

З пачаткам камп'ютарнай эры гукавыя сігналы пачалі аблічбоўваць, г. зн. хвалі сталі замяняць наборамі кропкавых (імпульсных) сігналаў, а велічыню імпульсаў — лікавымі кодамі (прыклад 1.2). Аўдыяінфармацыя атрымала **лічбавую** форму.

Аўдыяфайл — файл з аўдыяінфармацыяй у лічбавай форме.

Відэаінфармацыя — адлюстраванне рухомых аб'ектаў, запісанае якім-небудзь чынам, прыдатным для ўзнаўлення.

Спачатку відэаінфармацыю запісвалі **ў форме кінафільмаў.** Пры дэманстрацыі асобныя фотакадры на кінаплёнцы зліваліся на экране ў рухомы відарыс.

Відэаінфармацыя **ў лічбавай форме** з'яўляецца наборам электронных фатаграфій (прыклад 1.3).

Прыклад 1.1. Аналагавая форма гукавога сігналу.

Гісторыя запісу аналагавага гуку пачалася ў 1857 г., калі француз Эдуард Леон Скот дэ Мартэнвіль вынайшаў прыладу, якая прадзірала гукавыя дарожкі на закуродымленай паперы.

Прыклад 1.2. Схема лічбавання аналагавага гукавога сігналу.

Хваля замяняецца наборам сігналаў з пунктаў (імпульсных сігналаў), а велічыня імпульсаў задаецца лікамі. Набор пунктаў пераўтвараецца ў набор лікаў: **(2)(8)(11)(13)(14)(13)(12)(10)(7)...**

Гісторыя кінематографа пачалася ў 1885 г., калі французы Агюст і Луі Люм'еры ўпершыню правялі дэманстрацыю кінафільмаў.

Прыклад 1.3. Набор кадраў лічбавага відэа можна ўспрымаць як набор электронных кадраў кінафільма.

Відэафайл — файл з відэаінфармацыяй, якая суправаджаецца аўдыяінфармацыяй, у лічбавай форме.

1.2. Праграмныя сродкі запісу і прайгравання

У смартфонах праграмныя сродкі запісу гуку пры дапамозе мікрафона прадстаўлены дыктафонамі (прыклад 1.4).

На камп'ютарах з аперацыйнай сістэмай Windows стандартнай праграмай для запісу гуку пры дапамозе мікрафона з'яўляецца праграма **«Звукозапись»** (прыклад 1.5). Больш шырокія магчымасці мае бясплатная праграма UV SoundRecorder.

Для запісу відэаінфармацыі ў смартфонах шырока выкарыстоўваецца дадатак **«Камера»** ў рэжыме запісу відэа. Гэты рэжым маюць і лічбавыя фотаапараты.

Запісваць відэа дазваляюць камп'ютары з мікрафонам і вэб-камерай. Для камп'ютараў распрацаваны таксама праграмы для запісу гуку і відэа, якія прайграюцца іншымі праграмамі.

Праграмы для прайгравання аўдыя- і відэафайлаў называюцца **плэерамі. Медыяплэеры** — плэеры, якія прайграюць як гук, так і відэа (прыклады 1.6 і 1.7).

Аўдыя- і відэафайлы могуць мець ліцэнзійныя абмежаванні на бясплатнае капіраванне, прайграванне і распаўсюджванне.

1.3. Фарматы аўдыяфайлаў

Аўдыяфайлы, як і відэафайлы, могуць адрознівацца спосабамі лічбавага запісу — фарматамі.

Прыклад 1.4. Дадатак **«Диктофон»** у адным з сучасных смартфонаў.

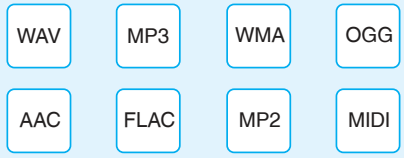
Прыклад 1.5. Акно стандартнай для Windows праграмы **«Звукозапись»**.

Прыклад 1.6. На камп'ютарах часцей за ўсё ўстанаўліваюцца аўдыяплэеры AIMP, Winamp Lite, медыяплэеры Windows Media Player (WMP), Winamp, Quicktime, KM Player, VLC Media Player і інш. Да цяперашняга часу распрацаваны цэлы шэраг аўдыя- і медыяплэераў для смартфонаў.

Прыклад 1.7. Акно медыяплэера WMP.

Кнопкі кіравання прайграваннем змешчаны ў ніжняй частцы акна.

Прыклад 1.8. Назвы асноўных фарматаў аўдыяфайлаў:



Лічбавы гуказапіс можа мець некалькі каналаў: мона (1 канал), стэрэа (2 каналы), Dolby Digital (6 каналаў) і г. д.

Прыклад 1.9. Імёны аўдыяфайлаў розных фарматаў:

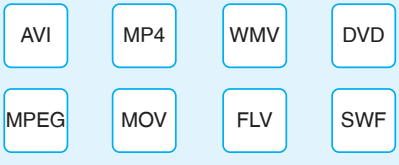
golos2.wav; karaoke.midi;
скрыпка.mp3; test31.wma.

Прыклад 1.10. Адна мінута запісу ў фармаце WAV мае аб'ём каля 10 Мб, у фармаце MP3 — ад 0,5 да 2,4 Мб.

Фармат WMA (Windows Media Audio) распрацаваны кампаніяй Microsoft як канкурэнт фармату MP3 і ўключае падтрымку сістэмы кіравання аўтарскімі правамі.

Гэта значыць, што праслухоўваць ахаваныя кампазіцыі можна толькі на камп'ютары, з якога кампазіцыя была загружана з музычнага магазіна.

Прыклад 1.11. Найбольш папулярныя фарматы відэафайлаў:



Кампанія Apple актыўна выкарыстоўвае ўласныя абазначэнні для фарматаў відэафайлаў, аналагаў MP4, напрыклад `m4a`, `m4b`, `m4v`, `m4p`, `m4r`.

Для мабільных тэлефонаў распрацаваны фармат 3GP, які выкарыстоўвае моцнае сціскание. Гэта дазваляе выкарыстоўваць яго на слабых мабільных тэлефонах.

Фармат аўдыяфайла — структура і асаблівасці запісу ў файле лічбавай аўдыяінфармацыі.

Існуе больш за 40 фарматаў аўдыяфайлаў (прыклад 1.8). Назва фармату служыць расшырэннем імя (тыпам) аўдыяфайла (прыклад 1.9).

Для высакаякаснага запісу гуку выкарыстоўваецца фармат WAV. У параўнанні з файламі іншых фарматаў файлы гэтага фармату маюць вельмі вялікія аб'ёмы.

Фармат MP3 самы распаўсюджаны. Выкарыстоўвае спецыяльныя метады сціскання аўдыяфайлаў за кошт невялікага зніжэння якасці гуку.

(Разгледзьце прыклад 1.10.)

Фармат MIDI (MID) з'яўляецца лічбавым уяўленнем нотных запісаў для выкарыстання на электронных музычных інструментах. Прайграванню розных нотных партый можна надаць афарбоўку гукаў фартэпіяна, скрыпкі, трубы і іншых інструментаў.

1.4. Фарматы відэафайлаў

Фармат відэафайла — структура і асаблівасці запісу ў файле лічбавай відэаінфармацыі, якая суправаджаецца аўдыяінфармацыяй.

Фарматы для запісу ў відэафайл толькі відэаінфармацыі не прадугледжаны. Але відэа можна захоўваць у файле і без гуку.

Вядома больш за 70 фарматаў відэафайлаў (прыклад 1.11). Назва фармату служыць расшырэннем імя (тыпам) відэафайла. Для запісу аўдыя- і відэафайлаў, акрамя праграм запісу, выкарыстоўваюць кодэкі.

Кодэк — спецыяльная праграма, якая сціскае (памяншае) і аднаўляе першапачатковы аб'ём аўдыя- або відэафайла.

Адрозніваюць **аўдыякодэкі** і **відэакодэкі**. Аўдыякодэкі носяць імёны фарматаў аўдыяфайлаў. Імёны відэакодэкаў з імёнамі фарматаў не супадаюць (прыклад 1.12).

Пры запісе і прайграванні відэафайла заўсёды выкарыстоўваецца пара з відэакодэка і аўдыякодэка. Фарматы відэафайлаў выкарыстоўваюць розныя пары кодэкаў (прыклад 1.13).

Прыклад 1.12. Назвы асноўных відэакодэкаў:

H.261 | XviD | DivX | MPEG4

Прыклад 1.13. Папулярны фармат відэафайлаў AVI можа выкарыстоўваць відэакодэк H.264 і аўдыякодэк MP3.

Магчымы і іншыя спалучэнні, напрыклад відэакодэк MPEG4 і аўдыякодэк AC3, відэакодэк XviD і аўдыякодэк MP3.

1. Што такое аўдыяінфармацыя?
2. У якім выглядзе запісваецца аўдыяінфармацыя ў лічбавай форме?
3. Што такое відэаінфармацыя?
4. Што такое відэаінфармацыя ў лічбавай форме?
5. Як называюцца праграмы для прайгравання лічбавых аўдыя- і відэафайлаў?
6. Што такое фармат аўдыяфайла?
7. Чым цікавы фармат аўдыяфайлаў MIDI?
8. Што такое фармат відэафайла?
9. Ці існуюць фарматы для запісу ў відэафайл толькі відэаінфармацыі?
10. Што такое кодэк?
11. Якія віды кодэкаў выкарыстоўваюцца для работы з відэафайламі?
12. Колькі кодэкаў патрабуецца для запісу відэафайла?

Практыкаванні

1 Прывядзіце прыклады фарматаў аўдыяфайлаў.

2 Прывядзіце прыклады фарматаў відэафайлаў.

3 Прывядзіце прыклады аўдыякодэкаў.

4 Прывядзіце прыклады відэакодэкаў.

5 Адкрыйце ў смартфоне дадатак **«Диктофон»**. Прагаварыце і запішыце азначэнне фармату відэафайла. Праслухайце запіс.

6 З дапамогай адпаведнага дадатка знайдзіце ў смартфоне папку з аўдыяфайламі і вызначыце іх фарматы.

7 Адкрыйце ў смартфоне дадатак «Камера». Папрасіце аднакласніка прачытаць услых азначэнне фармату аўдыяфайла перад камерай. Запішыце відэа і прагледзьце яго.

8 З дапамогай адпаведнага дадатка знайдзіце ў смартфоне папку з відэафайламі і вызначыце іх фарматы.

9 Уключыце камп'ютар, падключыце да яго мікрафон і навушнікі. Прагаварыце азначэнне фармату відэафайла і з дапамогай праграмы **«Звукозапись»** запішыце яго ў аўдыяфайл.

§ 2. Уводзіны ў рэдагаванне аўдыяфайлаў

Прыклад 2.1. Неабходнасць у рэдагаванні гуказапісу ўзнікае, калі яго працягласць трэба паменшыць або павялічыць. Напрыклад, калі працягласці падабранай музычнай кампазіцыі недастаткова для суправаджэння выступлення спевакоў або танцораў, якое рыхтуецца.

Выкарыстанне гукавога эфекту дазваляе змяніць стыль гучання гуказапісу, напрыклад гучнасць, хуткасць або тэмп прайгравання, вышыню тону. З дапамогай гукавых эфектаў можна выдаляць пстрычкі і трэск, дабаўляць рэха, выдаляць з музычнай кампазіцыі гучанне голасу.

Прыклад 2.2. Сярод бясплатных аўдыярэдактараў вылучым Audacity, WavePad Sound Editor, Wavosaur, FREE Wave MP3 Editor, Swifturn Free Audio Editor.

Прыклад 2.3. Канвертацыя аўдыяфайла можа спатрэбіцца, напрыклад, калі ў мультымедыйную прэзентацыю трэба ўставіць гукавы запіс з CD-дыска. Праграма для стварэння мультымедыйных прэзентацый не дапускае ўстаўку на слайд аўдыяфайлаў такога фармату.

Прыклад 2.4. Аўдыярэдактары дазваляюць захоўваць аўдыяфайлы ў розных фарматах, таму для канвертацыі дастаткова загрузіць аўдыяфайл у адным фармаце, а потым захаваць яго ў іншым.

2.1. Рэдагаванне і канвертацыя

Вядомыя два віды апрацоўкі аўдыяфайлаў: рэдагаванне і канвертацыя.

Рэдагаванне аўдыяфайла — працэс яго змянення, які складаецца ў выразанні, устаўцы, выдаленні і камбінаванні частак аўдыяфайла, што называюцца **фрагментамі**. Рэдагаванне ўключае таксама ўжыванне гукавых эфектаў да ўсяго гуказапісу і да яго фрагментаў (прыклад 2.1).

Для рэдагавання аўдыяфайлаў выкарыстоўваюцца праграмныя сродкі, якія называюцца **аўдыярэдактарамі** (прыклад 2.2).

Рэдагаваць аўдыяфайлы мы будзем з дапамогай аўдыярэдактара Audacity[1]. З інтэрфейсам дадзенай праграмы можна пазнаёміцца ў *Дадатку 1* (с. 153).

Канвертацыя аўдыяфайла — працэс змянення яго фармату (прыклад 2.3). Каб выканаць канвертацыю аўдыяфайлаў, можна выкарыстоўваць аўдыярэдактары (прыклад 2.4).

2.2. Загрузка і прайграванне гуказапісу ў аўдыярэдактары

Загрузку аўдыяфайла ў рэдактар Audacity пачынаюць камандай галоўнага меню **Файл** → **Открыть ...** .

[1] Даступны для спампоўвання на сайце https://www.audacityteam.org

Аўдыярэдактар аўтаматычна канвертуе файл у свой унутраны фармат, і ў акне з'яўляецца адлюстраванне гуказапісу ў выглядзе адной або дзвюх аўдыядарожак (трэкаў).

Прайграваць гуказапіс, прыспыняць і спыняць прайграванне дазваляюць першыя тры кнопкі **Панэлі прайгравання і запісу** (прыклад 2.5).

У час прайгравання направа па дарожках рухаецца вертыкальная лінія — **курсор**. Прайграванне гуказапісу заўсёды пачынаецца з зададзенага становішча курсора. Калі прайграванне спынена, курсор можна перанесці ў любое іншае месца дарожкі (прыклад 2.6).

2.3. Вылучэнне фрагмента гуказапісу

У аўдыярэдактары любы фрагмент гуказапісу можна вылучыць. Заўважым, што на **Панэлі інструментаў** рэдактара павінна быць націснута кнопка **Выделение** (прыклад 2.7).

Вылучаны фрагмент на дарожцы атрымлівае іншы колер фону (прыклад 2.8). Калі фрагмент гуказапісу вылучаны, то прайграць і праслухаць можна толькі яго. Вылучэнне фрагмента здымаецца пстрычкай мышы па свабодным месцы дарожкі.

Адрозніваюць два спосабы вылучэння фрагментаў: аглядны і дакладны.

Аглядны спосаб вылучэння фрагментаў выкарыстоўваецца з мэтай праслухоўвання фрагментаў і праводзіцца працяжкай паказальніка мышы па дарожцы.

Межы вылучанага фрагмента заўсёды можна перамясціць. Для гэтага

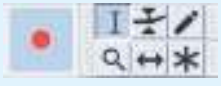
Прыклад 2.5. Кнопкі **Панэлі прайгравання і запісу.**

Назвы кнопак адлюстроўваюць іх функцыі.

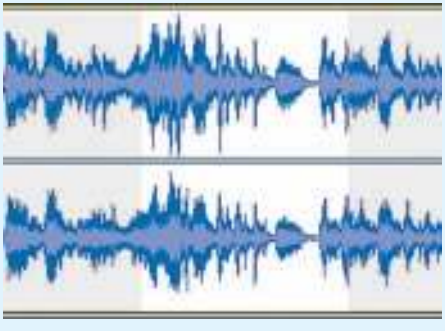
Прайграванне гуказапісу таксама можна пачаць/спыніць націсканнем на клавішу **Прабел** на клавіятуры.

Прыклад 2.6. Курсор можна перанесці ў іншае месца дарожкі пстрычкай мышы. Дакладна ў пачатак дарожкі курсор пераносіцца кнопкай **Перейти к началу дорожки** панэлі **Воспроизведения и записи**, а дакладна ў канец дарожкі — кнопкай **Перейти к концу дорожки.**

Прыклад 2.7. Кнопка **Выделение** на **Панэлі інструментаў.**



Прыклад 2.8. Відарыс вылучанага фрагмента гуказапісу.



Прыклад 2.9. Лікавыя палі на **Панэлі вылучэння фрагментаў.**

Поле **Начало выделения** паказвае адлік пачатку вылучанага фрагмента.

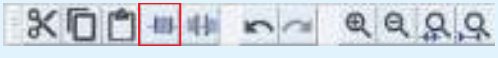
Поле **Конец выделения** паказвае адлік канца вылучанага фрагмента.

Поле **Позиция аудио** паказвае адлік становішча курсора.

Прыклад 2.10. Увядзём у лікавае поле **Начало выделения** адлік 6,9 с:

1. Пстрыкаем мышшу па разрадзе секунд у полі (разрад вылучаецца белым колерам).
2. Колам мышы ўстанаўліваем значэнне 6.
3. Аналагічна пасля пстрычкі ў разрад дзясятых долей секунды ўводзім лічбу 9.

Прыклад 2.11. Кнопка **Обрезать** на **Панэлі рэдагавання.**



Прыклад 2.12. Абрэжам гуказапіс да працягласці 2 мін 30 с.

Дакладным спосабам вылучаем фрагмент працягласцю 2 мін 30 с ад пачатку запісу і націскаем кнопку на **Панэлі рэдагавання.**

Прыклад 2.13. Для выкарыстання эфекту **Плавное затухание** ў канцы запісу вылучаюць фрагмент працягласцю 2—4 с і выкарыстоўваюць каманду **Эффекты.** Выпадае вялікі спіс эфектаў, у якім трэба знайсці патрэбны.

дастаткова падвесці паказальнік мышы да мяжы вылучанага фрагмента знутры (паказальнік прымае выгляд) і перацягнуць мяжу.

Дакладны спосаб вылучэння фрагментаў выкарыстоўваецца для іх капіравання. Гэтым спосабам фрагмент вылучаецца з дапамогай адліку часу ад пачатку запісу на **Панэлі вылучэння фрагментаў** (прыклад 2.9).

Каб вылучыць фрагмент, у палі **Начало** і **Конец выделения** ўводзяць час пачатку і канца фрагмента адпаведна (прыклад 2.10).

2.4. Абразанне фрагмента гуказапісу і выкарыстанне эфекту

Аперацыя абразання выкарыстоўваецца ў выпадках, калі працягласць гуказапісу трэба скараціць. У гуказапісе вылучаецца фрагмент патрэбнай працягласці і выкарыстоўваецца кнопка **Обрезать** на **Панэлі рэдагавання** (прыклад 2.11). У выніку ў гуказапісе застаецца толькі вылучаны фрагмент (прыклад 2.12).

Каб гук абрэзанага гуказапісу пры прайграванні не абрываўся рэзка, трэба выкарыстаць адзін з эфектаў, напрыклад **Плавное затухание** (эфект плаўнага памяншэння гучнасці гучання) (прыклад 2.13).

2.5. Захаванне аўдыяфайла

Для захавання аўдыяфайла ў рэдактары Audacity ёсць дзве магчымасці.

Калі работу з гуказапісам трэба працягнуць, то камандай **Файл** → → **Сохранить проект** гуказапіс захоўваюць ва ўнутраным фармаце рэдак-

тара як файл праекта з расшырэннем **.aup** (тып **AUP**). Праслухаць такі аўдыяфайл на плэерах немагчыма.

Каб захаваць аўдыяфайл у іншым фармаце, яго экспартуюць (канвертуюць). Для гэтага камандай **Файл** → → **Export Audio...** выклікаецца акно **Export Audio**, у якім уводзіцца імя аўдыяфайла і выбіраецца яго фармат (прыклад 2.14).

Кнопка **Параметры** ў акне **Export Audio** дазваляе выклікаць акно для ўстаноўкі параметраў якасці гуказапісу, які захоўваецца. Асноўны параметр якасці лічбавага запісу гуку і відэа носіць назву *бітрэйт*.

Бітрэйт (хуткасць патоку) — колькасць біт двайковага запісу, якая прыходзіцца на секунду праслухоўвання.

Бітрэйт вымяраецца ў кілабітах у секунду (кбіт/с, або kbps).

Чым больш бітрэйт, тым вышэйшая якасць запісу і большы аб'ём файла (прыклад 2.15).

Прыклад 2.14. Фарматы, якія падтрымлівае рэдактар Audacity пры загрузцы і экспарце аўдыяфайлаў.

AIFF (Apple) signed 16 bit PCM
WAV (Microsoft) signed 16 bit PCM
GSM 6.10 WAV (mobile)
Файлы MP3
Файлы Ogg Vorbis
Файлы FLAC
Файлы MP2
Передать внешней программе
M4A (AAC) Files (FFmpeg)
AC3 Files (FFmpeg)
AMR (narrow band) Files (FFmpeg)
WMA (version 2) Files (FFmpeg)
Custom FFmpeg Export

Прыклад 2.15. Сувязь велічыні бітрэйта і якасці двухканальнага гуказапісу ў фармаце MP3:

- **32 кбіт/с** — якасць запісу маўлення ў дыктафонах;
- **96 кбіт/с** — якасць запісу для перадачы маўлення або гуку нізкай якасці па каналах сувязі;
- **192 кбіт/с** — прымальны ўзровень якасці для запісу музыкі;
- **256 кбіт/с** — высокі ўзровень якасці для запісу музыкі;
- **320 кбіт/с** — найвышэйшы ўзровень якасці гуказапісу, падтрымліваемы фарматам MP3.

1. Што такое курсор аўдыярэдактара Audacity?
2. Якім чынам вылучаны фрагмент гуказапісу адлюстроўваецца ў рэдактары Audacity?
3. Чым адрозніваюцца аглядны і дакладны спосабы вылучэння фрагментаў?
4. Як ажыццяўляецца абразанне фрагмента гуказапісу?
5. Якое змяненне фрагмента ажыццяўляе эфект **Плавное затухание**?
6. Якія магчымасці для захавання аўдыяфайлаў мае рэдактар Audacity?
7. Што такое бітрэйт?

Практыкаванні

1 Адкрыйце ў аўдыярэдактары файл з музычнай кампазіцыяй (дадзеная кампазіцыя ліцэнзійных абмежаванняў не мае). Праслухайце яе.

2 Вылучыце любы фрагмент загружанага гуказапісу аглядным спосабам. Праслухайце яго з дапамогай кнопак **Панэлі прайгравання і запісу** і з дапамогай клавішы **Прабел**.

3 Устанавіце, чым адрозніваюцца дзеянні кнопкі **Остановить** і **Приостановить**?

4 Вылучыце і праслухайце фрагмент загружанага гуказапісу ад 31 с да 1 мін 27 с.

5 Скараціце гучанне загружанага гуказапісу да 1 мін з выкарыстаннем эфекту **Плавное затухание**.

6 Захавайце вынік папярэдняга практыкавання як праект фармату AUP і як аўдыяфайл фармату MP3 з бітрэйтам 192 кбіт/с.

§ 3. Асноўныя аперацыі рэдагавання аўдыяфайла

Асноўныя задачы рэдагавання аўдыяфайла зручна разгледзець на прыкладзе рэдагавання музычных кампазіцый. Задача скарачэння музычнай кампазіцыі вялікіх цяжкасцей не выклікае. Кампазіцыю дастаткова абрэзаць і ў канцы прыглушыць гук.

Структура музычнай кампазіцыі вызначаецца наборам яе фрагментаў. Многія музычныя кампазіцыі (песні) маюць форму, якую называюць куплетнай, г. зн., што яна складаецца з куплетаў. У куплеце, як правіла, два фрагменты: запеў і прыпеў. У кампазіцыі таксама магчымы фрагменты, якія называюць уступленнямі і пройгрышамі.

Прыклад 3.1. Музычныя кампазіцыі, якія выкарыстоўваюцца для суправаджэння выступленняў на канцэртах або капусніках (фанаграмы), часта патрабуюць павелічэння або памяншэння колькасці куплетаў. Гэта звязана з тым, што самаробныя тэксты песень рэдка супадаюць па колькасці куплетаў з зыходнымі фанаграмамі.

Акрамя таго, у такіх фанаграмах бывае неабходна выдаліць або памяняць месцамі прыпевы або пройгрышы.

Пры стварэнні фанаграм для відэафільмаў працягласці адной музычнай кампазіцыі часта таксама не хапае на ўвесь фільм. У такім выпадку яе трэба павялічваць дубліраваннем усёй кампазіцыі або некаторых яе фрагментаў.

3.1. Асноўныя задачы рэдагавання

Асноўнымі задачамі рэдагавання аўдыяфайла з'яўляюцца:

- скарачэнне аўдыяфайла;
- змяненне структуры фрагментаў аўдыяфайла.

Скарачэнне аўдыяфайла патрабуецца, калі працягласць яго гучання перавышае патрэбную, напрыклад неабходную працягласць гучання музычнага суправаджэння. Адзін са спосабаў рашэння гэтай задачы разгледжаны ў папярэднім параграфе.

Задача змянення структуры фрагментаў аўдыяфайла ўзнікае, калі асобныя фрагменты гуказапісу трэба выдаліць, пераставіць або прадубліраваць (прыклад 3.1).

Для музычных кампазіцый задача ўскладняецца тым, што вылучаць фрагменты (куплеты, прыпевы і г. д.) неабходна дакладным спосабам. Каб не парушыць плаўнасці гучання, адлікі часу для пачатку і канца фрагментаў вызначаюць максімальна дакладна.

3.2. Алгарытм знаходжання дакладнага адліку

Аўдыярэдактар Audacity дазваляе з высокай дакладнасцю знаходзіць адлікі для момантаў пачатку куплетаў, прыпеваў і іншых фрагментаў з выкарыстан-

нем інструментаў маштабавання. Інструменты маштабавання ў аўдыярэдактары ўжываюцца для павелічэння або памяншэння адлюстравання гуказапісу на дарожцы (прыклад 3.2).

Для знаходжання дакладнага адліку для момантаў пачатку музычных фрагментаў будзем выкарыстоўваць наступны алгарытм:

1. Праслухаць кампазіцыю і з любой дакладнасцю вылучыць фрагмент, які змяшчае патрэбны момант.

2. Пстрыкнуць па кнопцы **Уместить выделение**. Адлюстраванне фрагмента павялічваецца, але вылучэнне не здымаецца.

3. Праслухаць вылучаны фрагмент (можна некалькі разоў). У патрэбны момант, вызначаны на слых, пстрыкнуць па кнопцы **Приостановить**. Курсор замірае. Неабходны момант вызначаны, а адлік часу паказаны ў полі **Текущая позиция**.

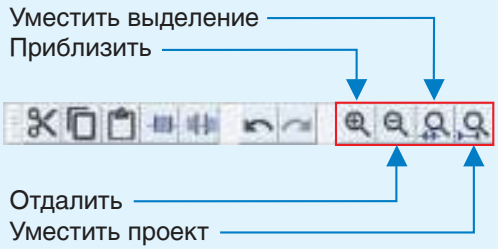
4. Для ўдакладнення адліку каля становішча курсора вылучыць невялікі фрагмент і пстрыкнуць па кнопцы **Остановить**. Курсор перамяшчаецца ў пачатак вылучанага фрагмента. Далей перайсці да кроку 2.

Удакладненне дастаткова правесці 2 разы (прыклад 3.3).

3.3. Асноўныя аперацыі рэдагавання

Рэдагаванне аўдыяфайлаў уключае наступныя асноўныя аперацыі з фрагментамі: **вылучэнне, абразанне, капіраванне, устаўку, выдаленне** і **выкарыстанне эфекту**.

Прыклад 3.2. Група кнопак інструментаў маштабавання на **Панэлі рэдагавання**:



1. Кнопка **Приблизить** павялічвае адлюстраванне гуказапісу.

2. Кнопка **Отдалить** памяншае адлюстраванне гуказапісу.

3. Кнопка **Уместить выделение** паказвае ўвесь вылучаны фрагмент.

4. Кнопка **Уместить проект** паказвае ўвесь гуказапіс.

Прыклад 3.3. Вызначым адлік часу для пачатку другога куплета ў некаторай музычнай кампазіцыі, выконваючы крокі алгарытму:

1. Кампазіцыя праслухоўваецца, і вылучаецца фрагмент з момантам пачатку другога куплета.

2. Кнопкай **Уместить выделение** адлюстраванне фрагмента павялічваецца.

3. Вылучаны фрагмент праслухоўваецца, і ў патрэбны момант праводзіцца пстрычка па кнопцы **Приостановить**.

4. Каля становішча курсора вылучаецца невялікі фрагмент і праводзіцца пстрычка па кнопцы **Остановить**.

5. Паўтараюцца крокі 2—4.

6. Паўтараюцца крокі 2—3.

Адлік — у полі **Текущая позиция**.

Прыклад 3.4. Кнопкі выдалення, капіравання і ўстаўкі фрагментаў на **Панэлі рэдагавання.**

Прыклад 3.5. Павялічым на адзін куплет працягласць музычнага суправаджэння да песні.

У адпаведнасці з алгарытмам з пункта 3.2 устанаўліваем адлікі для момантаў пачатку і канца другога куплета.

Па адліках дакладным спосабам вылучаем фрагмент запісу для капіравання. Фрагмент капіруем у буфер абмену з дапамогай кнопкі **Копировать** і націскаем клавішу-стрэлку **Управа** клавіятуры. У выніку вылучэнне знікае, а курсор зрушваецца ў канец выкарыстанага фрагмента.

Далей пстрыкаем па кнопцы **Вставить.** Фрагмент (куплет) устаўляецца.

Пры неабходнасці аперацыю ўстаўкі можна паўтарыць некалькі разоў.

З аперацыямі вылучэння, абразання і выкарыстання эфекту мы ўжо пазнаёміліся. Аперацыі капіравання, устаўкі і выдалення фрагментаў вядомыя вам па іншых праграмах і праводзяцца пры дапамозе кнопак на **Панэлі рэдагавання** (прыклад 3.4). Замест гэтых кнопак можна выкарыстоўваць камбінацыі клавіш клавіятуры, вядомыя вам з курса 6-га класа.

Аперацыі капіравання, устаўкі і выдалення фрагментаў выкарыстоўваюцца для змянення структуры фрагментаў аўдыяфайла (прыклад 3.5).

Аперацыі рэдагавання пры неабходнасці можна адмяніць і вярнуць, выкарыстоўваючы кнопкі **Отменить** і **Вернуть** на **Панэлі рэдагавання.** Гэтыя кнопкі дазваляюць адмяніць або вярнуць цэлы набор апошніх аперацый рэдагавання.

1. Якія задачы з'яўляюцца асноўнымі пры рэдагаванні аўдыяфайла?
2. Чаму пры рэдагаванні моманты пачатку куплета або прыпеву ў песні трэба знаходзіць з максімальнай дакладнасцю?
3. Для чаго прызначаны інструменты маштабавання ў аўдыярэдактары?
4. Якія аперацыі рэдагавання аўдыяфайла з'яўляюцца асноўнымі?
5. Якія аперацыі рэдагавання аўдыяфайла забяспечваюць змяненне структуры аўдыяфайла?
6. Якія кнопкі на Панэлі рэдагавання выкарыстоўваюцца пры капіраванні, устаўцы і выдаленні фрагментаў?

Практыкаванні

1 Адкрыйце ў аўдыярэдактары файл з музычнай кампазіцыяй (дадзеная кампазіцыя ліцэнзійных абмежаванняў не мае). Праслухайце яе.

2 Вылучыце і праслухайце фрагмент загружанага гуказапісу ад 2 мін 08 с да 2 мін 40 с.

3 Павялічце адлюстраванне фрагмента, вылучанага ў заданні 2, на ўсю дарожку. Вярніце адлюстраванне ўсяго гуказапісу.

4 Загружаная кампазіцыя пачынаецца з прыпеву, уключае два куплеты і заканчваецца паловай прыпева. Выдаліце другі куплет і палову прыпева пасля яго. На гэта месца ўстаўце два разы першы куплет.

5 Захавайце вынік пр. 4 як праект фармату AUP і як аўдыяфайл фармату MP3 з максімальным бітрэйтам.

§ 4. Уводзіны ў камп'ютарны відэамантаж

4.1. Відэамантаж і канвертацыя

Усе відэафайлы падзяляюцца на **відэафрагменты**, якія атрыманы ў выніку запісу, і **відэафільмы**, якія прызначаны для дэманстрацыі.

Для відэафайлаў вядомыя два віды апрацоўкі: відэамантаж і канвертацыя.

Камп'ютарны відэамантаж — гэта працэс стварэння відэафільма з відэафрагментаў з дапамогай спецыяльнага праграмнага сродку.

Праграмны сродак для відэамантажу называецца **відэарэдактарам** (прыклад 4.1). Мы будзем выкарыстоўваць відэарэдактар VideoPad[1]. З інтэрфейсам праграмы можна пазнаёміцца ў *Дадатку 1* (с. 154).

Канвертацыя відэафайла складаецца ў змяненні яго фармату (прыклад 4.2). Для канвертацыі відэафайлаў выкарыстоўваюцца праграмныя сродкі, якія называюцца **відэаканвертарамі** (прыклад 4.3). Мы будзем выкарыстоўваць відэаканвертар Convertilla[2]. З інтэрфейсам праграмы можна пазнаёміцца ў *Дадатку 1* (с. 155).

Прыклад 4.1. Сярод бясплатных відэарэдактараў для камп'ютараў вылучым рэдактары Videopad, Shotcut, OpenShot Video Editor, Windows Movie Maker (або **Кінастудыя**), Lightworks.

Распрацаваны відэарэдактары і для смартфонаў на Andriod: Video Editor, PowerDirector, KineMaster — Pro Video Editor. Ёсць яны на іншых платформах.

Прыклад 4.2. На сучасных камп'ютарах установлены кодэкі для вядомых фарматаў відэафайлаў. Пра партатыўныя ўстройствы такога сказаць нельга. Таму відэа, запампаванае на смартфон, часта не адкрываецца, і яго звычайна канвертуюць. Для відэамантажу часам даводзіцца канвертаваць відэафрагменты, знятыя на розных мабільных устройствах.

Прыклад 4.3. Шмат якія відэаканвертары дазваляюць канвертаваць і аўдыяфайлы. Сярод бясплатных відэаканвертараў для камп'ютараў неабходна вылучыць Convertilla, VSDC Free Video Converter, Any Video Converter Free, Format Factory і інш.

Канвертаваць відэафайлы можна і з дапамогай анлайн-відэаканвертараў[3].

---

[1] Даступны для спампоўвання на сайце https://www.nchsoftware.com/videopad/ru

[2] Даступны для спампоўвання на сайце http://convertilla.com/ru/

[3] Інфармацыя атрымана з сайтаў https://convert-video-online.com/ru/ і https://www.online-convert.com/ru (дата доступу 10.01.2018).

Прыклад 4.4. Адно з магчымых візуальных уяўленняў відэарада.

Ствараючы відэафільм з фрагментаў, відэарад можна змяняць — выдаляць, дабаўляць і перамяшчаць відэафрагменты.

Прыклад 4.5. Тэкставыя відэафрагменты дазваляюць уключаць у відэафільм тэкставыя надпісы (назву фільма, яго частак і інш.).

Прыклад 4.6. Усе відэафрагменты, як правіла, маюць уласнае гукавое суправаджэнне. У выніку наразання відэафрагментаў і зборкі фільма гэты від гукавога суправаджэння фільма становіцца фрагментарным і вельмі дрэнна ўспрымаецца. Тут патрабуецца дабаўленне фанаграмы.

Прыклад 4.7. Кнопка **Добавить файл(ы)** на ўкладцы **Главная**.

Загружаныя файлы аўтаматычна размяркоўваюцца па раздзелах (папках) **Видеофайлы** і **Аудиофайлы**. У раздзеле **Видеоряды** знаходзіцца файл праекта, які ствараецца аўтаматычна.

Прыклад 4.8. Відэакліп у акне перадпрагляду.

Дзеянне кнопкі ▶ **Воспроизведение/Пауза** дубліруе клавіша **Прабел**. Відэакліп можна праглядаць пакадрава.

4.2. Асноўныя аперацыі відэамантажу

Камп'ютарны відэамантаж уключае наступныя асноўныя аперацыі:

- дзяленне і абразанне відэафрагментаў;
- стварэнне відэафільма з фрагментаў;
- захаванне відэафільма.

Стварэнне відэафільма з фрагментаў выкарыстоўвае паняцце відэарада.

Відэарад — паласа з умоўных відарысаў відэафрагментаў, якая адлюстроўвае структуру відэастужкі (прыклад 4.4).

Пры неабходнасці камп'ютарны відэамантаж можа таксама ўключаць наступныя аперацыі:

- стварэнне і дабаўленне ў відэафільм тэкставых відэафрагментаў (прыклад 4.5);
- дабаўленне ў відэафільм музычнага суправаджэння (фанаграмы) са знешняга аўдыяфайла (прыклад 4.6).

4.3. Загрузка, дзяленне і абразанне відэафрагментаў

Відэарэдактар VideoPad дазваляе загружаць як відэафайлы (відэафрагменты), так і аўдыяфайлы (фанаграмы). Загрузку праводзяць з дапамогай кнопкі **Добавить файл(ы)** на ўкладцы **Главная** (прыклад 4.7).

Загружаныя файлы ў відэарэдактары VideoPad называюцца **кліпамі**.

Калі адкрыты ўкладкі **Главная** або **Клипы**, то пстрычка па відэакліпе аўтаматычна адкрывае яго ў акне перадпрагляду на ўкладцы **Предпросмотр клипа** (прыклад 4.8). У гэтым акне відэафрагмент можна прагледзець, падзяліць або абрэзаць.

Курсор у акне перадпрагляду кліпа — вертыкальная чырвоная лінія на паласе эскізаў і шкале часу, якая паказвае становішча бягучага кадра ў кліпе.

Відэакліп можна падзяліць на дзве часткі па становішчы курсора (кнопкай **Разделить**).

Для абразання кліпа дастаткова перацягнуць у новае становішча па шкале часу чырвоную і сінюю дужкі (прыклад 4.9).

Кліп абразаецца віртуальна. Гэта азначае, што межы абразання заўсёды можна ссунуць.

4.4. Стварэнне відэафільма з фрагментаў

Стварэнне відэафільма складаецца ў зборцы відэарада з відэафрагментаў.

Кліпы з раздзела **Видеофайлы** па адным пераносяцца ў акно відэарада, якое павінна знаходзіцца ў рэжыме **Раскадровка**. Кліп можна перацягнуць мышшу або вылучыць яго і выкарыстць камбінацыю клавіш: **Ctrl + Shift + End** — у канец відэарада, **Ctrl + Shift + Home** — у пачатак відэарада (прыклад 4.10).

У акне перадпрагляду цяпер можна прагледзець і відэарад. Пераключэнне паміж аб'ектамі прагляду праводзяць выбарам укладак у загалоўку акна перадпрагляду (прыклад 4.11).

4.5. Захаванне відэафільма

Захаванне відэафільма як праекта ў файле з расшырэннем **.vpj** унутранага фармату пачынаецца кнопкай **Сохранить проект** на ўкладцы **Главная**.

Прыклад 4.9. Абразанне відеакліпа ў акне перадпрагляду.

Лікавыя палі над кнопкамі **Начало** і **Конец** адкрываюцца для ўводу пстрычкай мышы. Адлікі часу пачатку і канца часткі кліпа, якую пакідаюць, можна ўвесці дакладна ў лікавых палях над кнопкамі **Начало** і **Конец**.

Прыклад 4.10. Відэафайлы, якія перанесены ў відэарад, у раздзеле **Видеофайлы** аўтаматычна адзначаюцца зялёнай птушкай.

Прыклад 4.11. Укладка **Просмотр видеоряда** акна перадпрагляду мае сваю сістэму кнопак кіравання.

У набор кнопак кіравання прайграваннем дабаўляюцца кнопкі: **Перейти к краю предыдущего клипа** і **Перейти к краю следующего клипа**.

Кнопка **Разделить** атрымлівае меню, у якім можна выбраць дарожкі для раздзялення. З'яўляецца кнопка **Снимок**, якая дазваляе зрабіць здымак кадра відэафільма. Кнопка **Включить 360 гр** прызначана для дабаўлення складанага відэаэфекту.

Курсор відэарада знаходзіцца ў акне відэарада, а кіраваць ім можна як мышшу ў акне відэарада, так і кнопкамі ў акне перадпрагляду.

Прыклад 4.12. Акно **Экспортировать видео** ў раздзеле **Настройки экспорта** файла змяшчае 7 параметраў, якія настройваюцца.

У полі 1-га радка **Имя файла:** уводзіцца імя ствараемага відэафайла.

У полі 2-га радка **Сохранить в:** уводзіцца імя папкі для захавання файла. Імя папкі выбіраецца ў акне, якое выклікаецца ў тым жа радку кнопкай **Обзор.**

У полі 4-га радка **Формат:** назва фармату будучага відэафайла выбіраецца ў спісе фарматаў. Пры неабходнасці можна выкарыстоўваць **Продвинутые настройки кодека.**

Астатнія настройкі экспарту рэкамендуецца не змяняць.

У акне **Сохранить проект как** выбіраецца папка і імя праекта.

Захаванне відэафільма ў іншым фармаце называецца *экспартам.*

Экспарт пачынаецца пстрычкай па кнопцы **Видеофайл** на ўкладцы **Экспорт.** У адказ адкрываецца акно **Экспортировать видео,** дзе праводзіцца настройка параметраў экспарту відэафайла (прыклад 4.12).

Пасля настройкі параметраў пстрычка па кнопцы **Создать** у гэтым акне пачынае даволі працяглы працэс захавання відэафільма, ход якога адлюстроўваецца ў іншым дыялогавым акне **Очередь на экспорт.**

1. Што такое камп'ютарны відэамантаж?
2. Якія асноўныя аперацыі ўключае камп'ютарны відэамантаж?
3. Якія аперацыі, акрамя асноўных, можа ўключаць камп'ютарны відэамантаж?
4. Як праводзяць загрузку відэа- і аўдыяфайлаў у рэдактар VideoPad?
5. Для чаго ў рэдактары VideoPad прызначана акно перадпрагляду?
6. Што такое курсор у акне перадпрагляду?
7. Як праводзіцца абразанне відэакліпаў у рэдактары VideoPad?
8. Ці можна змяніць абразанне відэакліпа ў рэдактары VideoPad?
9. У якім рэжыме акна відэарада праводзіцца стварэнне відэарада?
10. Якім чынам у рэдактары VideoPad кліпы перамяшчаюцца ў відэарад?
11. Які элемент акна перадпрагляду адказвае за выбар паміж праглядам кліпа і відэарада?
12. Што такое захаванне праекта ў рэдактары VideoPad?
13. Што такое экспарт відэафільма ў рэдактары VideoPad?

Практыкаванні

1 У відэарэдактары VideoPad стварыце новы праект і адкрыйце (загрузіце ў відэарэдактар) гатовыя відэафрагменты (усе відэафрагменты ліцэнзійных абмежаванняў не маюць).

2 Выканайце пералічаныя дзеянні, выкарыстоўваючы рэдактар VideoPad.

1. Абрэжце ўсе відэазапісы з раздзела **Видеофайлы** да працягласці 4 с.
2. Стварыце з фрагментаў відэарад у парадку іх нумарацыі.
3. Прагледзьце атрыманы відэарад.
4. Захавайце створаны відэафільм як праект у файле з імем **Вада.vpj.**
5. Захавайце створаны відэафільм у відэафайле **Вада.avi.**

3 Выкарыстоўваючы канвертар Convertilla, сканвертуйце файл Вада.avi ў фармат MP4.

Аперацыя	Вынік
Запусціць праграму Convertilla.	З'яўляецца акно праграмы.
Пстрыкнуць па кнопцы **Открыть.**	З'яўляецца акно **Выбор файла видео.**
У акне **Выбор файла видео** знайсці і вылучыць файл **Вода.avi**, а затым пстрыкнуць па кнопцы **Открыть.**	У ніжнім полі **Файл:** прапануюцца папка і імя файла для захавання выніка канвертацыі, якія можна змяніць.
У спісе **Формат:** выбраць MP4.	Кодэкі падбіраюцца аўтаматычна.
Пстрыкнуць па кнопцы **Конвертировать.**	Відэафайл канвертуецца і захоўваецца ў фармаце MP4.

§ 5. Камп'ютарны відэамантаж з тэкстамі і фанаграмай

5.1. Стварэнне тэкставых кліпаў

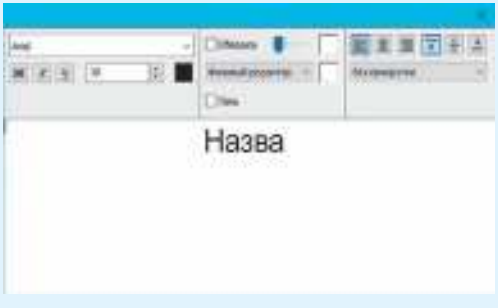
Тэкставы кліп — відэафрагмент з тэкставым надпісам на празрыстым або каляровым фоне. Для стварэння тэкставага кліпа на ўкладцы **Клипы** выкарыстоўваюць кнопку **Добавить текст.** Новы кліп пад імем **Текстовый клип** з'яўляецца ў раздзеле **Изображения** для зыходных файлаў. Для наступных тэкставых кліпаў да імя дабаўляецца нумар у дужках (прыклад 5.1).

Новы тэкставы кліп адразу адкрываецца ў акне перадпрагляду і мае празрысты фон, на што паказвае шахматная тэкстура. Адначасова адкрываецца дыялогавае акно для ўводу тэксту ў кліп (прыклад 5.2).

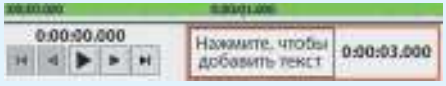
Тэкст надпісу ўводзіцца з клавіятуры. Параметры шрыфта змяняюцца інструментамі, якія размешчаны ў левай частцы панэлі акна. Памеры і становішча тэксту, які ўводзіцца, кантралююцца ў акне перадпрагляду. Там жа

Прыклад 5.1. Імёны некалькіх тэкставых кліпаў у раздзеле **Изображения.**

Прыклад 5.2. Дыялогавае акно для ўводу тэксту ў тэкставы кліп нагадвае акно простага тэкставага рэдактара.

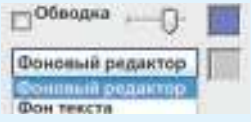


Прыклад 5.3. У акне перадпрагляду вылучаны: кнопка выкліку акна для ўводу тэксту (у форме тэкставага поля) і поле працягласці тэкставага кліпа.



Працягласць тэкставага кліпа можна змяніць пасля пстрычкі па лікавым полі. Звычайна ўводзіцца колькасць секунд. Лікавыя значэнні правей за колькасць секунд можна выдаляць.

Прыклад 5.4. Меню кнопкі **Фоновый редактор** у акне ўводу тэксту.



Выбар пункта **Заполненный фон** у меню кнопкі **Фоновый редактор** адначасова мяняе і надпіс на гэтай кнопцы. Кнопка атрымлівае назву **Заполненный фон**.

Прыклад 5.5. Паласа над **Видеодорожкой 1** для накладання тэкставага кліпа з празрыстым фонам зверху відэарада.

Прыклад 5.6. Зрух кліпа па відэадарожцы ўправа-ўлева перацягваннем мышы аблягчаецца ўтрымліваннем клавішы **Shift** клавіятуры.

Пры перацягванні тэкставага кліпа каля паказальніка мышы адлюстроўваецца велічыня зруху.

ў лікавым полі адлюстроўваецца працягласць тэкставага кліпа (прыклад 5.3).

Для стварэння ў тэкставым кліпе каляровага фону ў дыялогавым акне для ўводу тэксту выкарыстоўваюць кнопку **Фоновый редактор.** Пстрычка па кнопцы выклікае меню, у якім выбіраецца пункт **Заполненный фон**, а затым пстрычкай па квадратным полі правей пачынаецца выбар колеру (прыклад 5.4).

Цяпер ужо ў меню кнопкі **Заполненный фон** выбар пункта **Фоновый редактор** вяртае празрыстасць фону тэкставага кліпа. У дыялогавым акне для ўводу тэксту празрыстасць фону не адлюстроўваецца.

5.2. Устаўка і накладанне тэкставых кліпаў

Тэкставыя кліпы ўстаўляюцца ў відэарад як звычайныя відэафрагменты.

Тэкставыя кліпы з празрыстым фонам накладаюцца зверху відэарада ў рэжыме **Шкала времени** акна відэарада. У гэтым рэжыме бачная асноўная **Видеодорожка 1** і над ёй — паласа для новай відэадарожкі (прыклад 5.5). Менавіта ў гэту паласу (новую відэадарожку) тэкставы кліп перацягваецца мышшу з раздзела **Изображения** для файлаў.

Становішча дабаўленага тэкставага кліпа можна змяніць перацягваннем адлюстравання кліпа ўправа-ўлева па новай відэадарожцы пры дапамозе мышы (прыклад 5.6).

5.3. Відэапераходы паміж кліпамі

Як і паміж слайдамі ў камп'ютарных прэзентацыях, паміж кліпамі можна ўстаўляць эфекты змены, якія называ-

юцца **відэапераходамі.** У акне відэарада на адлюстраванні кожнага відэакліпа ёсць Х-падобны **значок відэаперахо-ду** (прыклад 5.7).

Пстрычка па значку відэаперахаду адкрывае шырокае меню для выбару віду пераходу да наступнага кліпа (прыклад 5.8).

Відэаперахад паміж кліпамі займае некаторы час, і ў гэты час кліпы павінны паказвацца адначасова. Таму рэдактар прапануе выбраць спосаб стварэння пераходу. Наступны кліп можна ссунуць лявей на час пераходу. А можна кліп не зрушваць (замарозіць яго становішча) і запоўніць час пераходу паказам толькі яго першага кадра.

5.4. Дабаўленне і настройка фанаграмы

Дабаўленне і настройку музычнай фанаграмы да фільма праводзяць у рэжыме **Шкала времени** акна відэарада. У гэтым рэжыме відэа- і аўдыядарожкі маюць па чатыры кнопкі кіравання (прыклад 5.9).

Аўдыяфайл (новую фанаграму) з раздзела **Аудиофайлы** перацягваюць у паласу пад **Аудиодорожкой 1**. Утворыцца новая аўдыядарожка. Фанаграму па ёй можна перацягваць мышшу ўправа-ўлева.

Настройка фанаграмы складаецца ў яе абразанні да працягласці відэарада і дабаўленні эфектаў **Появление** і **Исчезновение**, якія ў рэдактары прывязаны адпаведна да пачатку і да канца фанаграмы.

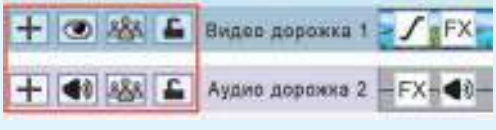
Прыклад 5.7. Значкі відэапераходу на відарысе відэафрагментаў у відэарадзе.

Значкі відэапераходу адлюстроўваюцца ў акне відэарада ў абодвух яго рэжымах.

Прыклад 5.8. Частка меню значка відэапераходу.

У меню таксама ёсць лікавае поле для змянення працягласці пераходу.

Прыклад 5.9. Кнопкі кіравання дарожкамі ў рэжыме **Шкала времени** акна відэарада. Іх прызначэнне пакажуць падказкі каля паказальніка мышы.

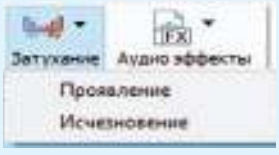


Калі кліпы ў відэарадзе маюць свае аўдыядарожкі, то ўсе яны размешчаны на **Аудиодорожке 1**. Гэту аўдыядарожку звычайна адключаюць кнопкай кіравання **Выключить звук дорожки.**

Прыклад 5.10. Курсор відэарада кнопкамі кіравання ў акне перадпрагляду ўстанаўліваюць у канец відэарада і ў гэтым жа акне пстрыкаюць па кнопцы **Разделить**. Фанаграма ў акне відэарада падзяляецца на дзве часткі.

У акне відэарада другую частку фанаграмы вылучаюць пстрычкай мышы і выдаляюць клавішай **Delete** клавіятуры.

Прыклад 5.11. Кнопка **Затухание** адкрывае меню з назвамі эфектаў.



Эфекты выкарыстоўваюцца па чарзе.

Абразанне фанаграмы да працягласці відэарада ўключае яе падзел на дзве часткі і выдаленне другой часткі (прыклад 5.10).

Каб дабавіць адзін з эфектаў, фанаграму ў акне відэарада вылучаюць і пстрыкаюць па кнопцы **Затухание** на ўкладцы **Видеоряд**. Назва эфекту выбіраецца ў выпадаючым меню (прыклад 5.11). Пад канец адкрываецца дыялогавае акно **Появление** або **Исчезновение**, у якім у тэкставае поле ўводзіцца працягласць эфекту. Практыка паказвае, што для працягласці эфектаў з'яўлення і знікнення 2 с цалкам дастаткова.

1. Што такое тэкставы кліп?
2. Якім чынам тэкставыя кліпы ўстаўляюць у відэарад?
3. Якім чынам тэкставыя кліпы накладаюць зверху відэарада?
4. Што такое пераходы паміж кліпамі?
5. Для чаго служыць значок пераходу на адлюстраванні кліпа?
6. Як праводзіцца дабаўленне музычнай фанаграмы да фільма?
7. У чым складаецца абразанне фанаграмы да працягласці відэарада?

Практыкаванні

1 Загрузіце ў відэарэдактар VideoPad файл праекта **Вада.vpj**.

2 Адкрыйце (загрузіце ў відэарэдактар) аўдыяфайл з музычнай кампазіцыяй (дадзеная кампазіцыя ліцэнзійных абмежаванняў не мае). Праслухайце яе.

3 На базе праекта **Вада.vpj** стварыце і захавайце ў файле **Вада.avi** відэафільм, пры стварэнні якога павінны быць выкананы пералічаныя патрабаванні.

1. Назва фільма «Вада» павінна быць змешчана на фоне першага кліпа.
2. Тэкставы кліп з надпісам «Канец» на чорным фоне павінен быць змешчаны ў канцы фільма.
3. Паміж кліпамі павінны быць устаноўлены пераходы.
4. У якасці фанаграмы да фільма павінна быць выкарыстана музычная кампазіцыя з загружанага аўдыяфайла.

Глава 2
АСНОВЫ АНІМАЦЫІ

§ 6. Асноўныя паняцці. Рэдактар для стварэння анімацыі

6.1. Анімацыя. Віды анімацыі

Чалавек са старажытнасці спрабаваў захаваць рух у рысунку. Сёння рух відарысаў можна бачыць у мультфільмах, у відэафільмах, у рэкламных банерах на вэб-старонках і інш. А слова *анімацыя* вядомае нават маленькім дзецям.

Анімацыя (ад лац. *animare* — ажывіць) — працэс змянення памеру, становішча, колеру або формы аб'екта з цягам часу.

Анімацыя ўяўляе сабой паслядоўную дэманстрацыю серыі відарысаў (кадраў). Кадр адлюстроўваецца некаторы час, пасля чаго знікае, а на яго месцы з'яўляецца новы.

Кадры — відарысы паслядоўных фаз руху аб'ектаў або іх частак.

Анімацыя заснавана на ўласцівасці чалавечага зроку «памятаць» відарыс на працягу некаторага часу, пасля таго як назіранне спыняецца. Чым больш кадраў змяшчае анімацыя, тым больш згладжаным будзе рух у працэсе яе прайгравання. Для стварэння ілюзіі бесперапыннага руху частата змены кадраў павінна быць не меншай за 12 кадраў у секунду.

Першапачаткова, пры падрыхтоўцы кадраў для анімацыі, кожны кадр рысаваўся асобна і цалкам, што аднімала шмат часу нават у вялікага калектыва

Самы ранні ўзор анімацыі створаны прыкладна 5000 гадоў назад. Ён змешчаны на кубку з неабпаленай гліны, знойдзеным у час раскопак у Іране. На кубку адлюстравана каза, якая падскоквае і зрывае лісты з пальмавага дрэва.

Кожны асобны відарыс казы на кубку ўяўляе сабой асобны кадр.

У 1877 г. была запатэнтавана вынаходка французa Шарля-Эміля Рэйно (1844—1918) — праксінаскоп.

Шарль-Эміль Рэйно

Праксінаскоп

Праксінаскоп уяўляў сабой устройства з адкрытага цыліндра, у цэнтры якога знаходзілася люстраная прызма. Колькасць граней прызмы адпавядала колькасці відарысаў-мініяцюр. Пры хуткім вярчэнні цыліндра на бачнай грані прызмы ствараліся ілюзія руху.

Уолт Дзіснэй

М. М. Канстанцінаў

Паслойную тэхніку ў мультыплікацыі ўпершыню выкарыстаў Уолт Дзіснэй (1901—1966) — амерыканскі мастак-мультыплікатар, заснавальнік кампаніі Walt Disney Productions.

У 1968 г. група савецкіх вучоных на чале з Мікалаем Мікалаевічам Канстанцінавым (савецкім і расійскім матэматыкам, нар. у 1932 г.) стварыла матэматычную мадэль руху жывёліны. Вылічальная машына БЭСМ-4, выконваючы праграму, прамалёўвала кадры мультфільма з анімацыяй рухаў кошкі. Для стварэння кінаплёнкі з мультфільмам кожны кадр быў раздрукаваны на прынтары, ролю пікселя адыгрывала літара «Ш».

У цяперашні час існуюць розныя тэхналогіі стварэння камп'ютарнай анімацыі. Напрыклад:

- **Запіс руху.** Акцёры ў спецыяльных касцюмах з датчыкамі выконваюць рухі, якія запісваюцца камерамі і аналізуюцца спецыяльным праграмным забеспячэннем. Выніковыя даныя пра перамяшчэнне суставаў і канечнасцей акцёраў выкарыстоўваюць у дачыненні да двухмерных скелетаў віртуальных персанажаў, дзякуючы чаму дасягаюць высокага ўзроўню дакладнасці іх руху.
- **Працэдурная анімацыя** цалкам або часткова разлічваецца камп'ютарам.
- Пры **анімацыі, якая праграміруецца,** рухі тых аб'ектаў, што аніміруюць, праграміруюцца з дапамогай браўзернай мовы JavaScript і мовы работы з Flash-дадаткамі ActionScript.

мастакоў. Затым стала выкарыстоўвацца паслойная тэхніка рысавання аб'ектаў і фонаў на празрыстых плёнках, якія накладаюцца адна на адну. Гэта знізіла працаёмкасць работ, бо не трэба было рысаваць кожны кадр цалкам. Сучасныя анімацыйныя тэхналогіі пераведзены на камп'ютарную аснову.

Камп'ютарная анімацыя — стварэнне анімацыі з дапамогай камп'ютара.

Працуючы над стварэннем камп'ютарнай анімацыі, мастак звычайна прамалёўвае пачатковае і канчатковае становішча рухомых аб'ектаў, а ўсе прамежкавыя станы разлічвае і адлюстроўвае камп'ютар. Аб'екты камп'ютарнай анімацыі змяшчаюцца на розных слаях (падобна празрыстым плёнкам у класічнай анімацыі).

Пры стварэнні камп'ютарнай анімацыі могуць выкарыстоўвацца растравыя відарысы (Gif-анімацыя) і вектарныя рысункі (Flash-анімацыя).

Вылучаюць два спосабы стварэння камп'ютарнай анімацыі:

- пакадравая анімацыя;
- разліковая анімацыя — анімацыя руху аб'ектаў і анімацыя формы.

Пры стварэнні **пакадравай анімацыі** прамалёўваюцца ўсе фазы руху аб'екта. Такая тэхналогія незаменная пры стварэнні складанай анімацыі з разнастайнай графікай.

Анімацыя руху або **формы** прадугледжвае маляванне толькі асобных кадраў. У гэтых кадрах аб'ект размяшчаецца ў пачатку і ў канцы руху. Усе

астатнія кадры — прамежкавыя. Відарыс у іх стварае камп'ютарная праграма, якая вылічае, дзе і ў які момант павінен знаходзіцца аб'ект. Разліковая анімацыя выкарыстоўваецца для стварэння анімацыйных эфектаў на вэб-старонках, а таксама пры стварэнні рэкламных, вучэбных і забаўляльных фільмаў.

6.2. Рэдактар Flash

З'яўленню камп'ютарнай анімацыі спрыяла развіццё праграм для работы з графікай.

Праграмы для работы з анімацыяй паказаны ў прыкладзе 6.1. Адным з найбольш папулярных рэдактараў для стварэння анімацыі з'яўляецца Flash. Перавага Flash у тым, што з яго дапамогай можна стварыць прыгожую анімацыю, а файлы будуць невялікага памеру. Рэдактар Flash мае інтэрфейс, шмат элементаў якога знаёмыя вам па вопыце работы ў графічных рэдактарах (гл. *Дадатак 2*, с. 156).

У пачатку работы ў рэдактары неабходна стварыць новы дакумент (**File** → → **New...**) або адкрыць той, які ўжо існуе (**File** → **Open...**).

Дакумент, створаны ў Flash, прынята называць **фільмам**. У акне рэдактара Flash можна адначасова адкрываць некалькі файлаў з фільмамі. Укладкі адкрытых файлаў размяшчаюцца пад радком меню: 

.

Для пераходу да патрэбнага файла дастаткова пстрыкнуць мышшу па ўкладцы з яго імем. Зорачка справа ад імя файла абазначае, што ў ім зроблены змяненні, якія не захаваны.

Прыклад 6.1. Для стварэння анімірованых відарысаў існуе мноства праграм, як платных, так і бясплатных. Напрыклад:

	Vectorian Giotto
	Adobe Animate
	Easy GIF Animator
	Pivot Stickfigure Animator

Гісторыя Flash пачалася ў 1996 г., калі кампанія Macromedia выпусціла прадукт пад назвай Flash.

У 2005 г. выйшла версія Macromedia Flash Professional 8. У гэтай версіі палепшана работа з графікай і анімацыяй. У тым жа годзе фірма Adobe купіла Macromedia разам з яе прадуктамі, уключаючы Flash.

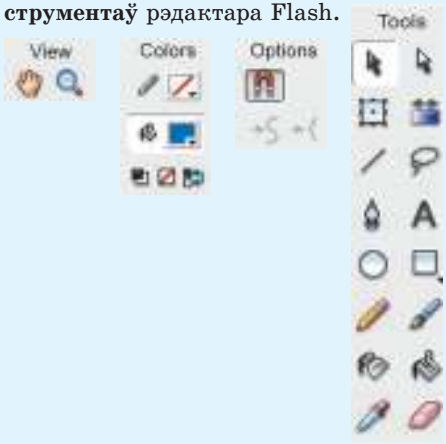
Значкі розных версій рэдактара Flash:

	Macromedia Flash MX 2002
	Macromedia Flash Professional 8 2005
	Adobe Flash Professional CC 2015

Для ўстройстваў, якія не падтрымліваюць Flash, можна захаваць фільм у фарматах HTML і GIF. У гэтым выпадку мультыплікацыя можа быць прагледжана практычна на ўсіх устройствах.

Рэдактар Flash падтрымлівае мову сцэнарыяў (апісання паводзін аб'екта) ActionScript. Выкарыстоўваючы гэту мову, можна стварыць інтэрактыўныя (якія змяшчаюць элементы ўзаемадзеяння з карыстальнікам) фільмы.

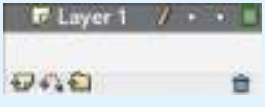
Прыклад 6.2. Раздзелы **Панэлі інструментаў** рэдактара Flash.



Прыклад 6.3. Змяненне памераў мантажнага стала.

Прыклад 6.4.

Шкала времени (Timeline)



Список слоев (Layers)

У левай частцы акна рэдактара Flash размяшчаецца **Панэль інструментаў**, якая складаецца з чатырох частак:

- **Инструменты (Tools)** — інструменты рысавання і рэдагавання;
- **Просмотр (View)** — спосаб прагляду;
- **Цвета (Colors)** — колеры абводкі і заліўкі;
- **Параметры (Options)** — настройка ўласцівасцей выбранага інструмента.

(Разгледзьце прыклад 6.2.)

У рабочай вобласці рэдактара Flash можна выконваць аперацыі стварэння і рэдагавання аб'ектаў (гл. *Дадатак 2*, с. 156). У кадр трапляюць толькі тыя аб'екты, якія размешчаны ў межах **мантажнага стала**. Астатняя частка рабочай вобласці патрэбна для папярэдніх рысункаў і для рэалізацыі эфекту паступовага ўваходу аб'екта ў кадр (або выхаду з кадра).

Памеры мантажнага стала можна змяніць (прыклад 6.3) у акне **Свойства документа (Document Properties)**.

Над рабочай вобласцю знаходзіцца **шкала времени** і **список слоев** (прыклад 6.4). **Шкала времени (Timeline)** прызначана для работы з кадрамі. **Слои (Layers)** — камп'ютарны аналаг празрыстых плёнак, якія выкарыстоўваюцца ў традыцыйнай мультыплікацыі.

У правай частцы акна змяшчаюцца дадатковыя панэлі (прыклад 6.5).

Рэдактар Flash падтрымлівае вектарную графіку, аднак дазваляе выкарыстоўваць і растравыя відарысы, імпартаваныя са знешніх файлаў. Пры захаванні фільма ў рэдактары Flash звычайна выкарыстоўваюць два тыпы

файлаў: **.fla** і **.swf**. Уласным фарматам Flash з'яўляецца фармат **FLA**. У гэтым фармаце фільм захоўваецца для наступнага рэдагавання. Для рэалізацыі магчымасці прагляду фільма яго трэба апублікаваць.

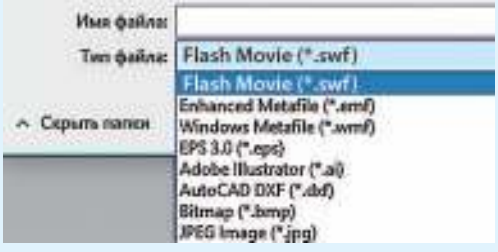
Публікацыя — захаванне фільма ў фармаце **SWF**. Апублікаваць файл можна, выканаўшы каманду **Файл** → → **Опубликовать (File** → **Publish)** або з дапамогай камбінацыі клавіш **Ctrl** + + **Enter**. Файл пры публікацыі захоўваецца ў папку, у якой знаходзіцца файл фармату FLA. Апублікаваны фільм можна прагледзець у любым прайгравальніку Flash, а таксама ў браўзеры.

Рэдактар Flash дазваляе экспартаваць вынікі работы ў іншыя фарматы. Для гэтага неабходна выканаць каманду **Файл** → **Экспорт** → **Экспорт фільма (File** → **Export** → **Export Movie)** і выбраць фармат файла (прыклад 6.6).

Прыклад 6.5. Дадатковыя панэлі.

Смеситель цветов (Color Mixer) **Библиотека (Library)**

Прыклад 6.6. Экспарт файлаў.



1. Што называюць анімацыяй?
2. Што ўяўляе сабой кадр?
3. З якой мінімальнай частатой павінна адбывацца змена кадраў, каб ствараласа ілюзія бесперапыннага руху?
4. У чым складаецца пакадравая анімацыя?
5. Чым разліковая анімацыя адрозніваецца ад пакадравай?
6. У якім фармаце трэба захаваць фільм, каб яго можна было рэдагаваць?
7. Што значыць апублікаваць фільм?

Практыкаванні

1 Адкрыйце рэдактар Flash. Стварыце новы дакумент. Выкарыстоўваючы матэрыял § 6 і Дадатка 2 (с. 156), вывучыце інтэрфейс акна рэдактара. Выканайце:

1. У меню выберыце **Вид** → **Сетка** → **Показать сетку (View** → **Grid** → → **Show Grid)**. Як змянілася старонка ў Рабочай вобласці?
2. Устанавіце памеры мантажнага стала 800 px (width) x 200 px (height).
3. Вывучыце інструменты раздзела **Просмотр (View) на Панэлі інструментаў**. З дапамогай інструмента **Рука (Hand Tool)** ( ) перамясціце старонку ў Рабочай вобласці. Павялічце бачны памер старонкі

і паменшыце яго з дапамогай інструмента **Лупа (Zoom Tool).** Настройце павелічэнне (памяншэнне) маштабу ў раздзеле **Параметры** ( ).

4. Адкрыйце (закрыйце) дадатковыя вокны **Смеситель Цветов (Color Mixer)** і **Библиотека (Library).**

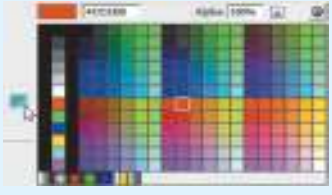
2 Адкрыйце ў рэдактары Flash файл. Захавайце файл пад новым імем. Выканайце публікацыю фільма. Прагледзьце фільм у прайгравальніку Flash.

§ 7. Стварэнне відарысаў і рэдагаванне аб'ектаў

Прыклад 7.1. Рысаванне ліній.

Перад рысаваннем лініі на **Панэлі ўласцівасцей** выбраць:

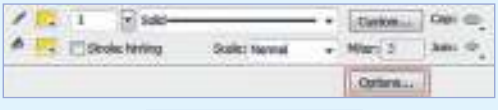
1. Таўшчыню: 

2. Колер: 

3. Стыль: 

Прыклад 7.2. Рысаванне прамавугольніка з закругленымі вугламі.

Прыклад 7.3. Дыялогавае акно **Tool Settings.**



7.1. Стварэнне відарысаў

Асноўным аб'ектам, які аніміруецца ў рэдактары Flash, з'яўляецца вектарны відарыс. Стварэнне вектарных відарысаў у Flash мае шмат агульнага з аналагічным працэсам у вектарных рэдактарах. Разгледзім асаблівасці выкарыстання інструментаў рысавання рэдактара Flash.

Для рысавання ліній у рэдактары Flash выкарыстоўваецца інструмент **Линия (Line).** Вызначыць колер, стыль і таўшчыню можна на **Панэлі ўласцівасцей** (прыклад 7.1). Лініі з вуглом нахілу, кратным 45°, рысуюцца пры націснутай клавішы Shift.

Авал можна нарысаваць з дапамогай інструмента **Овал (Oval).** Гэты інструмент таксама выкарыстоўваецца для рысавання круга і акружнасці (пры націснутай клавішы Shift).

Для рысавання прамавугольнікаў выкарыстоўваюць інструмент **Прямоугольник (Rectangle).** Каб задаць радыус закруглення вуглоў прамавугольніка, трэба на **Панэлі інструментаў** у раздзеле **Параметры** выбраць і ў дыялогавым акне **Настройки прямоугольника (Rectangle Settings)** увесці значэнне радыуса ў пікселях (прыклад 7.2).

Для рысавання многавугольнікаў і зорак трэба выбраць інструмент **Многоугольник/Звезда (PolyStar)**, разгарнуўшы спіс інструмента :

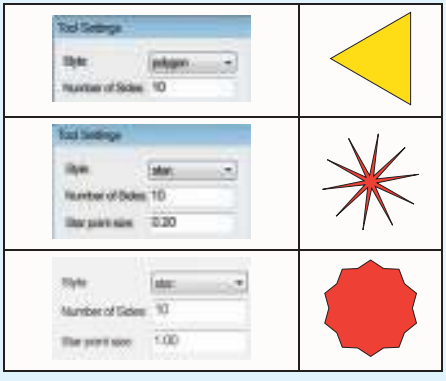
Калі інструмент актыўны, на **Панэлі ўласцівасцей** знаходзіцца кнопка Options..., якая выклікае акно настройкі інструмента (прыклад 7.3). У ім можна выбраць тып фігуры (многавугольнік/зорка), задаць колькасць старон (прамянёў) і іх памер (прыклад 7.4).

Для рысавання ліній і крывых выкарыстоўваецца інструмент **Перо (Pen)** (прыклад 7.5). З дапамогай гэтага інструмента лёгка рысаваць ломаную лінію. Пстрычкамі мышы рысуецца адрэзак або контур, які складаецца з адрэзкаў прамых ліній, злучаных вуглавымі (апорнымі) пунктамі.

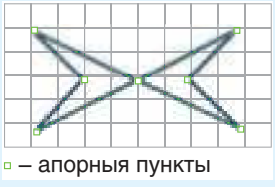
Для рысавання контураў адвольнай формы выкарыстоўваецца інструмент **Карандаш (Pencil)**. Для інструмента **Карандаш** можна выбраць розныя рэжымы рысавання (прыклад 7.6):

- **Выпрямление (Straighten)** дазваляе пераўтварыць зыходны відарыс у адну з геаметрычных фігур (прыклад 7.7);
- **Сглаживание (Smooth)** згладжвае лініі. Ступень згладжвання ліній задаецца на **Панэлі ўласцівасцей** у полі Smoothing 85;
- **Рисунок чернилами (Ink)** падобны да рэжыму **Smooth**. Ступень згладжвання нязначная і не змяняецца.

Прыклад 7.4. Рысаванне многавугольнікаў і зорак.



Прыклад 7.5. Рысаванне ломанай лініі інструментам **Перо**.



Прыклад 7.6. Выбар рэжыму рысавання для інструмента **Карандаш**.

Прыклад 7.7. Выкарыстанне інструмента **Карандаш** у рэжыме **Выпрямление**.

У час рысавання	Пасля завяршэння рысавання
	
	

Прыклад 7.8. Інструменты вылучэння:

— **Выбор (Selection);**

— **Выбор подобласти (Subselection);**

— **Петля (Lasso);**

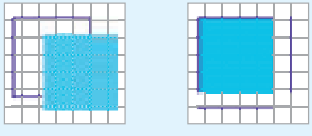
— **Произвольная трансформация (Free Transform);**

— **Перо (Pen Tool).**

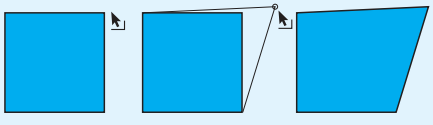
Прыклад 7.9. Вылучэнне аб'ектаў.

Увесь аб'ект	Заліўка
	
Частка абводкі	Усе часткі абводкі (с Shift)
	

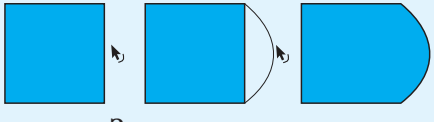
Прыклад 7.10. Выкарыстанне інструмента .



Перамяшчэнне



Змяненне становішча вяршынь



Змяненне контуру

7.2. Рэдагаванне відарысаў

Пры падрыхтоўцы кадраў анімацыі даводзіцца рэдагаваць існуючыя аб'екты. Уласцівасці абводкі і заліўкі (колер, форма, узаемнае размяшчэнне) могуць змяняцца карыстальнікам незалежна адна ад адной. Заліўку і абводку можна выдаляць паасобку.

Перш чым выканаць якія-небудзь дзеянні з аб'ектам, яго неабходна вылучыць. Рэдактар Flash валодае вялікім спектрам інструментаў вылучэння (прыклад 7.8). Асноўны інструмент вылучэння — **Выбор (Selection).** З яго дапамогай можна вылучыць увесь аб'ект, склаўшы яго ў прамавугольнік. Той жа вынік атрымаецца, калі выканаць двайную пстрычку па аб'екце. Некалькі аб'ектаў вылучаюцца пры націснутай клавішы Shift. Вылучаны аб'ект пакрываецца дробнай сеткай. Пстрычкай можна вылучаць заліўку або абводку паасобку (прыклад 7.9).

З дапамогай інструмента можна выконваць наступныя аперацыі рэдагавання аб'ектаў:

- **Перамяшчэнне.** Вылучыць аб'ект. Пры з'яўленні крыжыка з двунакіраваных стрэлак перамясціць аб'ект.
- **Змяненне становішча вяршынь.** Падвесці курсор да вяршыні і перамясціць вяршыню.
- **Змяненне контуру.** Не вылучаючы контур, падвесці курсор да контуру і перамясціць у патрэбным напрамку.

(Разгледзьце прыклад 7.10.)

Шмат якія аперацыі змянення аб'ектаў могуць быць выкананы з дапамогай інструмента **Произволь-**

ная трансформация (Free Transform). Пры рабоце з інструментам пасля вылучэння аб'екта на **Панэлі інструментаў** становяцца даступнымі кнопкі выбару рэжымаў:

- **Поворот и наклон (Rotate and Skew);**
- **Масштабирование (Scale);**
- **Искажение (Distort);**
- **Изгиб (Envelope).**

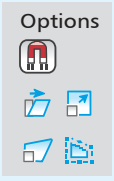
(Разгледзьце прыклад 7.11.)

Прыклад 7.12 ілюструе змяненне аб'екта пры выкарыстанні інструмента **Произвольная трансформация** ў розных рэжымах. Трансфармацыя выконваецца з дапамогай маркераў, размешчаных на вылучальнай рамцы. Кожны маркер злучаны з пэўнай аперацыяй. Розным маркерам адпавядае свой варыянт паказальніка мышы.

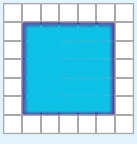
З дапамогай каманд меню **Изменить (Modify)** можна выканаць аперацыі над аб'ектамі:

- Пераўтварэнне (змяненне памеру, паварот, адлюстраванне і інш.) — **Трансформация (Transform);**
- Групоўку (аб'яднанне некалькіх аб'ектаў у адзін) — **Группировка (Group);**
- Выраўноўванне (па гарызанталі і па вертыкалі, адносна меж аб'ектаў і адносна цэнтра) — **Выравнивание (Align).**

Прыклад 7.11. Кнопкі выбару рэжыму для інструмента **Произвольная трансформация.**

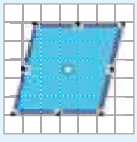


Прыклад 7.12. Выкарыстанне інструмента **Произвольная трансформация.**

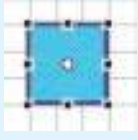


Зыходны відарыс

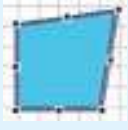
Паварот



Нахіл



Маштабаванне



Скажэнне

Выгін

1. Якія інструменты рэдактара Flash выкарыстоўваюцца для рысавання?
2. У якім рэжыме інструмента **Карандаш** рысуюцца геаметрычныя фігуры?
3. Для чаго прызначаны інструмент **Выбор**?
4. Якія пераўтварэнні аб'екта можна выканаць з дапамогай інструмента **Произвольная трансформация**?

Практыкаванні

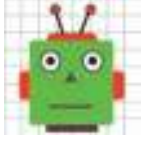
1 Адкрыйце файл. Дапоўніце відарыс аўтамабіля. Пры рысаванні акна выкарыстоўвайце градыентную заліўку. Скапіруйце відарыс акна, выкарыстайце ў дачыненні да копіі аперацыю адлюстравання. Захавайце змяненні ў файле.

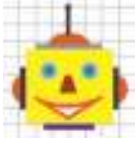
2 Стварыце відарысы, выкарыстоўваючы рэкамендацыі.

Відарысы	Рэкамендацыі
	Выкарыстоўвайце інструменты і з настройкамі:  Выкарыстайце інструмент для пераўтварэння ліній у дугі або нарысуйце дугі як частку акружнасці.
	Для рысавання выкарыстоўвайце інструменты і . Інструмент выкарыстоўвайце ў розных рэжымах. Карэкціруйце відарыс з дапамогай інструмента .
	Капіраванне акон выконвайце, утрымліваючы клавішу Shift. Для выгіну дома выкарыстоўвайце інструмент . Скарэкціруйце контур з дапамогай інструмента .

Захавайце відарысы ў фармаце FLA.

3 Нарысуйце аднаго з робатаў. Для рысавання выкарыстоўвайце інструменты , , і інструмент рэдагавання . Захавайце відарыс з імем, якое адпавядае эмоцыі робата.





§ 8. Слаі. Бібліятэка аб’ектаў. Імпарт аб’ектаў

8.1. Работа са слаямі

Слаі — найважнейшы элемент анімацыі. Выкарыстанне слаёў дазваляе ствараць складаныя шматпланавыя сцэны фільма, рэдагуючы кожны аб’ект на асобным слоі. Адзін са слаёў можа выкарыстоўвацца ў якасці фону, другі — змяшчаць аніміраваныя аб’екты, а трэці — элементы гукавога суправаджэння фільма (прыклад 8.1).

Імёны слаёў паказаны злева ад часавай шкалы. Пасля стварэння файла ў спісе слаёў знаходзіцца адзін слой з імем **Layer 1 (Слой 1)**.

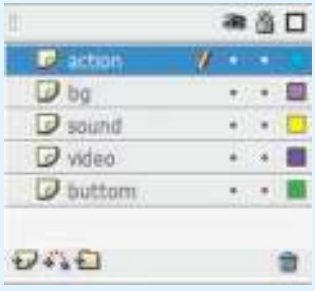
Для стварэння новага слоя патрабуецца вылучыць той слой, над якім вы хочаце змясціць новы, а затым націснуць кнопку **Вставить слой (Insert Layer)**. Новы слой з’яўляецца ў спісе слаёў над вылучаным слоем, як бачна з прыкладу 8.2.

Новаму слою прысвойваецца імя **Layer** з указаннем парадкавага нумара. Гэта імя звычайна замяняюць імем, якое тлумачыць прызначэнне або змест слоя.

Для зручнасці работы са слаямі ў Flash рэалізавана магчымасць захоўвання кожнага набору ўзаемазвязаных слаёў у асобнай папцы слаёў (прыклад 8.3).

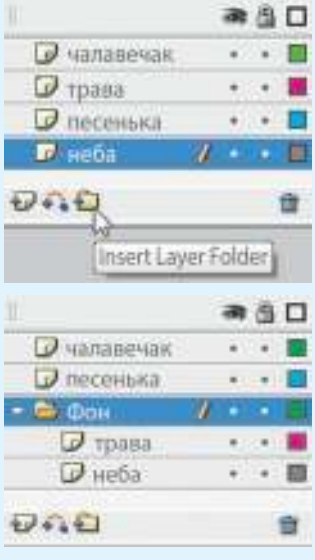
Усе слаі абсалютна празрыстыя. Аб’екты, размешчаныя на розных слаях, візуальна ўспрымаюцца як элементы адзінай сцэны. Аб’ект, змешчаны на верхнім слоі, засланяе аб’екты, якія

Прыклад 8.1. Выкарыстанне слаёў.

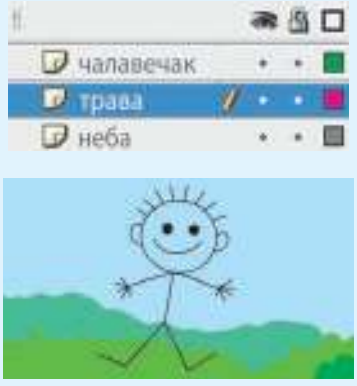


Прыклад 8.2. Стварэнне новага слоя.

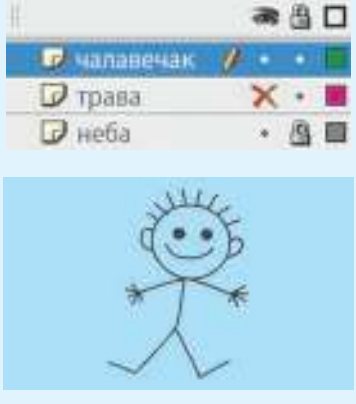
Прыклад 8.3. Стварэнне папкі слаёў.



Прыклад 8.4. Адлюстраванне слаёў у залежнасці ад іх размяшчэння.



Прыклад 8.5. Адлюстраванне слаёў з рознымі ўласцівасцямі.



знаходзяцца ў той жа пазіцыі на ніжніх слаях (прыклад 8.4).

Можна змяняць парадак размяшчэння слаёў. Аб'екты аднаго слоя рэдагуюцца незалежна ад аб'ектаў іншых слаёў. Пры неабходнасці можна адначасова выбраць аб'екты з розных слаёў і працаваць з імі як з адзіным цэлым. Напрыклад, змяніць іх колер.

Выдаліць слой (папку) можна пстрычкай па кнопцы **Удалить слой (Delete Layer)**, размешчанай у ніжнім правым вугле **Панэлі кіравання слаямі**. Пры выдаленні папкі выдаляюцца таксама і тыя слаі, якія ў яе ўваходзяць.

Значкі справа ад імя слоя адлюстроўваюць яго ўласцівасці:

- **Актыўнасць.** У гэтым слоі аб'екты ствараюць або рэдагуюць. Слой вылучаецца ў спісе колерам і вызначаецца значком .
- **Бачнасць.** Аб'екты схаванага слоя нябачныя, слой вызначаецца .
- **Блакіроўка.** Слой вызначаецца ў спісе значком .

(Разгледзьце прыклад 8.5.)

На заблакіраваным або схаваным слоі нельга рэдагаваць і ствараць аб'екты. Пры стварэнні анімацыі для нерэдагуемых слаёў устанаўліваецца блакіроўка і адключаецца бачнасць.

8.2. Бібліятэка аб'ектаў

Пры стварэнні анімацыі ўзнікае неабходнасць выкарыстоўваць некаторыя аб'екты некалькі разоў. Для захоўвання такіх аб'ектаў у рэдактары Flash прызначана **Библиотека (Library).**

Аб'екты, змешчаныя ў бібліятэку, называюць **сімваламі**. Выкарыстанне сімвалаў істотна паскарае працэс распрацоўкі фільма.

Адзін раз створаны відарыс можна шматразова выкарыстоўваць як у адным, так і ў розных фільмах.

Існуюць тры тыпы сімвалаў:

1. **Графика (Graphic).** Змяшчае відарыс аднаго кадра.

2. **Кнопка (Button).** Змяшчае кнопкі, якія рэагуюць на дзеянні карыстальніка і кіруюць прайграваннем фільма.

3. **Клип (MovieClip).** Можа змяшчаць анімацыю з любой колькасцю кадраў.

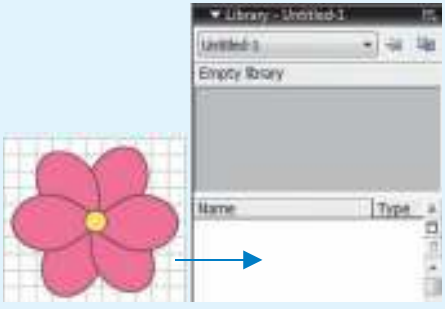
Для пераўтварэння аб'екта ў сімвал трэба націснуць клавішу **F8** або перацягнуць відарыс у акно **Библиотека**. Затым у вобласці прагляду ў верхняй частцы панэлі **Библиотека** можна ўбачыць відарыс сімвала, а ў спісе сімвалаў — імя сімвала (прыклад 8.6).

Копію сімвала, змешчаную ў рабочую вобласць, называюць **экзэмплярам**. Экзэмпляры могуць значна адрозніваца ад самога сімвала. Пры рэдагаванні экзэмпляра сімвал не змяняецца. І насупраць, любыя змяненні сімвала прыводзяць да адпаведных змяненняў усіх яго экзэмпляраў.

Кожны экзэмпляр мае ўласныя ўласцівасці, якія могуць рэдагавацца без змянення адпаведных уласцівасцей сімвала. Так, можна змяняць колер і празрыстасць экзэмпляра, перавызначаць яго тып (напрыклад, пераўтварыць графічны сімвал у кнопку).

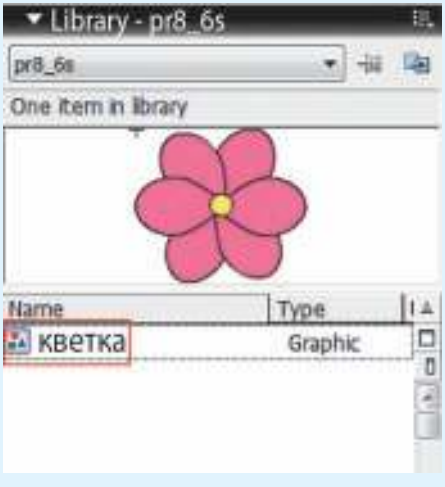
Прыклад 8.6. Стварэнне сімвала тыпу **Графика**.

1. Перацягнуць вылучаны відарыс у спіс сімвалаў панэлі **Библиотека**:



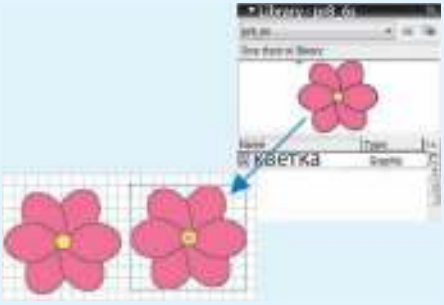
2. У акне **Convert to Symbol** выбраць тып **Graphic** і даць імя сімвалу:

3. Прагледзець змест бібліятэкі:

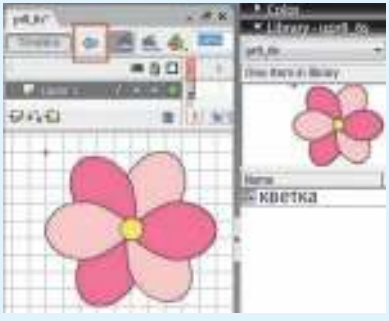


Прыклад 8.7. Стварэнне экзэмпляраў сімвала.

Перацягнуць відарыс сімвала з панэлі **Библиотека** на палатно.



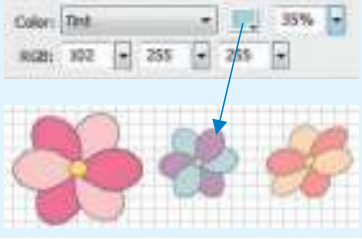
Прыклад 8.8. Рэдагаванне сімвала.



Прыклад 8.9. Рэдагаванне экзэмпляра.

1. З дапамогай інструментаў трансфармацыі.

2. Змяненне колеру на панэлі **Properties**.



Можна таксама нахіляць, круціць або маштабаваць экзэмпляр без таго, каб уздзейнічаць на сімвал.

Пры вылучэнні экзэмпляра вакол яго з'яўляецца блакітная рамка (прыклад 8.7).

Рэдагаванне сімвала ажыццяўляецца ў спецыяльным рэжыме, перайсці ў які можна з дапамогай двайной пстрычкі па відарысе сімвала на панэлі **Библиотека**. Для выхаду з гэтага рэжыму неабходна націснуць кнопку ⇦, размешчаную над спісам слаёў (прыклад 8.8).

Для рэдагавання ўласцівасцей экзэмпляра выкарыстоўваюцца інструменты трансфармацыі і інструменты **Панэлі ўласцівасцей (Properties)** (прыклад 8.9).

Пераўтварыць экзэмпляр сімвала ў звычайную вектарную графіку можна, выбраўшы каманду **Break Apart** з кантэкставага меню экзэмпляра або націснуўшы камбінацыю клавіш Ctrl + B.

8.3. Імпарт і выкарыстанне аб'ектаў

Рэдактар Flash падтрымлівае шмат якія фарматы для імпарту відарысаў — JPEG, GIF, PNG і PSD.

Існуюць наступныя варыянты імпарту відарысаў (меню **Файл** → **Импорт**):

- непасрэдна ў рабочую вобласць **(Import to Stage)**;
- у бібліятэку **(Import to Library)**.

Імпартаваныя ў бібліятэку відарысы ўяўляюць сабой сімвалы.

Для імпартавання відарысаў з празрыстасцю фону рэкамендуецца выкарыстоўваць фармат PNG (прыклад 8.10).

Імпартаваныя растравыя відарысы пераўтвараюцца ў вектарныя для наступнага рэдагавання з дапамогай каманды **Изменить (Modify) → Bitmap → Trace Bitmap** (прыклад 8.11).

Памер імпартаваных відарысаў можа не адпавядаць памеру мантажнага стала. Для ўстранення гэтай неадпаведнасці змяняюць або памер мантажнага стала, або памер відарыса.

Выбар каманды **Открыть внешнюю библиотеку (Open External Library)** меню **Файл (File) → Import (Импорт)** дазваляе адкрываць бібліятэку любога файла Flash (фармату .fla) і выкарыстоўваць яе сімвалы.

Прыклад 8.10. Імпарт відарысаў розных фарматаў.

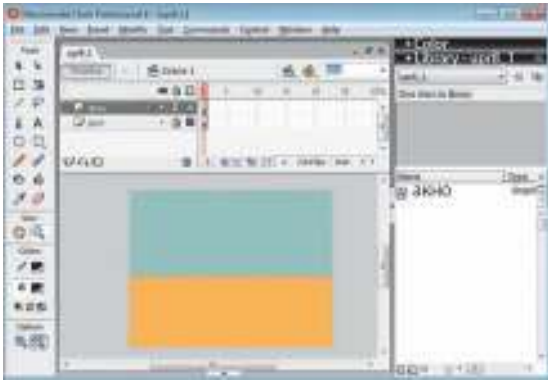
Прыклад 8.11. Рэдагаванне вектарнага відарыса, пераўтворанага з растравага.

1. З якой мэтай выкарыстоўваюць слаі?
2. Для чаго прызначана бібліятэка аб’ектаў?
3. Што ўяўляе сабой сімвал?
4. Як пераўтварыць аб’ект у сімвал?
5. Што такое экзэмпляр сімвала?
6. Як перайсці ў рэжым рэдагавання сімвала?
7. Куды можна імпартаваць відарысы ў рэдактары Flash?

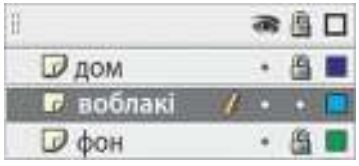
Практыкаванні

1 Адкрыйце файл і выканайце заданні.

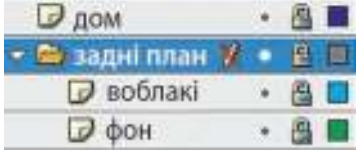
1. Змяніце паслядоўнасць слаёў (слой «дом» перацягніце ўверх). Зніміце бачнасць слоя «дом».
2. Паміж слаямі «дом» і «фон» стварыце слой «воблакі». Нарысуйце ў гэтым слоі воблака і пераўтварыце ў сімвал. Змясціце некалькі экзэмпляраў сімвала



«воблака» на слоі. Пераўтварыце экзэмпляры (памер, паварот, нахіл, колер). Адкрыйце бачнасць слоя «дом».



3. Стварыце ў спісе слаёў папку «задні план» і перацягніце ў яе слаі «фон» і «воблакі». Устанавіце блакіроўку папкі.



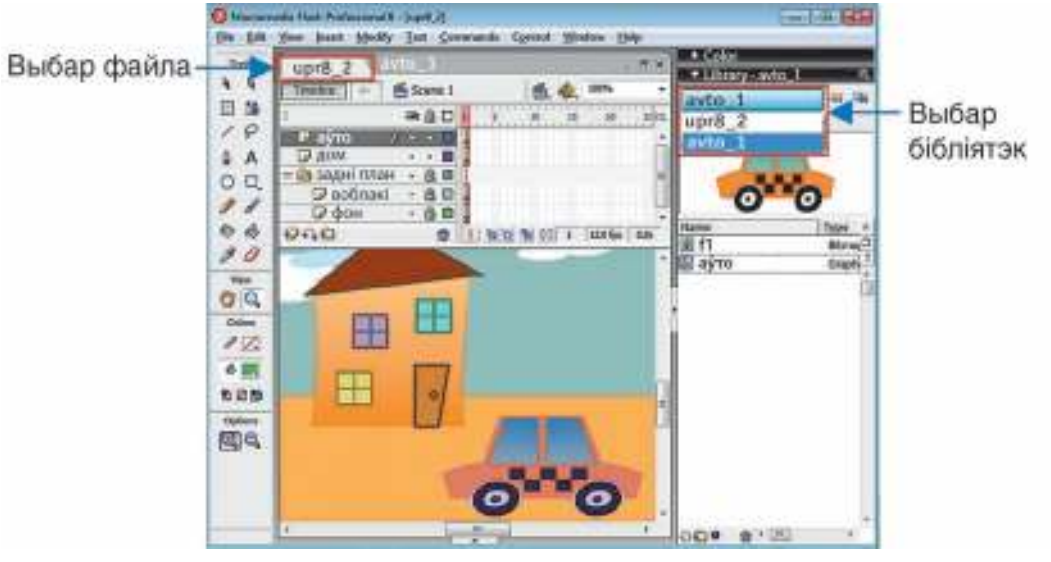
4. Актывізуйце слой «дом», зніміце яго блакіроўку. Змясціце на слой некалькі экзэмпляраў сімвала «акно» з бібліятэкі. Змяніце колер акон, выкарыстоўваючы магчымасці панэлі Properties.

5. Захавайце вынік работы ў файл з імем `upr8_2.fla`.

2 У рэдактары Flash адкрыйце адначасова файл `upr8_2.fla` і файл з відарысам аўтамабіля, атрыманы ў выніку выканання практыкавання 1 пасля § 7. Выканайце пералічаныя дзеянні:

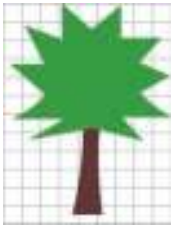
1. Стварыце сімвал відарыса аўтамабіля на новым слоі.

2. Вярніцеся ў файл `upr8_2.fla` і, выбраўшы бібліятэку файла з аўтамабілем, стварыце экзэмпляр аўтамабіля. Змяніце колер экзэмпляра.



3. Захавайце змяненні ў файле.

3* У файле `upr8_2.fla` дабаўце новыя слаі з відарысамі і пераўтварыце іх у сімвалы:



Дапоўніце відарыс у `upr8_2.fla` экзэмплярамі сімвалаў. Змяніце сімвалы ў адпаведнасці з рысункам. Захавайце змяненні ў файле.

§ 9. Пакадравая анімацыя

Асноўным інструментам пры стварэнні анімацыі з'яўляецца шкала часу. З яе дапамогай можна выконваць розныя аперацыі з кадрамі.

На шкале часу кожнаму слою адпавядае радок з сеткай. Кожнай ячэйцы адпавядае асобны кадр. Лікі над шкалой абазначаюць нумары кадраў. На кадр, які знаходзіцца на мантажным стале, паказвае **маркер кадра** — чырвоны прамавугольнік з лініяй (прыклад 9.1).

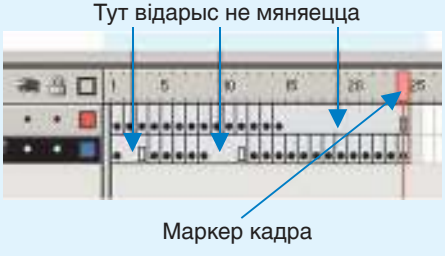
Кадры, змест якіх вызначаецца аўтарам анімацыі, называюцца **ключавымі**.

Адлюстраванне кадраў на шкале часу залежыць ад іх прызначэння.

(Разгледзьце прыклад 9.2.)

Пры выкананні аперацый з кадрамі можна выкарыстоўваць каманды кантэкставага меню кадра, а таксама «гарачыя» клавішы:

Прыклад 9.1. Адлюстраванне кадраў на шкале часу.



Прыклад 9.2. Адлюстраванне кадраў на шкале часу.

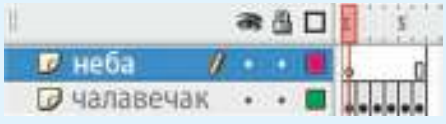
	Ключавы кадр са змесцівам (рэдагуецца, з'яўляецца зыходным)
	Ключавы кадр без змесціва
	Простыя кадры (падаўжаюць бачнасць папярэдняга ключавога кадра)
	Прамежкавыя кадры (адлюстроўваюць трансфармацыю аб'екта паміж двума ключавымі кадрамі)

Прыклад 9.3. Устаўка ключавога кадра.

1. Вылучыць ячэйку сеткі.

2. Націснуць F6.

Прыклад 9.4. Стварэнне фонавага слоя.



Пры стварэнні пакадравай анімацыі рэкамендуецца прытрымлівацца наступных правілаў:

1. Выконваючы сціскание і расцягванне, захоўвайце аб'ём (падаўжаючы аб'ект, не забывайце звужаць яго).

2. Аб'ект не павінен занадта рэзка спыняцца і замірацъ — колькасць кадраў павінна быць такой, каб рух аб'екта быў плаўным.

3. Павольны выхад і павольны ўваход.

4. Правільны разлік часу — важна задаць дастаткова часу, каб падрыхтаваць гледача да чакання дзеяння, самога дзеяння і рэакцыі на дзеянне.

Звычайна на мантажным стале знаходзіцца адзін кадр. Каб спрасціць размяшчэнне і рэдагаванне пакадравай анімацыі, можна праглядаць на мантажным стале два і некалькі кадраў адначасова. Такі рэжым называецца рэжымам калькавання.

- Insert Keyframe, F6 — устаўка копіі ключавога кадра;
- Insert Blank Keyframe, F7 — устаўка пустога ключавога кадра;
- Insert Frame, F5 — устаўка простага кадра;
- Clear Keyframe, Shift + F6 — ачыстка ключавога кадра;
- Remove Frames, Shift + F5 — выдаленне кадра.

Перад устаўкай кадра неабходна спачатку вылучыць ячэйку сеткі, прызначаную для змяшчэння новага кадра (прыклад 9.3). Аперацыі вылучэння групы кадраў выконваюцца гэтак жа, як і аперацыі вылучэння групы іншых аб'ектаў.

Для змянення становішча кадра ў шкале часу дастаткова перацягнуць яго ў патрэбную пазіцыю.

Анімацыя, якая цалкам складаецца з ключавых кадраў, называецца **пакадравай анімацыяй**.

Пры стварэнні пакадравай анімацыі для кожнага ключавога кадра трэба вызначыць працягласць прайгравання.

Чым больш кадраў у пакадравай анімацыі, тым натуральней рухі персанажаў. Каб рухі аб'ектаў у анімацыі не былі рэзкімі, а фільм занадта кароткім або хуткім, можна дабавіць:

- ключавыя кадры з прамежкавым становішчам аб'екта, які аніміруецца;
- простыя кадры пасля кожнага ключавога кадра.

Для дабаўлення да анімацыі фону трэба стварыць яшчэ адзін слой. Пры стварэнні новага слоя адразу ствара-

ецца пусты ключавы кадр і простыя кадры па даўжыні ўжо гатовай анімацыі. Аднаго ключавога кадра дастаткова, бо фон статычны (прыклад 9.4).

Перад публікацыяй анімацыю трэба пратэсціраваць — выбраць у меню рэдактара каманду **Контроль → Тестирование фильма (Control → Test Movie)**.

У прыкладзе 9.5 паказаны працэс стварэння пакадравай анімацыі.

Перавагай пакадравай анімацыі з'яўляецца яе натуральнасць, паколькі кожны наступны кадр не падобны да папярэдняга.

Недахопы пакадравай анімацыі:

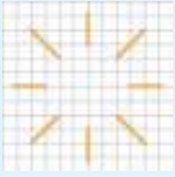
- пры неабходнасці змянення ўсёй анімацыі трэба змяняць кожны кадр;
- займае вялікі аб'ём, бо даводзіцца захоўваць інфармацыю пра кожны кадр.

У Flash ёсць інструменты для калькавання анімацыі. Напрыклад, функцыя **Луковая шелуха (Onion Skin)**, якая дазваляе аніматару праглядаць любую колькасць паслядоўных кадраў.

Пасля выбару рэжыму калькавання змест вылучанага ў дадзены момант кадра адлюстроўваецца ў поўным колеры, а змест суседніх кадраў адлюстроўваецца ў выглядзе напаўпразрыстага шлейфа.

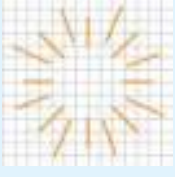
Прыклад 9.5. Стварэнне пакадравай анімацыі.

1. Адкрыць файл sun.fla з відарысам прамянёў сонца.



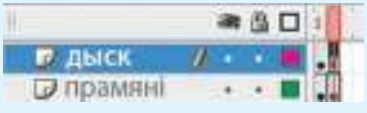
2. Стварыць копію кадра.

3. Змяніць другі кадр, дапоўніўшы яго копіяй прамянёў. Для змянення копіі выкарыстаць інструмент (паварот, маштабаванне). Сумясціць цэнтры копіі і зыходнага відарыса **(Modify → Align → Horizontal Center + Vertical Center)**:



4. Стварыць новы слой «дыск» з відарысам дыска сонца:

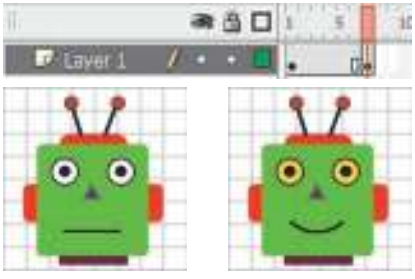
5. У кадры 2 слоя «дыск» стварыць просты кадр:



6. Захаваць змяненні ў файле.
7. Пратэсціраваць анімацыю.
8. Апублікаваць фільм.

1. Для чаго прызначана шкала часу?
2. Якія кадры называюцца ключавымі?
3. Як ствараецца дубль ключавога кадра?
4. Якую анімацыю называюць пакадравай?
5. Якую камбінацыю клавіш выкарыстоўваюць для тэсціравання анімацыі?

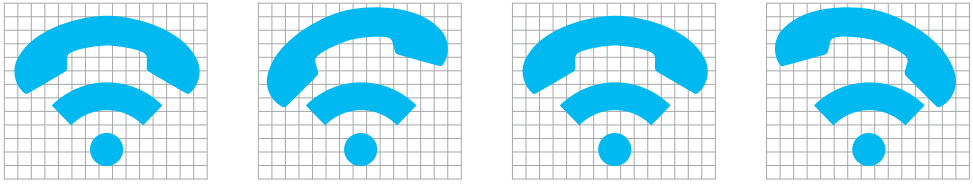
Практыкаванні



1 Адкрыйце файл. З дапамогай пакадравай анімацыі змяніце эмоцыю робата, як паказана на рысунку.

Захавайце файл. Апублікуйце атрыманую вамі анімацыю.

2 Адкрыйце файл (вынік выканання задання 1 з практыкавання 2 пасля § 7). Стварыце пакадравую анімацыю з чатырох ключавых кадраў.



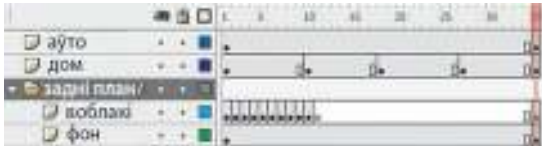
Захавайце змяненні ў файле. Апублікуйце анімацыю.

3 Адкрыйце файл (вынік выканання практыкавання 1 пасля § 8). Стварыце пакадравую анімацыю з пяці ключавых кадраў у слоі «дом».

1. У кадрах 2 і 4 змяніце колер акон.
2. Дабаўце простыя кадры ў адпаведных слаях.

4 Дапоўніце анімацыю, атрыманую пасля выканання практыкавання 3, анімацыяй перамяшчэння воблакаў.

Захавайце анімацыю ў файле `upr9_4.fla`. Апублікуйце анімацыю.



5* Дабаўце да анімацыі, атрыманай пасля выканання задання 4, анімацыю сонца з прыкладу 9.5.

§ 10. Анімацыя руху

10.1. Прамалінейны рух

Пры рабоце над пакадравай анімацыяй кадры, якія змяшчаюць прамежкавыя фазы руху аб'ектаў, вы стваралі самі. Стварэнне прамежкавых кадраў можна даручыць камп'ютару. У гэтым выпадку дастаткова задаць стан аб'екта анімацыі ў пачатку і ў канцы руху, а ўсе прамежкавыя фазы руху разлічыць праграма Flash.

Такая анімацыя называецца **анімацыяй руху** і выкарыстоўваецца толькі ў дачыненні да сімвалаў. На слоі можа знаходзіцца адзін аніміраваны сімвал.

Відарысы ў анімацыі руху знаходзяцца ў ключавых кадрах. Прамежкавыя кадры захоўваюць спасылкі на першы ключавы кадр, што дазваляе скараціць памер файла з анімацыяй.

Уключыць/выключыць рэжым анімацыі руху можна з дапамогай каманд кантэкставага меню пачатковага кадра:

- **Создать анимацию движения (Create Motion Tween);**
- **Удалить движение (Remove Tween).**

(Разгледзьце прыклад 10.1.)

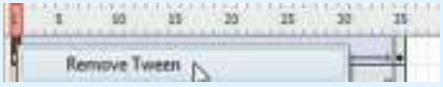
Самая простая анімацыя руху — рух па прамой (прыклад 10.2).

Пасля выканання каманды **Создать анимацию движения** паміж ключавымі кадрамі з'яўляецца суцэльная стрэлка, размешчаная на ліловаблакітным фоне. Прамежкавыя кадры адлюстроўваюць паслядоўнасць фаз

Прыклад 10.1. Выкарыстанне кантэкставага меню пачатковага кадра.

1. Стварэнне руху:

2. Выдаленне руху:

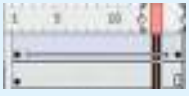


Прыклад 10.2. Стварэнне анімацыі прамалінейнага руху.

1. Змясціць фонавы рысунак і парашутыста на розныя слаі.
2. Відарыс парашутыста пераўтварыць у сімвал (F8).
3. Вылучыць у слоі з фонам кадр 45 і дабавіць простыя кадры (F5).
4. Вылучыць у слоі з парашутыстам кадр 45 і пераўтварыць яго ў ключавы (F6).
5. Змяніць памеры і становішча парашутыста ў кадры 45:

6. Стварыць анімацыю руху парашутыста па прамой.

Прыклад 10.3.

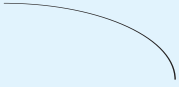


З'яўленне замест стрэлкі штрыхавой лініі азначае, што дапушчана памылка і анімацыя не створана.

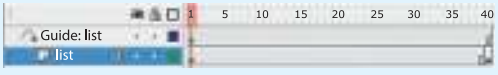
Прыклад 10.4. Стварэнне анімацыі руху па траекторыі.

1. Нарысаваць або імпартаваць відарыс.

2. Стварыць накіравальны слой і адлюстраваць на ім траекторыю.



3. У кадр 40 слоя з відарысам уставіць копію ключавога кадра (F6), а ў слой з траекторыяй дабавіць простыя кадры (F5).

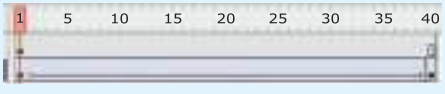


4. Упэўніцца, што на **Панэлі інструментаў** актыўны параметр **Защелка:** 

.

5. Перамясціць відарыс у ключавых кадрах у становішча, калі пункты трансфармацыі супадаюць з канцамі лініі траекторыі.

6. У слоі з відарысам уключыць рэжым анімацыі руху.



руху (прыклад 10.3). Пры стварэнні анімацыі руху важна, каб:

1. Пачатковая і канчатковая фазы анімацыі былі атрыманы з дакладных копій аднаго і таго ж аб'екта.

2. У пачатковым і ў канчатковым ключавых кадрах знаходзіліся толькі адзіныя аб'екты.

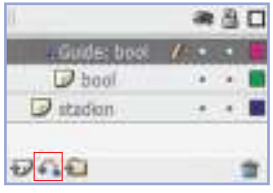
10.2. Рух па траекторыі

Для стварэння анімацыі руху па зададзенай траекторыі неабходны спецыяльны **накіравальны слой**.

Накіравальны слой змяшчаецца над слоем з аніміруемым аб'ектам. Для дабаўлення накіравальнага слоя трэба:

1. Вылучыць слой з аніміруемым аб'ектам.

2. Націснуць кнопку **Добавить движение (Add Motion Guide)**:



На накіравальным слоі адлюстроўваецца траекторыя, па якой будзе перамяшчацца аніміруемы аб'ект. Пры праглядзе фільма лінія траекторыі не адлюстроўваецца. У слоі з аніміруемым аб'ектам задаецца анімацыя руху, як у выпадку прамалінейнага руху (прыклад 10.4).

Для прывязкі аб'екта да траекторыі неабходна:

1. Уключыць параметр **Защелка (Snap)** на **Панэлі інструментаў**.

2. Сумясціць пункт трансфармацыі аб'екта (кружок у цэнтры) з пачаткам лініі траекторыі (у першым кадры) і канцом (у канчатковым кадры).

Калі аб'ект не прывязаць да траекторыі, то ён будзе рухацца прамалінейна.

Пры стварэнні анімацыі руху некалькіх аб'ектаў для кожнага слоя з аб'ектамі ствараецца свой слой накіравальных. Таксама можна прывязаць некалькі слаёў да аднаго слоя накіравальных, каб некалькі аб'ектаў рухаліся па адной і той жа траекторыі.

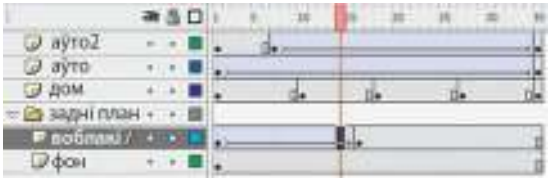
1. У дачыненні да якіх аб'ектаў можна выкарыстаць анімацыю руху?
2. Як стварыць анімацыю руху?
3. Як выдаліць анімацыю руху?
4. Для чаго прызначаны накіравальны слой?
5. Як ствараецца накіравальны слой?
6. Як звязаць траекторыю з аб'ектам руху?

Практыкаванні

1 Адкрыйце файл. Выканайце прыклад 10.2. Захавайце змяненні ў файле `upr10_1.fla`. Апублікуйце анімацыю.

2 Унясіце змяненні ў файл `upr9_4.fla` (вынік выканання практыкавання 4 пасля § 9).

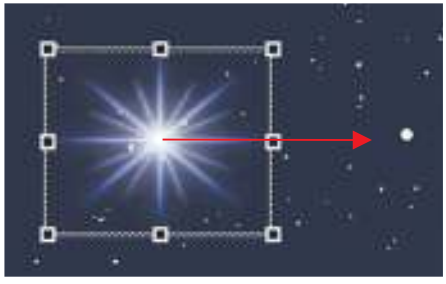
1. У слоі «воблакі» замяніце пакадравую анімацыю на анімацыю руху.
2. Стварыце анімацыю прамалінейнага руху аўтамабіля злева направа.
3. Дабаўце другі экзэмпляр аўтамабіля і стварыце прамалінейную анімацыю руху аўтамабіля справа налева.



Захавайце створаную анімацыю ў файле `upr10_2.fla`. Апублікуйце анімацыю.

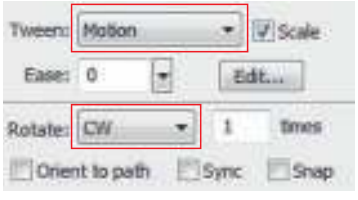
3 Адкрыйце файл. Стварыце анімацыю руху зорачкі па крузе, выконваючы дадзеныя ніжэй рэкамендацыі.

1. З дапамогай інструмента перанясіце пункт трансфармацыі (цэнтр будучага вярчэння) на некаторую адлегласць.



2. У кадр 30 устаўце копію ключавога кадра.

3. Перайдзіце на першы кадр і адкрыйце **Панэль уласцівасцей.** У спісе **Определение (Tween)** выберыце **Движение (Motion).** У спісе **Поворот (Rotate)** выберыце прымусовае вярчэнне **па** (CW) або **супраць (CCW) гадзіннікавай стрэлкі.**



4. Запусціце анімацыю на прагляд. Захавайце змяненні ў файле. Апублікуйце анімацыю.

4 Дапоўніце анімацыю з практыкавання 3 анімацыяй руху ракеты.

Захавайце змяненні ў файле `upr10_4.fla`. Апублікуйце анімацыю.

5 Стварыце анімацыю руху па траекторыі. Выканайце імпарт відарысаў з файлаў.

Фонавы відарыс	Аніміруемы відарыс	Траекторыя руху
		
		

§ 11. Анімацыя формы

Анімацыя формы — плаўнае змяненне аб'екта анімацыі.

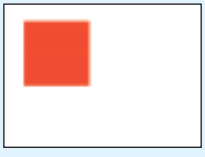
Пры стварэнні анімацыі формы аб'ектам з'яўляецца не экзэмпляр, як пры анімацыі руху, а звычайны вектарны відарыс. Колькасць прымітываў у відарысе можа быць рознай у пачатку і ў канцы анімацыі.

У працэсе анімацыі формы відарыс можа падзяліцца на некалькі незалежных фрагментаў, кожны з якіх будзе паступова трансфармавацца. Ці, наадварот, некалькі незалежных відарысаў пры анімацыі, паступова змяняючы абрыс (памеры, колер, форму), могуць стаць часткамі адзінага відарыса. Таму анімацыя формы лепш за ўсё падыходзіць для простых відарысаў без абводкі.

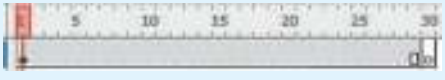
У рамках анімацыі формы можна таксама змяняць становішча і колер аб'ектаў (прыклад 11.1). Перамяшчэнне

Прыклад 11.1. Стварэнне анімацыі формы.

1. У першым кадры нарысаваць квадрат:

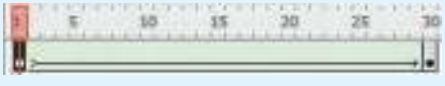


2. Дабавіць пусты ключавы кадр у кадры 30 (F7)



і нарысаваць у ім круг:

3. Стварыць анімацыю формы:



Прыклад 11.2. Адлюстраванне прамежкавых фаз.

Пераход адной формы ў іншую не заўсёды прадказальны. Калі зыходны і канчатковы аб'екты змяшчаюць некалькі фігур, то складана здагадацца, як будзе адбывацца трансфармацыя.

Прыклад 11.3. Выкарыстанне растравых відарысаў.

1. Імпартаваць відарыс на мантажны стол.

2. Пераўтварыць сімвал у звычайную графіку.

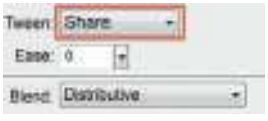
3. У кадрах 15 і 30 дабавіць копіі ключавых кадраў. Змяніць у кадры 15 відарыс:

4. Стварыць анімацыю формы.

аб'ектаў пры анімацыі формы заўсёды прамалінейнае. Для стварэння анімацыі формы патрабуецца:

1. Выбраць пачатковы ключавы кадр (або любы кадр паміж двума ключавымі).

2. На **Панэлі ўласцівасцей** выбраць **Форма (Shape):**



Паміж ключавымі кадрамі пасля стварэння анімацыі формы павінна з'явіцца суцэльная стрэлка на салатавым фоне. Прамежкавыя кадры будуць адлюстроўваць паслядоўнасць фаз змянення формы (прыклад 11.2). Пры заданні адмоўных значэнняў у полі параметра **Замедление (Ease)** анімацыя будзе ісці паскорана, а пры ўводзе дадатных значэнняў гэтага параметра — запаволена.

Вам ужо вядома, што імпартаваныя растравыя відарысы ўяўляюць сабой сімвалы. Іх можна выкарыстоўваць пры стварэнні анімацыі формы, спачатку пераўтварыўшы ў звычайную графіку (прыклад 11.3).

Пры анімацыі аб'ектаў з градыентнай заліўкай у зыходнага і канчатковага аб'екта павінен быць адзін і той жа тып градыентнай заліўкі. Flash не ўмее пераўтвараць лінейны градыент у радыяльны і наадварот.

Пры выкарыстанні анімацыі формы ў дачыненні да вельмі складанага аб'екта можна дамагчыся лепшага выніку, калі разбіць адну анімацыю на некалькі і самастойна нарысаваць прамежкавыя контуры.

1. У дачыненні да якіх аб'ектаў можна выкарыстаць анімацыю формы?
2. Якія ўласцівасці аб'ектаў могуць змяняцца пры анімацыі формы?
3. Як стварыць анімацыю формы?
4. Якім чынам можна выкарыстоўваць імпартаваныя відарысы пры стварэнні анімацыі формы?

Практыкаванні

1 Стварыце анімацыю формы «Сэрца на далоні». Відарыс для фону імпартуйце.

Нумар кадра	Відарыс
1	
10	

Захавайце анімацыю ў файле `upr11_1.fla`. Апублікуйце анімацыю.

2 Адкрыйце файл. Стварыце анімацыю формы «Мігатлівая зорка».

1. У кадры 1 пераўтварыце відарыс зоркі ў звычайную графіку.
2. У кадр 15 устаўце ключавы кадр.
3. Змяніце памеры зоркі, утрымліваючы клавішу Alt.
4. Стварыце анімацыю формы.
5. Захавайце змяненні ў файле `upr11 _ 2.fla` і выканайце публікацыю.

3 Стварыце анімацыю формы «Пераўтварэнне».

1. Нарысуйце ў кадры 1 круг з градыентнай заліўкай.
2. Перайдзіце ў кадр 12 і пераўтварыце яго ў пусты ключавы кадр.
3. Імпартуйце ў кадр 12 растравы відарыс кавуна. Пераўтварыце відарыс кавуна ў вектарны і выдаліце фон (гл. пункт 8.3, с. 40—41). Прывядзіце ў адпаведнасць памер і месца размяшчэння цэнтра круга і кавуна.
4. Кадр 30 пераўтварыце ў просты кадр.
5. Паміж ключавымі кадрамі стварыце анімацыю формы.
6. Захавайце вынік і выканайце публікацыю.

4* Дапоўніце вынікі выканання практыкаванняў 1 і 2 анімацыяй формы так, каб атрымаліся названыя сюжэты.

1. «Разбітае сэрца» — сэрца падае і разбіваецца.
2. «Зорнае неба» — некалькі мігатлівых зорак, перыяд і інтэнсіўнасць мігацення ў кожнай свае.

§ 12. Анімацыя тэксту

Прыклад 12.1. Тэкставае поле, якое пашыраецца.

Прыклад 12.2. Тэкставае поле фіксаванай шырыні.

Прыклад 12.3. Тэкставы блок.

Прыклад 12.4. Змяненне празрыстасці тэксту.

Прыклад 12.5. Трансфармацыя тэксту.

У любы фільм у рэдактары Flash можа быць дабаўлены тэкст. Для дабаўлення тэксту ў фільм выкарыстоўваецца інструмент **A Тэкст (Text Tool)**.

Месца, дзе будзе ўводзіцца тэкст, вызначаецца пстрычкай мышы. З'яўляецца рамка, якая вылучае тэкст, — тэкставае поле. У правым верхнім вугле рамкі маецца маркер, выгляд якога вызначае тып тэкставага поля:

- круглы маркер адпавядае тэкставаму полю, якое пашыраецца (прыклад 12.1). Гэта аднарадковае тэкставае поле, шырыня якога аўтаматычна павялічваецца пры ўводзе тэксту;
- прамавугольны маркер адпавядае тэкставаму полю фіксаванай шырыні (прыклад 12.2). Калі чарговы сімвал не змяшчаецца на бягучым радку, у такім полі выконваецца аўтаматычны пераносна наступны радок. Шырыню поля можна змяніць, перацягнуўшы маркер.

Пасля завяршэння ўводу ўтвараецца тэкставы блок з блакітнай рамкай (прыклад 12.3).

На **Панэлі ўласцівасцей** для тэксту могуць быць устаноўлены: памер, шрыфт, стыль, інтэрвал, колер і спосаб выраўноўвання. Можна змяняць празрыстасць колеру тэксту (прыклад 12.4).

Тэкставыя блокі гэтак жа, як і відарысы, можна трансфармаваць: паварочваць, маштабаваць, нахіляць (прыклад 12.5). Пры гэтым захоўваецца магчымасць рэдагавання сімвалаў тэксту.

Перад выкарыстаннем аперацыі скажэння або выгіну да ўсяго тэксту (прыклад 12.6) тэкставы блок неабходна пераўтварыць у графічны аб'ект. Тое ж самае выконваецца для трансфармацыі асобных сімвалаў. Магчымыя два варыянты выкарыстання каманды **Разбить (Break Apart)**:

1. Аднаразовае выкарыстанне каманды. Адбываецца падзел тэксту на сімвалы з захаваннем уласцівасцей кожнага з іх як асобнага фрагмента тэксту. Сімвалы тэксту змяшчаюцца ў асобныя рамкі. Да кожнага сімвала могуць быць ужыты любыя аперацыі, дапушчальныя для тэкставага блока (прыклад 12.7).

2. Каманда выкарыстоўваецца двойчы. Сімвалы тэксту пераўтвараюцца ў графічныя аб'екты і вылучаюцца як графічныя аб'екты. У гэтым выпадку ў дачыненні да кожнага сімвала тэксту можна выкарыстоўваць градыентную заліўку, скажэнне і выгін (прыклад 12.8).

Пасля выкарыстання каманды **Разбить** тэкст ужо нельга рэдагаваць як адзіны аб'ект.

У дачыненні да тэкставых блокаў выкарыстоўваецца толькі анімацыя руху (прыклад 12.9).

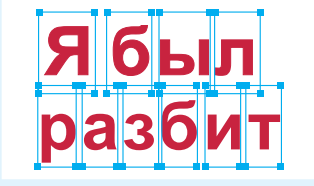
Анімацыю формы ў дачыненні да тэксту можна выкарыстаць толькі

Прыклад 12.6. Трансфармацыя графічнага аб'екта, які змяшчае тэкст.

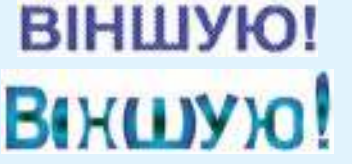
1. Скажэнне.

2. Выгін.

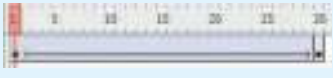
Прыклад 12.7. Змяненне ўласцівасцей кожнага сімвала і выкарыстанне аперацый трансфармацыі.



Прыклад 12.8. Градыентная заліўка і выгін сімвалаў.



Прыклад 12.9. Рух тэксту з вярчэннем.



Прыклад 12.10. Выкарыстанне анімацыі формы ў дачыненні да тэксту.

1. У стартавы кадр анімацыі імпартаваць відарыс і пераўтварыць яго ў графіку (Ctrl + B).

2. Праз некалькі кадраў уставіць пусты ключавы кадр (фінішны кадр анімацыі).

3. У фінішны кадр анімацыі дабавіць тэкст і пераўтварыць яго ў графіку (Ctrl + B двойчы).

4. Стварыць анімацыю формы.

пасля пераўтварэння сімвалаў тэксту ў графіку (прыклад 12.10).

Каб сімвалы тэксту аніміраваліся паасобку, неабходна:

1. Пераўтварыць тэкст у графіку.

2. Выканаць каманду кантэкставага меню **Распределить по слоям (Distribute to Layers)**. У выніку кожная літара, застаючыся ў першым кадры, апынецца ў сваім слоі. Слаі аўтаматычна атрымаюць адпаведныя імёны.

Калі некалькі літар, якія ідуць запар, павінны паводзіць сябе аднолькава, іх можна пераўтварыць у адзін графічны аб'ект.

1. Якія віды трансфармацыі выкарыстоўваюцца ў дачыненні да тэкставага блока?
2. Для чаго тэкст пераўтвараюць у графічны аб'ект?
3. У якім выпадку захоўваецца магчымасць рэдагавання тэксту?
4. Які від анімацыі выкарыстоўваецца ў дачыненні да тэкставых блокаў?
5. Калі можна выкарыстоўваць анімацыю ў дачыненні да асобных сімвалаў тэксту?

Практыкаванні

1 Выканайце прыклады 12.5—12.10. Захавайце вынікі работы.

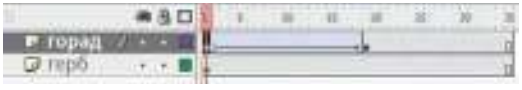
2 Стварыце анімацыю па апісанні.

1. Устанавіце памеры дакумента 700 × 200 пікселяў.
2. Стварыце два слоі — «герб» і «горад».
3. У першы кадр слоя «герб» імпартуйце відарыс герба вашага горада (раённага або абласнога цэнтра). Пры неабходнасці змяніце памеры відарыса, захоўваючы прапорцыі.
4. У першым кадры слоя «горад» стварыце тэкставы блок з назвай горада. Устанавіце шрыфт, памер і колер тэксту ў адпаведнасці з узорам.

5. Кадр 19 слоя «горад» пераўтварыце ў ключавы.

6. У стартавым кадры слоя «горад» змяніце памер і месца размяшчэння тэкставага блока.

7. У кадры 35 двух слаёў дабаўце простыя кадры.
8. У слоі «горад» стварыце анімацыю руху.
9. Захавайце вынік і выканайце публікацыю.



3 Адкрыйце файл `upr11_3.fla` (вынік выканання практыкавання 3 пасля § 11). Дабаўце ў фільм анімацыю тэксту.

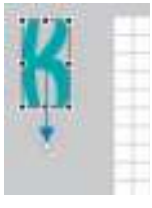
1. Стварыце тэкставы блок і напішыце слова «КАВУН» пад відарысам круга ў першым кадры.

2. Разбіце тэкст на асобныя літары і размяркуйце іх па слаях.

3. У слоі «К» пераўтварыце вылучаную літару ў графіку. Устанавіце ў якасці колеру літары адзін з адценняў зялёнага і надайце літары незвычайную форму.

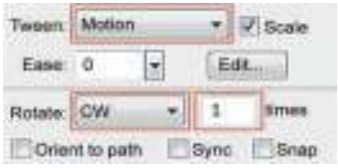
4. Пераўтварыце літару ў сімвал для стварэння анімацыі руху.
5. Стварыце канчатковую фазу будучай анімацыі. На некаторым выдаленні па шкале часу ўстаўце копію першага кадра: 

.



6. Вылучыце першы кадр у слоі «К». Змяніце стартавыя значэнні анімацыі. Для гэтага: вынесіце відарыс літары за межы мантажнага стала; змяніце прапорцыі літары і перанясіце пункт трансфармацыі (цэнтр будучага вярчэння).

7. Стварыце анімацыю руху. Літара, паступова змяняючы свае памеры і прапорцыі, павінна вылятаць за межы мантажнага стала і, здзейсніўшы абарот, заняць сваё месца.



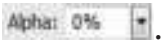
8. Кіруючыся дзеяннямі, апісанымі ў п. 3–7, стварыце анімацыю руху ўсіх літар. Змяняючы месца размяшчэння стартавых кадраў на шкале часу, дасягніце эфекту з'яўлення літар па чарзе. Літары могуць рухацца па розных кругавых або ламаных траекторыях, здзяйсняць павароты, змяняць свае памеры і прапорцыі. Час руху і колькасць паваротаў літар таксама могуць быць рознымі.

Захавайце змяненні ў файле `upr12_2.fla` і выканайце публікацыю.

4 Стварыце анімацыю «Вада».

1. Вызначыце памеры дакумента 400 × 200 пікселяў.
2. У першым кадры ў цэнтры мантажнага стала стварыце тэкставы блок, як паказана на рысунку. Устанавіце шрыфт, памер і колер тэксту ў адпаведнасці з узорам.

3. Разбіце тэкст на асобныя літары і размяркуйце іх па слаях.
4. У кадр 5 усіх чатырох слаёў дабаўце ключавы кадр. Пераўтварыце ў гэтым кадры кожную літару ў графічны аб'ект.
5. У кадр 20 слаёў «В», «А» і «Д» устаўце пусты ключавы кадр.
6. Дабаўце ў кожны слой у кадры 20 тэкст у адпаведнасці: «В» — «Н», «А» — «2» і «Д» — «О». Паменшыце памер сімвала «2». Літары ў кадрах 5 і 20 павінны мець прыблізна аднолькавае размяшчэнне.
7. Кадр 20 слоя «А» зрабіце ключавым. Устанавіце празрыстасць колеру



.

8. Пераўтварыце ў кадры 20 кожны сімвал у графічны аб'ект.
9. Стварыце анімацыю формы (кадр 5 — стартавы, кадр 20 — фінішны).
10. Стварыце слой «вада» ў канцы спіса слаёў. У кадр 1 слоя імпартуйце відарыс фону.
11. Ва ўсіх слаях кадра 35 устаўце просты кадр:

12. Захавайце вынік і выканайце публікацыю.

Глава 3
АСНОВЫ АЛГАРЫТМІЗАЦЫІ І ПРАГРАМІРАВАННЯ

§ 13. Асноўныя алгарытмічныя канструкцыі

13.1. Алгарытм і алгарытмічныя канструкцыі

У 7-м класе вы пазнаёміліся з асноўнымі алгарытмічнымі канструкцыямі. Для рашэння задач па праграмiраванні былі вылучаны асноўныя этапы (прыклад 13.1).

Алгарытм — канечная паслядоўнасць дакладных дзеянняў, фармальнае выкананне якіх дазваляе атрымаць рашэнне задачы для любога дапушчальнага набору зыходных даных.

Выканаўца — чалавек, група людзей або тэхнічнае ўстройства, здольныя правільна выконваць каманды алгарытмаў. У далейшым будзем разглядаць толькі выканаўцу-ўстройства з абмежаваным наборам каманд. Набор каманд аднаго выканаўцы называюць **сістэмай каманд выканаўцы**. Каманды камп'ютарнага выканаўцы могуць быць рэалізаваны ў выглядзе працэдур і функцый.

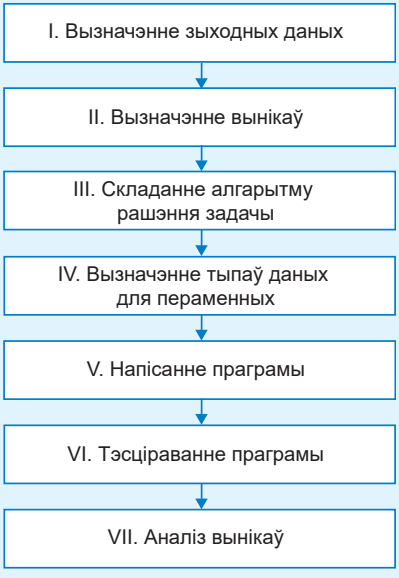
Усе каманды выканаўцы дзеляць на групы:

1. Каманды, якія непасрэдна выконвае выканаўца.

2. Каманды, якія змяняюць парадак выканання іншых каманд выканаўцы.

Любы алгарытм можа быць запісаны з выкарыстаннем трох базавых алгарытмічных канструкцый:

Прыклад 13.1. Этапы рашэння задачы па праграмiраванню:



У працэсе рашэння задачы некаторыя этапы даводзіцца паўтараць да таго часу, пакуль аналіз вынікаў не пакажа, што задача рэшана правільна.

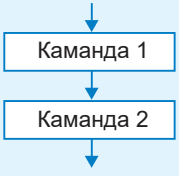
Для пошуку памылак можна выкарыстоўваць сродкі наладкі праграм (гл. *Дадатак* 3, с. 161—162).

У 1966 г. італьянскія матэматыкі Карада Бём (1923—2017) і Джузэпэ Джакапіні (1936—2001) сфармулявалі і даказалі становішча структурнага праграмiравання, згодна з якім любы выконваемы алгарытм можа быць пераўтвораны да структураванага выгляду, калі ход выканання алгарытму вызначаецца пры дапамозе трох структур кіравання: паслядоўнасцей, галінаванняў і цыклаў.

Цалкам канцэпцыя структурнага праграміравання была распрацавана ў сярэдзіне 70-х гг. пры ўдзеле Э. Дейкстры.

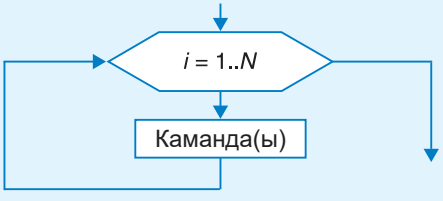
Прыклад 13.2 Блок-схемы алгарытмічных канструкцый:

1. Паслядоўнасць:

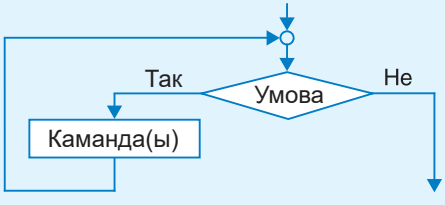


2. Цыкл:

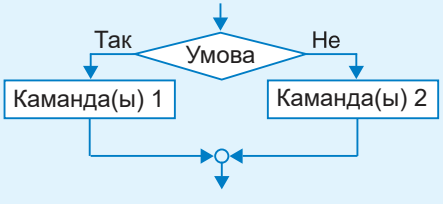
1) цыкл з параметрам (значэнне параметра змяняецца ад 1 да *N*)



2) цыкл з перадумовай



3. Каманда галінавання



паслядоўнасць, цыкл і **галінаванне** (прыклад 13.2).

Каманды цыкла і галінавання кіруюць парадкам выканання іншых каманд у праграме і належаць да **каманд кіравання (кіруючым канструкцыям).**

Паслядоўнасць каманд, выканаўцам якой з'яўляецца камп'ютар, называецца **праграмай.** Праграма ўяўляе сабой запіс на некаторай фармальнай мове — мове праграміравання. Камандамі ў мове праграміравання лічаць:

- аператары (аператар прысвойвання, аператар галінавання, аператар цыкла і інш.);
- выклікі дапаможных алгарытмаў (як убудаваных у бібліятэкі, так і створаных карыстальнікам).

13.2. Алгарытмічная канструкцыя *паслядоўнасць*

Алгарытмічная канструкцыя ***паслядоўнасць*** — паслядоўнасць каманд алгарытму, што выконваюцца ў тым парадку, у якім яны запісаны. Сярод каманд, якія ўтвараюць алгарытмічную канструкцыю *паслядоўнасць*, адсутнічаюць каманды, што змяняюць парадак выканання іншых каманд.

У 7-м класе, вывучаючы мову Pascal, вы выкарыстоўвалі наступныя каманды (прыклад 13.3):

- працэдуры для ўводу і выводу даных;
- аператар прысвойвання.

Для **ўводу даных** прызначана каманда `read()`. У дужках праз коску пералічваюцца імёны пераменных, значэнні якіх неабходна ўвесці.

Для **выводу даных** выкарыстоўваюць каманду `write()`. Яна дазваляе выводзіць тэкставыя паведамленні і лікавыя значэнні. Тэкставыя паведамленні запісваюцца ў двукоссі, выводзяцца ў выглядзе паслядоўнасці сімвалаў так, як запісаны, і не аналізуюцца пры выкананні.

Пры выкарыстанні каманды `writeln()`; пасля выводу паведамлення або ліку адбываецца перавод курсора на наступны радок.

Аператар прысвойвання прызначаны для таго, каб:

- задаваць значэнні пераменным;
- вылічаць значэнне выразу (вынік будзе запісаны як значэнне пераменнай).

Фармат запісу аператара прысвойвання:

`<імя пераменнай> := <выраз>;`

У запісе арыфметычнага выразу выкарыстоўваюцца знакі матэматычных дзеянняў: складанне (+), адніманне (–), множанне (*), дзяленне (/), а таксама цэлалікавага дзялення (`div`) і знаходжання астачы (`mod`). Неабходна памятаць, што аперацыя дзялення (/) выкарыстоўваецца пры вылічэннях з данымі тыпу `real`. Для даных тыпу `integer` выкарыстоўваюцца аперацыі `div` і `mod`.

Нярэдка ў адной праграме даводзіцца выконваць адну і тую ж паслядоўнасць каманд некалькі разоў.

Прыклад 13.3. Дадзены x, y. Напісаць праграму для вылічэння значэння выразу

$$a = \frac{2x}{7 + y^2}(x - y).$$

Этапы выканання задання

I. Вызначэнне зыходных даных: пераменныя x, y.

II. Вызначэнне вынікаў: пераменная a.

III. Алгарытм рашэння задачы.

1. Увод зыходных даных.
2. Вылічэнне значэння выразу.
3. Вывад выніку.

IV. Апісанне пераменных: усе пераменныя, вызначаныя для рашэння задачы, маюць тып real.

V. Праграма:

```
var x,y,a: real;
begin
  write('Увядзіце x =');
  read(x);
  write('Увядзіце y =');
  read(y);
  a:= 2 * x * (x-y)/(7 + y * y);
  writeln('a =',a);
end.
```

VI. Тэсціраванне праграмы:

Запусціць праграму і ўвесці значэнні $x = 3.8$, $y = 2.7$. Вынік:

Окно вывода

```
Увядзіце x = 3.8
Увядзіце y = 2.7
a = 0.585024492652204
```

VII. Правільнасць вылучэнняў праверыць на калькулятары.

У прыкладзе алгарытмічная канструкцыя *паслядоўнасць* утворана камандамі:

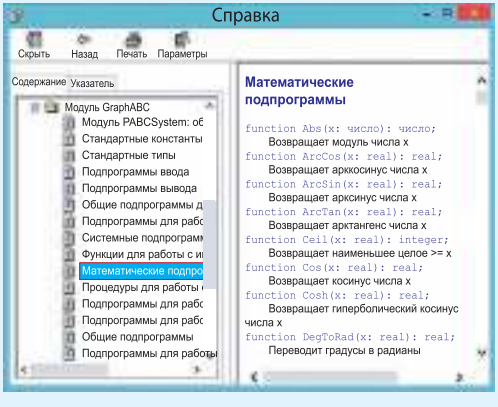
- вывад паведамленняў;
- увод значэнняў пераменных;
- каманда прысвойвання;
- вывад выніку.

Прыклад 13.4. Напісаць праграму для вываду наступнага відарыса:

Відарыс складаецца з двух аднолькавых фігур. Афармім дапаможны алгарытм `Figura` для адлюстравання адной фігуры. Праграма:

```
uses Drawman;
procedure Figura;
begin
  PenDown; OnVector(1, 0);
  OnVector(0, 3); OnVector(–1, 0);
  OnVector(0, –1); OnVector(3, 0);
  OnVector(0, 1); OnVector(–1, 0);
  OnVector(0, –2); OnVector(–2, 0);
  OnVector(0, –1); PenUp;
end;
begin
  Field(10, 5);
  ToPoint(1, 1); Figura;
  ToPoint(6, 1); Figura;
end.
```

Прыклад 13.5. Пералік матэматычных функцый можна паглядзець у даведцы PascalABC:



У гэтым выпадку зручна выкарыстоўваць дапаможны алгарытм, які можна выконваць патрэбную колькасць разоў, звяртаючыся да яго назвы.

Дапаможны алгарытм — алгарытм, які можна выкарыстоўваць у іншых алгарытмах, запісаўшы яго імя і, калі неабходна, значэнні параметраў.

Дапаможны алгарытм вырашае некаторую частку асноўнай задачы. Выклік дапаможнага алгарытму з'яўляецца камандай, якая можа замяняць некалькі каманд.

Дапаможныя алгарытмы вы выкарыстоўвалі пры напісанні праграм для вучэбных камп'ютарных выканаўцаў Чарцёжнік і Робат (прыклад 13.4). Каманды `read` і `write` таксама рэалізаваны як дапаможныя алгарытмы.

Пры вылічэннях часта выкарыстоўваюцца розныя матэматычныя функцыі (прыклад 13.5). Гэтыя функцыі рэалізаваны як убудаваныя дапаможныя алгарытмы і могуць ужывацца пры запісе арыфметычных выразаў. Аргументы функцый заўсёды запісваюцца ў дужках. Некаторыя з функцый прыведзены ў табліцы (іншыя можна паглядзець у *Дадатку 3*, с. 158).

Запіс на мове Pascal	Апісанне
`abs(x)`	Знаходзіць модуль ліку x
`sqr(x)`	Узводзіць лік x у квадрат
`sqrt(x)`	Знаходзіць корань квадратны з ліку x. Вынік — заўсёды лік тыпу `real`

Запіс на мове Pascal	Апісанне
`trunc(x)`	Знаходзіць цэлую частку рэчаіснага ліку x (`real`). Вынік — лік тыпу `integer`
`frac(x)`	Знаходзіць дробавую частку рэчаіснага ліку x (`real`). Вынік — лік тыпу `real`
`sin(x)`	Вылічае сінус ліку x. Лік x задаецца ў радыянах[1]
`cos(x)`	Вылічае косінус ліку x. Лік x задаецца ў радыянах
`RadToDeg(x)`	Пераводзіць радыяны ў градусы
`DegToRad(x)`	Пераводзіць градусы ў радыяны

Аргументам функцыі можа быць лік, пераменная, выраз або іншая функцыя: `sin(DegToRad(45))`, `sqrt(abs(-16))`.

У прыкладзе 13.6 выкарыстоўваюцца матэматычныя функцыі для ўзвядзення ліку ў квадрат і вылічэння квадратнага кораня.

Прыклад 13.6. Зададзена даўжыня стараны квадрата a. Напісаць праграму знаходжання плошчы квадрата і даўжыні яго дыяганалі.

Этапы выканання задання

I. Зыходныя даныя: даўжыня стараны, пераменная a.

II. Вынік: пераменныя S (плошча) і d (даўжыня дыяганалі).

III. Алгарытм рашэння задачы.

1. Увод зыходных даных.

2. Вылічэнне плошчы па формуле $S = a^2$.

3. Вылічэнне даўжыні дыяганалі па формуле $d = a\sqrt{2}$.

4. Вывад выніку.

IV. Апісанне пераменных: усе пераменныя, вызначаныя для рашэння задачы, маюць тып `real`.

V. Праграма:

```
var a, S, d: real;
begin
 write('Увядзіце a ='); read(a);
 S:= sqr(a); d:= a*sqrt(2);
 writeln('S =',s); writeln('d =',d);
end.
```

VI. Тэсціраванне праграмы.

Запусціць праграму і ўвесці значэнне $a = 5.6$. Вынік:

Окно вывода

```
Увядзіце a = 5.6
S = 31.36
d = 7.91959594928933
```

Правільнасць вылічэнняў можна праверыць на калькулятары.

? **1.** Што такое алгарытм?
2. Пералічыце асноўныя алгарытмічныя канструкцыі.
3. Якая каманда выкарыстоўваецца ў мове Pascal для ўводу даных?
4. Якія каманды выкарыстоўваюцца ў мове Pascal для вываду даных?
5. Для чаго патрэбна каманда прысвойвання?
6. У якіх выпадках зручна выкарыстоўваць дапаможны алгарытм?
7. Якія матэматычныя функцыі могуць выкарыстоўвацца пры запісе арыфметычных выразаў?

[1] Радыян — адзінка вымярэння вуглоў у Міжнароднай сістэме адзінак (π радыян адпавядае велічыні разгорнутага вугла 180°).

Практыкаванні

1 Расстаўце каманды праграмы ў правільным парадку так, каб можна было вылічыць значэнне выразу $a = \frac{2x}{x^2+4}$.

```
1. writeln('a = ',a);
2. write('Увядзіце значэнне x=');
3. End.
4. Var x,y,a: real;
5. Begin
6. a: = 2 * x/(x * x+4);
7. read(x);
```

2 Вызначыце тыпы даных для кожнай пераменнай, выкарыстанай у аператары прысвойвання.

```
1. y: = sqrt(a – 4)/16;
2. z: = sqr(3 * a + 2);
3. a: = abs(a – 4.2);
4. d: = x mod 2;
5. y: = int(a);
6. y: = trunc(a);
7. y: = frac(a);
8. s: = sin(3.14 * r).
```

3 Знайдзіце і выпраўце памылкі ў праграмах.

1.
```
var x, y, z1, z2:integer;
begin
 write('Увядзіце x =');
 read(x);
 write('Увядзіце y =');
 read(y);
 z1:= int(x/y);
 z2:= frac(x/y);
 write('Цэлая  частка =',z1);
write('Дробная частка=',z2);
end.
```

2.
```
var x, y, z1, z2:real;
begin
 write('Увядзіце x  =');
 read(x);
 write('Увядзіце y  =');
 read(y);
 z1:=x div y;
 z2:=x mod y;
write('Цэлая частка=',z1);
 write('Астача =',z2);
end.
```

4 Дадзены x і z. Змяніце праграму з прыкладу 13.4 так, каб вылічалася значэнне выразу $a = \frac{2x}{\sqrt{z^2+9}}$.

5 Зададзены тры лікі. Напішыце праграму для знаходжання сярэдняга арыфметычнага гэтых лікаў.

6 Зададзены два лікі. Напішыце праграму для знаходжання дзелі ад дзялення першага ліку на другі і акругліце вынік да найбліжэйшага цэлага.

7 Дадзены гіпатэнуза і катэт прамавугольнага трохвугольніка. Напішыце праграму для знаходжання другога катэта і плошчы трохвугольніка.

8* Зададзены два лікі. Напішыце праграму для знаходжання дзелі гэтых лікаў. Акругліце вынік да дзясятых, пакінуўшы ў дробавай частцы адну лічбу.

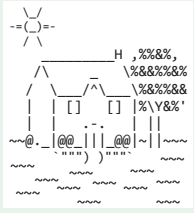
§ 14. Графічныя магчымасці асяроддзя праграміравання PascalABC

14.1. Асновы работы з графікай

Вы ўжо знаёмыя з графічнымі рэдактарамі, у якіх для пабудовы відарысаў на камп'ютары выкарыстоўваюцца графічныя прымітывы — простыя геаметрычныя фігуры: прамавугольнік, акружнасць, эліпс, адрэзак і г. д. Графічны рэдактар — праграма, напісаная на якой-небудзь мове праграміравання.

Для работы з графікай мовы праграміравання выкарыстоўваюць спецыяльныя бібліятэкі (модулі), якія змяшчаюць наборы каманд для пабудовы відарысаў. У PascalABC для работы з графікай выкарыстоўваецца бібліятэка GraphABC. Для падключэння гэтай бібліятэкі ў праграме запісваецца каманда **uses** `GraphABC`;

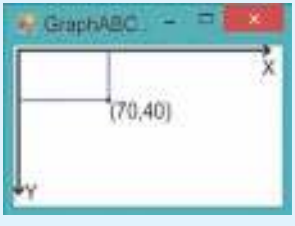
Становішча фігур задаецца каардынатамі ў графічным акне. Каардынатная плоскасць у ім адрозніваецца ад той, якую вы выкарыстоўваеце на ўроках матэматыкі. Пачаткам каардынат з'яўляецца верхні левы вугал графічнага акна — пункт (0; 0) (прыклад 14.1). Каардынаты задаюць парадкавы нумар пікселя па гарызанталі і па вертыкалі, таму яны могуць быць толькі цэлымі лікамі. Адлік значэнняў каардынаты x адбываецца злева направа, а каардынаты y — зверху ўніз. Па змоўчанні ствараецца графічнае акно памерам 640 × 480 пікселяў.



Першыя камп'ютары не мелі магчымасцей работы з графікай. На друкавальных устройствах выводзіліся «карцінкі», што складаліся з сімвалаў.

У 1958 г. быў запушчаны камп'ютар Lincoln TX-2, у якім упершыню выкарыстоўваўся графічны экран. У 1981 г. пачалі выкарыстоўваць колеры. У графічным рэжыме пры разрашэнні 320 × 200 пікселяў выкарыстоўваліся 4 колеры са стандартных палітр: пурпурны, сіне-зялёны, белы, чорны або чырвоны, зялёны, жоўты, чорны.

Прыклад 14.1. Графічнае акно асяроддзя праграміравання PascalABC з адлюстраваннем каардынатных восей і пункта з каардынатамі (70; 40).

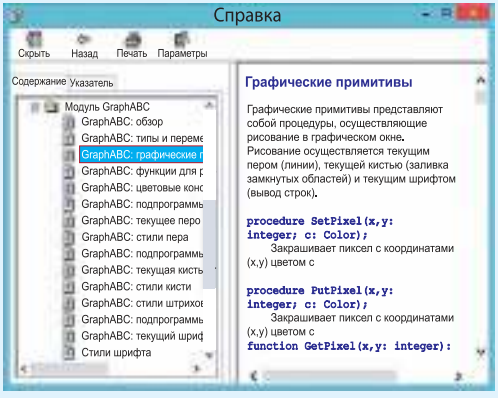


Пункт размешчаны на адлегласці 70 пікселяў ад левага краю акна і на адлегласці 40 пікселяў ад верхняга краю.

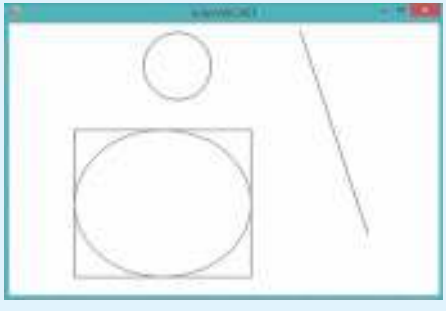
Памеры графічнага акна можна задаць камандай `SetWindowSize(n,m)`; У дужках пададзены памеры акна па гарызанталі і па вертыкалі.

[1] https://www.asciiart.eu/buildings-and-places/houses (дата доступу: 26.07.2018).

Прыклад 14.2. Работа з даведачнай сістэмай. Для пераходу ў даведнік неабходна націснуць клавішу F1 або выканаць каманду меню: **Помощь** → → **Справка.** У акне, якое адкрыецца, перайсці ў раздзел **Стандартные модули** і выбраць **Модуль GraphABC.**



Прыклад 14.3. Графічныя прымітывы:



Праграма для іх рысавання:

```
uses GraphABC;
begin
  //Круг
  Circle(250, 125, 30);
  //Прамавугольнік
  Rectangle(100,200,400,450);
  //Эліпс
  Ellipse(100,200,400,450);
  //Адрэзак
  Line(450, 50, 550, 350);
end.
```

14.2. Работа з даведачнай сістэмай асяроддзя праграміравання PascalABC

У бібліятэцы GraphABC змяшчаецца вялікая колькасць каманд. Гэтыя каманды апісаны ў даведачнай сістэме асяроддзя PascalABC (прыклад 14.2). Тут ёсць апісанне графічных прымітываў, назвы колеравых канстант, апісанне работы з пяром і пэндзлем, каманды работы з графічным акном.

Каманды бібліятэкі GraphABC — дапаможныя алгарытмы, запісаныя як асобныя працэдуры. Выкарыстанне каманды ў праграме азначае выклік адпаведнага алгарытму.

14.3. Асноўныя графічныя прымітывы

Разгледзім каманды для рысавання графічных прымітываў:

- `Line(x1,y1,x2,y2)` — адрэзак, які злучае пункты з каардынатамі ($x1$, $y1$) і ($x2$, $y2$);
- `MoveTo(x,y)` — устанаўлівае бягучую пазіцыю рысавання ў пункт (x, y);
- `LineTo(x,y)` — адрэзак ад бягучай пазіцыі да пункта (x, y);
- `Rectangle(x1,y1,x2,y2)` — прамавугольнік, зададзены каардынатамі супрацьлеглых вяршынь ($x1$, $y1$) і ($x2$, $y2$);
- `Circle(x1,y1,r)` — круг з цэнтрам у пункце ($x1$, $y1$) і радыусам r;
- `Ellipse(x1,y1,x2,y2)` — авал (эліпс), упісаны ў прамавугольнік з каардынатамі супрацьлегых вяршынь ($x1$, $y1$) і ($x2$, $y2$).

(Разгледзьце прыклад 14.3.)

Каманды для рысавання іншых графічных прымітываў маюцца ў даведачнай сістэме і ў *Дадатку 3* (с. 159).

Прыклад 14.4. Напісаць праграму, якая будуе відарыс доміка, вы-

карыстоўваючы працэдуры Line, Lineto, Rectangle, Circle.

Этапы выканання задання

I. Зыходныя даныя: вынік работы праграмы не залежыць ад зыходных даных.

II. Вынік: гатовы рысунак.

III. Алгарытм рашэння задачы.

Рысунак складаецца з: прамавугольнікаў, адрэзкаў, круга. Для разліку каардынат рэкамендуецца спачатку зрабіць рысунак на лісце паперы ў клетачку.

IV. Апісанне пераменных. Пераменныя не выкарыстоўваюцца.

14.4. Работа з пяром і пэндзлем

У графічных рэдактарах, перш чым рысаваць якія-небудзь фігуры, устанаўліваюць іх колер. Звычайна выбіраюць два колеры. Колер 1 вызначае колер ліній і контураў фігур, Колер 2 выкарыстоўваецца для заліўкі фігур. Акрамя таго, можна змяняць стыль ліній і заліўкі, а таксама вызначаць таўшчыню ліній.

У графічным рэжыме PascalABC настройкі лініі вызначае пяро (Pen), а настройкі ўнутранай вобласці фігур — пэндзаль (Brush). Каманды для работы з пэндзлем і пяром прыведзены ў табліцы.

Каманда	Апісанне
SetPenColor	Колер ліній
SetPenWidth	Таўшчыня ліній
SetPenStyle	Стыль ліній
SetBrushColor	Колер заліўкі
SetBrushStyle	Стыль заліўкі
SetBrushHatch	Від штрыхоўкі для заліўкі

Прыклад 14.4.

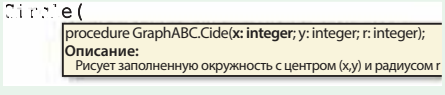
V. Праграма:

```
uses GraphABC;
begin
  //Дом
  Rectangle(100,200,400,450);
  //Акно
  Rectangle(200,250,300,350);
  Line(250, 250, 250, 350);
  Line(200, 300, 300, 300);
  //Дах
  MoveTo(100,200);
  LineTo(250, 0);
  LineTo(400, 200);
  Circle(250, 125, 30);
end.
```

VI. Тэсціраванне праграмы:

Запусіць праграму. Вынік:

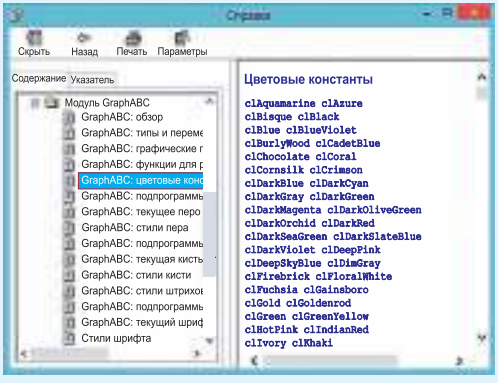
Звярніце ўвагу, што пры напісанні каманд у вас з'яўляюцца падказкі:



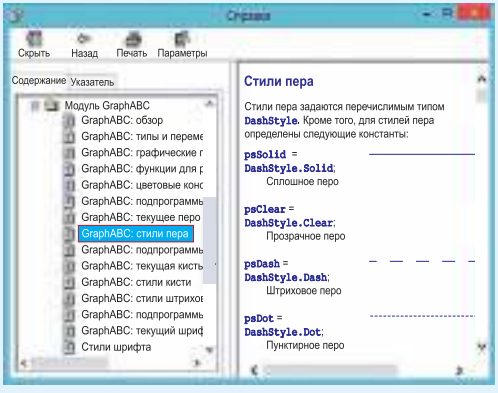
Падказка з'яўляецца таксама пры навядзенні мышы на ўжо напісаную каманду.

Калі ўстанавіць тэкставы курсор на каманду і націснуць F1, то адкрыецца старонка з даведніка з апісаннем гэтай каманды.

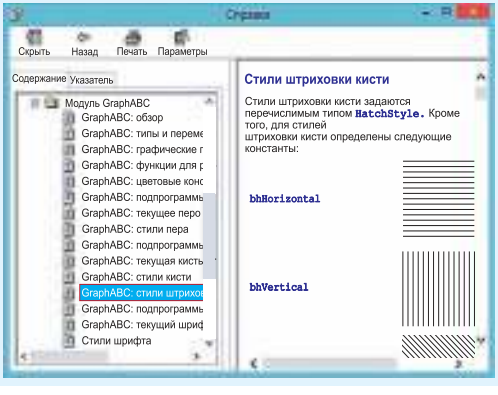
Прыклад 14.5. Колеравыя канстанты.



Прыклад 14.6. Стылі пяра.



Прыклад 14.7. Стылі штрыхоўкі пэндзля.



Значэнне, якое трэба ўстанавіць для кожнай з каманд, запісваецца ў дужках. Напрыклад:

- `SetPenColor(clRed)` — чырвоны колер рысавання ліній;
- `SetBrushColor(clBlue)` — сіні колер заліўкі фігур;
- `SetPenWidth(3)` — таўшчыня пяра ў 3 пікселі;
- `SetPenStyle(psDot)` — штрыхавая лінія;
- `SetBrushStyle(bsHatch)` — штрыхавая заліўка;
- `SetBrushHatch(bhCross)` — штрыхоўка ў клетачку.

Значэнні колеравых канстант, стыляў ліній і залівак можна знайсці ў даведачнай сістэме асяроддзя PascalABC (прыклады 14.5—14.7) і ў *Дадатку 3* (с. 160).

Каманды для ўстаноўкі колеру і стылю запісваюць перад камандай рысавання фігуры. Гэтыя каманды дзейнічаюць да таго часу, пакуль колер або стыль не будзе зменены. Калі, напрыклад, для пяра была ўстаноўлена таўшчыня ў 3 пікселі, то адрэзкі і межы фігур будуць мець таўшчыню ў тры пікселі да новай змены таўшчыні інструмента.

Для фігур па змоўчанні ўстаноўлена заліўка белым колерам. Калі колер пэндзля выбраць да рысавання фігуры, то фігура будзе зафарбавана ўстаноўленым колерам. Колер ужо нарысаванай фігуры можна змяніць з дапамогай каманды заліўкі:

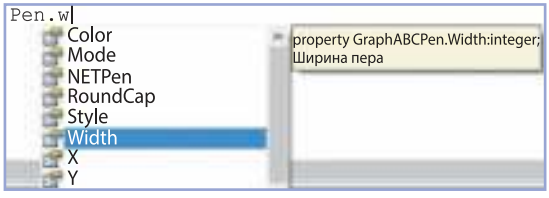
`FloodFill(x,y,c);` — залівае абмежаваную вобласць аднаго колеру колерам `c`, пачынаючы з пункта ўнутры вобла-

сці (x,y) (у прыкладзе 14.8 паказана, як расфарбаваны домік з прыкладу 14.4).

Акрамя таго, асяроддзе праграмірання PascalABC дазваляе звяртацца да ўласцівасцей пэндзля і пяра па-іншаму. Так, напрыклад, для змянення колеру (стылю, таўшчыні) можна запісаць

```
Pen.Color := clRed;
Pen.Style := psDot;
Pen.Width := 3;
```

Падказка асяроддзя выглядае наступным чынам:



Другую частку каманды можна выбраць з выпадаючага спіса.

Пры вывучэнні вектарнай графікі вы пазнаёміліся з колеравай мадэллю RGB, якая дазваляе запісаць любы колер трыма складнікамі: чырвоным, зялёным і сінім. Функцыя `RGB (r, g, b)` дазваляе вызначыць колер па трох складніках. Так, каманда `SetPenColor (clRed);` аналагічная камандзе `SetPenColor (RGB(255,0,0));` Такі спосаб задання колеру дазваляе задаваць колеры, значэнні якіх не апісаны колеравымі канстантамі. Значэнні складнікаў колеру можна паглядзець у графічным рэдактары Paint (прыклад 14.9).

У графічным рэжыме PascalABC можна выводзіць у графічнае акно тэксты і лікі.

`TextOut(x,y,z);` — выводзіць радок або лік `z` у прамавугольнік з каардынатамі левага верхняга вугла `(x,y)`.

Прыклад 14.8. Праграма:

```
uses GraphABC;
begin
  //Дом
  SetPenColor(RGB(255,0,0));
  SetBrushColor(clBlue);
  Rectangle(100,200,400,450);
  //Акно
  SetBrushColor(clYellow);
  Rectangle(200,250,300,350);
  SetPenColor(clRed);
  SetPenStyle(psDot);
  SetPenWidth(2);
  Line(250,250,250,350);
  Line(200,300,300,300);
  //Дах
  SetPenStyle(psSolid);
  SetPenWidth(1);
  Line(100, 200, 250, 0);
  Line(250, 0, 400, 200);
  SetBrushStyle(bsHatch);
  SetBrushColor(clLightGreen);
  SetBrushHatch(bhCross);
  Circle(250, 125, 30);
  FloodFill(250,70, clPlum);
end.
```

Вынік работы праграмы:

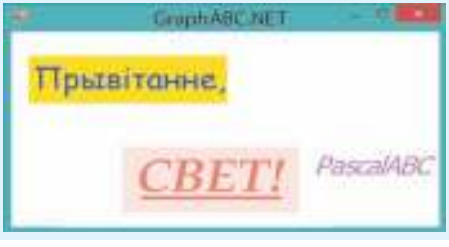
Прыклад 14.9. Складнікі колеру:

Прыклад 14.10. Выведзем у графічнае акно прывітанне свету, выкарыстоўваючы розныя ўласцівасці тэксту.

Праграма:

```
uses GraphABC;
begin
 //Колер фону для тэксту
 SetBrushColor(clYellow);
 SetFontName('Comic Sans MS');
 //Колер літар
 SetFontColor(clBlue);
 //Памер шрыфту
 SetFontSize(25);
 //Паўтлусты шрыфт
 SetFontStyle(fsBold);
 TextOut(20,30,'Прывітанне,');
 SetBrushColor(clPink);
 StFontName('PaladiumC
              Bold Italic');
 SetFontColor(clSalmon);
 SetFontSize(50);
 SetFontStyle(fsUnderline);
 TextOut(120,130,'СВЕТ!');
 //Празрысты фон
 SetBrushStyle(bsClear);
 SetFontName('Tahoma');
 SetFontColor(clViolet);
 SetFontSize(20);
 SetFontStyle(fsItalic);
 TextOut(300,130,'PascalABC');
end.
```

Вынік выканання праграмы:



Калі трэба вывесці радок, то яго сімвалы змяшчаюць у двукоссе, для вываду ліку можна выкарыстоўваць пераменныя або значэнні лікаў. Калі ў якасці *z* запісаць арыфметычны выраз, то будзе выведзена яго значэнне.

Для змянення параметраў тэксту ўжываюць наступныя каманды:

Каманда	Апісанне
SetFontColor	Колер сімвалаў
SetFontSize	Памер сімвалаў
SetFontName	Імя бягучага шрыфту
SetFontStyle	Стыль тэксту

Каманда SetBrushColor устанаўлівае колер прамавугольніка, унутры якога будзе знаходзіцца тэкст (прыклад 14.10). Каманда дзейнічае да таго часу, пакуль колер не будзе зменены. Для запісу тэксту без фону трэба ўстанавіць празрысты фон пэндзля: SetBrushStyle(bsClear).

Усе параметры для тэксту задаюцца да каманды вываду яго ў графічнае акно. Імя шрыфта, якім будзе выведзены тэкст, змяшчаюць у двукоссе. Магчымыя варыянты можна паглядзець у Word. Стыль тэксту можа мець наступныя значэнні: fsNormal (звычайны), fsBold (тлусты), fsItalic (нахільны), fsUnderline (падкрэслены) або іх камбінацыі: fsBoldUnderline (тлусты падкрэслены).

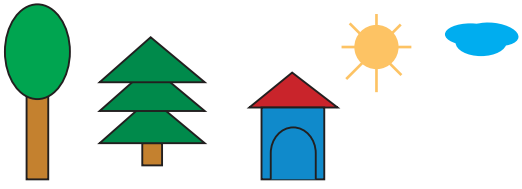
1. Якая бібліятэка выкарыстоўваецца для падключэння графікі ў PascalABC?
2. Як вызначана сістэма каардынат у графічным акне?
3. Як задаць памеры графічнага акна?
4. Як знайсці апісанне графічных прымітываў у даведнік?
5. Як можна змяніць колер ліній, заліўкі?
6. Як вывесці тэкст у графічным акне?

Практыкаванні

1 Падпішыце відарыс з прыкладу 14.8.

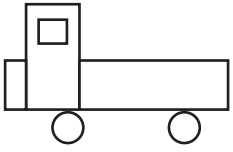
2 Дапоўніце відарыс доміка з прыкладу 14.8 відарысамі трубы і дыму з трубы ў выглядзе некалькіх авалаў:

3 Дапоўніце вынік, атрыманы пры выкананні задання 2, якімі-небудзь з прапанаваных відарысаў або прыдумайце свае.

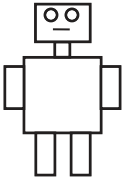


4 Напішыце праграму для стварэння відарыса. Размалюйце дадзены відарыс па сваім вырашэнні. Дадатковыя каманды для пабудовы графічных прымітываў можна знайсці ў даведачнай сістэме.

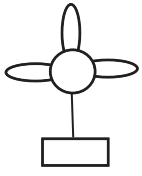
Грузавік
1



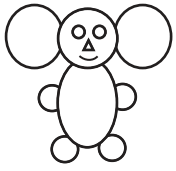
Робат
2



Кветка
3



Чабурашка
4



§ 15. Простыя і састаўныя ўмовы

15.1. Лагічны тып даных

Нагадаем вывучаныя ў 7-м класе паняцці *выказванне* і *ўмова для выканаўцы*.

Выказванне — апавядальны сказ (сцверджанне), пра які можна сказаць, праўдзівы ён ці непраўдзівы.

Умовай для выканаўцы з'яўляецца вядомае яму выказванне, якое можа выконвацца (быць праўдзівым) або не выконвацца (быць непраўдзівым).

Тып `Boolean` названы ў гонар англійскага матэматыка і логіка Джорджа Буля, які займаўся пытаннямі матэматычнай логікі ў XIX ст.

Дадзены тып прысутнічае ў пераважнай большасці моў праграміравання. У некаторых мовах рэалізуецца праз лікавы тып даных. Тады за значэнне «няпраўда» прымаецца 0, а за значэнне «праўда» — 1.

Прыклад 15.1. Прыклады лагічных выразаў:

- `3 < 7` — лагічны выраз, значэнне якога `true`;
- `2 + 2 * 2 = 8` — лагічны выраз, значэнне якога `false`;
- `abs(-5) > abs(3)` — лагічны выраз, значэнне якога `true`;
- `y >= sqr(x)` — лагічны выраз, значэнне якога можна вызначыць, толькі ведаючы значэнні пераменных x і y. Пры $x = 2$ і $y = 10$ значэнне выразу — `true`. Пры $x = 10$ і $y = 2$ — `false`.

Праверым праўдзівасць гэтых выразаў у праграме:

```
var a1, a2, a3, a4, a5: boolean;
  x, y: integer;
begin
  a1 := 3 < 7;
  writeln('a1 =', a1);
  a2 := 2 + 2 * 2 = 8;
  writeln('a2=', a2);
  a3 := abs(-5) > abs(3);
  writeln('a3 =', a3);
  x := 2; y := 10;
  a4:= y > = sqr(x);
  writeln('a4 = ', a4);
  x := 10; y := 2;
  a5 := y >= sqr(x);
  writeln('a5  =', a5);
end.
```

Вынік работы праграмы:

Окно вывода

```
a1 = True
a2 = False
a3 = True
a4 = True
a5 = False
```

Па змоўчанні `false < true`.

У мове праграміравання Pascal для работы з умовамі вызначаны **лагічны тып даных boolean.** Велічыні тыпу `boolean` могуць прымаць два значэнні — `false` (непраўдзіва) і `true` (праўдзіва).

Значэнні `false` і `true` атрымліваюцца ў выніку выканання аперацый параўнання над лікавымі данымі. Для параўнання выкарыстоўваюць знакі, паказаныя ў табліцы.

Аперацыя	PascalABC
Роўна (=)	=
Не роўна (≠)	<>
Больш (>)	>
Менш (<)	<
Больш або роўна (⩾)	>=
Менш або роўна (⩽)	<=

Параўноўваць можна канстанты, пераменныя, арыфметычныя і лагічныя выразы.

Лагічны выраз — выраз, які прымае адно з двух значэнняў: `true` або `false`.

Лагічныя выразы можна прысвойваць пераменным тыпу `boolean`, а таксама выводзіць іх значэнні на экран: будзе выведзена слова `false` або `true` адпаведна (прыклад 15.1). Умовы для выканаўцы з'яўляюцца прыватным выпадкам лагічных выразаў.

Прыклад 15.2. Напісаць праграму, якая выведзе на экран значэнне `true` або `false` у залежнасці ад таго, з'яўляецца ўведзены лік x цотным ці не.

Этапы выканання задання

I. Зыходныя даныя: x (уведзены лік).

II. Вынік: a (`true` або `false`).

III. Алгарытм рашэння задачы.

1. Увод зыходных даных.

2. Вылічэнне значэння лагічнай пераменнай. Лік з'яўляецца цотным, калі астача ад дзялення яго на 2 роўна нулю. Значэнне пераменнай a вызначаецца значэннем выразу x **mod** 2 = 0.

3. Вывад выніку.

IV. Апісанне пераменных: x — `integer`, a — `boolean`.

Прыклад 15.2.

V. Праграма:

```
var x: integer;
  a: boolean;
begin
  write('Увядзіце x =');
  read(x);
  a := x mod 2 = 0;
  write('Лік цотны — ',a);
end.
```

VI. Тэсціраванне

Запусціць праграму і ўвесці значэнне x = 6. Вынік:

```
Окно вывода
Увядзіце x = 6
Лік цотны - True
```

Запусціць праграму і ўвесці значэнні x = 11. Вынік:

```
Окно вывода
Увядзіце x = 11
Лік цотны - False
```

15.2. Састаўныя ўмовы

З выказваннямі можна выконваць лагічныя аперацыі **(НЕ, І, АБО)**. Для лагічных пераменных таксама вызначаны лагічныя аперацыі, якія адпавядаюць аперацыям над выказваннямі: **not**, **and**, **or**.

Лагічныя выразы, у якіх поруч з простымі ўмовамі (параўнаннямі) выкарыстоўваюцца лагічныя аперацыі, называюць **састаўнымі ўмовамі.**

Прывядзём табліцы праўдзівасці лагічных аперацый.

Лагічная пераменная		Вынік аперацыі		
A	*B*	**not** *A*	*A* **and** *B*	*A* **or** *B*
True	True	False	True	True
False	True	True	False	True
True	False	False	False	True
False	False	True	False	False

У мове PascalABC рэалізавана лагічная аперацыя **xor** — выключальнае **АБО**. Гэтай аперацыі адпавядае выказванне: «Толькі адно з двух выказванняў можа быць праўдзівым». Табліца праўдзівасці для аперацыі **xor**:

A	*B*	*A* **xor** *B*
True	True	False
False	True	True
True	False	True
False	False	False

Усе лагічныя аперацыі могуць выкарыстоўвацца ў дачыненні да лікаў тыпу integer. Лік разглядаецца ў двайковым уяўленні, і аперацыі выкарыстоўваюцца ў дачыненні да бітаў ліку. Біт, роўны 1, уяўляецца як праўда, а біт, роўны нулю, — як няпраўда.

Прыклад 15.3. Вызначэнне парядку дзеянняў для выразу (*a*, *d* — boolean, *c*, *b* — integer):

```
a or (c < b) and d
```

Першым выконваецца параўнанне *c* і *b*, затым лагічная аперацыя **and**, потым — **or**.

Прыклад 15.4*. Разгледзім выраз:

```
a < b and c < d
```

Калі *a*, *b*, *c*, *d* мае тып integer, то атрымаем памылку: «Операция '<' не применима к типам boolean и integer» (з дапамогай знака '<' нельга параўноўваць лік і лагічную пераменную). Калі пераменныя маюць тып real, то ўзнікне памылка: «Операция 'and' не применима к типу real». Правільны запіс выразу:

```
(a < b) and (c < d)
```

Усе вышэйпералічаныя памылкі ўзнікаюць таму, што аперацыя and валодае большым прыярытэтам у адносінах да аперацый <. Таму спачатку будзе ажыццяўляцца спроба выканаць аперацыю b **and** c, а затым параўнання.

Прыклад 15.5. Пабудова адмаўленняў:

```
not not a = a;
not (a and b) = (not a) or (not b);
not (a or b) = (not a) and (not b).
```

Разгледзім выраз **not** a < b з пераменнымі *a* і *b* тыпу integer. Тут аперацыя not належыць да пераменнай *a*, таму ў двайковым уяўленні ліку *a* біты, роўныя 1, будуць заменены на 0, а біты, роўныя 0, — на 1. Затым атрыманы вынік параўнаецца з лікам *b*. Для адмаўлення параўнання выраз трэба запісаць так: **not** (a < b).

У лагічных выразах могуць сустракацца як арыфметычныя аперацыі, так і лагічныя. Парадак выканання аперацый вызначаецца іх прыярытэтам:

1) **not**;

2) *, /, **div**, **mod**, **and**;

3) +, –, **or**;

4) =, <>, <, >, <=, >=.

(Разгледзьце прыклад 15.3.)

Аперацыі з аднолькавым прыярытэтам выконваюцца па парадку, злева направа. Для змянення парадку выканання аперацый ужываюць дужкі (прыклад 15.4).

Пры складанні праграм часта трэба будаваць адмаўленні складаным лагічным выразам. Для гэтага карысна выкарыстоўваць тоеснасці, вядомыя з алгебры логікі (прыклад 15.5), і наступную табліцу:

Умова	Супрацьлеглая ўмова (адмаўленне ўмовы)
$a < b$	$a >= b$
$a > b$	$a <= b$
$a = b$	$a <> b$

Прыклад 15.6. Напісаць праграму, якая выдасць на экран значэнне true або false ў залежнасці ад таго, ці знаходзіцца лік *B* паміж лікамі *A* і *C*.

Этапы выканання задання

I. Зыходныя даныя: пераменныя *A*, *B*, *C* (лікі, якія ўводзяцца).

II. Вынік: rez (True або False).

III. Алгарытм рашэння задачы.

1. Увод зыходных даных.

2. Вылічэнне значэння лагічнай пераменнай. Разгледзім два выпадкі.

Правільная няроўнасць: $A < B < C$. Гэтай няроўнасці адпавядае лагічны выраз: (A < B) **and** (B < C). Нададзім пераменнай r1 значэнне гэтага выразу.

Правільная няроўнасць: $A > B > C$. Гэтай няроўнасці адпавядае лагічны выраз: (A > B) **and** (B > C). Нададзім пераменнай r2 значэнне гэтага выразу.

Адказам на задачу будзе значэнне лагічнага выразу r1 **or** r2.

3. Вывад выніку.

IV. Апісанне пераменных: A, B, C — integer, r1, r2, rez — boolean.

Для работы з лагічнымі велічынямі могуць выкарыстоўвацца функцыі. Функцыя Ord (парадкавы нумар значэння) дазваляе пераўтвараць лагічнае значэнне ў лікавае: Ord(false) = **0**, а Ord(true) = **1**. Функцыі Pred (папярэдняе значэнне) і Succ (наступнае значэнне) дазваляюць пераўтвараць лагічныя значэнні:

```
D := Pred(true);  {D = false}
E := Succ(false);  {E = true}
```

Прыклад 15.6.

V. Праграма:

```
var A, B, C: integer;
 r1, r2, rez: boolean;
begin
 writeln('Увядзіце A, B, C');
 read(A, B, C);
 r1 := (A < B) and (B < C);
 r2 := (A > B) and (B > C);
 rez := r1 or r2;
 write('Лік B паміж лікамі
       A і C — ',rez);
end.
```

VI. Тэсціраванне.

Запусціць праграму і ўвесці значэнні A = 5, B = 0, C = −5. Вынік:

Окно вывода

```
Увядзіце A, B, C
5 0 -5
Лік B паміж лікамі A і C - True
```

Запусціць праграму і ўвесці значэнні A = −2, B = −7, C = 5. Вынік:

Окно вывода

```
Увядзіце A, B, C
-2 -7 5
Лік B паміж лікамі A і C - False
```

VII. Аналіз вынікаў. Для поўнага тэсціравання праграмы трэба праверыць ўсе магчымыя выпадкі ўзаемнага размяшчэння A, B, C (іх усяго 6).

1. Што такое састаўная ўмова?
2. Назавіце лагічныя аперацыі, якія выкарыстоўваюцца ў PascalABC.
3. Які прыярытэт у лагічнай аперацыі **not** (**and**, **or**)?

Практыкаванні

1 Сфармулюйце і рэалізуйце адваротную задачу для прыкладу 15.2: для ўсіх тых выпадкаў, для якіх у зыходнай задачы было true, трэба вывесці false і, наадварот, для ўсіх тых выпадкаў, у якіх у зыходнай задачы атрымлівалася false, атрымаць true.

2 У PascalABC вызначана лагічная функцыя odd(x). Значэнне гэтай функцыі true, калі лік x з'яўляецца няцотным, і false, калі x — цотны. Змяніце праграму прыкладу 15.2, выкарыстоўваючы функцыю odd.

3 Вызначыце, што робяць наступныя праграмы, і дапоўніце каманду вываду.

1.
```
var x: integer;
    a: boolean;
begin
  write('Увядзіце x =');
  read(x);
  a := x mod 10 = 0;
  write('Лік ... — ',a);
end.
```

2.
```
var x: integer;
    a: boolean;
begin
  write('Увядзіце x  =');
  read(x);
  a := (x>10) and (x < 100);
  write('Лік ... — ',a);
end.
```

4 Напішыце праграму, якая выведзе на экран значэнне `true` або `false`, у залежнасці ад таго, з'яўляецца ўведзены лік *x* дадатным ці не.

5 Напішыце праграму, якая выведзе на экран значэнне `true` або `false`, у залежнасці ад таго, з'яўляецца ўведзены лік *x* чатырохзначным ці не.

6* Зададзены два дадатныя лікі *x* і *y*. Вызначыце, ці дакладна, што першы лік меншы за другі і хоць бы адзін з іх няцотны. Выведзіце на экран `true` або `false`.

§ 16. Аператар галінавання

Выкарыстанне кіруючых канструкцый прадугледжвае запіс праграмы ў структураваным выглядзе. Структураванасць праграм дасягаецца за кошт водступаў, якія рэгулююць змяшчэнне ўкладзеных алгарытмічных канструкцый.

Можна выконваць наступнае правіла: пры руху курсора ўніз ад «пачатку» структуры да яе «канца» на шляху курсора могуць сустрэцца толькі прабелы. Усё, што знаходзіцца «ўнутры» структуры, змяшчаецца правей.

Кнопка дазваляе пераўтварыць код праграмы да структураванага выгляду.

Прыклад 16.1.
V. Праграма:
```
var x: integer;
begin
 write('Увядзіце x = '); read(x);
 if x > 0 then
   write('дадатнае')
 else
   write('не дадатнае');
end.
```

16.1. Запіс аператара галінавання

Алгарытмічная канструкцыя *галінаванне* (гл. блок-схему ў прыкладзе 13.2, с. 60) забяспечвае выкананне той ці іншай паслядоўнасці каманд у залежнасці ад праўдзівасці або непраўдзівасці некаторай умовы.

Аператар галінавання — каманда, якая рэалізуе алгарытмічную канструкцыю *галінаванне* на мове праграміравання.

Для запісу аператара галінавання выкарыстоўваюць каманды **if**. Фармат каманды:
```
if <умова> then
begin
  Каманды 1;
end
else
begin
  Каманды 2;
end;
```

Аператар галінавання можа быць у поўнай або ў скарочанай формах. У скарочанай форме адсутнічае блок else:

```
if <умова> then
begin
  Каманды;
end;
```

Умова ў запісе аператара галінавання бывае простай і састаўной. Аператарныя дужкі могуць быць апушчаны, калі ўнутры іх знаходзіцца адна каманда.

Прыклад 16.1. Зададзены лік x. Трэба вызначыць, з'яўляецца ён дадатным ці не, і вывесці адпаведнае паведамленне.

Этапы выканання задання

I. Зыходныя даныя: x (уведзены лік).

II. Вынік: адпаведнае паведамленне.

III. Алгарытм рашэння задачы.

1. Увод зыходных даных.

2. Праверка значэння выразу ($x > 0$).

3. Вывад выніку.

IV. Апісанне пераменных: x — integer.

16.2. Рашэнне задач з выкарыстаннем аператара галінавання

Прыклад 16.2. У момант часу 00:00 на святлафоры для пешаходаў уключылі зялёны сігнал. Далей сігнал святлафора змяняецца кожную мінуту: 1 мінуту гарыць зялёны сігнал, 1 мінуту — чырвоны. Вядома, што з моманту ўключэння святлафора прайшло m мінут. Патрабуецца нарысаваць святлафор з уключаным сігналам у

Прыклад 16.1. *Працяг.*

VI. Тэсціраванне.

Запусціць праграму і ўвесці значэнне $x = 5$. Вынік:

Окно вывода

```
Увядзіце x = 5
дадатны
```

Запусціць праграму і ўвесці значэнне $x = -1$. Вынік:

Окно вывода

```
Увядзіце час  x = -1
не дадатны
```

VII. Аналіз вынікаў. Для поўнай праверкі праграмы патрабуецца яшчэ праверыць значэнне $x = 0$.

Окно вывода

```
Увядзіце x = 0
не дадатны
```

Прыклад 16.2.

V. Праграма:

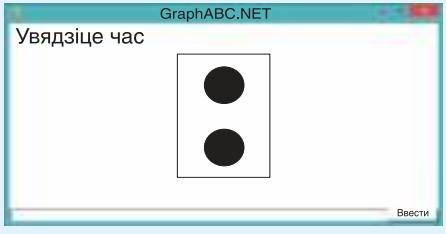
```
uses GraphABC;
var m:integer;
begin
  Rectangle(250,50,390,250);
  SetBrushColor(clBlack);
  Circle(320,100,30);
  Circle(320,200,30);
  SetBrushColor(clWhite);
  writeln('Увядзіце час');
  read(m);
  writeln(m)[1];
  if m mod 2 = 1 then
    FloodFill(320,100,clRed)
  else
    FloodFill(320,200,clGreen);
end.
```

[1] Пры ўводзе даных у графічным акне яны не адлюстроўваюцца на экране. Для таго каб бачыць, што ўвялі, неабходна дадаткова вывесці ўведзенае значэнне.

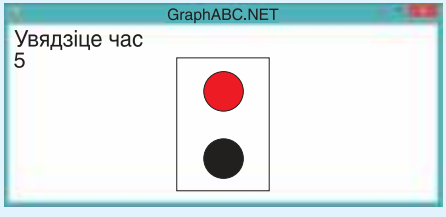
Прыклад 16.2. *Працяг.*

VI. Тэсціраванне.

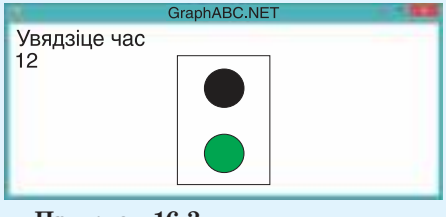
Выгляд графічнага акна да ўводу ліку:



Увесці значэнне $x = 5$. Вынік:



Увесці значэнне $x = 12$. Вынік:



Прыклад 16.3.

V. Праграма:

```
Var x1, y1, x2, y2, r_T, r_K: real;
begin
  writeln('Танін дом');
  read(x1,y1);
  writeln('Кацін дом');
  read(x2,y2);
  r_T := sqrt(x1*x1+y1*y1);
  r_K := sqrt(x2*x2+y2*y2);
  if r_T < r_K then
    writeln('Танін дом бліжэй')
  else
    writeln('Кацін дом бліжэй');
end.
```

VI. Запусціць праграму і ўвесці значэнні: Танін дом — $x1 = 2.3$, $y1 = 4.5$, Кацін дом — $x2 = -2.1$, $y2 = 4.9$ адпаведнасці з уведзеным значэннем часу.

Этапы выканання задання

I. Выходныя даныя: m (зададзены час).

II. Вынік: рысунак святлафора, які залежыць ад значэння m.

III. Алгарытм рашэння задачы.

1. Рысаванне святлафора (прамавугольнік і 2 кругі) з выключанымі сігналамі.

2. Увод зыходных даных.

3. Колер сігналу будзе залежаць ад таго, цотным ці няцотным будзе значэнне m. Калі m цотнае — сігнал зялёны (зафарбоўваем ніжні круг), калі няцотнае — чырвоны (зафарбоўваем верхні круг).

4. Зафарбуем патрэбны круг колерам у залежнасці ад цотнасці m.

IV. Апісанне пераменных: m — `integer`.

Прыклад 16.3. Таня і Каця жывуць у розных дамах. Ім стала цікава, хто з іх жыве бліжэй да школы. Яны змясцілі на карце прамавугольную сістэму каардынат так, каб школа мела каардынаты (0; 0). Вядома, што Танін дом мае каардынаты (x1; y1), а Кацін (x2; y2). Дзяўчынкі ходзяць у школу па прамой і праходзяць розныя адлегласці. Трэба напісаць праграму, якая вызначыць, чый дом бліжэй да школы.

Этапы выканання задання

I. Зыходныя даныя: каардынаты дамоў дзяўчынак $x1$, $y1$, $x2$, $y2$.

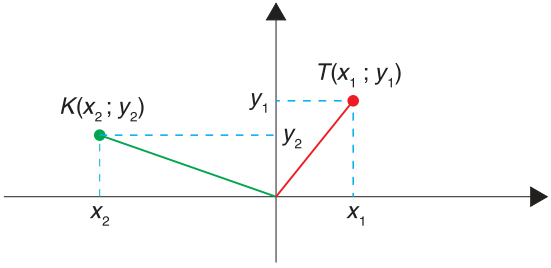
II. Вынік: паведамленне пра тое, чый дом бліжэй.

III. Алгарытм рашэння задачы.

1. Увод каардынат дамоў.

2. Вылічэнне адлегласцей да школы: r_T (ад Танінага дома) і r_K ад Кацінага дома). Для вылічэння выкарыстаем тэарэму Піфагора:

$$r\_T = \sqrt{x_1^2 + y_1^2} \text{ і } r\_K = \sqrt{x_2^2 + y_2^2}.$$



3. Параўнанне адлегласцей. Вывад адказу.

IV. Апісанне пераменных: $x1$, $y1$, $x2$, $y2$, r_T, r_K маюць тып `real`.

Прыклад 16.4. Вася пачаў займацца стральбой з лука. Для трэніроўкі ён вырашыў стварыць мадэль мішэні, якая будзе рэагаваць на лазер. Мішэнь уяўляе сабой два кругі (страляе Вася пакуль не вельмі добра) рознага радыуса з агульным цэнтрам. Калі Вася пападае у маленькі круг, то круг загараецца зялёным. Вялікі круг пры пападанні ў яго загараецца жоўтым. Калі Вася не папаў ні ў адзін з кругоў, то вобласць па-за кругамі загараецца чырвоным. Неабходна рэалізаваць камп'ютарную мадэль Васевай мішэні (пры пападанні на мяжу круга нічога не павінна адбывацца).

Этапы выканання задання

I. Выходныя даныя: каардынаты пункта стрэлу (x; y).

II. Вынік: рысунак мішэні.

III. Алгарытм рашэння задачы.

Прыклад 16.3. *Працяг.*
Вынік павінен быць такім:

Окно вывода
```
Танін дом
2.3 4.5
Кацін дом
-2.1 4.9
Танін дом бліжэй
```

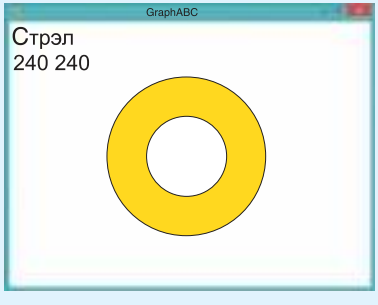
Прыклад 16.4.

V. Праграма:

```
uses GraphABC;
var x,y, x0, y0, R_b, R_m, z:
integer;
begin
  x0 := 320; y0 := 240;
  R_b := 150; R_m := 75;
  Circle(x0,y0,R_b);
  Circle(x0,y0,R_m);
  writeln('Стрэл');
  read(x,y);
  writeln(x,' ',y);
  z := sqr(x-x0)+sqr(y-y0);
  if z < sqr(R_m) then
    FloodFill(x,y,clLightGreen)
  else
    if z < sqr(R_b) then
      FloodFill(x,y,clYellow)
    else
      FloodFill(x,y,clRed);
end.
```

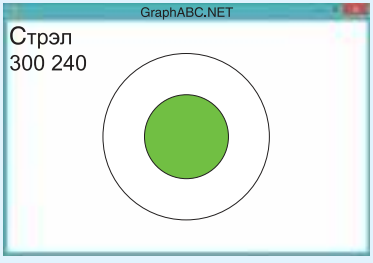
VI. Тэсціраванне.

Запусціць праграму і ўвесці каардынаты стрэлу (240; 240). Вынік:

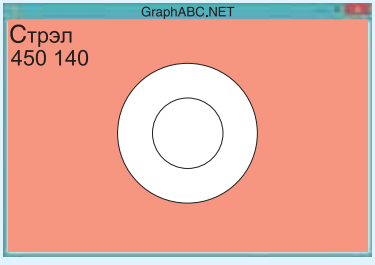


Прыклад 16.4. *Працяг.*

Запусціць праграму яшчэ раз і ўвесці каардынаты стрэлу (300; 240).



Запусціць праграму яшчэ раз і ўвесці каардынаты стрэлу (450; 140).



Прыклад 16.5.

V. Праграма:

```
var a, a1, a2, a3: integer;
begin
  write('Увядзіце a = ');
  read(a);
  if (a > 99) and (a < 1000) then
  begin
    //Першая лічба
    a1 := a div 100;
    //Другая лічба
    a2 := a mod 100 div 10;
    //Трэцяя лічба
    a3 := a mod 10;
    writeln(a1);
    writeln(a2);
    writeln(a3);
  end
  else
    writeln('не трохзначны');
end.
```

1. Рысаванне мішэні: 2 кругі радыусаў $R\_b = 150$ і $R\_m = 75$ з цэнтрам у пункце $(x_0; y_0)$, $x_0 = 320$, $y_0 = 240$. Спачатку рысуем круг большага радыуса.

2. Увод даных: каардынаты пункта стрэлу.

3. Колер рысунка будзе залежаць ад таго, у якую вобласць адносна кругоў трапіў пункт. Магчымы 3 выпадкі:

1) пункт унутры маленькага круга. Даўжыня адрэзка паміж пунктам і цэнтрам круга меншы за радыус. Па тэарэме Піфагора:

$(x - x_0)^2 + (y - y_0)^2 < R\_m^2$;

2) калі ўмова а) не выконваецца, правяраем, ці належыць пункт вялікаму кругу:

$(x - x_0)^2 + (y - y_0)^2 < R\_b^2$;

3) калі ўмовы а) і б) не выконваюцца, то Вася не трапіў у мішэнь.

4. Для скарачэння запісу вызначым пераменную $z = (x - x_0)^2 + (y - y_0)^2$.

5. Закрасім патрэбную вобласць колерам у залежнасці ад праверкі ўмоў.

IV. Апісанне пераменных: x, y, x_0, y_0, R_b, R_m, z маюць тып `integer`.

Прыклад 16.5. Праверыць, ці з'яўляецца ўведзены лік трохзначным, і калі так, то вывесці лічбы гэтага ліку ў асобных радках.

Этапы выканання задання

I. Зыходныя даныя: a (трохзначны лік).

II. Вынік: пераменныя $a1$, $a2$, $a3$ (лічбы ліку) або паведамленне «не трохзначны».

III. Алгарытм рашэння задачы.

1. Увод зыходнага ліку.

2. Праверка ліку. Лік a з'яўляецца трохзначным, калі $99 < a < 1000$.

3. Калі лік трохзначны, вылучаем яго лічбы:

1) для вылучэння першай лічбы $a1$ знаходзім цэлую частку ад дзялення ліку a на 100;

2) для вылучэння другой лічбы $a2$ ліку a знаходзім астачу ад яго дзялення на 100, а затым цэлую частку ад дзялення атрыманай астачы на 10;

3) апошняя лічба ліку $a3$ з'яўляецца астачай ад дзялення ліку a на 10.

4. Вывад выніку.

IV. Апісанне пераменных: усе пераменныя маюць тып `integer`.

Прыклад 16.5. *Працяг.*

VI. Тэсціраванне.

Запусціць праграму і ўвесці значэнне 345.

Вынік наступны:

Окно вывода
```
Увядзіце a = 345
3
4
5
```

Іншыя варыянты зыходных даных:

Окно вывода
```
Увядзіце a = 24
не трохзначны
```

1. Што такое аператар галінавання?
2. Чым адрозніваецца поўны запіс аператара галінавання ад скарочанага?
3. Ці можна выкарыстоўваць састаўныя ўмовы ў аператары галінавання?

Практыкаванні

1 Ці можна змяніць лагічны выраз у аператары галінавання ў прыкладзе 16.1 так, каб паведамленні `'дадатны'` і `'не дадатны'` прыйшлося памяняць месцамі? Калі так, то як гэта зрабіць?

2* Якія змяненні трэба ўнесці ў праграму прыкладу 16.1, каб для ліку разглядаліся тры выпадкі: `'дадатны'`, `'адмоўны'`, `'роўны нулю'`?

3 Падключыце графічны рэжым у праграме прыкладу 16.1. Змяніце праграму так, каб паведамленне `'дадатны'` выводзілася чырвоным колерам, а паведамленне `'не дадатны'` — сінім.

4* Змяніце праграму ў прыкладзе 16.2 так, каб цотнасць (няцотнасць) ліку правяралася з выкарыстаннем функцыі `odd`.

5 Напішыце праграму. Зададзены лік x. Калі лік цотны, то нарысаваць на экране зялёны прамавугольнік, а калі няцотны, то чырвоны круг (гл. прыклад 16.2).

6 Дабаўце ў праграму з прыкладу 16.3 праверку карэктнасці зыходных даных: каардынаты дамоў павінны быць такімі, каб адлегласці да школы былі рознымі. Калі адлегласці аднолькавыя, то вывесці паведамленне `'Каардынаты ўведзены няправільна'`, а калі розныя, то вывесці адказ.

7 Якія змяненні спатрэбіцца ўнесці ў праграму з прыкладу 16.3, калі дапусціць, што дзяўчынкі могуць праходзіць аднолькавыя адлегласці? Унясіце змяненні ў праграму і праверце правільнасць яе работы.

8 Для ўскладнення трэніровак Вася (прыклад 16.4) вырашыў змяняць месцазнаходжанне мішэні і радыусы кругоў. Дабаўце ў праграму магчымасць уводу радыусаў вялікага і маленькага кругоў, а таксама цэнтра мішэні. Праверце правільнасць работы праграмы на розных наборах зыходных даных.

9 Як вядома, шмат якія задачы маюць не адзінае рашэнне. Так, Юля знайшла іншы спосаб вылічэння другой лічбы трохзначнага ліку для прыкладу 16.5. Якую з каманд выкарыстоўвала Юля? Растлумачце, што атрымаецца пры выкананні кожнай з прыведзеных каманд.

1) `a2:=a mod 10 div 10;` 2) `a2:=a div 10 mod 10;` 3) `a2:=a div 100 mod 10.`

10 Праграму з прыкладу 16.5 змянілі. Сфармулюйце ўмову задачы, якая рашаецца з дапамогай гэтай праграмы.

```
var a, a1, a2, a3: integer;
begin
  write('Увядзіце  a = '); read(a);
  if (a > 99) and (a < 1000) then
  begin
    // Першая  лічба
    a1 := a div 100;
    // Другая  лічба
    a2 := a mod 100 div 10;
    // Трэцяя  лічба
    a3 := a mod 10;
    if a1 mod 2 = 0 then
       writeln(a1,'— цотная ');
    if a2 mod 2 = 0 then
       writeln(a2,'— цотная ');
    if a3 mod 2 = 0 then
       writeln(a3, '— цотная);
    if odd(a1) and odd(a2) and odd(a3) then
       writeln('няма  цотных  лічбаў');
  end
  else
    writeln('не  трохзначны');
end.
```

11 Праграму з задання 10 праверылі для некаторых выпадкаў. Ці ўсе магчымыя сітуацыі разгледзелі? Што трэба дабавіць?

Окно вывода	Окно вывода	Окно вывода	Окно вывода
Увядзіце a = 246 2 - цотная 4 - цотная 6 - цотная	Увядзіце a = 103 0 - цотная	Увядзіце a = 537 няма цотных лічбаў	Увядзіце a = 26 не трохзначны

12 Пеця вырашыў удасканаліць праграму з задання 10 і праверку лічбаў у ліку запісаў наступным чынам:

```
if a1 mod 2 = 0 then
 writeln(a1,' — цотная')
else
 if a2 mod 2 = 0 then
   writeln(a2,' — цотная')
 else
   if a3 mod 2 = 0 then
     writeln(a3, ' — цотная')
   else
     writeln('няма цотных лічбаў');
```

Чаму Пецева адзнака аказалася невысокай? Прывядзіце прыклады, для якіх праграма выдае няправільны адказ. Прывядзіце прыклады, у якіх праграма выдае правільны адказ, калі такое магчыма.

13 Дадзены натуральны лік. Напішыце праграму, якая правярае, ці з'яўляецца ён трохзначным і ці кратная 7 сума яго лічбаў.

14* Дадзены натуральны лік. Напішыце праграму, якая правярае, ці з'яўляецца ён чатырохзначным і ці размешчаны яго лічбы ў парадку змяншэння.

15* Вася навучыўся трапляць у цэнтр мішэні з прыкладу 16.4 і вырашыў перайсці да больш складаных трэніровак. Цяпер яго мішэнь уяўляе сабой тры ўкладзеныя кругі з радыусамі $R1$, $R2$, $R3$ (вядома, што $R1 < R2 < R3$). Рэалізуйце камп'ютарную мадэль гэтай мішэні. Колеры выбярыце самастойна.

§ 17. Аператар цыкла

17.1. Аператар цыкла з перадумовай

Алгарытмічная канструкцыя *паўтарэнне* (*цыкл*) уяўляе сабой паслядоўнасць дзеянняў, якія выконваюцца шматразова (гл. блок-схему ў прыкладзе 13.2, с. 60). Саму паслядоўнасць называюць **целам цыкла.**

Аператар цыкла — каманда, якая рэалізуе алгарытмічную канструкцыю *паўтарэнне* на мове праграмірования.

У Pascal існуюць розныя магчымасці кіраваць тым, колькі разоў будзе паўтарацца цела цыкла. Можа быць зададзена ўмова працягвання або

Цыкл з зададзенай умовай заканчэння работы ў PascalABC запісваецца наступным чынам:

```
repeat
  цела цыкла;
until <умова>;
```

Цыкл працуе, пакуль умова непраўдзівая, і спыняе работу, калі ўмова становіцца праўдзівай.

Гэты цыкл называюць **цыклам з постумовай**, паколькі праверка ўмовы ажыццяўляецца пасля выканання цела цыкла. Цыкл з постумовай заўсёды выконваецца хоць бы адзін раз.

Цыклы `repeat` і `while` ў PascalABC узаемазамяняльныя, таму пры напісанні праграм дастаткова выкарыстання толькі аднаго з іх.

Прыклад 17.1.
V. Праграма:

```
uses GraphABC;
var x, y, r: integer;
begin
  r := 10;
  x := 10; y := 10;
  while x < 640 do
  begin
    Circle(x, y, r);
    x := x + 20;
  end;
end.
```

VI. Тэсціраванне.
Запусціць праграму. Вынік:

VII. Лікавае значэнне ва ўмове цыкла можна замяніць функцыяй, якая вызначае гарызантальнае разрашэнне акна: `WindowWidth`:

```
while x < WindowWidth do
```

Функцыі `RedColor`, `GreenColor`, `BlueColor` дазваляюць мяняць інтэнсіўнасць адпаведнага колеру.

```
uses GraphABC;
var x, y, r, c: integer;
begin
  r := 10;
  x := 10; y := 10;
  c := 255;
  while x < 640 do
  begin
   //Інтэнсіўнасць чырвонага
    SetBrushColor(RedColor(c));
    Circle(x,y,r);
    x := x + 20;
   //Памяншэнне інтэнсіўнасці
    c := c-5;
  end;
end.
```

заканчэння работы цыкла, а таксама лік паўтарэнняў цела цыкла.

Цыкл з перадумовай выкарыстоўваецца ў тым выпадку, калі вядома ўмова працягвання работы. Для запісу аператара цыкла з перадумовай выкарыстоўваецца каманда `while`. Фармат каманды:

```
while <умова> do
begin
  цела цыкла;
end;
```

Прыклад 17.1. Напісаць праграму для рысавання рада акружнасцей з радыусам 10 пікселей уздоўж верхняга краю графічнага акна.

Этапы выканання задання

I—II. Вынікам работы праграмы, якая не залежыць ад зыходных даных, будзе рысунак, што адлюстроўвае рад акружнасцей уздоўж верхняга краю графічнага акна.

III. Алгарытм рашэння задачы.

1. *Становішча першай акружнасці*. Акружнасць размесцім ў верхнім левым вугле. Для гэтага задаецца радыус $r = 10$ і каардынаты цэнтра $x = 10$, $y = 10$.

2. *Становішча любой іншай акружнасці*, якая задавальняе ўмову задачы, будзе залежаць ад кардынаты x. У цыкле будзем змяняць значэнне x. Кожнае новае значэнне будзе на 20 (на памер дыяметра) большым за папярэдняе.

3. Цыкл павінен завяршыцца, калі значэнне каардынаты x стане большым, чым 640 — гарызантальны памер акна.

IV. Апісанне пераменных: x, y, r — `integer`.

17.2. Аператар цыкла з параметрам

Цыкл з параметрам выкарыстоўваецца тады, калі вядома колькасць паўтарэнняў.

Для запісу аператара цыкла з параметрам выкарыстоўваецца каманда **for**. Фармат каманды:

```
for var[1] i:= N1 to N2 do
begin
  цела  цыкла;
end;
```

Або

```
for var i:= N2 downto N1 do
begin
  цела  цыкла;
end;
```

У першым варыянце параметр цыкла i змяняецца ад *N1* да *N2*, кожны раз павялічваючыся на 1. У другім — параметр i памяншаецца на 1 пры кожным выкананні цела цыкла ад *N2* да *N1*. Калі *N1* > *N2*, цыкл не выконваецца ні разу. Змяняць значэнне параметра ўнутры цела цыкла нельга.

Прыклад 17.2. Напісаць праграму для вываду табліцы множання на зададзены лік x.

Этапы выканання задання

I. Зыходныя даныя: x (уведзены лік).

II. Вынік: 9 радкоў выгляду $a * x = c$.

III. Алгарытм рашэння задачы.

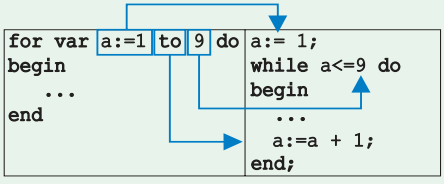
1. Значэнне пераменнай a змяняецца ў цыкле ад 1 да 9.

2. Значэнне пераменнай $c = a \cdot x$.

3. Паколькі колькасць паўтарэнняў загадзя вядомая, выкарыстаем цыкл for.

IV. Апісанне пераменных: x, c — integer.

Любы цыкл **for** можа быць заменены на цыкл **while**:



Адваротнае не заўсёды магчыма.

Прыклад 17.2.

V. Праграма:

```
var x, c : integer;
begin
 write('Увядзіце x = '); read(x);
 for var a : = 1 to 9 do
 begin
   c := a * x;
   writeln(a,' * ',x,' = ',c);
 end;
end.
```

VI. Тэсціраванне.

Запусціць праграму. Увесці $x = 7$.

Окно вывода

```
Увядзіце x = 7
1 * 7 = 7
2 * 7 = 14
3 * 7 = 21
4 * 7 = 28
5 * 7 = 35
6 * 7 = 42
7 * 7 = 49
8 * 7 = 56
9 * 7 = 63
```

Рашэнне з дапамогай цыкла while:

```
var a, x, c : integer;
begin
 write('Увядзіце x = '); read(x);
 a := 1;
 while a < = 9 do
 begin
   c := a * x;
   writeln(a,' * ',x,'= ',c);
   a := a + 1;
 end;
end.
```

VII. Праверыць вынік.

[1] Ключавое слова var можа быць прапушчана, тады пераменная і павінна быць апісана (як integer) у раздзеле апісання var да пачатку праграмы.

Пры рашэнні задач з выкарыстаннем аператара цыкла важна правільна выбраць від цыкла. Калі вядома колькасць паўтарэнняў цела цыкла, то выбіраюць цыкл **for**, а інакш — цыкл **while**.

Унутры цыкла можна выкарыстоўваць аператары break (неадкладны выхад з бягучага цыкла) і аператар continue (пераход да канца цела цыкла).

Прыклад 17.3.

V. Праграма:

```
uses GraphABC;
var a,x1,y1,x2,y2: integer;
begin
  write('Увядзі  a = ');
  read(a); write(a);
  x1 := 50; y1 := 50;
  x2 := 450; y2 := 450;
  for var i := 1 to 20  do
  begin
    Rectangle(x1,y1, x2,y2);
    x1 := x1 + a;   y1 := y1 + a;
    x2 := x2 - a;   y2 := y2 - a;
  end;
end.
```

VI. Тэсціраванне.

Запусціць праграму і ўвесці значэнне $a = 10$. Вынік:

17.3. Рашэнне задач з выкарыстаннем аператара цыкла

Прыклад 17.3. Нарысаваць 20 квадратаў з агульным цэнтрам. Даўжыня стараны самага вялікага квадрата 400, верхні левы вугал размешчаны ў пункце (50; 50). Каардынаты верхняга левага і ніжняга правага вуглоў кожнага наступнага квадрата змяняюцца на a (a — уводзіцца).

Этапы выканання задання

I. Зыходныя даныя: a (уведзены лік).

II. Вынік: рысунак, які адлюстроўвае квадраты.

III. Алгарытм рашэння задачы.

1. Першым рысуецца самы вялікі квадрат. Каардынаты яго верхняга левага вугла $x1 = 50$, $y1 = 50$. Каардынаты ніжняга правага вугла $x2 = 450$, $y2 = 450$.

2. Для вызначэння становішча іншага квадрата трэба каардынаты верхняга левага вугла павялічыць на a, а ніжняга правага — паменшыць на a.

3. Будзем выкарыстоўваць цыкл `for`, паколькі зададзена колькасць квадратаў.

IV. Апісанне пераменных: a, $x1$, $y1$, $x2$, $y2$ — `integer`.

Прыклад 17.4*. Вывесці на экран найбольшы натуральны лік з прамежка $[n, m]$, які дзеліцца на зададзены лік x.

Этапы выканання задання

I. Зыходныя даныя: n, m (межы прамежка), x (зададзены лік).

II. Вынік: шуканы лік або паведамленне «Няма такіх лікаў».

III. Алгарытм рашэння задачы.

1. Няхай i — гэта бягучы лік з прамежка.

2. Паколькі нас цікавіць найбольшы лік з прамежка, то прагляд лікаў пачнём са значэння $i = m$. На кожным кроку будзем памяншаць i на 1.

3. Цыкл завяршыцца, калі мы знайшлі лік, які дзеліцца на x без астачы (астача роўна нулю), або прагледзелі ўсе лікі з прамежка $[n, m]$.

4. Паколькі колькасць паўтарэнняў загадзя невядомая, выкарыстаем цыкл `while`.

Цыкл будзе працягваць работу да таго часу, пакуль умова, сфармуляваная ў пункце 3, будзе непраўдзівай. А менавіта: непраўдзівай павінна быць умова `(i < n) or (i mod x = 0)`. Тады ўмова `not ((i < n) or (i mod x = 0))` будзе праўдзівай. Згодна з правіламі пабудовы адмаўленняў (гл. прыклад 15.5) гэту ўмову можна замяніць умовай: `(i >= n) and (i mod x <> 0)`. Яе і будзем выкарыстоўваць у якасці ўмовы цыкла.

5. Калі па заканчэнні цыкла $i = n - 1$, то няма лікаў, якія задавальняюць умову задачы.

IV. Апісанне пераменных: n, m, x, i — `integer`.

Прыклад 17.4*.

V. Праграма:

```
var i, n, m, x : integer;
begin
 writeln('Увядзіце межы n, m');
 read(n,m);
 write('Увядзіце  x = ');
 read(x);
 i := m;
 while (i >= n) and
       (i mod x <> 0) do
  i := i - 1;
 if i = n - 1 then
  writeln('Няма такіх лікаў')
 else
  writeln('Шуканы лік — ',i);
end.
```

VI. Тэсціраванне.

Запусціць праграму і ўвесці значэнні $n = 10$, $m = 20$, $x = 3$. Вынік:

```
Окно вывода
Увядзіце межы n, m
10 20
Увядзіце x = 3
Шуканы лік - 18
```

Запусціць праграму і ўвесці значэнні $n = 38$, $m = 45$, $x = 37$. Вынік:

```
Окно вывода
Увядзіце межы n, m
38 45
Увядзіце x = 37
Няма такіх лікаў
```

1. Што такое аператар цыкла?
2. Якім чынам можна кіраваць колькасцю выкананняў цела цыкла?
3. Як запісваецца аператар цыкла з перадумовай?
4. Як запісваецца аператар цыкла з параметрам?

Практыкаванні

1 Змяніце праграму з прыкладу 17.1.

1. Радыусы акружнасцей роўны 20.
2. Акружнасці размяшчаюцца ўздоўж левага краю акна.
3. Радыус акружнасці ўводзіцца карыстальнікам.

4. Акружнасці ўтвараюць рамку вакол акна.

5*. Карыстальнік задае мяжу акна, уздоўж якой будуць размяшчацца акружнасці (напрыклад: 1 — верхняя, 2 — левая, 3 — правая, 4 — ніжняя).

2 Якія змяненні трэба ўнесці ў праграму з прыкладу 17.1 для таго, каб рысунак выглядаў наступным чынам?

3 Унясіце змяненні ў праграму з прыкладу 17.2. Карыстальнік задае значэнне другога множніка, а таксама пачатковае і канчатковае значэнні першага множніка.

4 Пры якім максімальным значэнні *a* на экране будуць бачныя ўсё 20 квадратаў з прыкладу 17.3? Чаму пры вялікіх значэннях *a* не бачныя ўсе квадраты? Змяніце праграму так, каб квадраты адлюстроўваліся ад самага маленькага да самага вялікага (устанавіце празрыстую заліўку).

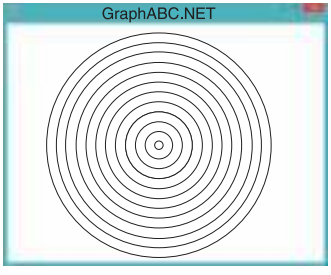
5 Якія змяненні трэба ўнесці ў праграму з прыкладу 17.3, каб атрымаць наступны відарыс? Функцыі для змянення інтэнсіўнасці колеру гл. у прыкладзе 17.1.

6 Змяніце праграму з прыкладу 17.3. Даўжыня стараны самага вялікага квадрата 400, а даўжыня стараны кожнага наступнага квадрата на *x* меншая (*x* уводзіцца).

7 Напішыце праграму, якая рысуе шэраг акружнасцей зададзенага радыуса, размешчаных па дыяганалі графічнага акна. Разгледзьце два варыянты:

1. Графічнае акно квадратнае.

2*. Графічнае акно прамавугольнае.



8 Напішыце праграму, якая рысуе канцэнтрычныя акружнасці з цэнтрам у сярэдзіне графічнага акна. Радыус самай маленькай акружнасці — 10 пікселяў. Розніца радыусаў — 20 пікселяў. Выкарыстоўвайце змяненне інтэнсіўнасці якога-небудзь колеру (або двух адначасова) для заліўкі кругоў.

9 У магазіне прадаюць цукеркі ва ўпакоўках па 0.1 кг, 0.2 кг, ... 0.9 кг, 1 кг. Вядома, што 1 кг цукерак каштуе *x* рублёў. Выведзіце кошты кожнай упакоўкі ў выглядзе:

0.1 кг цукерак каштуе ... р.;

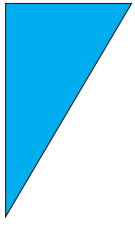
0.2 кг цукерак каштуе ... р. ... .

10* Выведзіце на экран такі найменшы натуральны лік з прамежка [*n*, *m*], які з'яўляецца няцотным і не дзеліцца на ўведзенае значэнне *x*.

§ 18. Складанне алгарытмаў для работы з графікай

18.1. Разлікі ў графічных пабудовах

Прыклад 18.1. Нарысаваць прамавугольны трохвугольнік, які адпавядае рысунку (катэты трохвугольніка паралельныя восям каардынат). Даўжыні катэтаў і каардынаты прамога вугла ўводзяцца.



Этапы выканання задання

I. Зыходныя даныя: a і b (даўжыні катэтаў), x і y (каардынаты вяршыні прамога вугла).

II. Вынік: адлюстраванне прамавугольнага трохвугольніка.

III. Алгарытм рашэння задачы.

1. Увод зыходных даных.

2. Каб адлюстраваць трохвугольнік, трэба выканаць наступныя дзеянні:

1) пабудаваць лініі з пункта з каардынатай $(x; y)$ у пункты з каардынатамі $(x + a; y)$ і $(x; y + b)$;

2) злучыць лініяй пункты $(x + a; y)$ і $(x; y + b)$;

3) зафарбаваць трохвугольнік. Для зафарбоўвання трохвугольніка трэба ведаць каардынаты якога-небудзь пункта ўнутры трохвугольніка. Такім пунктам

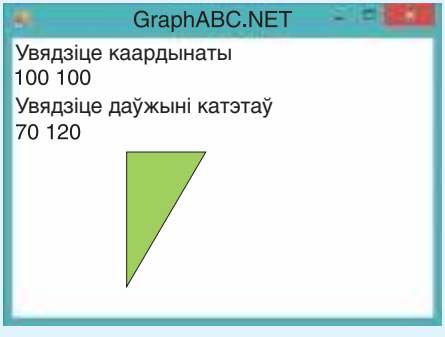
Прыклад 18.1.

V. Праграма:

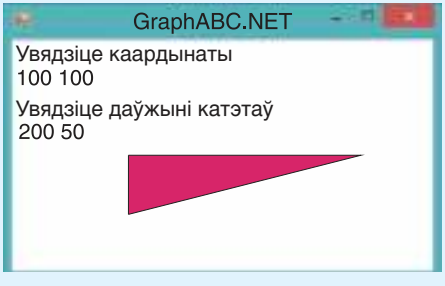
```
uses GraphABC;
var a,b,x,y,x_c, y_c:integer;
begin
 writeln('Увядзіце каардынаты');
  read(x,y); writeln (x,' ',y);
  writeln('Увядзіце  даўжыні
            катэтаў');
  read(a,b); writeln (a,' ',b);
  Line(x,y,x+a,y); Line(x,y,x,y+b);
  Line(x+a,y,x,y+b);
  //Каардынаты пункта
  //Унутры трохвугольніка
  x_c := x + 2; y_c := y + 2;
  FloodFill(x_c,y_c,clRandom);
end.
```

VI. Тэсціраванне.

Запусціць праграму і ўвесці значэнні: каардынаты (100; 100), даўжыні катэтаў 70 і 120. Вынік:



Іншы варыянт:



Прыклад 18.2*.

V. Праграма:

```
uses GraphABC;
var x,y,d: integer;
begin
  writeln('Каардынаты');
  read(x,y);
  writeln (x,' ',y);
  writeln('Старана');
  read(d);
  writeln (d);
  SetBrushColor(clYellow);
  Ellipse (x + d div 3,y,
           x + 2 * d div 3,y + d);
  SetBrushColor(clBrown);
  Pie(x + d div 2,y + d div 3,
      d div 3,0,180);
end.
```

VI. Тэсціраванне.

Запусціць праграму і ўвесці значэнні: каардынаты (100; 100), старана 150. Вынік:

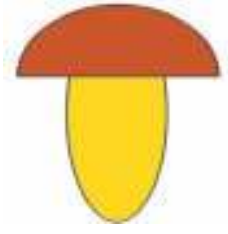
* Грыб можна ўпісаць не ў квадрат, а ў прамавугольнік. У гэтым выпадку трэба задаваць дзве велічыні, якія вызначаюць памеры прамавугольніка: даўжыню ($d1$) і шырыню ($d2$).

у дадзеным выпадку можа быць пункт з каардынатамі $(x + 2; y + 2)$[1].

IV. Апісанне пераменных: усе пераменныя маюць тып `integer`.

Шмат якія графічныя пабудовы можна падагульніць, калі дапусціць, што фігура павінна быць упісана ў квадрат. У гэтым выпадку для пабудовы фігуры дастаткова задаць каардынаты $(x; y)$ верхняга левага вугла квадрата і даўжыню яго стараны — d. Выкарыстоўваючы гэтыя велічыні, можна атрымаць каардынаты іншых вяршынь квадрата: $(x + d; y)$, $(x; y + d)$, $(x + d; y + d)$. Можна атрымаць кардынаты сярэдзіны стараны $(x + d \text{ div } 2; y)$ або цэнтра квадрата $(x + d \text{ div } 2; y + d \text{ div } 2)$.

Прыклад 18.2*. Нарысаваць у графічным акне грыб. Задаць каардынаты верхняга левага вугла квадрата і даўжыню яго стараны для вызначэння месцазнаходжання і памераў грыба.



Этапы выканання задання

I. Зыходныя даныя: x і y — каардынаты верхняга левага вугла квадрата, у які ўпісаны грыб, d — даўжыня яго стараны.

II. Вынік: відарыс грыба.

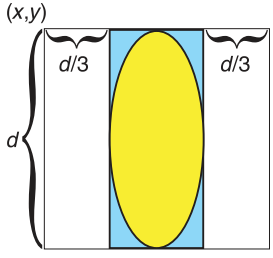
[1] Для адвольнага трохвугольніка можна выкарыстаць формулы: $\left(\frac{x_1 + x_2 + x_3}{3}; \frac{y_1 + y_2 + y_3}{3}\right)$.

III. Алгарытм рашэння задачы.

1. Увод зыходных даных.

2. Для таго каб пабудаваць грыб, трэба выканаць наступныя дзеянні:

1) пабудаваць авал для адлюстравання ножкі грыба. Параметры каманды для адлюстравання эліпса вызначым наступным чынам: `ellipse(x + d div 3, y, x + 2*d div 3, y + d);`



2) нарысаваць шапачку грыба. Для гэтага можна выкарыстоўваць каманду `Pie` (пабудова сектара круга). Каардынаты цэнтра $\left(x + \frac{d}{2}; y + \frac{d}{3}\right)$, радыус — $\frac{d}{3}$. Вуглы роўны 0° і 180° адпаведна.

IV. Апісанне пераменных: усе пераменныя маюць тып `integer`.

Выпадковыя лікі маюць шырокае выкарыстанне ў праграмiраванні. Яны выкарыстоўваюцца ў шыфраванні і ў мадэляванні. Шмат якія камп'ютарныя гульні выкарыстоўваюць выпадковыя лікі. На аснове выпадковых лікаў генерыруюцца капчы і паролі, рэалізуюцца розныя латарэі.

В PascalABC для атрымання выпадковага ліку выкарыстоўваюць функцыю `random`. Спосабы запісу функцыі:

`Random(a, b);` — вяртае выпадковы цэлы ў дыяпазоне ад a да b;

`Random(a);` — вяртае выпадковы цэлы ў дыяпазоне ад 0 до $a - 1$;

Прыклад 18.2*. *Працяг.*

У праграму рысавання грыба трэба ўнесці змяненні, якія дазваляюць разлічыць становішча ножкі і шапачкі грыба адносна каардынат $(x; y)$ і велічынь $d1$ і $d2$.

```
var x,y,d1,d2: integer;
  ...
  writeln('Стораны');
  read(d1, d2);
  writeln (d1, ' ',d2);
  SetBrushColor(clYellow);
  Ellipse(x+d1 div 3, y,
    x + 2 * d1 div 3, y + d2);
  SetBrushColor(clBrown);
  Pie(x + d1 div 2, y + d2 div 3,
     max(d1,d2) div 3, 0, 180);
```

Вынік:

Выпадковы лік — лік, які прымае адно значэнне з мноства, прычым з'яўленне таго ці іншага значэння нельга дакладна прадказаць. Напрыклад, калі б лікі з'явіліся ў выніку выцягвання бочачак у лато, то такая паслядоўнасць лікаў была б выпадковай.

У мовах праграмiравання выкарыстоўваюць **псеўдавыпадковыя лікі**, якія атрымліваюць з выкарыстаннем генератара выпадковых лікаў — алгарытму, што спараджае паслядоўнасць лікаў, элементы якой амаль незалежныя адзін ад аднаго і звычайна размеркаваны раўнамерна на зададзеным інтэрвале.

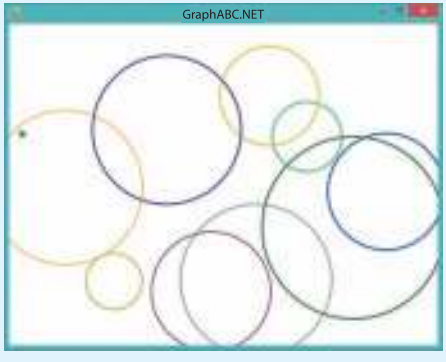
Прыклад 18.3.

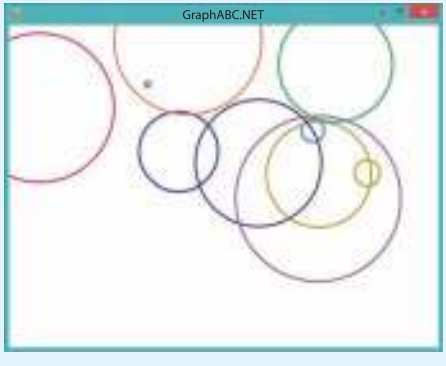
V. Праграма:

```
uses GraphABC;
var x,y,r: integer;
begin
  SetPenWidth(3);
  SetBrushStyle(bsClear);
  for var i:= 1 to 10 do
  begin
    x := random(600);
    y := random(400);
    R := random(150);
    SetPenColor(clRandom);
    circle(x,y,r);
  end;
end.
```

VI. Тэсціраванне.

Запусціць праграму. Павінна быць нарысавана 10 акружнасцей. Розныя варыянты работы праграмы:





`Random;` — вяртае выпадковы рэчыўны ў дыяпазоне [0..1).

Функцыя `clRandom` дазваляе задаць выпадковы колер.

Прыклад 18.3. Напісаць праграму для рысавання на экране 10 рознакаляровых акружнасцей. Размяшчэнне акружнасцей, іх радыусы і колер вызначаюцца выпадковым чынам.

Этапы выканання задання

I—II. Вынікам работы праграмы, якая не залежыць ад зыходных даных, будзе адлюстраванне 10 акружнасцей.

III. Алгарытм рашэння задачы.

1. Установім таўшчыню ліній у 3 пікселі і празрыстую заліўку.

2. Значэнні каардынат цэнтра акружнасці і яе радыуса вызначаюцца функцыяй `random`. Значэнне колеру для мяжы круга — `clRandom`.

3. Паколькі колькасць паўтарэнняў вядомая, будзем выкарыстоўваць цыкл **`for`**.

IV. Апісанне пераменных: *x*, *y* (каардынаты цэнтра), *r* (радыус) маюць тып `integer`.

18.2. Выкарыстанне дапаможных алгарытмаў

Пабудову фігур можна афармляць у выглядзе дапаможных алгарытмаў. Гэта дазволіць выкарыстоўваць такія алгарытмы для рашэння іншых задач.

Усе графічныя працэдуры, якія выкарыстоўваліся раней, мелі параметры. Яны дазвалялі вызначаць месцазнаходжанне і памер фігур. Карыстальнік таксама можа скласці свой дапаможны алгарытм з параметрамі.

Агульны выгляд працэдуры з параметрамі:

```
procedure <імя> (<спіс
          параметраў>:тып);
var ...
begin
  <каманды>
end;
```

Пры выкліку працэдуры важна памятаць, што колькасць параметраў і іх парадак павінны адпавядаць таму, як працэдура апісана.

Прыклад 18.4. Напісаць праграму для пабудовы n раўнабедраных прамавугольных трохвугольнікаў з даўжынёй катэта a. Размяшчэнне трохвугольнікаў вызначаецца выпадковым чынам.

Этапы выканання задання

I. Зыходныя даныя: n (колькасць трохвугольнікаў), a (даўжыня катэта).

II. Вынік: адлюстраванне n трохвугольнікаў.

III. Алгарытм рашэння задачы.

1. У прыкладзе 18.3 адлюстроўвалі акружнасці, размешчаныя выпадковым чынам, а ў прыкладзе 18.1 — прамавугольныя трохвугольнікі. Выкарыстаем праграмы гэтых прыкладаў.

2. Увод значэнняў пераменных n і a.

3. Паколькі колькасць паўтарэнняў вядомая, будзем выкарыстоўваць цыкл **for**.

4. Зменім праграму з прыкладу 18.3. Для гэтага каманду `cirle` (пабудова акружнасці) заменім на каманду пабудовы трохвугольніка:

1) месцазнаходжанне трохвугольніка задаецца каардынатамі прамога вугла, якія вызначым выпадковым чынам;

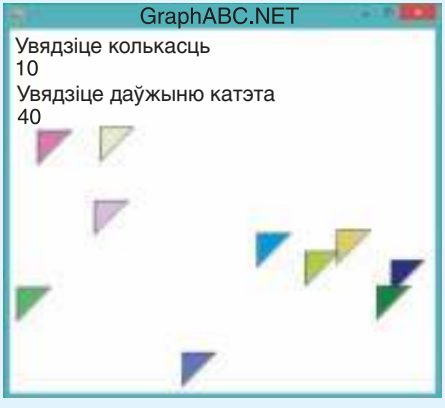
Прыклад 18.4.

V. Праграма:

```
uses GraphABC;
var n, x, y, a : integer;
procedure pr_treug (x, y, a,
          b : integer);
var x_c, y_c:integer;
begin
 line(x, y, x + a,y);
 line(x, y, x, y + b);
 line(x + a, y, x, y + b);
 x_c := x + 2; y_c := y + 2;
 FloodFill(x_c,y_c,clRandom);
end;
begin
 writeln('Увядзіце колькасць');
 read(n); writeln (n);
 writeln('Увядзіце даўжыню
          катэта');
 read(a); writeln (a);
 for var i:= 1 to n do
 begin
  x:= random(500);
  y:= random(400);
  pr_treug(x, y, a, a);
 end;
end.
```

VI. Тэсціраванне.

Запусціць праграму. Вынік:



Пры ўводзе іншых значэнняў вынікі будуць іншымі.

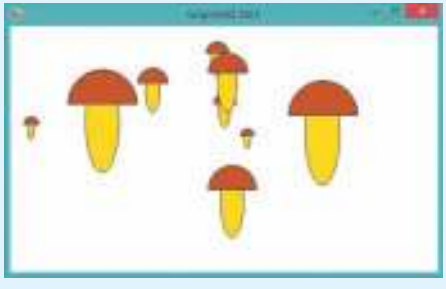
Прыклад 18.5*.
V. Праграма:

```
uses GraphABC;
var x,y,d: integer;

procedure grib(x,y,d:integer);
begin
  SetBrushColor(clYellow);
  Ellipse(x + d div 3, y,
       x + 2*d div 3, y + d);
  SetBrushColor(clBrown);
  Pie(x + d div 2, y + d div 3,
        d div 3, 0, 180);
end;

Begin
  for var i: = 1 to 10 do
  begin
    x:= random(400);
    y:= random(200);
    d:= random(150);
    grib(x,y,d);
  end;
end.
```

VI. Тэсціраванне.
Вынік можа быць наступным:



Можна дабавіць размалёўванне выпадковым колерам:

2) катэты прамавугольнага трохвугольніка маюць аднолькавую даўжыню — значэнне a.

5. Пабудову аднаго прамавугольнага трохвугольніка апішам у дапаможным алгарытме `pr _ treug`. Параметры працэдуры пабудовы трохвугольніка — каардынаты вяршыні прамога вугла і даўжыні катэтаў. Алгарытм апісаны ў прыкладзе 18.2.

IV. Апісанне пераменных: усе пераменныя маюць тып `integer`.

Прыклад 18.5*. Нарысаваць 10 грыбоў. Размяшчэнне і іх памеры вызначаюцца выпадковым чынам.

Этапы выканання задання

I—II. Вынікам работы праграмы, які не залежыць ад зыходных даных, будзе адлюстраванне 10 грыбоў.

III. Алгарытм рашэння задачы.

1. Пабудову аднаго грыба апішам у дапаможным алгарытме. Алгарытм апісаны ў прыкладзе 18.2.

2. Значэнні каардынат верхняга левага вугла і памер грыба вызначаюцца функцыяй `random`.

3. Паколькі колькасць паўтарэнняў вядомая, будзем выкарыстоўваць цыкл **`for`**.

IV. Апісанне пераменных: x, y (каардынаты верхняга левага вугла), d (памер) — `integer`.

Прыклад 18.6. Запоўніць графічнае акно акружнасцямі з радыусам 10.

Этапы выканання задання

I—II. Зыходныя даныя адсутнічаюць. Акружнасці павінны запоўніць усё графічнае акно.

III. Алгарытм рашэння задачы.

1. Задача з'яўляецца абагульненнем задачы з прыкладу 17.1. Каманды праграмы неабходна паўтарыць для нека-

лькіх радоў акружнасцей. Колькасць радоў вызначаецца вышынёй акна. Рысаванне аднаго рада аформім як дапаможны алгарытм `row`.

2. Становішча любога рада акружнасцей вызначаецца каардынатай *y*. Для кожнага значэння *y*, пакуль ён не стане большым, чым вертыкальны памер экрана, выконваем у цыкле наступнае:

1) рысуем рад акружнасцей;

2) змяняем *y*.

IV. Апісанне пераменных: *x*, *y*, *r* — `integer`.

У прыкладзе 18.6 паказана, як запоўніць графічнае акно акружнасцямі. Унёсшы невялікія змяненні ў гэту праграму, можна запаўняць графічнае акно любымі іншымі фігуркамі. Для гэтага дастаткова замяніць каманду `Circle(x,y,r)` у працэдуры `row` на іншую каманду. Можна выбраць графічны прымітыў з бібліятэкі `GraphABC` або самастойна напісаць працэдуру рысавання фігуркі (напрыклад, выкарыстаць працэдуру рысавання грыба з прыкладу 18.5).

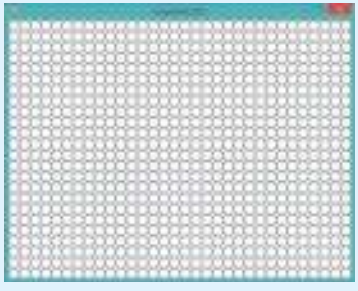
Прыклад 18.6.

V. Праграма:

```
uses GraphABC;
var x, y, r : integer;
procedure row(y : integer);
begin
  x :=10; R := 10;
  while x < WindowWidth do
  begin
    Circle(x,y,r); x := x+20;
  end;
end;
begin
  y := 10;
  while y < = WindowHeight do
  begin
    Row(y); y := y + 20;
  end;
end.
```

VI. Тэсціраванне.

Вынік:



1. Як задаць выпадковы лік?
2. Як задаць выпадковы колер?
3. Як апісаць працэдуру з параметрамі?

Практыкаванні

1 Выканайце заданні для прыкладу 18.1.

1. Паэксперыментуйце з праграмай, уводзячы розныя значэнні зыходных даных.

2. Растлумачце, што адбываецца пры ўводзе адмоўных значэнняў даўжынь катэтаў.

3. Што адбудзецца, калі ўвесці адмоўныя значэнні каардынат? Растлумачце вынік.

2 Выканайце заданні для прыкладу 18.3.

1. Выканайце праграму некалькі разоў. Зніміце празрыстую заліўку. Растлумачце, чаму некаторыя акружнасці нябачныя.

2. Унясіце ў праграму такія змяненні, каб можна было адлюстраваць 20 кругоў; 100 кругоў.
3. Які максімальны памер можа мець радыус круга ў праграме? Унясіце ў праграму змяненні так, каб рысаваліся кругі з радыусам, не большым за 20. Колькасць кругоў устанавіце роўнай 10 000.
4. Унясіце змяненні ў праграму так, каб карыстальнік мог уводзіць колькасць кругоў, якія адлюстроўваюцца на экране.

3 Выканайце заданні для прыкладу 18.4.

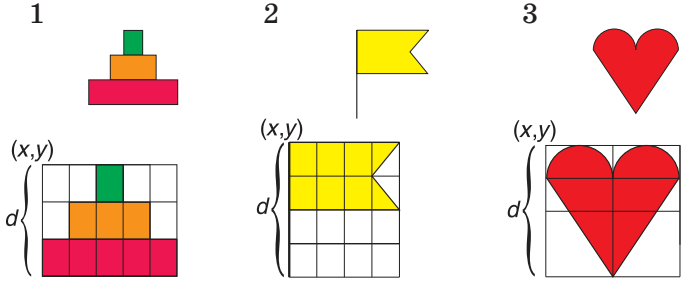
1. Запусціце праграму некалькі разоў. Растлумачце, чаму пры некаторых запусках трохвугольнікі рысуюцца зверху тэкста ў верхнім левым вугле экрана. Змяніце праграму так, каб трохвугольнікі рысаваліся ніжэй за тэкст (правей за тэкст).
2. Дабаўце ў праграму магчымасць уводу даўжыні другога катэта.
3. Змяніце праграму так, каб даўжыні катэтаў задаваліся выпадковым чынам.

4 Напішыце праграму, якая будуе выпадковым чынам відарысы 20 гарызантальных адрэзкаў даўжынёй 30 пікселяў. Распрацуйце два варыянты рашэння задачы. Адзін з выкарыстаннем цыкла **`while`**, а іншы — цыкла **`for`**.

1. Параўнайце дзве праграмы рашэння задачы. Які варыянт рашэння дадзенай задачы ўяўляецца вам лепшым? Чаму?
2. Задайце ў праграме таўшчыню адрэзка ў 3 пікселі.
3. Якія змяненні патрэбны ў праграме, каб таўшчыня адрэзка была выпадковым лікам з прамежека [2; 8]?
4. Унясіце змяненні ў праграму так, каб карыстальнік мог уводзіць колькасць адрэзкаў, якія адлюстроўваюцца на экране.
5. Якія змяненні трэба ўнесці ў праграму, каб замест гарызантальных адрэзкаў адлюстроўваліся вертыкальныя? Дыяганальныя?

5 Выкарыстоўваючы працэдуру рысавання трохвугольніка з прыкладу 18.4, нарысуйце рад трохвугольнікаў уздоўж верхняга (левага) краю графічнага акна.

6* Напішыце праграму для рысавання адной з фігурак. Задаюцца каардынаты верхняга левага вугла і даўжыня стараны квадрата (даўжыні старон прамавугольніка):



7* Напішыце праграму, якая нарысуе рад фігурак уздоўж краю графічнага акна. Выкарыстоўвайце фігуркі, якія рысавалі ў заданні 6.

8 Выканайце заданні для прыкладу 18.6.

1. Змяніце ў праграме значэнне $r = 10$ на $r = 20$. Чаму атрымаўся такі рысунак? Паэксперыментуйце са значэннямі радыуса, устанавіўшы празрыстую заліўку.
2. Якія змяненні трэба ўнесці ў праграму, каб экран запаўняўся кругамі з радыусам 20 без перасячэнняў?
3. Унясіце змяненні ў праграму так, каб усе кругі былі чырвонымі або рознакаляровымі.
4. Унясіце ў праграму змяненні так, каб графічнае акно можна было запаўняць кругамі ўведзенага радыуса.

9* Напішыце праграму, якая запоўніць усё графічнае акно:

1. Грыбамі (прыклад 18.2).
2. Фігуркамі з задання 6.

§ 19. Выкарыстанне асноўных алгарытмічных канструкцый для рашэння практычных задач

19.1. Выкарыстанне лікавых паслядоўнасцей

Лікавыя паслядоўнасці дазваляюць апісваць шмат якія працэсы, што адбываюцца ў прыродзе і грамадстве.

Напрыклад, паслядоўнасць лікаў 2, –1, 0, 2, 0, 1, –2 можа задаваць значэнні тэмпературы па днях тыдня; паслядоўнасць 746, 751, 758 — значэнні сярэдняй заработнай платы супрацоўнікаў прадпрыемства за квартал і г. д.

Паслядоўнасці могуць задавацца формулай, у якой значэнне элемента залежыць ад таго, які ў яго нумар у паслядоўнасці (прыклад 19.1).

Іншым спосабам задання элементаў паслядоўнасці з'яўляецца вызначэнне

Пацвярджэннем важнасці лікавых паслядоўнасцей з'яўляецца той факт, што створана цэлая энцыклапедыя лікавых паслядоўнасцей[1].

Прыклад 19.1. Элементы паслядоўнасці няцотных дадатных лікаў можна апісаць з дапамогай формулы $a_n = 2n - 1$. У гэтай формуле n — нумар элемента ў паслядоўнасці. Мінімальнае значэнне ліку $n = 1$. Выкарыстоўваючы формулу, атрымаем паслядоўнасць: 1, 3, 5, 7... .

Элементы паслядоўнасці могуць быць рэчаіснымі лікамі. Напрыклад, формула $a_n = \frac{n}{n^2 + 1}$ задае наступную паслядоўнасць: 0.5, 0.4, 0.3, 0.235, 0.192, ...

[1] Анлайн-энцыклапедыя цэлалікавых паслядоўнасцей. Рэжым доступу: http://oeis.org/?language=russian (дата доступу: 17.01.2018).

Прыклад 19.2. Адной з найбольш вядомых паслядоўнасцей з'яўляецца паслядоўнасць Фібаначы: 1, 1, 2, 3, 5, 8, 13, ... Нескладана заўважыць, што кожны яе элемент, пачынаючы з трэцяга, роўны суме двух папярэдніх. Гэта можна запісаць так: $f_n = f_{n-1} + f_{n-2}$, $f_1 = 1$, $f_2 = 1$.

Прыклад 19.3. Разгледзім паслядоўнасць 2, 4, 8, 16, ... Кожны лік у гэтай паслядоўнасці з'яўляецца ступенню ліку 2, таму можна задаць паслядоўнасць формулай $a_n = 2^n$. З іншага боку, кожны элемент паслядоўнасці, пачынаючы з другога, у два разы большы за папярэдні. Атрымаем формулу $a_n = 2a_{n-1}$ (для $n > 1$, $a_1 = 2$).

Прыклад 19.4.

V. Праграма:

```
var k, a: integer;
begin
 write('Колькасць k = '); read(k);
  for var n := 1 to k do
  begin
    a:= 2*n; write(a, ' ');
  end;
end.
```

VI. Тэсціраванне.

Запусціць праграму і ўвесці значэнне $k = 5$. Вынік:

Окно вывода

```
Колькасць k = 5
2 4 6 8 10
```

Запусціць праграму і ўвесці значэнне $k = 100$. Вынік:

Окно вывода

```
Колькасць k = 100
2 4 6 8 10 12 14 16 18 20 22 24 26 28
30 32 34 36 38 40 42 44 46 48 50 52
54 56 58 60 62 64 66 68 70 72 74 76
78 80 82 84 86 88 90 92 94 96 98 100
102 104 106 108 110 112 114 116 118
120 122 124 126 128 130 132 134 136
138 140 142 144 146 148 150 152 154
156 158 160 162 164 166 168 170 172
174 176 178 180 182 184 186 188 190
192 194 196 198 200
```

значэння новага элемента праз значэнне папярэдняга (прыклад 19.2).

Ёсць паслядоўнасці, якія можна задаваць як першым, так і другім спосабам (прыклад 19.3). Паслядоўнасці могуць будавацца з выпадковых лікаў.

Прыклад 19.4. Вывесці на экран першыя k цотных лікаў.

Этапы выканання задання

I. Зыходныя даныя: k (колькасць лікаў).

II. Вынік: k цотных лікаў, пачынаючы з 2.

III. Алгарытм рашэння задачы.

1. Увод ліку k.

2. Для атрымання цотнага ліку запішам формулу $a_n = 2n$.

3. Паколькі колькасць лікаў загадзя вядомая, то для іх атрымання можна выкарыстаць цыкл **for**.

4. Бягучы лік будзем захоўваць ў пераменнай a. Значэнне a вылічаецца па формуле і залежыць ад значэння n — лічыльніка цыкла. Пераменная n будзе змяняцца ад 1 (нумар першага цотнага ліку) да k (нумар апошняга ліку).

5. Атрыманыя лікі будзем выводзіць у цыкле праз прабел.

IV. Апісанне пераменных: k, a — `integer`.

Прыклад 19.5. Вывесці на экран усе элементы паслядоўнасці Фібаначы, меншыя за x (x уводзіца).

Этапы выканання задання

I. Зыходныя даныя: x (мяжа для лікаў).

II. Вынік: лікі Фібаначы, меншыя за x.

III. Алгарытм рашэння задачы.

1. Увод ліку x.

2. Вывад першых двух элементаў.

3. Лікі Фібаначы, пачынаючы з трэцяга, атрымліваюць па формуле $a_n = a_{n-1} + a_{n-2}$ ($a_1 = a_2 = 1$). Для вываду лікаў спатрэбяцца тры пераменныя: значэнне a, якое трэба вывесці, і два папярэднія значэнні — b і c.

n	1	2	3	4	5	6	7
Лікі Фібаначы	1	1	2	3	5	8	13
Бягучы крок			c	b	a		
Наступны крок				c	b	a	

Пасля вываду значэння a трэба «зрушыць» значэнні пераменных: $c := b$; $b := a$. Пачатковыя значэнні: $c := 1$; $b := 1$; $a := 2$.

4. Паколькі колькасць лікаў загадзя невядомая, то для іх вылічэння трэба выкарыстоўваць цыкл **while**. Умова працягвання работы цыкла: $a < x$.

5. У цыкле выводзім бягучае значэнне a, «зрушваем» значэнні пераменных і атрымліваем новае значэнне a.

IV. Апісанне пераменных: x, a, b, c — `integer`.

Лікі Фібаначы валодаюць мноствам цікавых матэматычных уласцівасцей. Напрыклад:

1. Могуць быць вылічаны па формуле Бінэ: $f_n = \dfrac{\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n}{\sqrt{5}}$.

2. Адносіна наступнага ліку да папярэдняга з'яўляецца пастаяннай велічынёй ≈ 1.618 (пачынаючы з 13-га). Гэты лік называюць залатым сячэннем.

3. Для трох паслядоўных лікаў Фібаначы правільныя суадносіны Касіні: $f_{n+1}f_{n-1} - f_n^2 = (-1)^n$.

Прыклад 19.5.

V. Праграма:

```
var a, b, c, x: integer;
begin
  write('Мяжа x = '); read(x);
  c := 1; b := 1; a := 2;
  write(c, ' ', b, ' ');
  while a < x do
  begin
    write(a, ' ');
    c := b; b := a; a := b + c;
  end;
end.
```

VI. Тэсціраванне.

Запусціць праграму і ўвесці значэнне $x = 100$. Вынік:

Окно вывода

```
Мяжа x = 100
1 1 2 3 5 8 13 21 34 55 89
```

Каб даведацца, колькі лікаў атрымалі ў якасці выніку, вызначым пераменную k. Пераменная будзе павялічваць сваё значэнне на 1 кожны раз, калі выводзіцца чарговы лік. Пасля завяршэння цыкла можна вывесці значэнне k.

```
var a, b, c, x, k: integer;
begin
  write('Мяжа x= ');
  read(x);
  c := 1; b := 1; a := 2;
  write(c, ' ', b, ' ');
  k := 2;
  while a < x do
  begin
    write(a, ' '); k := k+1;
    c := b; b := a; a := b + c;
  end;
  writeln;
  write('k = ',k);
end.
```

Вынік для $x = 1000$.

Окно вывода

```
Мяжа x = 1000
1 1 2 3 5 8 13 21 34 55 89 144 233
377 610 987
k = 16
```

Прыклад 19.6.

V. Праграма:

```
var n, k, a:integer;
begin
  write('Колькасць білетаў n = ');
  read(n); k:= 0;
  for var i:= 1 to n do
  begin
    a:= random(1,100); write(a,' ');
    if a mod 5 = 0 then
      k := k+1;
  end;
  writeln;
  writeln('Выйграла ',k,
          ' білеты(-а,аў');
end.
```

VI. Тэсціраванне. Запусціць праграму і ўвесці значэнне $n = 20$. Вынік:

Окно вывода

```
Колькасць білетаў  n = 20
56 81 10 34 3 20 43 34 57 54 92 59 47
23 44 90 83 79 29 91
Выйграла 3 білеты (-аў)
```

Пры адным і тым жа значэнні n праграма можа выдаваць розныя вынікі, паколькі лікі атрымліваюцца выпадковым чынам.

Прыклад 19.7.

V. Праграма:

```
var m : integer;
 a, S: real;
begin
 write('Колькасць дзён m = ');
 read(m); S := 0;
 for var n := 1 to m do
 begin
   a := n*n*n/(sqrt(n*n*n)-n+1);
   S := S+a;
 end;
 writeln('Усяго бактэрый =
         = ',S,' млн');
end.
```

VI. Тэсціраванне.

Запусціць праграму і ўвесці значэнне $m = 3$. Вынік:

Окно вывода

```
Колькасць дзён m = 3
Усяго бактэрый = 13.8230024772972 млн
```

Прыклад 19.6. Каця і Пеця вырашылі арганізаваць дабрачынную латарэю. Для гэтага яны выпадковым чынам генерыруюць нумар білета. Нумары білетаў належаць прамежку [1..100]. Выйгрышным білетам будзе той, нумар якога кратны 5. Вызначыць, колькі будзе выйгрышных білетаў сярод n згенерыраваных Кацяй і Пецем.

Этапы выканання задання.

I. Зыходныя даныя: n (колькасць білетаў).

II. Вынік: k — колькасць выйгрышных білетаў.

III. Алгарытм рашэння задачы.

1. Увод ліку n.

2. Да пачатку генерацыі колькасць выйгрышных білетаў роўна нулю ($k := 0$).

3. Паколькі колькасць білетаў загадзя вядомая, то для атрымання іх нумароў можна выкарыстаць цыкл **for**.

4. Бягучы нумар білета будзем захоўваць у пераменнай a. Нумары білетаў будзем атрымліваць па формуле `a = Random(1,100)` і выводзіць на экран. Для кожнга нумара будзем правяраць, ці роўна 0 астача ад дзялення ліку на 5. І калі роўна, то павялічым на 1 значэнне пераменнай k.

5. Вывад выніка.

IV. Апісанне пераменных: n, k, a — `integer`.

19.2. Знаходжанне сумы элементаў лікавай паслядоўнасці

Прыклад 19.7. У лабараторыі выводзяць карысныя бактэрыі. Эксперыментальна было ўстаноўлена, што ко-

лькасць бактэрый (у млн) залежыць ад нумара дня, у які праводзіцца эксперымент, наступным чынам: $a_n = \frac{n^3}{\sqrt{n^3} - n + 1}$. Вызначыце, колькі бактэрый вывелі за m дзён.

Этапы выканання задання

I. Зыходныя даныя: m (лік дзён).

II. Вынік: S (агульная колькасць бактэрый).

III. Алгарытм рашэння задачы.

1. Увод ліку m.

2. Для вылічэння агульнай колькасці бактэрый неабходна паслядоўна дадаваць колькасць бактэрый, выведзеных у дадзены дзень, да ўжо атрыманай колькасці бактэрый. Пачатковае значэнне сумы роўна 0.

3. Паколькі колькасць бактэрый загадзя вядомая, для вылічэння сумы можна выкарыстаць цыкл **for**.

4. Колькасць бактэрый у дадзены дзень будзем захоўваць у пераменнай a. Значэнне a залежыць ад значэння n — лічыльніка дзён. Пераменная n змяняецца ад 1 да m.

5. Вывад выніку S.

IV. Апісанне пераменных: m — `integer`, S, a — `real`.

19.3. Узвядзенне ліку ў ступень

Прыклад 19.8. Узвесці рэчыўны лік a ў цэлую ступень n.

Этапы выканання задання

I. Зыходныя даныя: a (аснова ступені), n (паказальнік ступені).

II. Вынік: S (значэнне ступені).

III. Алгарытм рашэння задачы.

1. Увод лікаў a, n.

Прыклад 19.7. *Працяг.*

Для праверкі правільнасці выніку можна палічыць значэнне сумы на калькулятары:

$$a_1 = 1; a_2 = \frac{2^3}{\sqrt{8} - 2 + 1} \approx 4.38;$$

$$a_3 = \frac{3^3}{\sqrt{27} - 3 + 1} \approx 8.44; S \approx 13.82.$$

Запусціць праграму і ўвесці значэнне $m = 30$. Вынік:

Окно вывода

```
Колькасць дзён m = 30
Усяго бактэрый = 2613.36198051392 млн
```

Прыклад 19.8.

V. Праграма:

```
var n, m: integer;
    a, S: real;
begin
  write('Аснова a = '); read(a);
  write('Паказчык n = '); read(n);
  S := 1; m := abs(n);
  for var i:= 1 to m do
    S := S*a;
  if n<0 then S:= 1/S;
  writeln('Ступень = ',S);
end.
```

VI. Тэсціраванне праграмы.

Запусціць праграму і ўвесці значэнні $a = 5$, $n = 3$. Вынік:

Окно вывода

```
Аснова a = 5
Паказчык n = 3
Ступень = 125
```

Запусціць праграму і ўвесці значэнні $a = 3$, $n = 0$. Вынік:

Окно вывода

```
Аснова a = 3
Паказчык n = 0
Ступень = 1
```

Запусціць праграму і ўвесці значэнні $a = 5$, $n = -2$. Вынік:

Окно вывода

```
Аснова a = 5
Паказчык n = -2
Ступень = 0.04
```

VII. Правільнасць вылічэнняў праверыць на калькулятары.

Прыклад 19.9.

V. Праграма:

```
var k:integer;
   x,y,h:real;
begin
  writeln('Колькасць значэнняў');
  readln(k); x := -3; h := 0.5;
  for var n:= 1 to k do
  begin
    y := (x+2)/(x*x+3);
    writeln(x:7:2,y:10:3); x := x+h;
  end;
end.
```

VI. Тэсціраванне праграмы.

Запусціць праграму і ўвесці значэнне $k = 5$. Вынік:

```
Окно вывода
Колькасць значэнняў
5
  -3.00      -0.083
  -2.50      -0.054
  -2.00      -0.000
  -1.50      -0.059
  -1.00      -0.250
```

Дабавім у праграму вывад меж табліцы:

```
var k:integer;
   x,y,h: real;
begin
  writeln('Колькасць значэнняў');
  readln(k); x := -3; h := 0.5;
  writeln('------------');
  writeln('|  x  |  y  |');
  writeln('------------');
  for var n := 1 to k do
  begin
   y := (x+2)/(x*x+3);
   writeln('|',x:7:2,'|',y:10:3,'|');
   x := x+h;
  end;
  writeln('------------');
end.
```

Вынік пры $k = 4$:

```
Окно вывода
Колькасць значэнняў
4
---------------------------
|      x      |      y      |
---------------------------
|    -3.00    |    -0.083   |
|    -2.50    |    -0.054   |
|    -2.00    |     0.000   |
|    -1.50    |     0.095   |
---------------------------
```

2. Для ўзвядзення ліку ў цэлую неадмоўную ступень трэба паслядоўна памнажаць на аснову ступені тое значэнне, якое атрымалі на папярэднім кроку. Гэта выцякае з роўнасці: $a^n = a^{n-1} \cdot a$.

3. Калі ступень адмоўная, то вынік можна атрымаць з роўнасці: $a^{-n} = \frac{1}{a^n}$. Колькасць паўтарэнняў цыкла m можна вызначыць як $m = |n|$.

4. Для вылічэння здабытку можна выкарыстаць цыкл **for**. Пачатковае значэнне ступені роўна 1.

5. У канцы праверым, дадатным ці адмоўным з'яўляецца значэнне n. Калі яно адмоўнае, то трэба змяніць значэнне пераменнай S.

6. Вывад выніку S.

IV. Апісанне пераменных: m, n — `integer`, a, S — `real`.

19.4. Пабудова табліцы значэнняў функцыі

Прыклад 19.9. Вывесці на экран табліцу значэнняў функцыі $y = \frac{x+2}{x^2+3}$. Колькасць значэнняў уводзіцца. Пачатковае значэнне $x = -3$, значэнні аргумента выводзяцца з крокам $h = 0{,}5$.

Этапы выканання задання

I. Зыходныя даныя: k (колькасць пунктаў).

II. Вынік: k значэнняў аргумента і значэнняў функцыі, якія ім адпавядаюць.

III. Алгарытм рашэння задачы.

1. Увод ліку k.

2. Для атрымання табліцы трэба ў цыкле вылічаць і выводзіць значэнне

аргумента і значэнне функцыі, якое яму адпавядае:

1) пачатковае значэнне аргумента $x = -3$. Для атрымання чарговага значэння аргумента трэба да бягучага значэння дадаць крок h;

2) значэнне функцыі вылічаецца па формуле $y = \frac{x+2}{x^2+3}$;

3) атрыманыя значэнні выводзяцца на экран. Для таго каб значэнні выводзіліся строга адно пад адным, трэба выкарыстоўваць фарматны вывад. Для гэтага задаць колькасць пазіцый для выводу значэння і вызначыць колькасць лічбаў пасля коскі. Запіс x:7:2 азначае, што для выводу пераменнай выкарыстоўваецца 7 пазіцый, пасля коскі выводзяцца 2 лічбы.

3. Паколькі колькасць пунктаў вядомая, выкарыстаем цыкл **for**.

IV. Апісанне пераменных: k — integer, x, y, h — real.

19.5. Вылучэнне лічбаў з ліку

Прыклад 19.10. Дадзены натуральны лік n. Вывесці лічбы ліку па адной у радку (пачынаючы з разраду адзінак). Вызначыць, колькі лічбаў у ліку.

Этапы выканання задання

I. Выходныя даныя: n (лік).

II. Вынік: z (бягучая лічба ліку), k (колькасць лічбаў у ліку n).

III. Алгарытм рашэння задачы.

1. Увод зыходных даных — лік n.

2. Вызначэнне пачатковага значэння лічыльніка для колькасці лічбаў ($k := 0$).

3. Колькасць лічбаў ліку роўна колькасці дзесятковых разрадаў у ліку.

Паняцце ліку ўзнікла ў глыбокай старажытнасці з практычнай патрэбы людзей. Для запісу лікаў выкарыстоўваюць лічбы. Любая інфармацыя ў камп'ютары ўяўляецца з дапамогай усяго дзвюх лічбаў (0 і 1).

Лікавы код мае кожная краіна свету, лічбы задаюць пін-код банкаўскай карты. Сёння з дапамогай лічбаў можна атрымаць лікавы вобраз практычна любога аб'екта.

Прыклад 19.10.

V. Праграма:

```
var k,n,z: integer;
begin
  write('Увядзіце n = ');
  read(n);
  k := 0;
  while n > 0 do
  begin
    //Бягучая  лічба
    z := n mod 10;
    writeln(z);
    //Памяншэнне ліку ў 10 разоў
    n := n div 10;
    //Падлік колькасці лічбаў
    k := k + 1;
  end;
  writeln('у ліку ', k,
          'лічбаў(-а/-ы))');
end.
```

VI. Тэсціраванне.

Запусціць праграму і ўвесці значэнне $n = 13579$. Вынік:

Окно вывода

```
Увядзіце  n = 13759
9
7
5
3
1
У ліку 5 лічбаў(-а/-ы)
```

Запусціць праграму і ўвесці значэнне $n = 1$. Вынік:

Окно вывода

```
Увядзіце  n = 10
1
У ліку 1 лічбаў(-а/-ы)
```

Алгарытм Эўкліда — алгарытм для знаходжання найбольшага агульнага дзельніка двух цэлых лікаў. Алгарытм названы ў гонар старажытнагрэчаскага матэматыка Эўкліда (III ст. да н. э.), які ўпершыню апісаў яго ў кнігах «Пачаткі». Гэта адзін з найбольш старых лікавых алгарытмаў, якія выкарыстоўваюцца ў наш час.

Эўклід выкарыстоўваў дадзены алгарытм не толькі для лікаў, але і для геаметрычных велічынь: даўжынь, плошчаў, аб'ёмаў. У XIX ст. алгарытм быў абагульнены на іншыя матэматычныя аб'екты. Сёння алгарытм Эўкліда выкарыстоўваецца пры шыфраванні даных.

Прыклад 19.11. Алгарытм Эўкліда для лікаў 42 і 24:

№	a	b
1	42	24
2	18 (42–24)	24
3	18	6 (24–18)
4	12 (18–6)	6
5	6 (12–6)	6

Прыклад 19.12.

V. Праграма:

```
var a,b:integer;
begin
  write('Значкі ў Іры a = ');
  read(a);
  write('Значкі ў Ігара b = ');
  read(b);
  while a<>b do
    if a>b then
      a := a - b
    else
      b := b - a;
  writeln('Усяго сяброў = ',a);
end.
```

VI. Тэсціраванне.

Запусціць праграму і ўвесці значэнні: $a = 42$, $b = 24$. Вынік:

Окно вывода

```
Значкі ў Іры a = 42
Значкі ў Ігара b = 24
Усяго сяброў = 6
```

Для знаходжання кожнай лічбы ліку трэба:

а) падзяліць лік на 10;

б) знайсці цэлую частку ад дзялення і астачу (астача і будзе чарговай лічбай) і павялічыць лічыльнік колькасці лічбаў;

в) вывесці атрыманую лічбу;

г) паколькі колькасць лічбаў у ліку загадзя невядомая, будзем выкарыстоўваць цыкл **while**. Пакуль цэлая частка ад дзялення большая за 0, у ліку яшчэ ёсць лічбы і трэба перайсці да выканання пункта а), інакш усе лічбы знойдзены.

4. Вывад значэння пераменнай k.

IV. Апісанне пераменных: k, n, z — `integer`.

19.6. Найбольшы агульны дзельнік двух лікаў

Найбольшым агульным дзельнікам (НАД) для двух цэлых лікаў называюць найбольшы з іх агульных дзельнікаў. Прыклад: для лікаў 42 і 24 найбольшы агульны дзельнік роўны 6.

Існуюць некалькі алгарытмаў знаходжання НАД. З адным з іх вы знаёміліся на ўроках матэматыкі. Трэба расклясці кожны з лікаў на простыя множнікі, выбраць агульныя і перамножыць.

Разгледзім іншы алгарытм, які называецца алгарытм Эўкліда.

1. Ад большага ліку аднімаем меншы.

2. Калі атрымаецца 0, то лікі роўныя адзін аднаму і гэта значэнне з'яўляецца НАД.

3. Калі вынік аднімання не роўны 0, то большы лік замяняем на рознасць большага і меншага.

4. Пераходзім да пункта 1.

(Разгледзьце прыклад 19.11.)

Прыклад 19.12. Іра і Ігар калекцыніруюць значкі. У Іры ў калекцыі a значкоў, а ў Ігара — b. Паколькі значкоў шмат, рабяты вырашылі падзяліцца сваімі значкамі з сябрамі. Якая найбольшая колькасць агульных сяброў можа быць у Іры і Ігара, калі кожны з іх хоча падзяліць усе свае значкі паміж сябрамі без астачы? Напрыклад, калі $a = 42$ і $b = 24$, то значкі можна падзяліць, калі ў Іры і Ігара 1, 2, 3 або 6 агульных сяброў. Найбольшая колькасць — 6.

Этапы выканання задання

I. Зыходныя даныя: a і b (колькасць значкоў у Іры і ў Ігара).

II. Вынік: найбольшая колькасць агульных сяброў.

III. Алгарытм рашэння задачы.

1. Увод лікаў a, b.

2. Паколькі значкі трэба дзяліць без астачы, то адказам на задачу можа быць толькі агульны дзельнік лікаў a і b. Сярод усіх дзельнікаў трэба знайсці найбольшы.

3. Для рашэння задачы напішам праграму вылічэння НАД(a, b) па алгарытме Эўкліда. Пакуль лікі a і b не роўныя, выканаем наступнае:

1) параўноўваем два лікі;

2) калі $a > b$, замяняем a на рознасць $a - b$, інакш замяняем b на рознасць $b - a$.

4. Вывад выніку. Вывесці можна як значэнне a, так і b (паколькі яны роўныя).

IV. Апісанне пераменных: a, b — `integer`.

Прыклад 19.13*. Напісаць праграму вылічэння НАД(x, y, z).

Прыклад 19.12. *Працяг.*

Для значэнняў $a = 1449$, $b = 596$ атрымаем:

```
Окно вывода
Значкі ў Іры a = 1449
Значкі ў Ігара b = 596
Усяго сяброў = 1
```

Дадзены вынік азначае, што лікі 1449 і 596 узаемна простыя і ўсе значкі можна аддаць толькі аднаму сябру.

*У Pascal, акрамя працэдур, часта выкарыстоўваюцца дапаможныя алгарытмы ў выглядзе функцый. Функцыя заўсёды вяртае значэнне, якое трэба прысвоіць якой-небудзь пераменнай або выкарыстоўваць у любым выразе: `b := abs(x), t := 2 + sqrt(a)`.

Агульны выгляд функцыі:

```
function <імя>(<спіс параметраў>:тып): тып выніку;
var ... //Можа адсутнічаць
begin
  <каманды>
  <імя>:= <значэнне>;
end;
```

Прыклад 19.13*.

V. Праграма:

```
var x,y,z,d,f:integer;
function NOD
         (a,b:integer):integer;
begin
  while a<>b do
    if a>b then
    a := a-b
    else
    b := b-a;
  NOD := a;
end;
begin
  write('Увядзіце x = ');
  read(x);
  write('Увядзіце y = ');
  read(y);
  write('Увядзіце z = ');
  read(z);
  d := NOD(x,y);
  f := NOD(d,z);
  writeln('НАД = ',f);
end.
```

Прыклад 19.13*. *Працяг.*

VI. Тэсціраванне.

Запусціць праграму і ўвесці значэнні: $x = 26$, $y = 143$, $z = 65$. Вынік:

```
Окно вывода
Увядзіце x = 26
Увядзіце y = 143
Увядзіце z = 65
НАД = 13
```

Загрузіць праграму і ўвесці значэнні: $x = 354$, $y = 847$, $z = 125$. Вынік:

```
Окно вывода
Увядзіце x = 354
Увядзіце y = 847
Увядзіце z = 125
НАД = 1
```

Дадзены вынік азначае, што лікі 354, 847 і 125 узаемна простыя.

Этапы выканання задання

I. Зыходныя даныя: x, y і z (тры лікі).

II. Вынік: НАД (x, y, z).

III. Алгарытм рашэння задачы.

1. Увод лікаў x, y, z.

2. Выкарыстаем тое, што НАД (x, y, z) = НАД (НАД (x, y), z). Гэта значыць, спачатку вылічым d = НАД (x, y), а затым f = НАД (d, z).

3. Для вылічэння НАД двух лікаў складзём функцыю NOD (a, b). Каманды функцыі NOD разгледжаны ў прыкладзе 19.12.

4. Вывад выніку.

IV. Апісанне пераменных: x, y, z, d, f — `integer`.

Практыкаванні

1 Выканайце заданні для прыкладу 19.4.

1. Унясіце ў праграму змяненні так, каб лікі выводзіліся ў адваротным парадку — ад большага да 2.
2. Якія змяненні трэба ўнесці ў праграму, каб вывесці ўсе цотныя лікі, меншыя за ўведзены лік x?
3. Унясіце змяненні ў праграму так, каб выводзіліся лікі, кратныя 3, 5; кратныя ўведзенаму ліку x.
4. Змяніце праграму так, каб можна было вывесці ўсе лікі паслядоўнасці, зададзенай формулай $a_n = \frac{n}{n^2+1}$ (прыклад 19.1).

2 Выканайце заданні для прыкладу 19.5.

1. Запусціце праграму для розных значэнняў x.
2. Якое максімальнае значэнне x можна ўвесці?
3. Замяніце ў праграме тып `integer` на тып `int64`. Колькі лікаў паслядоўнасці Фібаначы можна знайсці цяпер?
4. Змяніце праграму так, каб яна выводзіла значэнне ліку Фібаначы па ўведзеным нумары.

3 Выканайце заданні для прыкладу 19.6.

1. Запусціце праграму некалькі разоў для аднаго і таго ж значэння (напрыклад, 20). Які лік атрымліваецца ў адказе часцей за ўсё? Чаму?

2. Змяніце дыяпазон выпадковых лікаў на [1, 1000] і праверце, які лік атрымліваецца ў выніку часцей за іншыя для тых жа 20 элементаў.

3. Змяніце праграму так, каб вылічалася колькасць лікаў, кратных уведзенаму ліку x.

4. Змяніце праграму так, каб можна было палічыць колькасць лікаў з прамежку $[a; b]$, a і b уводзяцца.

4* Напішыце праграму, якая будзе выводзіць на экран элементы паслядоўнасці трыбаначы — першыя элементы паслядоўнасці: 0, 0, 1, 1, 2, 4, 7, 13, 24, 44, 81, 149, ... Кожны элемент, пачынаючы з чацвёртага, роўны суме трох папярэдніх: $a_n = a_{n-1} + a_{n-2} + a_{n-3}$.

1. Па зададзеным n вывесці элемент паслядоўнасці.

2. Для зададзенага x вывесці элементы паслядоўнасці, меншыя за x.

5 Выканайце заданні для прыкладу 19.7.

1. Замяніце ў рашэнні задачы цыкл **for** на цыкл **while**.

2. Выканайце праграму для $m = 2000$. Чаму ў адказе сума = NaN? Якія змяненні трэба ўнесці ў праграму для атрымання выніку? Знайдзіце значэнне для $m = 2000$, $m = 10\,000$.

6 Знайдзіце суму першых m элементаў паслядоўнасці. Лік m уводзіцца. Элементы паслядоўнасці задаюцца формулай $a_n = \frac{1}{n^3}$.

1. На колькі большым павінна быць значэнне m, каб праграма выдала адказ «сумма = Infinity»? Чаму так адбылося?

2. Якія змяненні трэба ўнесці ў праграму для атрымання правільнага выніку?

3. Замяніце ў рашэнні задачы цыкл **for** на цыкл **while**. Знайдзіце суму для $m = 20\,000$, $m = 1\,000\,000$.

7 Выканайце заданне для прыкладу 19.8. Змяніце форму вываду выніку такім чынам, каб вынік выводзіўся ў выглядзе a ^ n = S (напрыклад, для значэнняў $a = 2$, $n = 3$ павінна быць надрукавана 2 ^ 3 = 8).

8 Фактарыялам ліку n называюць здабытак усіх натуральных лікаў, якія не пераўзыходзяць n. Абазначаюць фактарыял так: $n!$ Па азначэнні фактарыял ліку 0 роўны 1. Напішыце праграму, якая вылічыць значэнне фактарыяла цэлага неадмоўнага ліку n. Для праверкі можна выкарыстоўваць наступнае: 0! = 1; 2! = 2; 5! = 120, 10! = 3 628 800.

9 Выканайце заданне для прыкладу 19.9. Замяніце ў рашэнні задачы цыкл **for** на цыкл **while**. У якасці ўмовы ў цыкле **while** можна выкарыстоўваць наступнае: $x <= 3$.

10 Пабудуйце табліцы значэнняў для дадзеных функцый.

1. $y = x^2 - 5x - 3$, $x \in [-3, 3]$, уводзіцца значэнне кроку h.

2. $y = 2 + \frac{3x^3 - 7}{x}$, $x \in [a, b]$[1], уводзяцца значэнні a, b і колькасць пунктаў.

[1] Падказка: $h = \frac{a + b}{k - 1}$.

11 Выканайце заданні для прыкладу 19.10.

1. Каманду `writeln('у ліку ', k, ' лічбаў)` замянілі камандай `writeln('лік ',n,' складаецца з ', k, ' лічбаў')`. Які вынік будзе атрыманы і чаму? Якія змяненні трэба ўнесці ў праграму для атрымання правільнага выніку?
2. Ці зменіцца вынік работы праграмы, калі замест умовы цыкла $n < 0$ выкарыстаць умову $n > 1$?
3. Праверце работу праграмы для $n = 0$. Чаму атрымаўся такі вынік? Што трэба змяніць у праграме для атрымання правільнага выніку?

12 Праграму з прыкладу 19.10 змянілі. Сфармулюйце задачу, якая рашаецца з дапамогай дадзенай праграмы.

```
var i,k,n,z: integer;
begin
  write('Увядзіце n = ');
  read(n);
  write('Увядзіце i = ');
  read(i);
  k:=0;
while n > 0 do
begin
  z : = n mod 10; //Бягучая  лічба
  k : = k + 1;
  if k = i then
    writeln('У  разрадзе ', i, ' стаіць  лічба ', z);
  N : = n div 10; //Памяншэнне ліку ў 10 разоў
end;
if i > k then
  writeln('У ліку  ', k, ' лічбаў, у разрадзе ', i, ' няма  лічбаў')
else
  writeln('У  ліку ', k, ' лічбаў);
end.
```

13 Дадзены натуральны лік *n*. Вызначыце, якіх лічбаў у ліку больш — цотных або няцотных.

14 Дадзены натуральны лік *n*. Выведзіце нумары разрадаў, у якіх стаяць лічбы, кратныя 3, або паведамленне, што такіх лічбаў няма.

15 Зададзены натуральны лік. Напішыце праграму, якая для ліку з няцотнай колькасцю лічбаў выведзе на экран лічбу, што стаіць на сярэдняй пазіцыі ліку. Калі ў ліку цотная колькасць лічбаў, то вывесці адпаведнае паведамленне. Напрыклад,

для ліку 23 452 адказам будзе 4, а для ліку 56 — паведамленне «У ліку цотная колькасць лічбаў».

16 Лік з'яўляецца паліндромам, калі ён аднолькава чытаецца злева направа і справа налева. Напішыце праграму, якая па ўведзеным натуральным ліку вызначыць, з'яўляецца ён паліндромам ці не. Прыклады: 12321, 6776 — паліндромы, 12335 — не.

17* Натуральны лік n называецца лікам Армстранга, калі сума лічбаў ліку, узведзеных у n-ю ступень, дзе n — колькасць лічбаў у ліку, роўная самому ліку. Напрыклад, $153 = 1^3 + 5^3 + 3^3$. Напішыце праграму, якая знойдзе:

1. Усе трохзначныя лікі Армстранга.
2. Усе чатырохзначныя лікі Армстранга.

18 У 1626 г. індзейцы прадалі востраў Манхэтэн за 20 долараў. Калі б гэтыя грошы былі змешчаны ў банк на бягучы рахунак і штогадовы прырост складаў бы x %, якая была б вартасць капіталу ў гэтым годзе? Напішыце праграму, якая адкажа на дадзенае пытанне.

19 Маюцца дзве пасудзіны. У першай знаходзіцца C_1 л вады, а ў другой C_2 л вады. З першай пасудзіны пераліваюць палову вады ў другую, а затым з другой пераліваюць палову вады ў першую (адно пераліванне) і г. д. Напішыце праграму, якая вызначыць, колькі вады апынецца ў кожнай з пасудзін пасля k пераліванняў.

20 Дадзены натуральны лік n. Напісаць праграму, якая выведзе ўсе лікі, узаемна простыя з n, а таксама лікі, меншыя за яго.

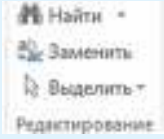
21* Змяніце праграму з прыкладу 19.13:

1. Знайсці НАД чатырох лікаў.
2. Знайсці НАК (найменшае агульнае кратнае) двух лікаў.
3. Уводзяцца лічнік і назоўнік правільнага дробу. Скараціце дроб.
4*. Два правільныя дробы зададзены сваімі лічнікамі і назоўнікамі. Знайдзіце іх суму. Адказ выведзіце ў выглядзе змешанага дробу.

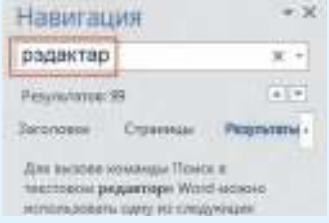
Глава 4
ТЭХНАЛОГІЯ АПРАЦОЎКІ ТЭКСТАВЫХ ДАКУМЕНТАЎ

§ 20. Рэдагаванне тэксту

Прыклад 20.1. Група **Редактирование**.



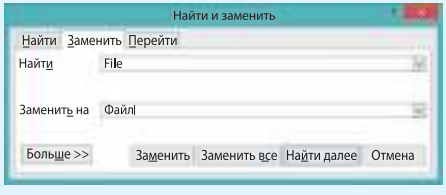
Прыклад 20.2. Панэль **Навигация**.



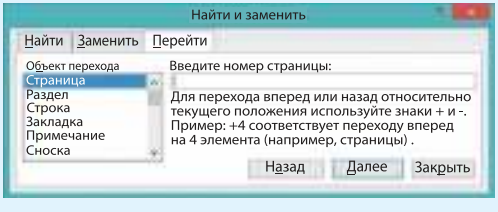
Знойдзеныя словы вылучаны колерам:

Для выкліку каманды **Поиск** у тэкставым рэдактары Word можна выкарыстоўваць адну з наступных магчымасцей:

Прыклад 20.3. Укладка **Заменить** у акне **Найти и заменить**.



Укладка **Перейти** акна **Найти и заменить** змяшчае каманды для навігацыі па структурных элементах тэкставага дакумента.



20.1. Пошук і замена ў тэксце

Тэкставыя рэдактары дазваляюць ажыццяўляць у тэксце пошук слоў або словазлучэнняў і пры неабходнасці замяняць іх іншымі словамі, словазлучэннямі.

Для выкліку каманды **Поиск** у тэкставым рэдактары Word можна выкарыстоўваць адну з наступных магчымасцей:

- каманду **Найти** на ўкладцы **Главная** ў групе **Редактирование** (прыклад 20.1);
- камбінацыю клавіш Ctrl + F.

Пасля выкліку каманды **Поиск** неабходна ўвесці ўзор для пошуку ў адпаведным полі панэлі **Навигация** (прыклад 20.2).

Для выкліку каманды **Заменить** выкарыстоўваецца адна з наступных магчымасцей:

- каманда **Заменить** на ўкладцы **Главная** ў групе **Редактирование** (гл. прыклад 20.1);
- камбінацыя клавіш Ctrl + H.

Пасля таго як выбрана каманда **Заменить**, неабходна запоўніць узор пошуку ў полі **Найти** (што шукаем) і ўзор замены ў полі **Заменить** (на што замяняем). Выгляд акна пры замене тэксту паказаны ў прыкладзе 20.3.

У *Дадатку 4* (с. 163) можна азнаёміцца з іншымі магчымасцямі пошуку і замены.

20.2. Праверка правапісу

Пры стварэнні тэкставага дакумента магчымы памылкі. Сучасныя тэкставыя рэдактары маюць убудаваныя сістэмы праверкі правапісу.

Сістэма праверкі правапісу — камп'ютарная праграма, якая ажыццяўляе праверку зададзенага тэксту на наяўнасць у ім арфаграфічных памылак.

Тэкставы рэдактар Word ажыццяўляе пошук памылак з дапамогай убудаваных слоўнікаў (прыклад 20.4). Калі слова з тэксту адсутнічае ў слоўніку, то яно падкрэсліваецца хвалістай чырвонай лініяй. Часткі тэксту, у якіх дапушчана стылістычная памылка або памылка фармаціравання тэксту, падкрэсліваюцца хвалістай блакітнай або зялёнай лініяй.

Пра магчымыя правільныя варыянты напісання слова або расстаноўкі знакаў прыпынку можна даведацца з кантэкставага меню падкрэсленага слова (прыклад 20.5). Аналагічны вынік можна атрымаць, калі пстрыкнуць па значку  , які размешчаны ў радку стану. Справа адкрыецца панэль, у якой паказаны варыянты выпраўлення памылкі (прыклад 20.6) або тлумачэнне таго, чаму тэкст падкрэслены блакітнай (зялёнай) хвалістай лініяй (прыклад 20.7).

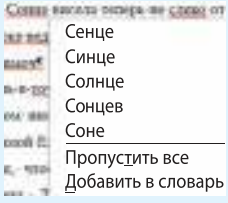
Успомнім некаторыя правілы ўводу камп'ютарнага тэксту:

1. Раздзяляльнікам паміж двума словамі з'яўляецца **адзін** прабел. Большая колькасць прабелаў успрымаецца як памылка і падкрэсліваецца блакітнай хвалістай лініяй.

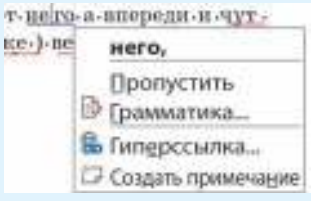
Прыклад 20.4. Вылучэнне памылак у Word.

Но · влева · некакой · дороги ·· не · было. · Сонце · висела · теперь · не · слево · от · него · а · впереди · и · чут · справо. · Вправа · (· на · бугарке ·) · веднелся · хутор.¶

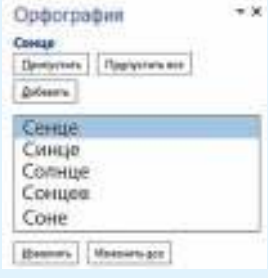
Прыклад 20.5. Варыянты правільнага напісання слова ў кантэкставым меню:



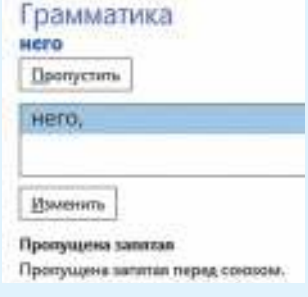
Расстаноўка знакаў прыпынку:



Прыклад 20.6. Панэль **Орфография**.



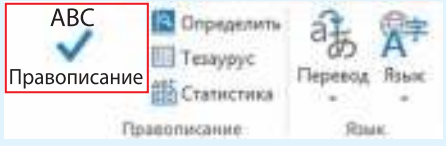
Прыклад 20.7. Панэль **Грамматика**.



Сістэмы праверкі правапісу на персанальных камп'ютарах з'явіліся ў 1980 г. і былі аўтаномнымі праграмамі. У 1980-х гг. іх уключылі ў склад праграм для работы з тэкстам.

Сёння праверка правапісу існуе не толькі для тэкставых рэдактараў, але і для вэб-браўзераў.

Прыклад 20.8. Каманды арфаграфічнага кантролю на ўкладцы **Рецензирование**.

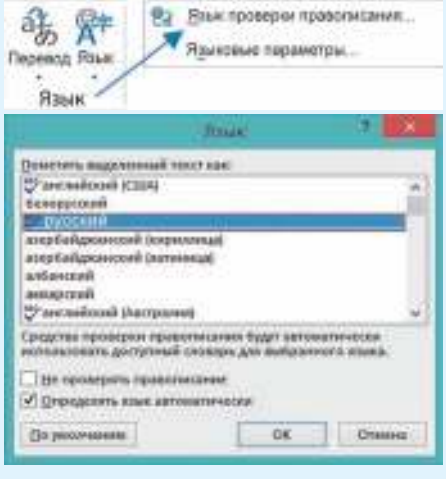


У Word можна шукаць сінонімы або антонімы для вылучанага слова (каманда **Рецензирование** → **Тезаурус**).

Для расстаноўкі пераносаў у тэксце выкарыстоўваецца каманда **Разметка страницы** → **Расстановка переносов**.

Для тых моў, для якіх ажыццяўляецца арфаграфічны кантроль, магчымы пераклад тэксту або асобных слоў. Для гэтага выкарыстоўваецца каманда **Рецензирование** → **Перевод**.

Прыклад 20.9. Окно **Язык**.



2. Пасля знакаў прыпынку («.», «,», «:», «;», «!», «?») абавязкова павінен быць прабел (але не перад імі). Калі пасля знака прыпынку прабелу няма, а адразу запісана новае слова, то Word успрымае гэтыя два словы як адно і падкрэслівае як арфаграфічную памылку.

3. Працяжнік аддзяляецца прабеламі з двух бакоў. Злучок прабеламі не аддзяляецца. Для пастаноўкі доўгага працяжніка (—), а не кароткага (–) можна выкарыстаць камбінацыю клавіш Ctrl + «-» (на лічбавай клавіятуры).

4. Прабел ставіцца перад адкрытай дужкай і пасля закрытай. Пасля адкрытай дужкі і перад закрытай прабел не ставіцца. Гэта ж правіла ўжываецца пры выкарыстанні двукоссяў.

Калі аўтаматычная праверка арфаграфіі адключана, знайсці памылкі ў тэксце можна, запусціўшы праверку правапісу камандай **Правописание** ўкладкі **Рецензирование** (прыклад 20.8) або клавішай F7.

Арфаграфічны кантроль ажыццяўляецца не для ўсіх даступных моў. Спіс даступных моў можна паглядзець, выбраўшы каманду **Язык проверки правописания** з выпадаючага спіса каманды **Язык** (прыклад 20.9). Калі побач з назвай мовы стаіць значок ABC, то для дадзенай мовы магчымы арфаграфічны кантроль. З прыкладу 20.9 бачна, што для рускай і англійскай моў арфаграфічны кантроль ажыццяўляецца, а для беларускай мовы — не.

1. Як ажыццяўляецца пошук у тэксце?
2. Як замяніць адно слова іншым?
3. Што разумеюць пад сістэмай праверкі правапісу?
4. Як Word вылучае памылкі ў тэксце?
5. Якія правілы неабходна выконваць пры ўводзе камп'ютарнага тэксту?

Практыкаванні

1 Адкрыйце тэкставы дакумент. Выпраўце памылкі, выкарыстоўваючы магчымасці арфаграфічнага кантролю. Растлумачце, чаму не падкрэсліваюцца чырвонымі лініямі некаторыя словы, напісаныя няправільна.

1. Но влева некакой дороги не было. Сонце висела теперь не слево от него а впереди и чут справо. Вправа на бугарке веднелся хутор.

2. У тэксці можна асставіць знакі перыносаў аўтаматычна або уручную. Пры ручным уводзе Word шукае слова, якія махчыма перынесці, і запытвае дазвол устаўкі перыносу. Пры афтаматычным уводзе перыносаў Word сам асстаўляе знакі там, дзе гэта трэба.

Правільны варыянт

1. Но влево никакой дороги не было. Солнце висело теперь не слева от него, а впереди и чуть справа. Справа на бугорке виднелся хутор[1].

2. У тэксце можна расставіць знакі пераносаў аўтаматычна або ўручную. Пры ручным уводзе Word шукае словы, якія магчыма перанесці, і запытвае дазвол устаўкі пераносу. Пры аўтаматычным уводзе пераносаў Word сам расстаўляе знакі там, дзе гэта трэба.

2 Адкрыйце тэкставы дакумент. Пры наборы тэксту на рускай мове выпадкова ўключылі беларускую раскладку клавіятуры. Выкарыстоўваючы функцыю **Замена**, выпраўце тэкст.

Інструкція по поіску сінонімов і антонімов

Для начала небольшая справка (с сайта Грамота.ру).

Сінонім — слово, отлічаюўееся от другого по звучанію ілі напісанію, но совпадаюўее ілі блізкое ему по значенію.

Антонім — слово с протівоположным по отношенію к другому слову значеніем.

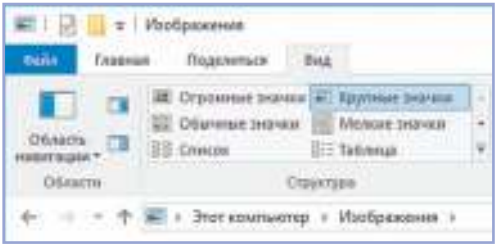
Тезаурус — словарь какого-лібо языка, представляюўій его лексіку в полном об'еме.

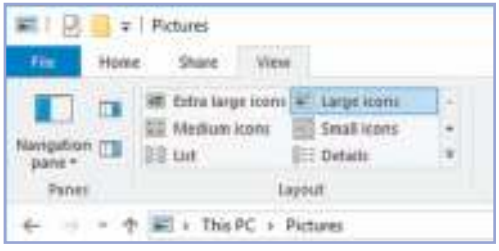
Выбор сінонімов і/ілі антонімов для определенного слова осуўествляется ўелчком правой кнопкі мыші на слове і наведеніем указателя мыші на команду **Сінонімы** (ілі камандай **Тезаурус**).

Прі выборе команды **Тезаурус** в редакторе Word открывается панель **Тезаурус**, в которой будут разліčные варіанты подходяўіх слов.

[1] Цытуецца паводле твора А. Гайдара «Пусть светит». Рэжым доступу: http://modernlib.net/books/gaydar_arkadiy/pust_svetit/read. Дата доступу: 10.02.2018.

3 Адкрыйце тэкставы дакумент з інструкцыяй па рабоце з праграмай **Explorer** (**Проводник**) для англамоўнай версіі Windows 10. Выпраўце інструкцыю так, каб яна дазволіла карыстальніку працаваць з рускай версіяй Windows.





Для гэтага з дапамогай каманды **Заменить** выканайце наступныя замены ва ўсім тэксце адразу: *Home — Главная, View — Вид, Icons — значки.* Замяніце: *Explorer — Проводник, New folder — Создать папку, Delete — Удалить, Move to — Переместить в, Copy to — Копировать в, Extra Large — Огромные, Large — Крупные, Medium — Обычные, Small — Мелкие.*

Кароткая інструкцыя па рабоце з праграмай Explorer

1. Стварыць папку: **Home → New Folder.**

2. Выдаліць: **Home → Delete.**

3. Перамясціць: **Home → Move to.**

4. Капіраваць файл: **Home → Copy to.**

5. Адлюстраванне файлаў у выглядзе велізарных значкоў: **View → Extra Large Icons.**

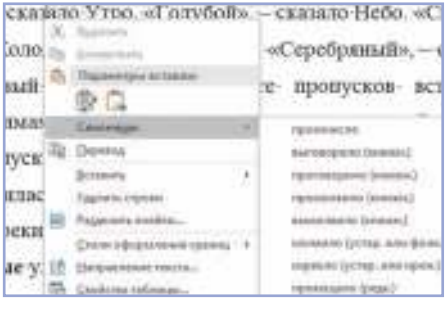
6. Адлюстраванне файлаў у выглядзе вялікіх значкоў: **View → Large Icons.**

7. Адлюстраванне файлаў у выглядзе звычайных значкоў: **View → → Medium Icons.**

8. Адлюстраванне файлаў у выглядзе дробных значкоў: **View → Small Icons.**

4* Загрузіце тэкставы дакумент. Замяніце слова *сказаць*, якое сустракаецца ў тэксце, яго сінонімамі.

Заспорили пуночки (северные воробьи), не могут решить, какой бывает снег. «Золотой», — сказало Утро. «Голубой», — сказало Небо. «Синий-синий», — сказали Тени. «Холодный», — сказала Утка. «Серебряный», — сказала Луна.



5* Загрузіце тэкставы дакумент. На месцы пропускаў устаўце словы, якія з'яўляюцца антонімамі да вылучаных. Указанне: спачатку скапіруйце вылучанае слова на месца пропуску, а затым замяніце яго антонімам.

Погода испортилась: **ясные** и **теплые** дни сменились ... и ... .
Правый берег реки был **пологий**, а левый ... .
Узкие и **кривые** улицы старой Москвы сменились ... и ... .
Из **тесных**, **темных** бараков рабочие переселились в ... и ... квартиры.
В конце долгого пути даже **легкий** труд становится ... .

§ 21. Спісы і калонкі

21. 1. Стварэнне і фармацiраванне спісаў

Пры стварэнні тэксту некаторыя абзацы даводзіцца нумараваць (прыклад 21.1) або вылучаць з дапамогай розных маркераў (прыклад 21.2).

Абзацы, адзначаныя маркерамі або нумарамі, утвараюць **спіс**.

Спіс абзацаў, адзначаных нумарамі, называюць **нумараваным**. Спіс абзацаў, адзначаных маркерамі, называюць **маркіраваным**.

Для стварэння спісаў можна выкарыстоўваць кнопкі на ўкладцы **Главная**:

— маркіраваны спіс. Элементы такога спіса пачынаюцца з малой літары і заканчваюцца коскай або кропкай з коскай;

— нумараваны спіс (з кропкай пасля нумара). Элементы спіса пачынаюцца з вялікай літары і заканчваюцца кропкай.

Гэтыя ж каманды прысутнічаюць у кантэкставым меню абзаца (прыклад 21.3).

Для афармлення гатовага тэксту ў выглядзе спіса трэба вылучыць тыя

Прыклад 21.1. Нумараваныя спісы.

Расклад урокаў на панядзелак:
1. Англійская мова.
2. Матэматыка.
3. Фізкультура.
4. Хімія.
5. Руская літаратура.
6. Гісторыя.

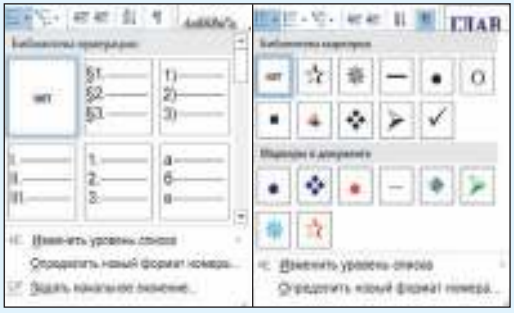
Якія тыпы эфектаў анімацыі рэалізаваны ў праграме PowerPoint? Выберыце правільныя адказы:
а) эфекты выхаду;
б) эфекты ўваходу;
в) эфекты раздзялення;
г) эфекты змяшчэння;
д) эфекты вылучэння;
е) эфекты змянення.

Прыклад 21.2. Маркіраваны спіс.

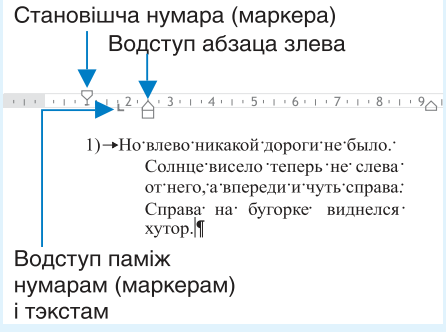
Сёння ў школьнай сталовай піражкі з начынкай з:
- капусты;
- бульбы;
- рыбы;
- тварагу;
- курагі.

Прыклад 21.3. Каманды для стварэння спісаў у кантэкставым меню.

Прыклад 21.4. Выбар фармату нумара і маркера для спіса.



Прыклад 21.5. Маркеры на лінейцы (уключаны рэжым адлюстравання недрукуемых сімвалаў).



Прыклад 21.6. Шматузроўневы спіс.

Поры года

1. ЗІМА:
   - снежань;
   - студзень;
   - люты.
2. ВЯСНА:
   - сакавік;
   - красавік;
   - май.
3. ЛЕТА:
   - чэрвень;
   - ліпень;
   - жнівень.
4. ВОСЕНЬ:
   - верасень;
   - кастрычнік;
   - лістапад.

Прыклад 21.7. Шматузроўневы спіс — частка зместу вучэбнага дапаможніка.

абзацы, якія будуць утвараць спіс, а затым выбраць від спіса. Больш дэталёва пра стварэнне спісаў і пра правілы іх афармлення можна прачытаць у *Дадатку 4* (с. 163—165).

Нумараваць абзацы можна арабскімі або рымскімі лічбамі, рускімі або лацінскімі літарамі. Розныя маркеры для спісаў можна выбраць з бібліятэкі ці выкарыстаць у якасці маркераў рысункі (прыклад 21.4). Нумар або маркер можна фармаціраваць, як і сімвалы: змяніць шрыфт, памер ці колер шрыфту, выкарыстаць напісанне.

Калі пры ўводзе тэксту першым сімвалам набраць лік, то пасля націску клавішы Enter гэты абзац будзе аўтаматычна афармлены ў выглядзе спіса. Наступны абзац атрымае нумар, на 1 большы. Кіраваць змяшчэннем спіса на старонцы можна з дапамогай маркераў на лінейцы (прыклад 21.5).

Тэкставы рэдактар Word дазваляе ствараць спісы складанай структуры.

Спісы, у якіх выкарыстоўваюцца адначасова розныя віды нумароў і/або маркераў, называюць **шматузроўневымі**.

У шматузроўневых спісах кожны ўзровень мае сваю нумарацыю (прыклад 21.6). Нумар новага ўзроўню можа складацца з нумара папярэдняга ўзроўню, які звычайна аддзяляецца кропкай. Прыкладам такога спіса з'яўляецца змест вучэбнага дапаможніка (прыклад 21.7).

21.2. Калонкі ў тэкставым дакументе

Word дазваляе змяшчаць тэкст на старонцы як у газетных калонках, у якіх тэкст бесперапынна перацякае з ніжняй часткі адной калонкі ў верхнюю частку наступнай (прыклад 21.8). З дапамогай калонак можна афармляць рэкламныя праспекты, буклеты, брашуры.

Для стварэння калонак набраны тэкст вылучаюць і выконваюць каманду **Колонки**, якая знаходзіцца на ўкладцы **Разметка страницы** (прыклад 21.9). Колькасць радкоў у створаных калонках будзе прыкладна аднолькавай (магчымае выключэнне — апошняя калонка).

Каманда **Другие колонки** адкрывае акно **Колонны**, дзе можна настройваць выгляд калонак (прыклад 21.10). Максімальная колькасць калонак залежыць ад шырыні ліста (для стандартнага ліста А4 — 12). Каб зрабіць калонкі рознай шырыні, трэба зняць птушку ў полі **Столбцы одинаковой ширины**. Поле **Разделитель** дазваляе ўставіць паміж калонкамі лінію падзелу.

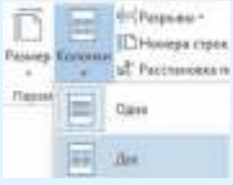
Калі тэкст пры наборы афармляюць у выглядзе калонак, то пад канец першай калонкі ўстаўляюць разрыў калонкі: каманда **Калонка** ў выпадаючым спісе **Разрывы** на ўкладцы **Разметка страницы** (прыклад 21.11). Камандзе адпавядае камбінацыя клавіш Ctrl + Shift + Enter. Дадзеная каманда ўстаўляе спецыяльны сімвал — **«Разрыв столбца»**, які паказвае канец бягучай калонкі. Ён адлюстроўваецца ў рэжыме паказу недрукуемых сімвалаў (прыклад 21.12).

Прыклад 21.8. Калонкі ў тэксце.

Акрамя курсаў, дысцыпліна стане абавязковай часткай праграмы ўстаноў адукацыі, што істотна павысіць IT-пісьменнасць насельніцтва.

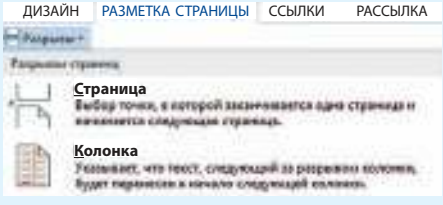
Галоўнай мэтай бліжэйшага будучага будзе навучыць пісьменнікаў, біёлагаў і бухгалтараў выкарыстоўваць камп'ютары для вырашэння сваіх задач. Напрыклад, урач, які выкарыстоўвае камп'ютарныя праграмы для больш дакладнай пастаноўкі дыягназу, будзе больш адпавядаць рэаліям будучага.

Прыклад 21.9. Стварэнне калонак.

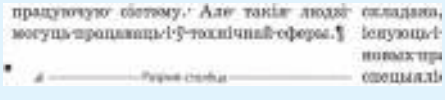


Прыклад 21.10. Акно з параметрамі кіравання калонкамі.

Прыклад 21.11. Устаўка разрыву калонкі.



Прыклад 21.12. Прымусовы канец калонкі.



1. Што такое спіс?
2. Якія існуюць віды спісаў?
3. Як стварыць спіс?
4. Як афармiць тэкст у выглядзе калонак?

Практыкаванні

1 Набярыце тэкст (або адкрыйце тэкставы дакумент). Афармiце маркіраваны спіс і выкарыстайце патрэбнае фармацiраванне.

Музыканты-раманісты чэрпалі натхненне ў народных песенных матывах і танцавальных рытмах. Часта звярталіся ў сваёй творчасці да літаратурных твораў Шэкспіра, Гётэ, Шылера. У XIX ст. шмат якія еўрапейскія краіны далі свету вялікіх кампазітараў:

- *Аўстрыя* і *Германія* — **Франц Шуберт** і **Рыхард Вагнер;**
- *Польшча* — **Фрэдэрык Шапэн;**
- *Венгрыя* — **Ферэнц Ліст;**
- *Італія* — **Джаакіна Расіні** і **Джузэпэ Вердзі;**
- *Чэхія* — **Бедржых Сметана;**
- *Нарвегія* — **Эдвард Грыг;**
- *Расія* — **Міхаіл Глінка, Аляксандр Барадзін, Пётр Чайкоўскі.**

2 Набярыце тэксты на адной з замежных моў (або адкрыйце тэкставы дакумент). Афармiце нумараваны спіс і выкарыстайце патрэбнае фармацiраванне ў дачыненні да тэксту.

Тэкст на англійскай мове:

Colour Idioms

1) To give a **black** look — гнеўна зірнуць;

2) once in a **blue** moon — вельмі рэдка, амаль ніколі;

3) to be (feel) **blue** — сумаваць, быць у дрэнным настроі;

4) to be like a **red** rag to a bull — дзейнічаць на каго-небудзь, як чырвоная ануча на быка;

5) a **white** elephant — дарагі, але бескарысны падарунак;

6) to be ***yellow*** — спужацца, быць баязліўцам.

Тэкст на французскай мове:

Des mots en couleurs

1) Mettre dans le **noir** — самы раз;
2) passer du **blanc** au noir — пераходзіць з адной крайнасці ў іншую;
3) couper le bois à **blanc** — вырубіць увесь лес;
4) avoir une peur **bleue** — жудасна спалохацца;
5) en dire de ***vertes*** — дазволіць сабе вольнасці ў размове;
6) faire *gris* mine à qu — сустрэць каго-небудзь з кіслай мінай.

Тэкст на іспанскай мове:

Colores en expresiones

1) suerte **negra** — горкая доля;
2) no distinguir lo blanco de lo **negro** — зусім не разбірацца ў чым-небудзь;
3) dejar en **blanco** — абвесці вакол пальца;
4) al **rojo** **blanco** — да шаленства;
5) ponerse **rojo** — пачырванець ад сораму;
6) el que quiera **azul** celeste, que le cueste — любіш катацца — любі і саначкі вазіць.

Тэкст на нямецкай мове:

Die geflügelte bunte Worte

1) die **schwarze** Kunst — тыпаграфская справа;
2) **blauen** Montag machen — прагуляць;
3) es wurde **grün** und **blau** vor den Augen — пацямнела ў вачах;
4) sich *gelb* und *grün* ärgern — не помнячы сябе;
5) **gelbe** Neid — чорная зайздрасць;
6) das wirkt auf him wie ein **rotes** Tuch — гэта дзейнічае на яго, як чырвоная ануча на быка.

3 Адкрыйце тэкставы дакумент «Якой можа быць праца IT-спецыяліста будучага». Афармеце тэкст у тры калонкі ў адпаведнасці з узорам. Кожная калонка павінна пачынацца з падзагалоўка (тэкст вылучаны паўтлустым напісаннем).

Якой можа быць праца IT-спецыяліста будучага[1]

Існуюць розныя прагнозы, але адно ясна дакладна — камп'ютарныя навукі нікуды не дзенуцца. Па якім бы сцэнарыі ні развіваўся свет, такія навыкі можна будзе выкарыстаць у многіх прафесіях і ў будучым.

Даступнае праграмiраванне

На папулярызацыю праграмiравання сярод дзяцей і дарослых накiравана мноства праектаў.

Галоўная мэта найбліжэйшага будучага — навучыць прадстаўнікоў розных прафесій. Напрыклад, урач, які выкарыстоўвае камп'ютарныя праграмы для дакладнай пастаноўкі дыягназу, або менеджар па продажы, які можа фільтраваць даныя кліентаў для больш эфектыўнай работы, будуць больш адпавядаць рэаліям будучага.

У кодзе толькі праграмісты

Колькасць праграмістаў ва ўсім свеце дасягнула 15 млн. Зараз актыўна развіваюцца дзіцячыя курсы праграмiравання, дзе вучаць пісаць код на мове Scratch. Таму колькасць спецыялістаў у далейшым будзе толькі расці.

Сёння з'яўляецца ўсё больш новых моў праграмiравання. Яны неабходны для папаўнення прабелаў ва ўжыванні ўжо існуючых моў. Таму патрэба ў праграмістах на сённяшні дзень даволі высокая.

Спецыяльнае праграмiраванне

Можна вылучыць сферы, у якіх у найбліжэйшым будучым спатрэбяцца IT-спецыялісты.

Ужо сёння развіваецца такі напрамак, як біржавы аналіз. Спецыяліст дадзенага профілю па сутнасці з'яўляецца праграмістам у фінансавай сферы.

Усё большую папулярнасць набывае інтэрнэт рэчаў. Яшчэ адзін перспектыўны напрамак — штучны інтэлект і робататэхніка. Таксама развіваюцца воблачныя тэхналогіі.

[1] Паводле матэрыялаў сайта https://nabr.com/company/1cloud/blog/316930/ (дата доступу 20.01.2018).

4* Набярыце тэкст (або адкрыйце дакумент). Стварыце копію набранага тэксту. Размяркуйце тэкст па двух калонках. Афармеце спісы паводле ўзору.

Сістэмнае ПЗ
Аперацыйныя сістэмы
Файлавыя менеджары
Сеткавыя праграмы
Драйверы
Утыліты
Прыкладное ПЗ
Праграмы агульнага прызначэння
Праграмы спецыяльнага прызначэння
Сістэмы навучання
Камп'ютарныя гульні
Інструментальнае ПЗ
Сістэмы праграмiравання
Транслятары
Наладчыкі
Наборы бібліятэк

Вынік:

I. Сістэмнае ПЗ:
- ➢ аперацыйныя сістэмы;
- ➢ файлавыя менеджары;
- ➢ сеткавыя праграмы;
- ➢ драйверы;
- ➢ утыліты.

II. Прыкладное ПЗ:
- ❋ праграмы агульнага прызначэння;
- ❋ праграмы спецыяльнага прызначэння;
- ❋ сістэмы навучання;
- ❋ камп'ютарныя гульні.

III. Інструментальнае ПЗ:
- ☆ сістэмы праграміравання;
- ☆ транслятары;
- ☆ наладчыкі;
- ☆ наборы бібліятэк.

1. Сістэмнае ПЗ:
1.1. Аперацыйныя сістэмы.
1.2. Файлавыя менеджары.
1.3. Сеткавыя праграмы.
1.4. Драйверы.
1.5. Утыліты.

2. Прыкладное ПЗ:
2.1. Праграмы агульнага прызначэння.
2.2. Праграмы спецыяльнага прызначэння.
2.3. Сістэмы навучання.
2.4. Камп'ютарныя гульні.

3. Інструментальнае ПЗ:
3.1. Сістэмы праграміравання.
3.2. Транслятары.
3.3. Наладчыкі.
3.4. Наборы бібліятэк.

§ 22. Табліцы

22.1. Стварэнне табліц

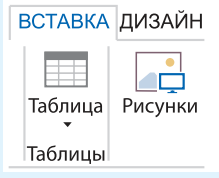
Сучасныя тэкставыя рэдактары дазваляюць устаўляць у тэкст дакумента розныя аб'екты: табліцы, рысункі, дыяграмы, формулы і інш. Большасць гэтых аб'ектаў можна дабавіць у тэкставы дакумент, выкарыстоўваючы каманды ўкладкі **Вставка**.

Табліцы дазваляюць структураваць тэкст, прэзентаваць яго больш наглядна.

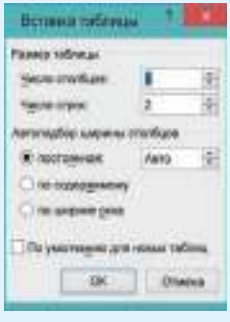
Уставіць табліцу ў дакумент можна з дапамогай каманды **Таблица** на ўкладцы **Вставка** (прыклад 22.1). Пасля націскання на кнопку трэба вылучыць мышшу вобласць, якая вызначае колькасць радкоў і слупкоў табліцы. Колькасць радкоў і слупкоў можна задаць лікамі, калі выбраць каманду **Вставить таблицу** (прыклад 22.2).

Шмат якія дзеянні, што можна ажыццяўляць з табліцай, вызначаны ў

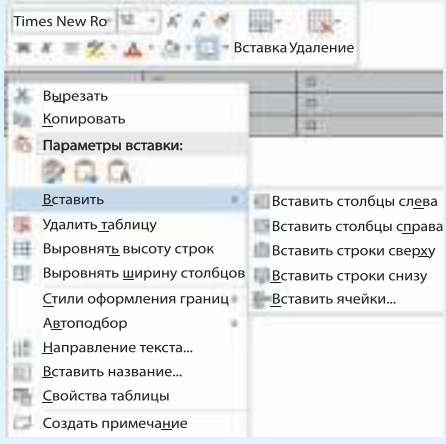
Прыклад 22.1. Устаўка табліцы.



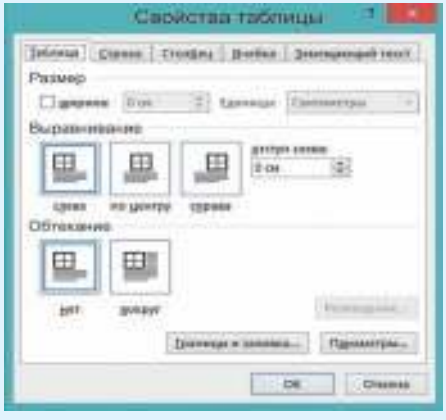
Прыклад 22.2. Вызначэнне памераў табліцы.



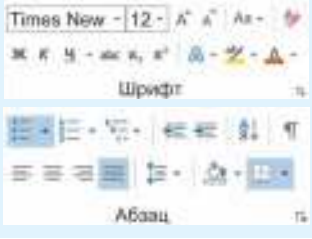
Прыклад 22.3. Кантэкставае меню табліцы.



Прыклад 22.4. Акно **Свойства таблицы**.



Прыклад 22.5. Каманды для змянення абзацных і сімвальных уласцівасцей тэксту ячэйкі табліцы (укладка **Главная**).



яе кантэкставым меню (прыклад 22.3). Кантэкставае меню можа змяшчаць і іншыя каманды, гэта залежыць ад таго, якія элементы табліцы вылучаны. У кантэкставым меню можна выбраць, прагледзець і змяніць уласцівасці табліцы (прыклад 22.4). Дадзенае акно дазваляе задаць размяшчэнне табліцы на старонцы дакумента: памер, выраўноўванне, водступ, абцяканне. Укладкі **Строка**, **Столбец** і **Ячейка** дазваляюць кіраваць памерамі адпаведных структурных элементаў табліцы.

22.2. Фармаціраванне табліц

Да фармаціравання табліцы належыць змяненне:

- абзацных і сімвальных уласцівасцей тэксту ўнутры ячэйкі;
- напрамку тэксту;
- шырыні слупка або вышыні радка;
- знешняга выгляду меж і фону ячэйкі;
- структуры табліцы: аб'яднанне ячэек, выдаленне і дабаўленне радкоў або слупкоў.

(Разгледзьце прыклады 22.5.—22.8.)

Некаторыя з каманд фармаціравання табліцы размешчаны на ўкладцы **Главная**. Дадаткова для фармаціравання табліцы выкарыстоўваюцца каманды, змешчаныя на ўкладках **Работа с таблицами** → **Конструктор** і **Работа с таблицами** → **Макет** (гл. *Дадатак 4*, с. 166).

У дачыненні да табліцы памерам 8 × 3 (прыклад 22.9, *а*) выкарысталі наступнае фармаціраванне:

1. У першым радку вылучылі ўсе ячэйкі і аб'ядналі іх. Атрымалі ячэй-

ку для ўводу загалоўка табліцы. У дачыненні да загалоўка выкарысталі наступнае фармаціраванне: шрыфт Arial, памер 14, паўтлусты. Выраўноўванне па цэнтры. Вакол ячэйкі, што змяшчае назву табліцы, знялі межы зверху, злева і справа.

2. Аб'ядналі ячэйкі ў трэцім і шостым радках. Зафарбавалі фон ячэек. У дачыненні да тэксту выкарысталі шрыфт Times New Roman, памер 12, курсіўнае паўтлустае напісанне.

3. Увялі тэкст у астатнія ячэйкі табліцы, шрыфт Times New Roman, памер 12.

4. Для ўсіх ячэек табліцы, што змяшчаюць лікі, выкарысталі выраўноўванне па цэнтры.

У прыкладзе 22.9, *б* у дачыненні да гэтай жа табліцы выкарысталі адзін з убудаваных стыляў табліц.

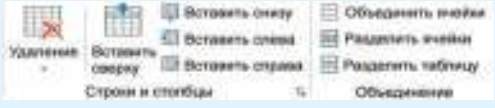
Часам пасля стварэння табліцы даводзіцца змяняць шырыню слупкоў або вышыню радкоў. Для змянення шырыні слупка (вышыні радка) трэба навесці паказальнік мышы на мяжу слупка (радка). Паказальнік мышы прыме выгляд: ↔ (↕). Далей, утрымліваючы націснутай левую клавішу мышы, трэба змяніць памер радка (або слупка).

Для выдалення слупка (радка) з табліцы неабходна яго вылучыць. Пасля гэтага выбраць каманду **Удаление** (прыклад 22.10). Для вылучаных слупкоў (радкоў) каманду **Удалять столбцы (строки)** можна выбраць з кантэкставага меню табліцы (прыклад 22.11).

Прыклад 22.6. Каманды для змянення напрамку тэксту, шырыні слупка або вышыні радка ў табліцы, выраўноўвання ў ячэйцы (укладка **Работа с таблицами → Макет**).

Прыклад 22.7. Каманды для змянення знешняга выгляду меж і фону ячэйкі (укладка **Работа с таблицами → → Конструктор**).

Прыклад 22.8. Каманды для змянення структуры табліцы (укладка **Работа с таблицами → Макет**).



Прыклад 22.9. Прыклады табліц.

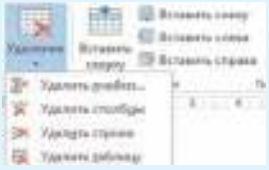
а

Шчыльнасць рэчываў		
Рычывы	кг/м³	г/см³
Рэчывы ў цвёрдым стане		
Серабро	10 500	10,5
Фарфор	2300	2,3
Рэчывы ў вадкім стане		
Ртуць	13 600	13,60
Вада	1000	1,00

б

Шчыльнасць рэчываў		
Рэчывы	кг/м³	**г/см³**
Рэчывы ў цвёрдым стане		
Серабро	10 500	**10,5**
Фарфор	2300	**2,3**
Рэчывы ў жыдкім стане		
Ртуць	13 600	**13,60**
Вада	1000	**1,00**

Прыклад 22.10. Каманды для выдалення (устаўкі) радкоў (слупкоў) табліцы. Укладка **Работа с таблицами → Макет**.



Прыклад 22.11. Каманды для выдалення радкоў (слупкоў) табліцы. Кантэкставае меню.

Прыклад 22.12. Каманды для ўстаўкі радкоў (слупкоў) з кантэкставага меню.

Калі проста націснуць клавішу Del, то выдаляцца толькі даныя з вылучаных ячэек табліцы.

Для ўстаўкі радкоў (слупкоў) можна выкарыстаць каманды для ўстаўкі (прыклад 22.10). Устаўляецца столькі радкоў (гл. слупкоў), колькі іх вылучана. Для ўстаўкі радкоў або слупкоў можна выкарыстаць і кантэкставае меню табліцы (прыклад 22.12). Калі націснуць клавішу Enter у канцы якога-небудзь радка табліцы (за мяжой апошняй ячэйкі ў радку), то пасля яго будзе дабаўлены новы радок табліцы.

Выкарыстоўваючы каманды ўкладкі **Работа с таблицами → Макет**, можна змяняць напрамак уводу тэксту, змяшчэнне тэксту адносна меж ячэйкі, задаваць палі ячэйкі. Пры неабходнасці табліцу можна пераўтварыць у тэкст.

1. Як уставіць табліцу ў дакумент?
2. Што разумеюць пад фармаціраваннем табліцы?
3. Як змяніць вышыню радка? Як змяніць шырыню слупка?
4. Як выдаліць радок (слупок) табліцы?
5. Як уставіць радок (слупок) у табліцу?

Практыкаванні

1 Стварыце адну з табліц. Тэксты вершаў можна загрузіць з файла. Выкарыстайце ў дачыненні да табліцы фармаціраванне па сваім выбары.

Басня І. Крылова[1]	Пераклад на **беларускую** мову Я. Купалы[2]
ЛЕБЕДЬ, ЩУКА И РАК Когда в товарищах согласья нет, На лад их дело не пойдет, И выйдет из него не дело, только мука.	**ЛЕБЕДЗЬ, ШЧУПАК І РАК** Там, дзе еднасці і згоды Няма шчырых у людзей, Праца іхняя заўсёды Ліха марна прападзе.

[1] Крылов, И. А. Басни; повести / [Сост. и предисл. П. Ткачева]. — Минск: БГУ, 1980.
[2] http://yankakupala.ru/lebedz-shchupak-i-rak (дата доступу 11.01.2018).

Басня И. Крылова	Пераклад на **беларускую** мову Я. Купалы
Однажды Лебедь, Рак да Щука Везти с поклажей воз взялись, И вместе трое все в него впряглись: Из кожи лезут вон, а возу все нет ходу! Поклажа бы для них казалась и легка: Да Лебедь рвется в облака, Рак пятится назад, Щука тянет в воду. Кто виноват из них, кто прав — судить не нам; Да воз и ныне там.	Раз ў калёсы запрагліся Лебедзь ды Шчупак і Рак; З усіх сіл вязці ўзяліся, Дый не зрушаць воз ніяк. Пад нябёсы Лебедзь рвецца, Шчупак цягне ў ваду; Распусціўшы клюшні, пхнецца Рак назад, як на бяду! ---------- Хто з іх вінен, хто не вінен, Не нам гэта раз'ясняць, Толькі ж і дасюль калёсы Як стаялі, так стаяць!

Верш Я. Купалы[1]	Пераклад верша на **французскую** мову M. Zakharkévitch[2]
А хто там ідзе? А хто там ідзе, а хто там ідзе У агромністай такой грамадзе? — Беларусы. А што яны нясуць на худых плячах, На руках у крыві, на нагах у лапцях? — Сваю крыўду. А куды ж нясуць гэту крыўду ўсю, А куды ж яны нясуць напаказ сваю? — На свет цэлы. А хто гэта іх, не адзін мільён, Крыўду несць наўчыў, разбудзіў іх сон? — Бяда, гора. А чаго ж, чаго захацелась ім, Пагарджаным век, ім, сляпым, глухім? — Людзьмі звацца.	**Qui vient par ici?** Qui vient par ici, qui vient par ici, Et qui est ce flot humain qui grossit? — Les Bélarussiens. Que portent-ils donc sur leur dos voûte? Que tiennent leurs bras tout ensanglantés? — L'injustice. Où la portent-ils, sur leur dos chargé, Qui doit la connaître et peut la juger? — Tout le monde. Mais tous ces millions, qui leur enseigna A voir l'injustice, et les réveilla? — La misère. Qui réclament-ils donc, aveugles, sourds, Asservis, esclaves depuis toujours? — Le nom d'hommes.

[1] Паводле матэрыялаў сайта http://yankakupala.ru/khto-tam-idze (дата доступу 11.01.2018).

[2] Вадюшина, Д. С. Французский язык: учеб. пособие для 8-го кл. — Минск: Вышэйшая школа, 2016. С. 163.

Johan Wolfgang Goethe[1] Des Wanderers NachtLied	
Über allen Gipfel ist Ruh, in allen Wipfeln spürest du kaum einen Hauch; die Vögelein schweigen im Walde. Warte nut, balde ruhest du auch.	
Пераклад[2] на **рускую** мову М. Ю. Лермантава	Пераклад[3] на англійскую мову H. W. Longfellow
Горные вершины Спят во тьме ночной, Тихие долины Полны свежей мглой. Не пылит дорога, Не дрожат листы. Подожди немного — Отдохнешь и ты.	O'er all the hill-tops Is quiet now, In all the tree-tops Hearst thou Hardly a breath; The birds are asleep in the trees: Wait; soon like these Thou too shaft rest.

2 Афармеце ў выглядзе табліцы расклад урокаў на тыдзень.

3 Стварыце табліцу змянення дзеяслова і афармеце яе па ўзоры.

1. Have ў англійскай мове

Person	Singular		Plural	
1-е лицо	I have		we	have
2-е лицо	you have		you	
3-е лицо	he	has	they	
	she			
	it			

2. Haben у нямецкай мове

Person	Singular		Plural
1-е лицо	ich habe		wir haben
2-е лицо	du hast		ihr habt
3-е лицо	er	hat	sie haben Sie haben
	sie		
	es		

[1] Паводле матэрыялаў сайта http://www.poetarium.info/goethe/wn.htm (дата доступу 11.01.2018).

[2] Лермонтов, М. Ю. Сочинения в 2 т. Т. 1 / сост. и ком. И. С. Чистовой. М.: Правда, 1988 (с.197).

[3] Паводле матэрыялаў сайта http://www.bartleby.com/356/524.html (дата доступу: 11.01.2018).

3. Avoir у французскай мове

Personne	Singulier		Pluriel
1-е лицо	j'ai		nous avons
2-е лицо	tu as		vous avez
3-е лицо	il	a	ils ont
	elle		
	on		

4. Tener у іспанскай мове

Persona	Singular		Plural	
1-е лицо	yo tengo		nosotros (as) tenemos	
2-е лицо	tú tienes		vosotros (as) tenéis	
3-е лицо	él	tiene	ellos	tienen
	ella		ellas	
	Usted (Vd.)		Ustedes (Vds)	

4 Стварыце табліцы, што змяшчаюць звесткі з розных прадметных абласцей. Афармеце табліцу ў адпаведнасці з узорам або выкарыстайце сваё афармленне.

1. Біялогія.

ПАРАЎНАЛЬНАЯ ХАРАКТАРЫСТЫКА КЛАСАЎ ЧЛЕНІСТАНОГІХ

	Ракападобныя	**Павукападобныя**	**Насякомыя**
Асяроддзе існавання	У асноўным воднае	Наземнае	У асноўным наземнае
Расчляненне цела	Часцей галавагрудзі і брушка	Галавагрудзі і брушка	Галава, грудзі і брушка
Колькасць канечнасцей	Розная	Чатыры пары	Тры пары
Колькасць вусікаў	Дзве пары	Няма	Адна пара
Крылы	Няма	Няма	Дзве пары, радзей — адна або няма
Вочы	Часцей складаныя	Простыя	Складаныя і простыя

2. Гісторыя.

	Еўропа			США
	Англія	Францыя	Германія	
Заработная плата ў 1850 г. за роўны рабочы час (у %)	100	64	75	240
Працягласць рабочага часу ў 1850 г. (у %)	100	117	111	96
Працягласць рабочага тыдня ў 1850 г. (у гадзінах)	84			72

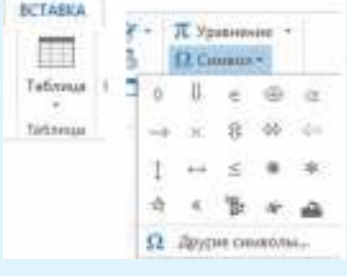
3*. Геаграфія.

КЛІМАТЫЧНЫЯ ПАЯСЫ

	Сярэднія тэмпературы (°С)		Ападкі		
	Зіма	Лета	Гадавыя, мм		Калі выпадаюць
			min	max	
Умераны	0	+16	500	1000	На працягу года
Субтрапічны	+8	+24	250	1000	Зімой
Экватарыяльны	+24	+24	500	2000	На працягу года

§ 23. Устаўка сімвалаў і формул

Прыклад 23.1. Выбар сімвала для ўстаўкі.



23.1. Устаўка і размяшчэнне сімвалаў у тэкставым дакуменце

Сучасныя камп'ютары выкарыстоўваюць табліцу сімвалаў Unicode, якая змяшчае 65 536 сімвалаў.

У Word падтрымліваецца магчымасць устаўкі сімвалаў: каманда **Символ** на ўкладцы **Вставка** (прыклад 23.1). Устаўку сімвалаў выконваюць тады, калі неабходна дабавіць у

тэкст сімвал, які адсутнічае на клавіятуры, але ёсць у табліцы сімвалаў.

Калі ў выпадаючым спісе адсутнічае патрэбны сімвал, то можна адкрыць акно **Символ**, выбраўшы каманду Ω Другие символы... (прыклад 23.2). Для пошуку сімвала ў табліцы можна выкарыстаць выпадаючыя спісы **Шрифт** і **Набор** (прысутнічае не для ўсіх шрыфтоў). У прыкладзе 23.3 апісана паслядоўнасць дзеянняў для ўстаўкі ў тэкст сімвала π.

У табліцы сімвалаў можна знайсці сімвалы еўрапейскіх і ўсходніх моў, матэматычныя і нотныя сімвалы, сімвалы грашовых адзінак і сімвалы-рысункі, якія можна ўставіць у тэкст (прыклад 23.4). У дачыненні да гэтых сімвалаў можна выкарыстоўваць фарматаванне.

23.2. Стварэнне і рэдагаванне формул

Матэматычныя (фізічныя, хімічныя і інш.) формулы могуць змяшчаць у сабе дастаткова складаныя элементы: дробы, знакі кораня, сістэмы ўраўненняў або няроўнасцей. Для стварэння такіх формул адных сімвалаў бывае недастаткова.

Для ўводу формул у Word выкарыстоўваюць каманду **Вставка** → → **Уравнение** → **Вставить новое уравнение** (прыклад 23.5). Пасля выканання каманды ў тэксце з'явіцца вобласць для ўводу формулы (ураўнення):

Дадаткова адкрыецца ўкладка **Работа с уравнениями** → **Конструктор**,

Прыклад 23.2. Выбар сімвала для ўстаўкі ў акне **Символ**.

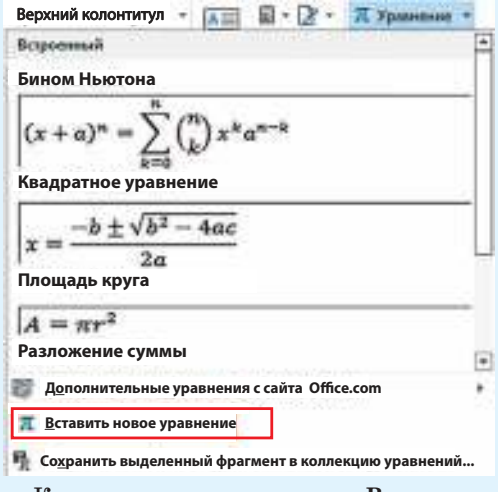
Прыклад 23.3. Устаўка сімвала π.

1. Выканаць каманду **Вставка** → → **Символ** → **Другие символы**.
2. У полі **Шрифт** выбраць **Symbol**.
3. Знайсці сімвал π і націснуць кнопку **Вставить**.

Лік π (пі) — матэматычная канстанта, роўная адносіне даўжыні акружнасці да яе дыяметра.

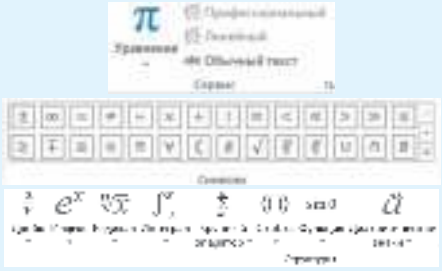
Прыклад 23.4. Розныя сімвалы з табліцы сімвалаў:

Прыклад 23.5. Устаўка ўраўнення.

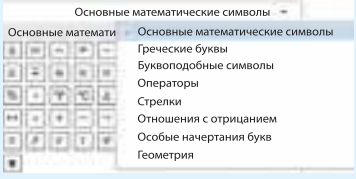


Каманда можа называцца **Вставка** → **Формула** → **Вставить новую формулу**.

Прыклад 23.6. Групы ўкладкі **Работа с уравнениями → Конструктор.**



Прыклад 23.7. Катэгорыі сімвалаў.



Прыклад 23.8. Запіс сістэмы няроўнасцей $\begin{cases} 2b-3\geqslant 13, \\ 3b^2-1<1. \end{cases}$

1. **Вставка → Уравнение → Вставить новое уравнение.**

2. У групе **Структуры** адкрыць шаблон Скобка, у спісе, які з'явіцца, выбраць **Отдельная скобка:** $\{\square$.

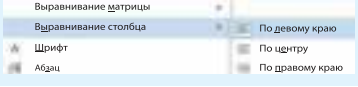
3. Адкрыць шаблон Матрица. Выбраць са спіса, які адкрыецца, **Пустая матрица 2 × 1:** $\begin{matrix}\square\\\square\end{matrix}$. Атрымаем $\left\{\begin{matrix}\square\\\square\end{matrix}\right.$.

4. Увесці першую няроўнасць. Знак $\geqslant$ можна знайсці ў групе **Символы.**

5. Увесці другую няроўнасць. Знак $<$ размешчаны на клавіятуры.

6. Для ўводу b^2 выкарыстоўваць шаблон $\square^{\square}$.

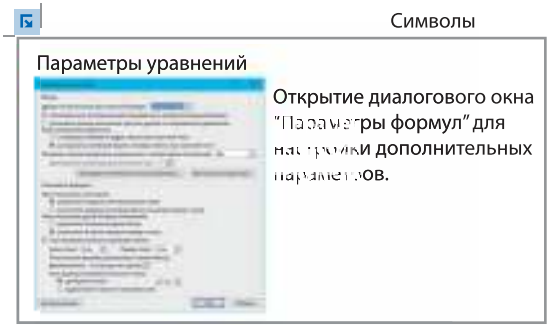
7. Выраўноўваць няроўнасці ў сістэме па левым краі, выкарыстоўваючы кантэкставае меню аб'екта **Уравнение.**



на якой пададзены каманды і шаблоны для ўводу розных элементаў формулы.

На ўкладцы вылучаны тры групы: **Сервис**, **Символы** і **Структуры** (прыклад 23.6).

Каманды групы **Сервис** дазваляюць вызначыць, як будзе выглядаць формула, або ўставіць гатовае ўраўненне з наяўных шаблонаў. Разгарнуўшы групу з дапамогай маленькай кнопкі са стрэлачкай у ніжнім правым вугле ↘, атрымаем доступ да агульных настроек параметраў ураўненняў.



Большасць сімвалаў, якія адсутнічаюць на клавіятуры, але выкарыстоўваюцца для ўводу формул, размяшчаюцца ў групе **Символы.** Сімвалы аб'яднаны па катэгорыях: асноўныя матэматычныя сімвалы, грэчаскія літары, стрэлкі і інш. (прыклад 23.7).

У групе **Структуры** знаходзяцца шаблоны для ўводу дробаў, індэксаў (верхніх і ніжніх), каранёў, дужак і інш.

У прыкладзе 23.8 паказана, як з дапамогай інструмента **Уравнение** запісаць сістэму лінейных няроўнасцей:

$$\begin{cases} 2b-3\geqslant 13, \\ 3b^2-1<1. \end{cases}$$

Для змянення формулы дастаткова клікнуць па ёй. Зноў стане даступная ўкладка **Работа с уравнениями** → → **Конструктор.**

Кантэкставае меню аб'екта ўраўненне змяшчае каманды, якія дазваляюць рэдагаваць і фармаціраваць ужо наяўную формулу.

Для ўстаўкі формул у тэкст дакумента, акрамя рэдактара Word, можна выкарыстаць іншыя тэкставыя рэдактары:

- LaTex[1] (мае ўласную мову вёрсткі формул);
- MathType[2] (уяўляе сабой невялікую праграму, якая запускаецца разам з Word на асобнай укладцы).

1. Як уставіць у тэкст сімвал, які адсутнічае на клавіятуры?
2. Як уставіць формулу ў тэкст дакумента?

Практыкаванні

1 Набярыце тэксты (або адкрыйце тэксты з файла). Выкарыстайце ўстаўку сімвалаў для тых сімвалаў, якія адсутнічаюць на клавіятуры. Сачыце за фармаціраваннем сімвалаў.

1. Калі $a > 0$, то няроўнасць $|x| \leqslant a$ раўназначная няроўнасці $-a \leqslant x \leqslant a$.
2. Няхай у $\triangle ABC$ і $\triangle A_1B_1C_1$ роўныя стораны AB і A_1B_1, BC і B_1C_1, але $AC > A_1C_1$. Дакажам, што $\triangle ABC > \triangle A_1B_1C_1$.
3. Даўжыня акружнасці вылічаецца па формуле $l = 2\pi r$, а плошча круга — па формуле $S = \pi r^2$.
4. У фізіцы папулярная шкала Кельвіна. У ёй 0 °C адпавядае 273 K, а 100 °C — гэта 373 K.
5. Пры пераходзе 1 кг рэчыва з цвёрдага стану ў вадкі паглынаецца колькасць цеплыні, лікава роўная ўдзельнай цеплыні плаўлення λ, і роўна столькі ж вылучаецца пры яго пераходзе з вадкага стану ў цвёрды.
6. Схема хімічнай рэакцыі: $Cu + O_2 \rightarrow CuO$.
7. Пры напісанні хімічных ураўненняў выкарыстоўваюць таксама знак ↓, калі рэчыва ўтварае асадак, або знак ↑, калі ў выніку рэакцыі ўтвараецца газ. Напрыклад:

$$AgNO_3 + HCl = AgCl\downarrow + HNO_3$$
$$C + O_2 = CO_2\uparrow$$

[1] Можна бясплатна спампаваць на афіцыйным сайце www.latex-project.org
[2] Спасылка для спампоўвання MathType: www.dessci.com. Маецца бясплатная 30-дзённая версія.

2 Набярыце тэксты з формуламі (або адкрыйце тэксты і ўстаўце формулы).

1. Пры рашэнні ўраўнення $\sqrt{2x+3} = x$ увядзём абедзве яго часткі ў квадрат. Атрымаем: $\left(\sqrt{2x+3}\right)^2 = x^2$, або $2x + 3 = x^2$.

2. Плошчу ромба можна вылічыць па формуле $S = \frac{d_1 d_2}{2}$, дзе d_1, d_2 — даўжыні дыяганалей ромба.

3. Вынікі доследаў дазваляюць запісаць формулу для разліку супраціўлення правадніка: $R = \rho\frac{l}{S}$. Каэфіцыент ρ называюць удзельным супраціўленнем рэчыва.

4. Паралельнае злучэнне дазваляе падключаць да крыніцы незалежна адна ад адной розныя прылады, нягледзячы на іх рабочы ток. Калі паралельных правадніковў толькі два, то: $\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2} = \frac{R_1 + R_2}{R_1 R_2}$, што прыводзіць да простага выразу: $R = \frac{R_1 R_2}{R_1 + R_2}$.

5. У прамысловасці вадарод атрымліваюць, прапускаючы вадзяную пару над распаленым вугалем: $C + H_2O \xrightarrow{t} CO + H_2$.

3 Стварыце аб'явы (адкрыйце файлы з тэкстам), выкарыстоўваючы сімвал шрыфтоў Webdings або Wingdings. Аб'явы зручна афармляць табліцай, робячы некаторыя межы ячэек нябачнымі.

§ 24. Графічныя аб’екты ў тэкставым дакументе

24.1. Устаўка рысункаў

Word дазваляе ўстаўляць у дакумент рысункі з розных крыніц:

- рысункі, якія захоўваюцца на дыску ў графічных файлах;
- відарысы з Інтэрнэту;
- вектарныя рысункі, створаныя з дапамогай фігур (графічных прымітываў);
- рысункі — графічныя копіі экрана.

Уставіць рысунак можна, карыстаючыся буферам абмену. Для гэтага ў іншай праграме вылучаем рысунак або яго фрагмент і выконваем каманду **Копировать**. Потым вяртаемся ў Word і выконваем каманду **Вставить**.

Укладка **Вставка** мае каманды для змяшчэння ў тэкставым дакументе розных відаў ілюстрацый (прыклад 24.1). Стварэнне вектарнага відарыса з аўтафігур выконваецца аналагічна стварэнню відарыса ў вектарным графічным рэдактары.

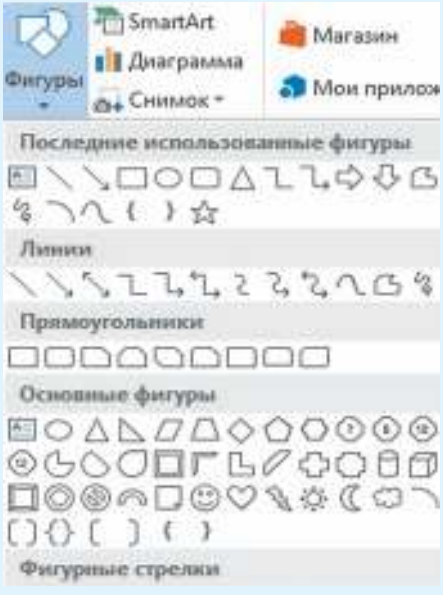
Для ўстаўкі рысунка з графічнага файла трэба выканаць каманду **Рисунки**. Затым неабходна паказаць імя файла. Рысунак будзе ўстаўлены ў пазіцыю курсора.

Пры выбары каманды **Изображения из Интернета** ў радку ўводзім запыт, які адлюстроўвае змест рысунка (прыклад 24.2).

Для ўстаўкі копіі экрана існуюць некалькі магчымасцей. Выкарыстоўваючы клавішу PrtScr[1] (копія ўсяго экрана) або камбінацыю клавіш

Прыклад 24.1. Каманды для ўстаўкі ілюстрацый на ўкладцы **Вставка**:

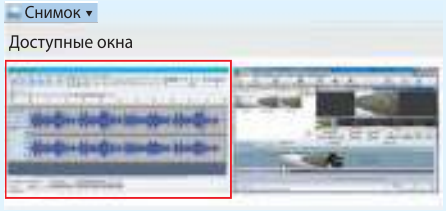
Калекцыя графічных прымітываў **Фигуры**:



Прыклад 24.2. Пошук рысунка па камандзе **Изображения из Интернета**.

[1] На некаторых клавіятурах гэтая клавіша падпісана Print Screen.

Прыклад 24.3. Выкарыстанне каманды **Снимок**.



Прыклад 24.4. Выкарыстанне каманды **Вырезка экрана**. У дакумент будзе ўстаўлена вылучаная частка экрана.

Прыклад 24.5. Устаўка аб'екта WordArt і выбар стылю (з выпадаючага спіса).

Прыклад 24.6. Устаўка аб'екта SmartArt і выбар стылю (з акна **Выбор графического элемента SmartArt**).

Alt + PrtScr (копія толькі актыўнага акна), змяшчаем відарыс у буфер абмену, а потым устаўляем яго ў дакумент. Каманда **Снимок** дазваляе ўставіць у дакумент копію любога з адкрытых акон. Апошняе з адкрытых акон адлюстроўваецца першым у спісе каманды **Снимок** (прыклад 24.3). З яго можна выразаць частку экрана (прыклад 24.4).

24.2. Устаўка аб'ектаў WordArt і SmartArt

З аб'ектамі WordArt і SmartArt вы пазнаёміліся ў 6-м класе, калі стваралі прэзентацыі. Работа з гэтымі аб'ектамі ў рэдактары Word адбываецца аналагічна.

Для ўстаўкі аб'ектаў у тэкст выбіраецца адпаведная каманда на ўкладцы **Вставка**. У прыкладзе 24.5 паказана, як уставіць аб'ект WordArt, а ў прыкладзе 24.6 — аб'ект SmartArt.

Пры вылучэнні аб'екта WordArt дабаўляецца ўкладка **Средства рисования** → **Формат**, на якой можна настроіць выгляд аб'екта. Для аб'екта WordArt можна змяніць наступныя параметры (прыклад 24.7):

- колер, таўшчыню і стыль лініі контуру вакол сімвалаў тэксту;
- колер або градыент для заліўкі;
- варыянты ценю;
- адлюстраванне і рэльеф сімвалаў;
- падсветку вакол сімвалаў;
- скрыўленне тэксту.

Пры вылучэнні аб'екта SmartArt (прыклад 24.8) дабаўляюцца дзве ўкладкі **Работа с рисунками SmartArt** → **Конструктор** і **Работа с рисунками SmartArt** → **Формат**. Каманды першай укладкі дазваляюць

змяняць структуру аб’екта, а другой — яго выгляд. Больш дэталёва пра настройкі аб’екта гл. *Дадатак 4* (с. 167).

24.3. Фармаціраванне аб’ектаў

Мы разгледзелі ўстаўку ў тэкставы дакумент розных аб’ектаў: WordArt, SmartArt, формул, рысункаў. Пасля ўстаўкі любога з гэтых аб’ектаў робіцца актыўнай укладка **Формат**. На ёй змяшчаюцца каманды, якія дазваляюць выбраць параметры фармаціравання адпаведнага аб’екта. Для розных аб’ектаў спіс гэтых каманд розны. Аднак ёсць каманды, што з’яўляюцца агульнымі для розных аб’ектаў. Гэтыя каманды аб’яднаны ў дзве групы: **Упорядочение** і **Размер** (прыклад 24.9).

Каманды групы **Упорядочение** дазваляюць кіраваць становішчам аб’екта і абцяканнем тэксту.

Рысунак, устаўлены ў тэкставы дакумент, можна абрэзаць з выкарыстаннем інструмента **Обрезка**.

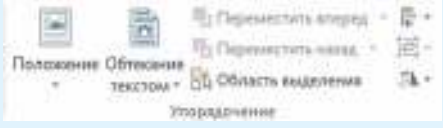
Пры разгортванні групы **Размер** (кнопка са стрэлачкай ↘) атрымаем асобнае акно **Макет**, якое дазваляе задаваць параметры фармаціравання аб’ектаў. Акно мае тры ўкладкі: **Положение**, **Обтекание текстом** і **Размер**.

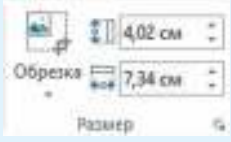
Укладка **Размер** (прыклад 24.10) дазволіць змяніць памер аб’екта: задаць патрэбны памер у сантыметрах або ў працэнтах адносна зыходнага памеру. Змяняць памер рысунка можна з дапамогай мышы.

Прыклад 24.7. Прыклад тэксту WordArt:

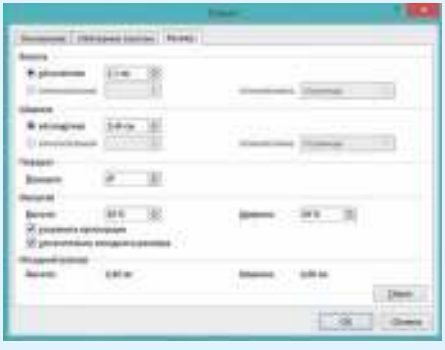
Прыклад 24.8. Прыклад рысунка SmartArt: арганізацыйная дыяграма.

Прыклад 24.9. Каманды груп **Упорядочение** і **Размер** на ўкладках **Форматирование**:





Прыклад 24.10. Укладка **Размер**:



Прыклад 24.11. Укладка **Обтекание текстом**:

Прыклад 24.12. Абцяканне рысунка **Сквозное**, тэкст — вакол:

А. С. Грин. Алые паруса[1]

— Не знаю, сколько пройдет лет, — только в Каперне расцветет одна сказка, памятная надолго. Ты будешь большой, Ассоль. Однажды утром в морской дали под солнцем сверкнет алый парус. Сияющая громада алых парусов белого корабля двинется, рассекая волны, прямо к тебе.

Прыклад 24.13. Абцяканне рысунка **Вокруг рамки**, тэкст — злева:

В. Быкаў. Альпійская балада[2]

Сям-там сінелі лапікі буйных духмяных незабудак, калыхаліся на ветры званочкі, ад густога водару жоўтай азаліі п'янела ў галаве. Мясцінамі ў кветкавых чашчобах трапляліся жарсцявыя плешыны, тырчалі з травы шэрыя камяні, каля іх заўжды было многа калючага шчэбню, ён шкодзіў ступням.

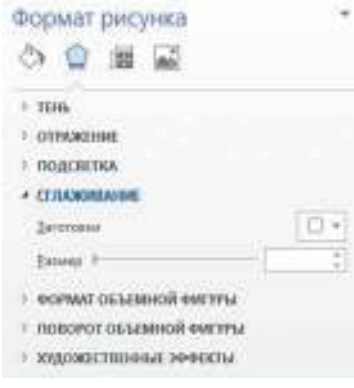
На ўкладцы **Обтекание текстом** (прыклад 24.11) можна ўстанавіць абцяканне аб'екта тэкстам (у тэксце, вакол рамкі, скразное і інш.) і выраўноўванне адносна тэксту. Становішча **в тексте** размяшчае аб'ект як сімвал тэксту.

Укладка **Положение** дазволіць вызначыць становішча аб'екта ў дакументе адносна старонкі, абзаца або калонкі.

Прыклады 24.12 і 24.13 дэманструюць адзін з варыянтаў абцякання тэкстам.

Абцяканне тэксту можна вызначыць з дапамогай значка , які з'яўляецца побач з аб'ектам пры яго вылучэнні.

Для кожнага аб'екта можна выбраць каманду **Формат** з кантэкставага меню, якая адкрывае дадатковую панэль. На гэтай панэлі сабраны ўсе параметры фармацiравання адпаведнага аб'екта. Панэль дублюе адпаведныя каманды ўкладкі **Формат**. Знешні выгляд панэлі можа адрозніваццa для розных аб'ектаў. Панэль **Формат** рысунка выглядае наступным чынам:



[1] Грин, А. С. Алые паруса. Минск: Наука и техника, 1979, 384 с.

[2] Быкаў, В. Збор твораў. У 4 т. Т. 1. Аповесці. Мінск: Мастацкая літаратура, 1980, 432 с.

1. Як можна ўставіць рысунак у тэкставы дакумент?
2. Якія спосабы ўстаўкі копіі экрана ў тэкставы дакумент вам вядомыя?
3. Якія параметры аб’екта WordArt можна змяняць?
4. Якія параметры аб’екта SmartArt можна змяняць?
5. Як змяніць памеры рысунка?
6. Якія спосабы абцякання тэкстам вы ведаеце?
7. Як адкрыць панэль **Формат рисунка**?

Практыкаванні

1 Выкарыстоўваючы пошук рысункаў (або загадзя падрыхтаваныя рысункі), стварыце віншаванні. Для афармлення надпісаў выкарыстайце аб’ект WordArt.

З Днём працы

З Днём Перамогі

2 Стварыце з дапамогай аб’ектаў WordArt і SmartArt (тып **Рисунок**) наступную схему. Відарысы атрымайце з экраннай копіі.

3 Праілюструйце тэкст, названы настаўнікам. Для ілюстрацый можна выкарыстаць рысункі, якія захоўваюцца на камп’ютары, відарысы з Інтэрнэту або самастойна падрыхтаваць рысунак у графічным рэдактары і ўставіць яго з файла або выкарыстоўваючы буфер абмену.

4 Адкрыйце файл з тэкстам. Праілюструйце тэкст, выкарыстоўваючы копіі экрана або гатовыя рысункі.

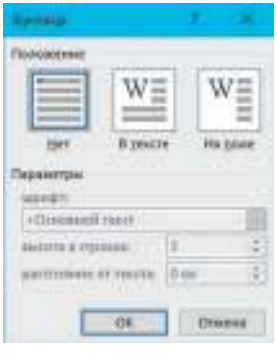
Як уставіць буквіцу ў дакуменце Word[1]

Хочаце дабавіць іскрынку ў дакументы Word? Буквіца і ёсць такая іскрынка, якая дазваляе адлюстроўваць першую літару абзаца вялікім шрыфтам.

1. Змясціце курсор у пачатак абзаца, у які хочаце ўставіць буквіцу.
2. Адкрыйце ўкладку **Вставка** і ў групе **Текст** націсніце **Буквица**.

3. Выберыце тып буквіцы: **В тексте** або **На поле**.

4. У раздзеле **Параметры буквицы** выберыце шрыфт буквіцы.



5. Задайце вышыню ў радках і адлегласць ад тэксту ў адпаведных палях у акне **Буквица**.
6. Націсніце **ОК**, каб уставіць буквіцу.

Біткоін (англ. *Bitcoin*, ад *bit* — «біт» і *coin* — «манета») — плацежная сістэма, якая выкарыстоўвае аднайменную адзінку для ўліку аперацый і аднайменны пратакол перадачы даных.

[1] Паводле матэрыялаў сайта https://hi-news.ru/gadgets (дата доступу 16.01.2018).

Гаджэты[1]

Гаджэты — тэхнічныя прыстасаванні для абсалютна розных мэт — з'яўляюцца лепшымі сябрамі чалавека. Па статыстыцы на кожнага, хто жыве на Зямлі, прыпадае мінімум тры тэхнічныя ўстройствы, г. зн. карысныя гаджэты. З'яўляючыся цалкам рознымі і прызначанымі для розных мэт (гадзіннікі і тэлефоны, док-станцыі і мікрафоны, праектары і мініяцюрныя робаты), гаджэты істотна палягчаюць жыццё чалавека.

Квадракоптары навучылі лятаць чародамі без GPS

У Інтэрнэце шмат ролікаў, дзе чароды дронаў прыгожа і зладжана ляцяць, адначасова выконваючы розныя трукі. Ну і што? Яны ж абапіраюцца на даныя, атрыманыя ад GPS. Але што рабіць, калі ніякага GPS выкарыстоўваць не атрымліваецца? Інжынеры ўніверсітэта Пенсільваніі вырашылі навучыць дроны абыходзіцца без іх.

Бесправадное зараднае ўстройства WattUp працуе на адлегласці да 4 м

Інжынеры ўжо даўно спрабуюць стварыць зараднае ўстройства, якое дазволіць людзям раз і назаўсёды пазбавіцца ад правадоў і пастаяннай неабходнасці знаходзіцца побач з разеткай. Улічваючы, колькі ў нашым жыцці ўсёмагчымых гаджэтаў і электронных устройстваў, нам даводзіцца рэгулярна падзаражаць іх, каб заставацца на сувязі, у курсе апошніх навін і г. д. Новае зараднае ўстройства WattUp, распрацаванае ў межах стартапу Energous, дазваляе падзаражаць па паветры адразу некалькі гаджэтаў на адлегласці больш за 4 м.

Паказаны самы кампактны 3D-прынтар памерам з рукзак

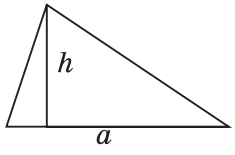
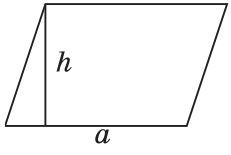
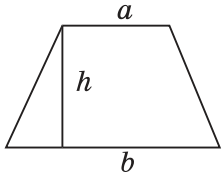
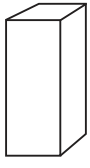
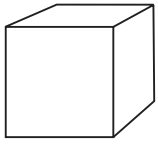
Нягледзячы на тое што 3D-прынтары ўсё больш актыўна ўваходзяць у наша жыццё, яны ўсё ж такі застаюцца дастаткова грувасткімі ўстройствамі. Аднак кітайскія інжынеры з карпарацыі MakeX паказалі самы кампактны на дадзены момант 3D-прынтар. Цікава, што гэты прынтар убудаваны ў звычайны рукзак.

[1] Паводле матэрыялаў сайта https://hi-news.ru/gadgets (дата доступу: 23.01.2018).

5 Устаўце табліцу з трох радкоў і двух слупкоў. У адной ячэйцы табліцы стварыце сваю візітную картку. Запішыце прозвішча, імя, школу і клас. Устаўце сваю фатаграфію. Скапіруйце візітоўку ва ўсе ячэйкі табліцы.

6 Устаўце табліцу з пяці радкоў і двух слупкоў. У адной ячэйцы табліцы стварыце бэйдж дзяжурнага. Скапіруйце бэйдж ва ўсе ячэйкі табліцы.

7 Стварыце наступную табліцу. Формулы ствараюцца з дапамогай устаўкі ўраўненняў, рысункі — з дапамогай устаўкі фігур.

Формулы для вылічэння плошчы			
Трохвугольнік	$S = \frac{ah}{2}$	a — аснова трохвугольніка h — вышыня трохвугольніка	
Паралелаграм	$S = ah$	a — аснова паралелаграма h — вышыня паралелаграма	
Трапецыя	$S = \frac{a+b}{2}h$	a, b — асновы трапецыі h — вышыня трапецыі	
Формулы для вылічэння аб’ёму			
Прамавугольны паралелепіпед	$V = abc$	a, b, c — даўжыня, шырыня, вышыня паралелепіпеда	
Куб	$V = a^3$	a — даўжыня канта куба	

8* Афармеце з дапамогай тэкставага рэдактара Word рашэнне задачы па геаметрыі, фізіцы або хіміі. Выкарыстайце табліцы, формулы, рысункі.

§ 25. Выкарыстанне стыляў

25.1. Паняцце стылю

Пры рабоце з вялікімі тэкставымі дакументамі заданне ўласцівасцей сімвалаў і абзацаў з'яўляецца даволі працаёмістым працэсам. Выкарыстанне стыляў дазваляе значна паскорыць фармаціраванне тэксту.

Пад **стылем** разумеюць набор параметраў фармацiравання тэксту. Стыль мае сваё імя.

Акрамя чыста афармляльніцкай задачы, стылі дазваляюць вырашыць таксама задачу структуравання тэксту. Для гэтага кожны са стыляў супастаўляюць з раздзелам, назвай загалоўка, тэкстам асноўнай часткі дакумента і інш. Вылучаюць наступныя віды стыляў:

- стыль абзаца;
- стыль сімвала;
- стыль спіса.

Фармаціраванне з выкарыстаннем стыляў (стылявое фармаціраванне) мае шэраг пераваг перад ручным:

1. Дазваляе эканоміць час. Выкарыстаць стыль як набор параметраў фармацiравання значна хутчэй, чым паслядоўна ўжываць асобныя параметры.

2. Спрыяе аднастайнаму афармленню тэкставага дакумента.

3. Дазваляе хутка змяніць выгляд асобных элементаў ва ўсім дакуменце. У гэтым выпадку дастаткова зрабіць змяненні ў стылі, і афармленне аўтаматычна выкарыстоўваецца ў дачыненні да ўсяго дакумента.

Тэкставы працэсар Word дазваляе карыстальніку ствараць свае стылі або выкарыстоўваць ужо створаныя. Для работы са стылямі на ўкладцы **Главная** ёсць група **Стили** (прыклад 25.1).

Звычайна новыя дакументы ствараюцца на аснове шаблону, які змяшчае пэўныя стылі.

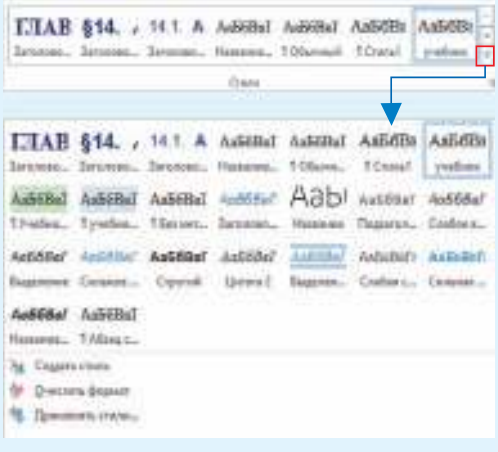
Стылі тэксту (абзаца або сімвала). Як правіла (або часта), выкарыстоўваюць стылі **Обычный** і **Основной** текст, а таксама іх мадыфікацыі. На іх аснове вызначаюцца стылі абзацаў і сімвалаў.

Стылі загалоўкаў (асаблівы стыль абзаца). Выкарыстанне загаловачных стыляў дазваляе аўтаматызаваць стварэнне зместу. Загаловак разглядаецца як абзац.

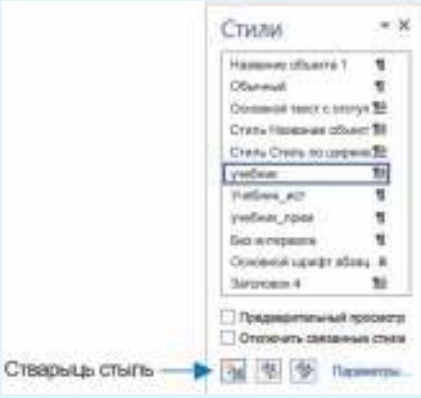
Стылі спісаў. Дазваляюць афармляць маркіраваныя і нумараваныя спісы.

Кожны тып стылю вызначае толькі ўласцівыя яму параметры. Адпаведна стылі розных тыпаў могуць накладацца адзін на адзін. Напрыклад, стылі спісаў кіруюць выглядам маркераў, структурай спіса і велічынёй водступаў пунктаў, але не памерам шрыфту. Такім чынам, тэкст у спісе кіруецца (як мінімум) двума стылямі — стылем абзаца і стылем спіса.

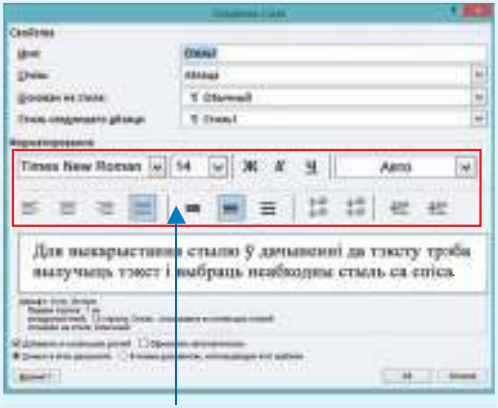
Прыклад 25.1. Група **Стили** ўкладкі **Главная**:



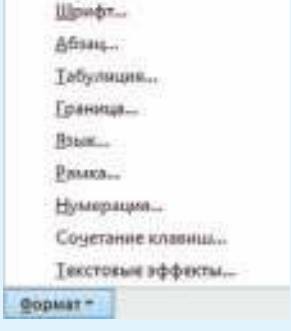
Прыклад 25.2. Панэль **Стили**.



Прыклад 25.3. Акно **Создание стиля**.



Прыклад 25.4. Меню кнопкі **Формат**.



Спіс наяўных стыляў адлюстроўваецца таксама на панэлі **Стили** (прыклад 25.2), якая разгортваецца з дапамогай кнопкі са стрэлачкай ↘. Для выкарыстання стылю ў дачыненні да тэксту трэба вылучыць тэкст і выбраць патрэбны стыль са спіса.

Для стварэння стылю выкарыстоўваюць кнопку **Создать стиль** у акне **Стили** (або каманду **Создать стиль** у разгорнутым спісе групы **Стили**). У акне **Создание стиля**, якое адкрыецца (прыклад 25.3), уводзяць неабходныя параметры стылю.

У полі **Имя** задаецца назва стылю (па змоўчанні **Стиль с номером**).

У полі **Стиль** выбіраецца абзац, знак або табліца ў залежнасці ад таго, для якога аб'екта вызначаем стыль.

У полі **Основан на стиле** выбіраюць стыль, найбольш блізкі па афармленні і прызначэнні да створанага. Так, напрыклад, стылі асноўнага тэксту дакумента засноўваюць на стылі **Обычный**, а стылі для афармлення назваў глаў, параграфаў і г. д. засноўваюць на стылях **Заголовок**.

У якасці **Стиля следующего абзаца** можна выбраць любы з ужо наяўных стыляў. Як правіла, для стыляў, заснаваных на стылі **Обычный**, выбіраюць імя ствараемага стылю. Тады ўсе абзацы дакумента будуць афармлены адным стылем (параметры стылю для новага абзаца пераносяцца з папярэдняга). Для стыляў, заснаваных на стылі **Заголовок**, у якасці стылю наступнага абзаца вызначаюць стыль загалоўка іншага ўзроўню або стыль тэксту, заснаваны на стылі **Обычный**.

Параметры фармаціравання сімвалаў і абзацаў устанаўліваюцца з дапамогай адпаведных кнопак. Можна таксама выкарыстаць кнопкай **Формат** (прыклад 25.4).

Патрабаванні, якія прад'яўляюцца да афармлення асноўнага тэксту рукапісу вучэбнага дапаможніка, уключаюць наступныя параметры тэксту: Шрыфт Times New Roman, памер 14. Выраўноўванне абзаца па шырыні, абзацны водступ — 1 см, міжрадковы інтэрвал — 1,5 радка. Стварэнне стылю з імем **«Учебник»**, які адпавядае названым патрабаванням, апісана ў прыкладзе 25.5.

Створаны стыль можна выдаліць, змяніць або абнавіць на аснове вылучанага фрагмента. Для гэтага выкарыстоўваюць каманды кантэкставага меню для выбранага стылю (прыклад 25.6).

25.2. Стылёвае афармленне загалоўкаў

Кожны, нават самы просты, дакумент складаецца з розных раздзелаў. Пад раздзелам разумеюць частку тэксту, якая нясе пэўны функцыянальны сэнс. Разуменне структуры дакумента дае магчымасць пісьме́нна яго афармліць і без цяжкасці перафармаціраваць у выпадку неабходнасці.

Напрыклад, вучэбны дапаможнік складаецца з глаў, глава змяшчае параграфы, у параграфах могуць быць пункты і падпункты (прыклад 25.7).

Паколькі адны раздзелы з'яўляюцца часткамі іншых (глава складаецца з параграфаў, параграфы складаюцца з пунктаў), то раздзелы адрозніваюць

Прыклад 25.5. Стварэнне стылю **«Учебник»**.

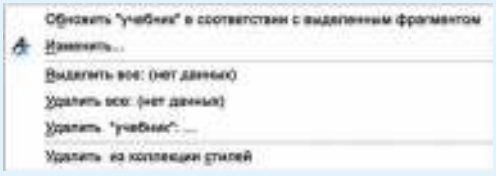
1. У акне **Стили** націснуць кнопку **Создать стиль** і вызначыць наступныя параметры стылю: **Имя** — **Учебник**, **Стиль** абзаца, **Основан на стиле** — **Обычный**, **Стиль следующего абзаца** — **Учебник**.

2. Вызначыць шрыфт (Times New Roman), памер шрыфту (14), выраўноўванне (па шырыні) і міжрадковы інтэрвал (1,5 радка).

3. Для вызначэння водступу першага радка націснуць кнопку **Формат** і выбраць **Абзац...** У акне **Абзац**, якое адкрыецца, задаць памер водступу першага радка (1 см).

4. Скончыць стварэнне стылю, націснуўшы кнопку OK. Прагледзець спіс стыляў, знайсці створаны стыль. Стыль можна выкарыстоўваць для афармлення дакументаў.

Прыклад 25.6. Кантэкставае меню стылю **«Учебник»**.



Інструмент **Формат по образцу** (🖌) дазваляе хутка скапіраваць стыль з аднаго аб'екта тэкставага дакумента на іншы.

Інструмент **Удалить все форматирование** (A🧹) выкарыстоўвае ў дачыненні да вылучанага тэксту стыль **Обычный**.

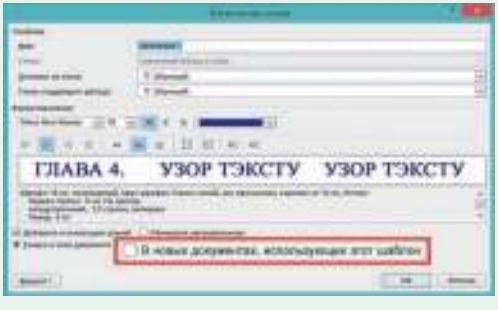
Прыклад 25.7. Структурныя элементы вучэбнага дапаможніка «Інфарматыка» для 7-га класа.

Глава 1. Інфармацыя і інфармацыйныя працэсы
§ 1. Інфармацыя ў жыцці чалавека 8
1.1. Віды інфармацыі ..... —
1.2. Носьбіты інфармацыі 10
1.3. Інфармацыйныя працэсы 11
§ 2. Уяўленне інфармацыі ў камп'ютары 14
2.1. Кадзіраванне інфармацыі ..... —
2.2. Адзінкі вымярэння аб'ёму інфармацыі ... 16

Прыклад 25.8. Параметры стылю **Заголовок 1** (структуру стылю можна прагледзець пры навядзенні паказальніка мышы на яго назву ў панэлі **Стили**).

Прыклад 25.9. Афармленне **Заголовков** (змяненне параметраў стылю).

Калі створаны ў дакуменце стыль неабходна выкарыстоўваць у іншых дакументах, то яго захоўваюць у шаблоне normal.dotx. Для гэтага выбіраюць пункт **В новых документах, использующих этот шаблон** у акне **Создание** або **Изменение стиля**. Пры закрыцці дакумента будзе прапанавана захаваць шаблон.



па ўзроўнях. Раздзел, які ўваходзіць у склад іншага, мае больш нізкі ўзровень.

Вылучэнне структурных элементаў тэксту забяспечвае структураванне дакумента, што палягчае яго ўспрыманне. Назвы раздзелаў афармляюць загалоўкамі адпаведнага ўзроўню: загаловак 1-га ўзроўню, загаловак 2-га ўзроўню і г. д. Аднолькавыя структурныя элементы (напрыклад, назва глаў, параграфаў, пунктаў) павінны афармляцца аднолькава — адным стылем.

Для загалоўкаў выкарыстоўваюць стылі **Заголовок 1**, **Заголовок 2** (або стылі, заснаваныя на стылях **Заголовок**) і г. д. Нумар у назве стылю загалоўка адпавядае яго ўзроўню. Напрыклад, для афармлення загалоўкаў глаў у вучэбным дапаможніку выкарыстоўваюць стыль **Заголовок 1**, для загалоўкаў параграфаў — **Заголовок 2** (прыклад 25.8) і г. д.

На аснове наяўных стыляў загалоўкаў ствараюць свае стылі (прыклад 25.9). Для афармлення загалоўкаў, як правіла, выкарыстоўваюць шрыфт большага памеру, чым асноўны тэкст дакумента, і паўтлустага напісання (гэта дазваляе скараціць час пошуку загалоўкаў у тэксце). Выраўноўваюць загалоўкі звычайна па цэнтры (магчыма таксама па левым або правым краях). Загалоўкі больш высокага ўзроўню афармляюцца больш важка, чым загалоўкі менш высокага ўзроўню (буйнейшы памер, больш тлустае напісанне і г. д.).

25.3. Генерацыя зместу

Змест з'яўляецца абавязковым элементам дакумента, у якім больш за 10 старонак. Ён спрашчае працу з дакументам.

Змест — паказальнік загалоўкаў у дакуменце, які адлюстроўвае яго структуру і паскарае пошук частак твора па нумарах старонак.

Перад тым як ствараць змест, усе загалоўкі глаў, параграфаў павінны быць афармлены адпаведнымі стылямі. У змест змяшчаюцца абзацы, афармленыя стылем **Заголовок**, з указаннем нумара старонак, з якіх яны былі ўзяты.

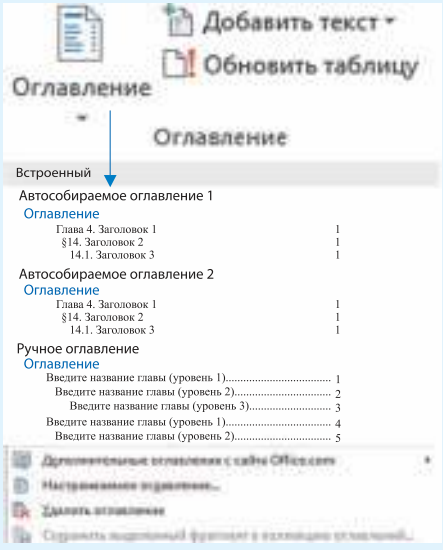
Змест ствараецца з дапамогай каманд групы **Оглавление** ўкладкі **Ссылки** (прыклад 25.10). У выпадаючым спісе каманды **Оглавление** можна выбраць выгляд зместу.

Автособираемое оглавление можна выбіраць, калі загалоўкі структурных элементаў былі афармлены стылямі **Заголовок 1**, **Заголовок 2** і **Загаловок 3**. Калі ўжываліся іншыя стылі для загалоўкаў (**Заголовок 4—9**) або трэба змяніць выгляд зместу, выкарыстоўваюць каманду **Настраиваемое оглавление**, якая адкрывае акно **Оглавление** (прыклад 25.11).

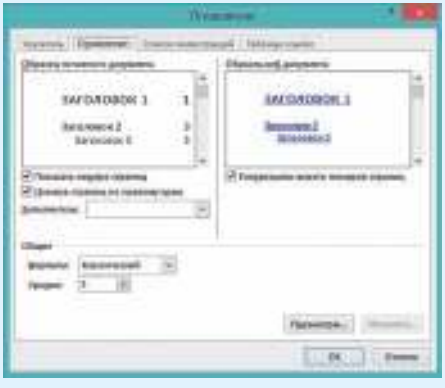
Знешні выгляд зместу можна выбраць са спіса **Форматы** (прыклад 25.12). З дапамогай кнопкі **Параметры** (прыклад 25.13 на с. 146) задаюць тыя стылі, на аснове якіх будзе пабудаваны змест. Колькасць узроўняў загалоўкаў, якія будуць уключаны ў змест, выбіраюць у выпадаючым спісе **Уровни**.

Па змоўчанні змест устаўляецца з нумарамі старонак, размешчанымі па правым краі. Птушкі ў адпаведных палях дазваляюць змяшчаць нумар

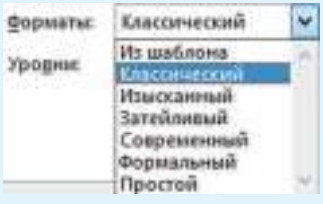
Прыклад 25.10. Група **Оглавление** ўкладкі **Ссылки**.



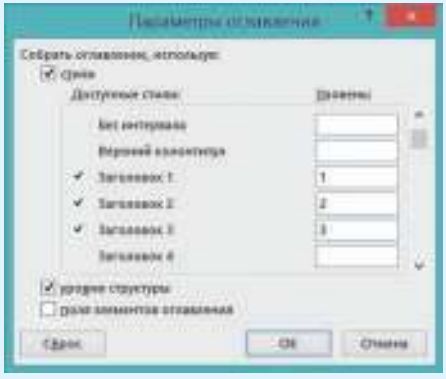
Прыклад 25.11. Акно **Оглавление**.



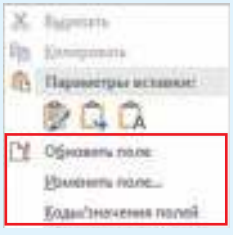
Прыклад 25.12. Фарматы зместу.



Прыклад 25.13. Параметры зместу.



Прыклад 25.14. Кантэкставае меню зместу.



старонкі побач з тэкстам або зусім знімаць нумары старонак.

У якасці запаўняльніка паміж загалоўкам і нумарам старонкі звычайна выкарыстоўваюць кропкі, але ў выпадаючым спісе **Заполнитель** можна выбраць працяжнік або знак падкрэслівання. Па змоўчанні запаўняльнік адсутнічае.

Змест звычайна ўстаўляюць на асобную старонку ў пачатку або ў канцы дакумента. Для ўстаўкі новай старонкі можна выкарыстаць каманду **Вставка → Пустая страница (Разрыв страницы)** або камбінацыю клавіш Ctrl + Enter.

Калі ў дакуменце зрабілі змяненні, то змест трэба абнавіць. Кіруюць змяненнямі з кантэкставага меню зместу (прыклад 25.14) або з дапамогай каманды **Обновить таблицу** групы **Оглавление** на ўкладцы **Ссылки**.

1. Што разумеюць пад стылем?
2. Для чаго выкарыстоўваюць стылі?
3. Як стварыць і змяніць стыль?
4. У чым асаблівасць выкарыстання загаловачных стыляў?
5. Як згенераваць змест дакумента?

Практыкаванні

1 Адкрыйце дакумент. Выканайце наступныя заданні.

1. Ачысціце тэкст ад выкарыстаных раней стыляў.
2. Стварыце стыль **Доклад** (да назвы **Доклад** прыпісаць сваё прозвішча) з наступнымі параметрамі: заснаваны на стылі **Обычный**, стыль абзаца; шрыфт Times New Roman, памер шрыфту — 14; міжрадковы інтэрвал — паўтарачны, выраўноўванне абзаца — па шырыні, водступ першага радка — 1 см; стыль наступнага абзаца — **Доклад**.
3. Выкарыстайце створаны стыль у дачыненні да асноўнага тэксту дакумента.
4. Афармеце ў выглядзе нумаванага спіса агульны сцэнарый паводзін карыстальнікаў у сацыяльных сетках (пункт **Як працуе сацыяльная сетка**) і спіс крыніц.

5. Афарміце ў выглядзе маркіраванага спіса паслядоўнасць падвідаў сацыяльных сетак.
6. Адфармацыруйце табліцу.
7. Захавайце дакумент пад новым імем.

2 Працягніце работу з дакументам з практыкавання 1.

1. Стварыце стыль загалоўкаў першага ўзроўню. Стыль **Заголовок 1 Доклад** з параметрамі:
- заснаваны на стылі **Заголовок 1**, стыль абзаца;
- шрыфт — Comic Sans MS, памер шрыфту — 22, напісанне — паўтлустае, колер — цёмна-сіні;
- міжрадковы інтэрвал — паўтарачны, выраўноўванне абзаца — па цэнтры, водступ першага радка — няма, інтэрвал перад абзацам — 18, пасля — 12;
- стыль наступнага абзаца — **Доклад**.

2. Выкарыстайце стыль для загалоўкаў: «Уводзіны», «Сацыяльныя сеткі» і «Заключэнне».
3. Стварыце стыль загалоўкаў другога ўзроўню **Заголовок 2 Доклад** з параметрамі:
- заснаваны на стылі **Заголовок 2**, стыль абзаца;
- шрыфт — Arial, памер шрыфту — 18, напісанне — паўтлустае, курсіўнае, колер — сіні;
- міжрадковы інтэрвал — паўтарачны, выраўноўванне абзаца — па цэнтры, водступ першага радка — няма, інтэрвал перад абзацам — 12, пасля — 6;
- стыль наступнага абзаца — **Доклад**.

4. Выкарыстайце стыль для загалоўкаў: «Што такое сацыяльныя сеткі?», «Як працуе сацыяльная сетка?», «Падвіды сацыяльных сетак», «Небяспекі сацыяльных сетак», «Найбуйнейшыя сацыяльныя сеткі», «Спіс крыніц».
5. Устаўце нумарацыю старонак.
6. Згенерыруйце змест.
7. Захавайце дакумент.

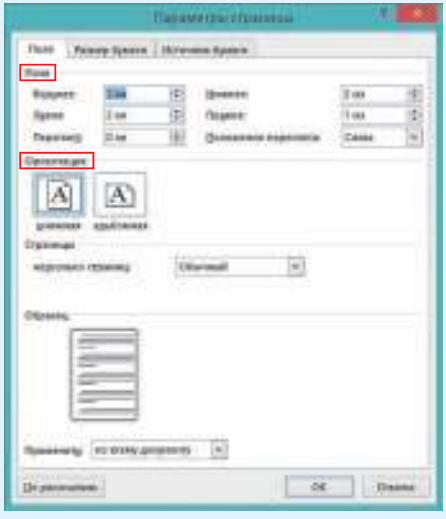
3 Адкрыйце дакумент, захаваны ў папярэднім заданні.

1. Змяніце ў стылі **Доклад** памер шрыфту з 14 на 15.
2. Змяніце ў стылі **Заголовок 1 Доклад** колер шрыфту з цёмна-сіняга на любы іншы, а памер шрыфту з 22 на 24.
3. Унясіце якое-небудзь змяненне ў фармаціраванне стылю **Заголовок 2 Доклад**.
4. Абнавіце змест.
5. Захавайце дакумент пад новым імем.

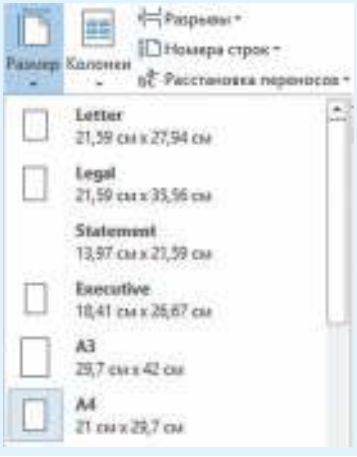
§ 26. Фармаціраванне старонкі

Прыклад 26.1. Каманды групы **Параметры страницы**.

Прыклад 26.2. Акно **Параметры страницы**.



Прыклад 26.3. Каманды кнопкі **Размер**.



26.1. Параметры старонкі

Падрыхтоўка тэкставага дакумента да друку пачынаецца з фармаціравання старонак дакумента. Устанавіць параметры фармаціравання старонак можна з дапамогай каманд групы **Параметры страницы** на ўкладцы **Разметка страницы** (прыклад 26.1). Кнопка са стрэлачкай ↘ разгортвае аднайменнае акно (прыклад 26.2).

Пры афармленні тэкставага дакумента, прызначанага для друку на паперы, важным параметрам дакумента з'яўляецца памер друкаванага ліста (укладка **Размер бумаги** або выпадаючы спіс каля кнопкі **Размер** — прыклад 26.3). У большасці выпадкаў выкарыстоўваецца папера стандартных памераў: А4 — 210 × 297 мм.

Для паперы стандартнага памеру, акрамя памераў, вызначаюць і параметр **Ориентация** (кнопкі **Поля** або **Ориентация** на ўкладцы **Разметка страницы**, прыклад 26.4). Адрозніваюць кніжную (вертыкальную) арыентацыю, пры якой вышыня ліста большая за яго шырыню, і альбомную (гарызантальную), пры якой шырыня ліста большая за яго вышыню.

Яшчэ адным параметрам старонкі дакумента, прызначанага для вываду на друк, з'яўляецца паняцце поля (укладка або кнопка **Поля**). Поле тэкставага дакумента — гэта адлегласць ад краю ліста да мяжы змяшчэння тэксту на старонцы. Задаюць верхняе, ніжняе, левае і правае палі. Пры выбары палёў неабходна ўлічваць магчымасці прын-

тара, патрабаванні стандартаў (прыклад 26.5) і прызначэнне дакумента.

Палі злева пакідаюць для пераплёту. Палі зверху і знізу звычайна выкарыстоўваюць для калонтытулаў і нумарацыі старонак.

Для афармлення выгляду старонак можна выкарыстоўваць розныя межы (прыклад 26.6). Акно **Границы и заливка** можна адкрыць з дапамогай адпаведнай каманды з выпадаючага спіса кнопкі **Границы** (⊞) на ўкладцы **Главная**. Для меж можна выкарыстоўваць лініі рознага тыпу, колеру і шырыні (укладка **Страница**). Акрамя таго, у якасці мяжы можна выбраць розныя відарысы. У выпадаючым спісе **Рисунок** можна выбраць карцінкі, арнаменты і іншыя шаблоны меж старонкі.

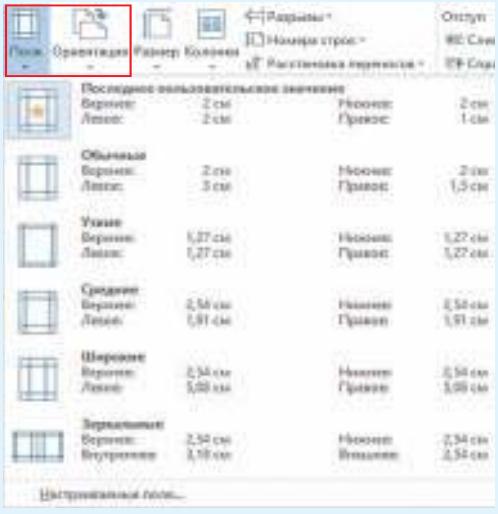
26.2. Калонтытулы

Асобнымі параметрамі тэкставага дакумента з'яўляюцца верхні і ніжні калонтытулы.

Калонтытул — тэкст або відарыс, якія размяшчаюцца на краі кожнай старонкі дакумента і паўтараюцца на ўсіх старонках.

Калонтытул звычайна размяшчаюць на ўсіх старонках дакумента, акрамя тытульных. Традыцыйна ўжываюць верхні і ніжні калонтытулы. У ніжнім калонтытуле часта ставяць нумарацыю старонак дакумента, а ў верхнім выводзяць назву дакумента і (або) прозвішча аўтара.

Прыклад 26.4. Каманды кнопкі **Поля**.



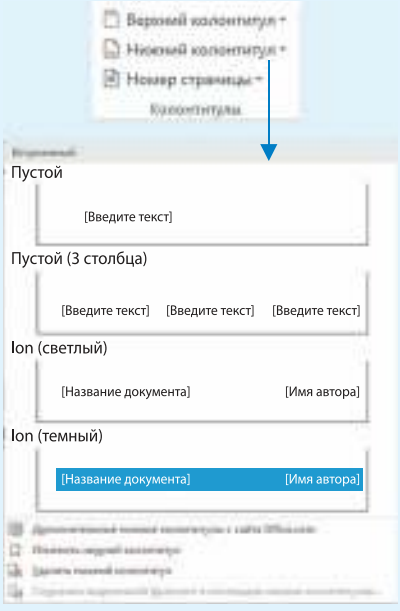
Прыклад 26.5. Стандарты афармлення палёў.

Для падачы работ на XXII Рэспубліканскі конкурс работ даследчага характару (канферэнцыя) навучэнцаў[1] вызначаны наступныя патрабаванні: ліст фармату А4; памер левага поля 30 мм, правага — 10 мм, верхняга і ніжняга — 20 мм. Звычайна такія ж палі ўстанаўліваюць для рэфератаў і іншых вучэбных работ.

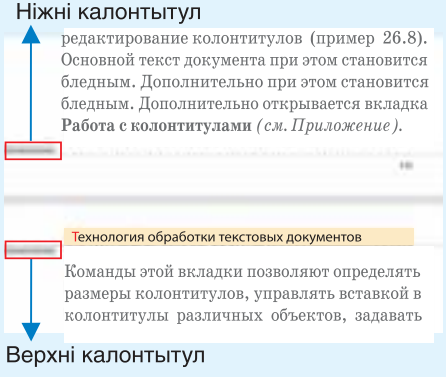
Прыклад 26.6. Акно **Границы и заливка**.

[1] http://www.uni.bsu.by/arrangements/conf/index.html (дата доступу: 16.01.2018).

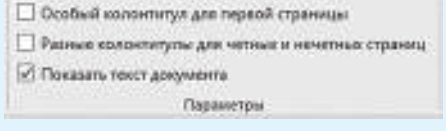
Прыклад 26.7. Група **Колонтитулы** ўкладкі **Вставка**.



Прыклад 26.8. Рэжым работы з калонтытуламі.



Прыклад 26.9. Група **Параметры** ўкладкі **Работа с колонтитулами**.



Унесці або змяніць інфармацыю ў раздзеле калонтытулаў можна з дапамогай каманд групы **Колонтитулы** ўкладкі **Вставка** (прыклад 26.7). Каманды **Верхний (Нижний) колонтитул** змяшчаюць некалькі шаблонаў афармлення калонтытулаў.

Двайны клік мышы ў верхнім або ніжнім полі старонкі дазваляе таксама перайсці да рэдагавання калонтытулаў (прыклад 26.8). Асноўны тэкст дакумента пры гэтым робіцца бледным. Дадаткова адкрываецца ўкладка **Работа с колонтитулами** (гл. *Дадатак 4*, с. 167). Каманды гэтай укладкі дазваляюць вызначаць памеры калонтытулаў, кіраваць устаўкай у калонтытулы розных аб'ектаў, задаваць параметры (прыклад 26.9). Параметр **Особый колонтитул для первой страницы** дазваляе выдаліць калонтытул з першай старонкі дакумента. Гэта важна, калі першая старонка з'яўляецца тытульнай.

Параметр **Разные колонтитулы для четных и нечетных страниц** дазваляе ўносіць у калонтытулы розны тэкст (напрыклад, на левую старонку — загаловак, на правую — прозвішча аўтара). Кнопка **Закрыть окно колонтитулов** вяртае ў асноўны тэкст дакумента.

Для ўстаўкі нумара старонкі неабходна выканаць каманду **Номер страницы** (прыклад 26.10). Для кожнага месцазнаходжання нумара старонкі ёсць магчымасці дадаткова выбраць, дзе будзе змяшчацца нумар у калонтытуле і як ён будзе выглядаць (прыклад 26.11).

Каманда **Формат номеров страниц** адкрывае аднайменнае акно. Тут можна выбраць фармат нумара: лічбы або лацінскія літары. Да нумара старонкі можна дабавіць нумар главы. Тут жа можна вызначыць, ці трэба працягваць нумарацыю старонак або пачаць яе нанова (напрыклад, для новага раздзела). Калі нумарацыя пачынаецца нанова, то можна паказаць, з якога нумара.

26.3. Падрыхтоўка дакумента да друку

Перад тым як адправіць дакумент на друк, рэкамендуецца выканаць яго папярэдні прагляд.

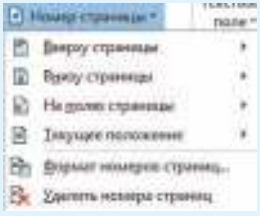
Папярэдні прагляд дакумента дазваляе карыстальніку ўбачыць, як будзе выглядаць кожная старонка дакумента пры друку. Пасля папярэдняга прагляду пры неабходнасці можна зрабіць змяненні ў афармленні дакумента. Напрыклад, выдаліць лішнія пустыя старонкі або змяніць змяшчэнне абзацаў.

Для вываду дакумента на друк трэба выканаць каманду **Файл** → **Печать.** На панэлі **Печать** можна:

- задаць колькасць копій дакумента;
- задаць нумары старонак, якія трэба вывесці на друк;
- выбраць арыентацыю і памер ліста;
- выбраць колькасць старонак для друку ў зменшаным выглядзе на адным лісце;
- змяніць памеры палёў;
- выбраць прынтар і настроіць яго ўласцівасці.

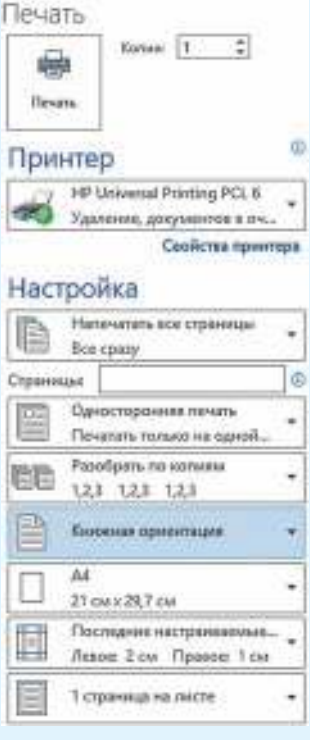
(Разгледзьце прыклад 26.12.)

Прыклад 26.10. Устаўка нумара старонкі.



Прыклад 26.11. Нумар унізе старонкі.

Прыклад 26.12. Настройкі друку.



1. Якія параметры старонкі вы можаце назваць?
2. Што такое палі старонкі?
3. Якую арыентацыю можна задаць для старонкі?
4. Што такое калонтытул?
5. Як уставіць мяжу на старонку?
6. Як уставіць нумар старонкі?

Практыкаванні

1 Устанавіце для ліста паперы А4 альбомную арыентацыю, усе палі — па 2 см. Набярыце тэкст. Падбярыце фармаціраванне сімвалаў і абзацаў так, каб тэкст заняў увесь ліст. Выканайце папярэдні прагляд дакумента. Раздрукуйце.

Увага! Увага! Увага!

17 мая ў школе праводзіўся ЗБОР МАКУЛАТУРЫ.

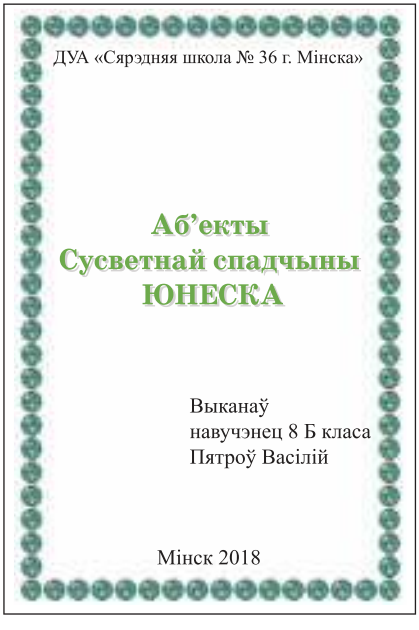
ВЫНІКІ

1-е месца — 8 Б клас — 300 кг
2-е месца — 6 А клас — 250 кг
3-е месца — 7 Г клас — 150 кг

Віншуем пераможцаў, якія атрымаюць *салодкія прызы*.

Савет школы

2 Афармеце тытульны ліст да рэферата па геаграфіі па ўзоры на рысунку справа і раздрукуйце яго на прынтары. Для вызначэння становішча элементаў тытульнага ліста выкарыстайце лінейку і каманды ўкладкі **Разметка страницы**.



3 Адкрыйце дакумент, створаны ў практыкаванні 1 з папярэдняга параграфа.

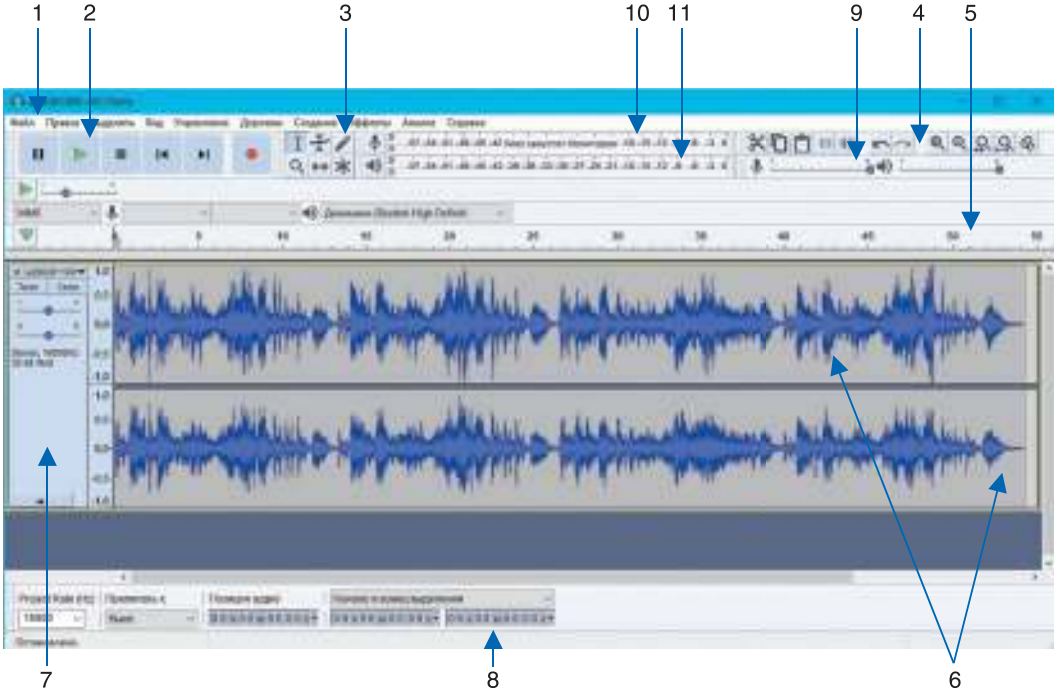
1. Устаўце нумары старонак унізе старонкі, выраўноўванне — ад цэнтра.
2. Устанавіце для дакумента наступныя палі: левае — 25 мм, правае — 10 мм, верхняе і ніжняе — 15 мм.
3. Устаўце тытульны ліст, створаны ў заданні 2. Замяніце назву.
4. Дабаўце ў верхні калонтытул сваё прозвішча. Устанавіце параметр **Особый колонтитул для первой страницы**.

5*. Праілюструйце дакумент.

ДАДАТКІ

Дадатак 1

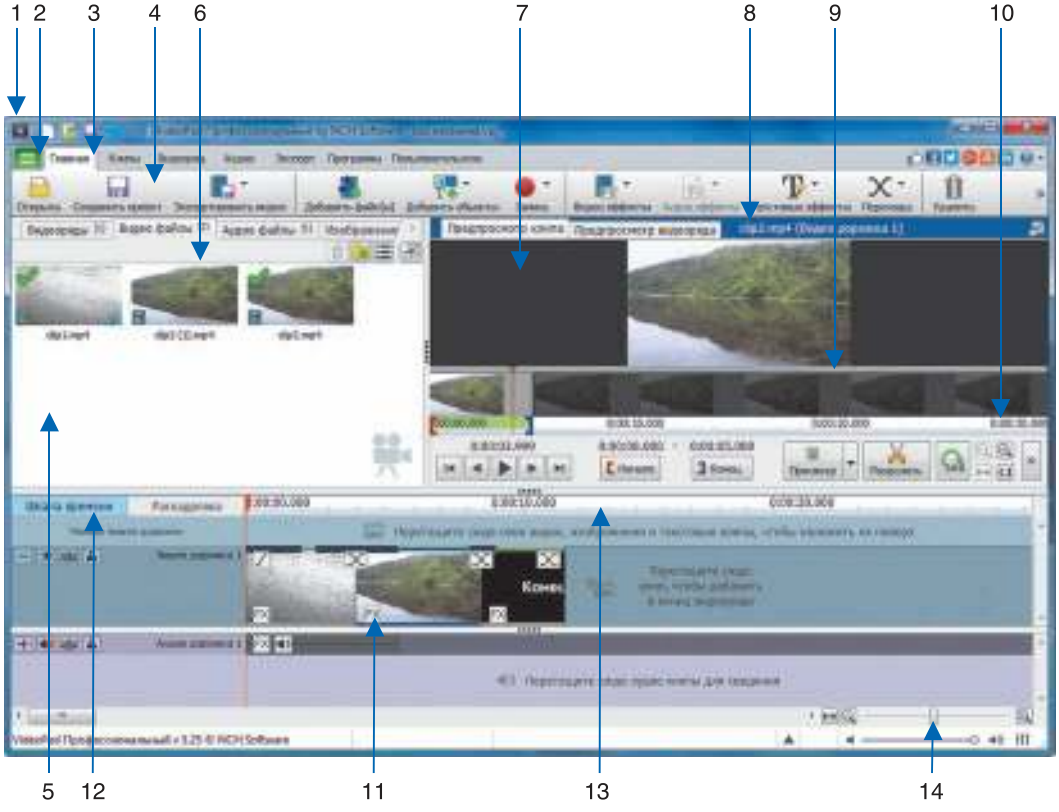
Элементы інтэрфейса аўдыярэдактара Audacity



1. Галоўнае меню.
2. Панэль прайгравання і запісу.
3. Панэль інструментаў.
4. Панэль рэдагавання.
5. Шкала часу.
6. Дарожкі.
7. Панэль кіравання дарожкамі.
8. Панэль вылучэння фрагментаў.
9. Мікшар.
10. Панэль маніторынгу ўзроўню запісу.
11. Панэль маніторынгу ўзроўню прайгравання.

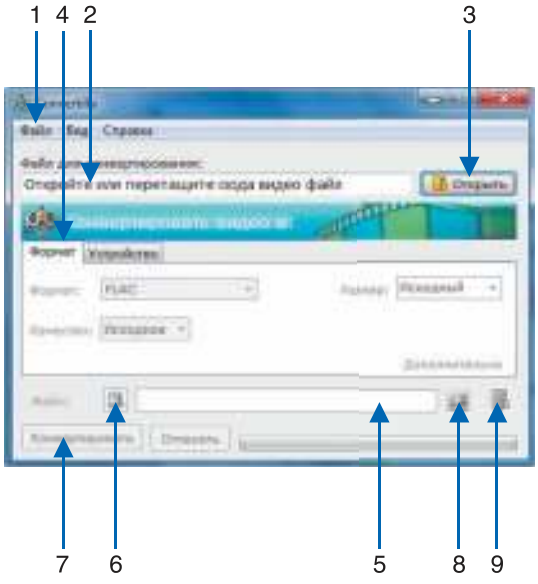
Панэлі можна перамяшчаць.

Элементы інтэрфейса відэарэдактара VideoPad



1. Панэль хуткага доступу.
2. Кнопка выкліку дадатковага меню.
3. Радок з назвамі ўкладак.
4. Панэль інструментаў выбранай укладкі.
5. Акно раздзелаў з файламі.
6. Укладкі раздзелаў з файламі.
7. Акно перадпрагляду.
8. Загаловак акна перадпрагляду з дзвюма ўкладкамі.
9. Паласа эскізаў.
10. Шкала часу ў акне перадпрагляду.
11. Акно відэарада.
12. Кнопкі выбару рэжыму акна відэарада.
13. Шкала часу ў акне відэарада.
14. Інструмент маштабавання відарыса ў акне відэарада.

Элементы інтэрфейса відэаканвертара Convertilla



1. Меню.
2. Поле для адлюстравання поўнага імя зыходнага файла.
3. Кнопка выкліку акна **Выбор файла видео**.
4. Укладка **Формат** для ўводу параметраў выхаднога файла.
5. Поле для адлюстравання поўнага імя выхаднога файла.
6. Кнопка выкліку акна **Открыть** для ўводу імя выхаднога файла і выбару папкі.
7. Кнопка запуску канвертацыі.
8. Кнопка запуску прайгравання выхаднога файла.
9. Кнопка адкрыцця папкі з выхадным файлам.

Магчымасці ўкладкі Формат

Калі загружаны аўдыяфайл:

- на ўкладцы можна выбраць абазначэнне аўдыяфармату і значэнне параметра якасці выхаднога файла;
- спасылка **Дополнительно** не працуе.

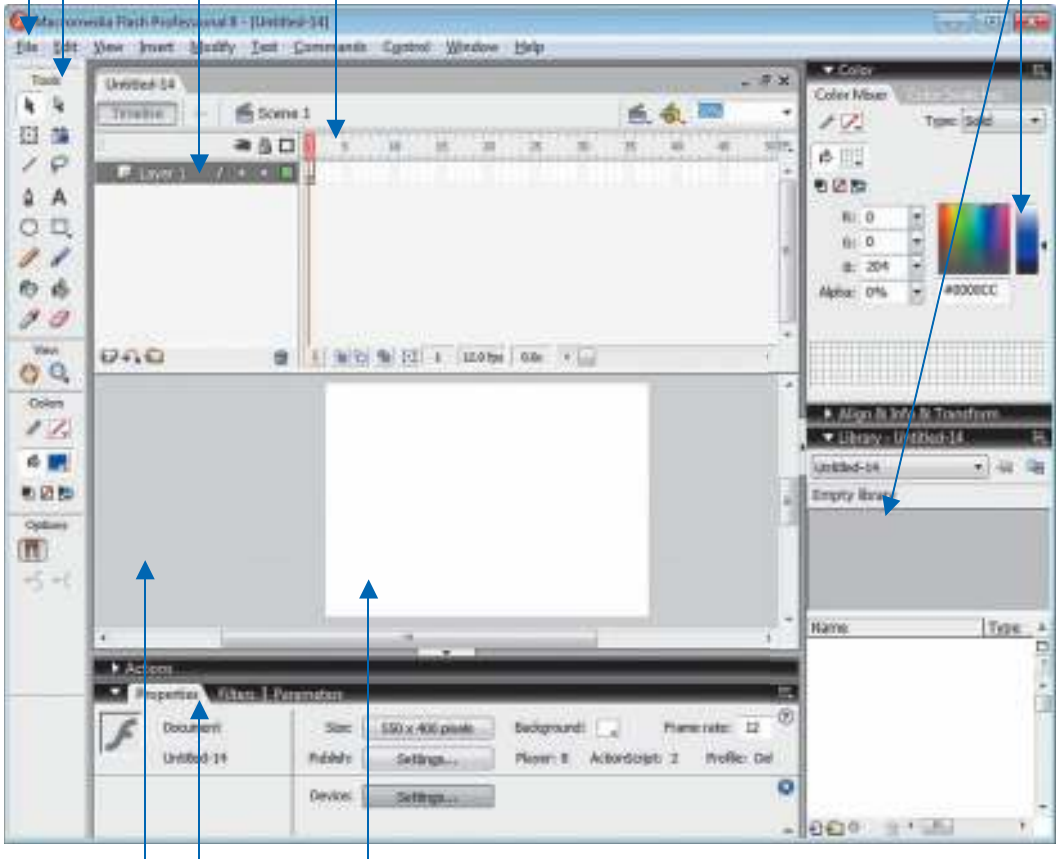
Калі загружаны відэафайл:

- на ўкладцы можна выбраць абазначэнне відэафармату, значэнне параметра якасці і разрашэнне (у полі **Размер**);
- спасылка **Дополнительно** на ўкладцы адкрывае меню з параметрамі **Без звука** і **Сохранить соотношение сторон**;
- выбар аўдыякодэка для відэафайла не прадугледжаны.

Дадатак 2

Элементы інтэрфейса рэдактара Flash

Акно рэдактара



1. Радок меню.
2. Панэль інструментаў.
3. Рабочая вобласць.
4. Мантажны стол.
5. Панэль уласцівасцей.
6. Спіс слаёў.
7. Шкала часу.
8. Дадатковыя панэлі.

Стартавая старонка

Па змоўчанні пасля загрузкі рэдактара Flash адкрываецца стартавая старонка, якая дае доступ да трох разделаў.

Каб адкрыць ужо існуючы файл, у раздзеле **Open a Recent Item** стартавай старонкі неабходна выбраць імя файла або папкі, у якой ён знаходзіцца. Для стварэння новага фільма трэба выбраць **Flash Document** у раздзеле **Create New**. У трэцім раздзеле знаходзіцца спіс шаблонаў.

Разгарнуць і згарнуць дадатковыя панэлі можна з дапамогай меню **Окно (Window)** або націснуўшы на значок спіса, які раскрыецца (▶ або ▼) побач з назвай панэлі.

Панэль уласцівасцей (Properties) адлюстроўвае і дазваляе рэдагаваць уласцівасці або вылучаных аб'ектаў, або рабочай вобласці, або актыўнага інструмента. Разгортваецца і згортваецца пры націсканні на значок спіса, які раскрываецца, побач са словам Properties.

Дадатак 3

Матэматычныя функцыі мовы Pascal

Запіс на Pascal	Апісанне
`abs(x)`	Знаходзіць модуль ліку x.
`ceil(x)`	Знаходзіць найменшае цэлае $\geqslant x$ (`real`). Вынік — лік тыпу `integer`.
`cos(x)`	Вылічае косінус ліку x. Лік x задаецца ў радыянах.
`DegToRad(x)`	Пераводзіць градусы ў радыяны.
`floor(x)`	Знаходзіць найбольшае цэлае $\leqslant x$ (`real`). Вынік — лік тыпу `integer`.
`frac(x)`	Знаходзіць дробную частку рэчаіснага ліку x (`real`). Вынік — лік тыпу `real`.
`int(x)`	Знаходзіць цэлую частку рэчаіснага ліку x (`real`). Вынік — лік тыпу `real`.
`Max(a,b)`	Знаходзіць максімальны з лікаў a і b.
`Min(a,b)`	Знаходзіць мінімальны з лікаў a і b.
`Odd(i)`	Лагічная функцыя, якая вяртае значэнне `True`, калі і няцотнае, і `False` у адваротным выпадку.
`Power(x, y)`	Знаходзіць значэнне xy (x, y — `real`). Вынік — лік тыпу `real`.
`RadToDeg(x)`	Пераводзіць радыяны ў градусы.
`round(x)`	Акругляе лік x да бліжэйшага цэлага. Калі лік знаходзіцца пасярэдзіне паміж двума цэлымі, то акругляецца да бліжэйшага цотнага (банкаўскае акругленне): `round(2.5)=2`, `round(3.5)=4`.
`sin(x)`	Вылічае сінус ліку x. Лік x задаецца ў радыянах.
`sqr(x)`	Узводзіць лік x у квадрат.
`sqrt(x)`	Знаходзіць корань квадратны з ліку x. Вынік — заўсёды лік тыпу `real`.
`tan(x)`	Вылічае тангенс ліку x. Лік x задаецца ў радыянах.
`trunc(x)`	Знаходзіць цэлую частку рэчаіснага ліку x (`real`). Вынік — лік тыпу `integer`.

Некаторыя графічныя прымітывы

Запіс на Pascal	Апісанне
`SetPixel(x,y,c)`	Зафарбоўвае піксел з каардынатамі (x, y) колерам c.
`MoveTo(x,y)`	Устанаўлівае бягучую пазіцыю рысавання ў пункт (x, y).
`LineTo(x,y)`	Рысуе адрэзак ад бягучай пазіцыі да пункта (x, y). Бягучая пазіцыя пераносіцца ў пункт (x, y).
`Line(x1,y1,x2,y2)`	Рысуе адрэзак ад пункта $(x1, y1)$ да пункта $(x2, y2)$.
`Line(x1,y1,x2,y2,c)`	Рысуе адрэзак ад пункта $(x1, y1)$ да пункта $(x2, y2)$ колерам c.
`Circle(x,y,r)`	Рысуе запоўнены круг з цэнтрам (x, y) і радыусам r.
`Ellipse(x1,y1,x2,y2)`	Рысуе запоўнены эліпс, абмежаваны прамавугольнікам, зададзеным каардынатамі супрацьлеглых вяршынь $(x1, y1)$ і $(x2, y2)$.
`Rectangle(x1,y1,x2,y2)`	Рысуе запоўнены прамавугольнік, зададзены каардынатамі супрацьлеглых вяршынь $(x1, y1)$ і $(x2, y2)$.
`RoundRect(x1,y1,x2,y2,w,h)`	Рысуе запоўнены прамавугольнік са скругленымі краямі; $(x1, y1)$ і $(x2, y2)$ задаюць пару супрацьлеглых вяршынь, а w і h — шырыню і вышыню эліпса, як і выкарыстоўваецца для скруглення краёў.
`Arc(x,y,r,a1,a2)`	Рысуе дугу акружнасі з цэнтрам у пункце (x, y) і радыусам r, заключаную паміж двума прамянямі, якія ўтвараюць вуглы $a1$ і $a2$ з воссю OX ($a1$ і $a2$ — рэчыўныя, задаюцца ў градусах і адлічваюцца супраць гадзіннікавай стрэлкі).
`Pie(x,y,r,a1,a2)`	Рысуе запоўнены сектар круга з цэнтрам у пункце (x, y) і радыусам r, заключаны паміж двума прамянямі, якія ўтвараюць вуглы $a1$ і $a2$ з воссю OX ($a1$ і $a2$ — рэчыўныя, задаюцца ў градусах і адлічваюцца супраць гадзіннікавай стрэлкі).
`TextOut(x,y,z)`	Выводзіць радок або лік z у прамавугольнік з каардынатамі левага верхняга вугла (x, y).
`DrawTextCentered(x,y,x1,y1,z)`	Выводзіць радок або лік z, адцэнтраваныя ў прамавугольніку з каардынатамі $(x, y, x1, y1)$.
`FloodFill(x,y,c)`	Залівае вобласць аднаго колеру колерам c, пачынаючы з пункта (x, y).

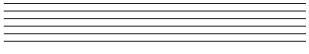
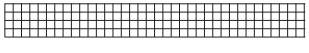
Стылі пяра (Pen)

Запіс на Pascal	Апісанне
psSolid	Суцэльнае пяро (па змоўчанні)
psClear	Празрыстае пяро
psDash	Штрыхавое пяро
psDot	Пункцірнае пяро
psDashDot	Штрыхпункцірнае пяро
psDashDotDot	Альтэрнатыўнае штрыхпункцірнае пяро (штрых і два пункціры)

Стылі пэндзля (Brush)

Запіс на Pascal	Апісанне
bsSolid	Суцэльны пэндзаль (па змоўчанні)
bsClear	Празрысты пэндзаль
bsHatch	Штрыхавы пэндзаль
bsGradient	Градыентны пэндзаль
Для ўсіх пэндзляў выкарыстоўваецца ўласцівасць Color. Для штрыхавога пэндзля дадаткова можна ўстанаўліваць уласцівасці Hatch і HatchBackgroundColor, для градыентнага — уласцівасць GradientSecondColor.	

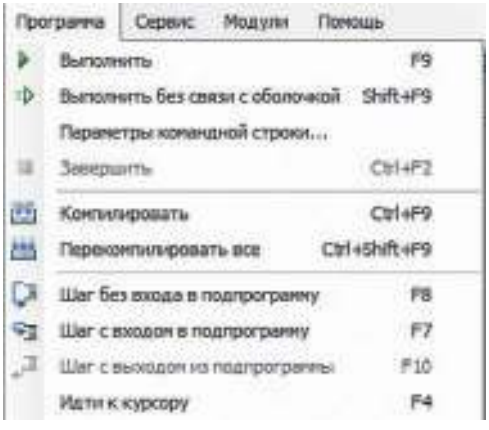
Некаторыя стылі штрыхоўкі пэндзля (HatchStyle)

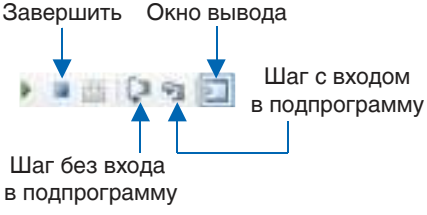
Запіс на Pascal	Апісанне
bhHorizontal	
bhVertical	
bhForwardDiagonal	
bhCross	
bhZigZag	

Наладка праграм у асяроддзі PascalABC

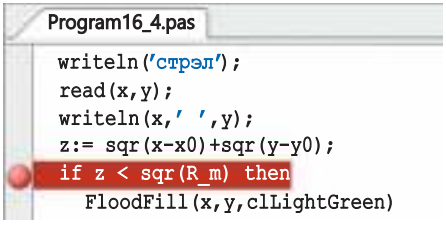
Праграма можа змяшчаць лагічныя памылкі, якія дазваляюць выканаць праграму, аднак вынік выканання будзе адрозніваца ад чаканага. Наяўнасць такіх памылак вызначаецца з дапамогай тэсціравання, а выпраўляць іх дазваляе наладчык, які ўваходзіць у склад асяроддзя праграмірования PascalABC. Ён дазваляе выконваць праграму, назіраючы за змяненнем значэнняў пераменных.

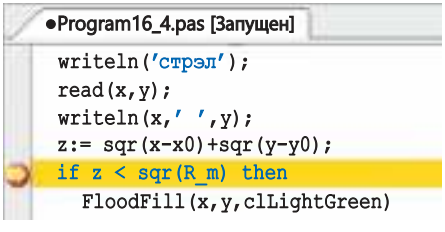
Каманды наладчыка сабраны ў меню **Программа** і вынесены на **Панэль інструментаў**.





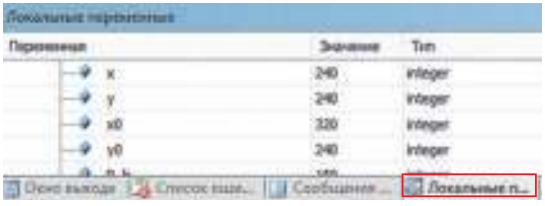
Некаторыя радкі праграмы могуць быць пазначаны як **пункты перарыву**. Для гэтага дастаткова клікнуць мышшу злева ад радка праграмы. Калі ў працэсе выканання праграмы дасягаецца пункт перарыву, выкананне праграмы прыпыняецца. Радок, на якім спынілася выкананне праграмы, падсвечваецца жоўтым.

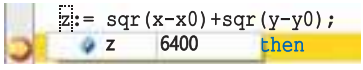




Пасля гэтага можна прагледзець значэнні пераменных у акне выведу на ўкладцы **Локальные переменные**, пачаць пакрокавае выкананне праграмы або выканаць праграму да наступнага пункта перарыву. У рэжыме наладкі,

навёўшы паказальнік мышы на імя пераменнай, у акне рэдактара кода можна адразу ўбачыць яе значэнне.





Каманды наладкі		
Каманда	Функцыя	Клавіша
Выканаць	Выконвае праграму	F9
Завяршыць	Завяршае сеанс наладкі або выкананне праграмы	Ctrl+F2
Ісці да курсора	Выконвае праграму да радка, на які ўстаноўлены курсор	F4
Крок без уваходу ў падпраграму	Выконвае радок праграмы. Пры наяўнасці выкліку падпраграмы ў гэтым радку выконвае падпраграму цалкам	F8
Крок з уваходам у падпраграму	Выконвае радок праграмы. Пры наяўнасці выкліку падпраграмы ў гэтым радку пераходзіць да выканання каманд падпраграмы	F8
Крок з выхадам з падпраграмы	Выконвае падпраграму да канца і перадае кіраўніцтва на каманду, якая ідзе за выклікам дадзенай падпраграмы ў асноўнай праграме	F10

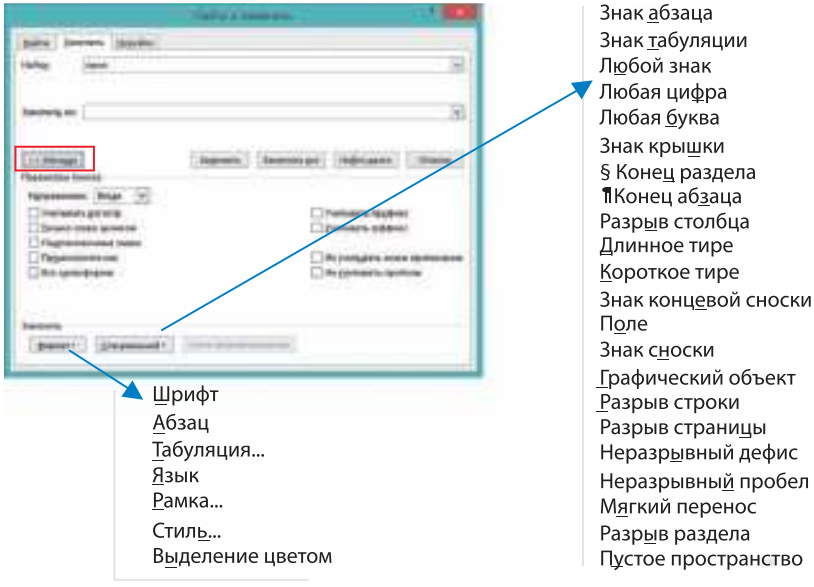
Дадатак 4

Работа з тэкставым дакументам

Пошук і замена

У акне **Заменить** ёсць кнопка **Больше**, націснуўшы на якую атрымаем дадатковыя магчымасці замены ў тэксце. Пасля націскання надпіс на кнопцы мяняецца на **Меньше**.

Кнопка **Формат** дазваляе паказаць параметры фармаціравання тэксту, якія трэба ўлічваць пры пошуку і замене.

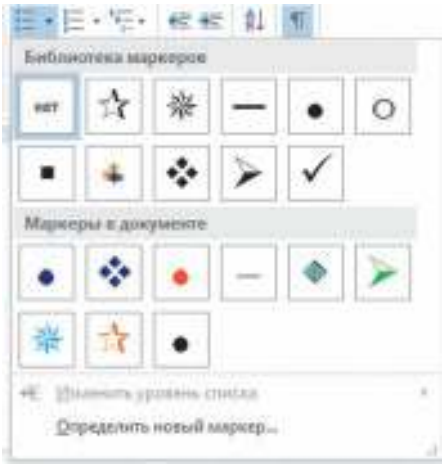


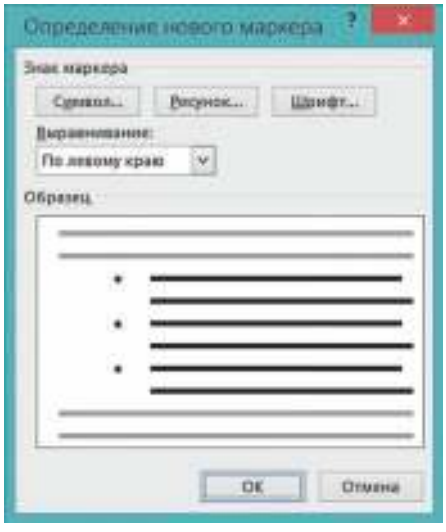
Кнопка **Специальный** дазваляе шукаць і замяняць недрукуемыя сімвалы: сімвалы абзаца, разрыву радка, пераносу і інш.

Стварэнне спісаў

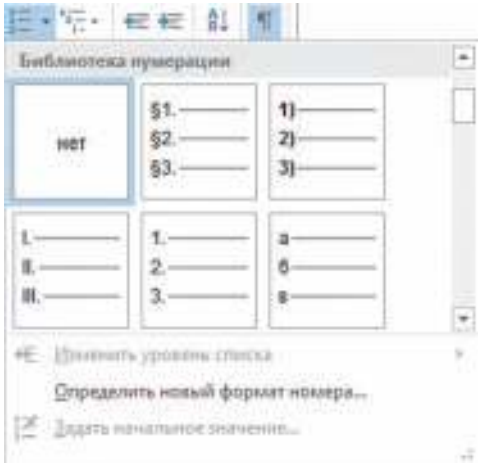
Пры стварэнні спісаў можна не толькі выкарыстоўваць для іх гатовыя стылі, але і ствараць свае.

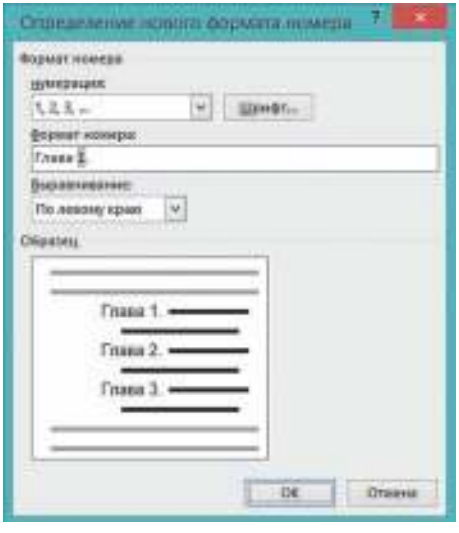
Каманда **Определить новый маркер** адкрывае адпаведнае акно, у якім у якасці маркера можна вызначыць кожны сімвал з табліцы сімвалаў або адвольны рысунак. З дапамогай кнопкі **Шрифт** можна задаць параметры сімвала: колер, памер, напісанне.





Для нумараваных спісаў можна вызначаць свой фармат нумара з дапамогай каманды **Определить новый формат номера.** У радку **Формат номера** перад нумарам можна ўпісаць кожнае слова (глава, прыклад і да т. п.), якое будзе прыпісвацца перад нумарам у выбраным спісе.





Пры стварэнні спісаў прытрымліваюцца наступных правілаў:

1. Сказ перад спісам можа заканчвацца двукроп'ем або кропкай. Двукроп'е ставіцца, калі ў гэтым сказе змяшчаецца слова ці словазлучэнне, якое паказвае на тое, што далей ідзе спіс або спіс тлумачыць тое, пра

што гаворыцца ў папярэднім сказе. У адваротным выпадку перад спісам ставіцца кропка.

2. У спісах, у якіх для нумарацыі выкарыстоўваюць лікі (арабскія або рымскія), тэкст пачынаецца з вялікай літары, калі пасля нумара стаіць кропка, і з малой літары, калі пасля нумара стаіць дужка.

3. У спісах, у якіх для нумарацыі выкарыстоўваюць літары (рускія або лацінскія), тэкст пачынаецца з вялікай літары, калі выкарыстоўваецца вялікая літара з кропкай, і з малой літары, калі для нумарацыі выкарыстоўваюцца малыя літары з дужкай.

4. У маркіраваных спісах тэкст пачынаецца з малой літары.

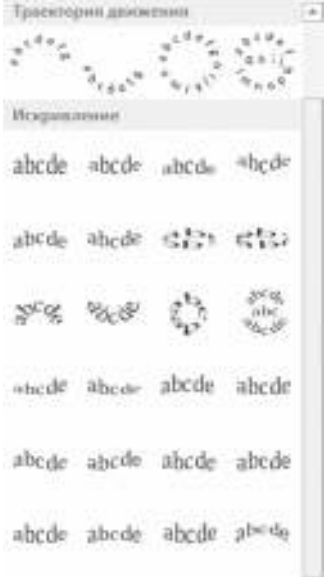
5. Пасля элемента спіса можа стаяць:

- коска, калі элементам спіса з'яўляецца адно слова;
- кропка з коскай, калі элемент спіса пачынаўся з малой літары;
- кропка, калі элемент спіса пачынаўся з прапісной літары.

6. У канцы спіса ставіцца кропка.

7. Пры выкарыстанні спісаў неабходна абавязкова звяртаць увагу на тое, каб пачатковыя словы кожнага элемента спіса былі ўзгоднены паміж сабой у родзе, ліку і склоне.

Параметры WordArt

Стылі WordArt	Эфекты WordArt	Каманда **Преобразовать**
		

Укладка Вставка

Укладка Работа с таблицами

Канструктар

Макет

Параметры SmartArt

Канструктар

Фармат

Укладка Работа с колонтитулами

(Назва і нумар установы адукацыі)

Вучэбны год	Імя і прозвішча навучэнца	Стан вучэбнага дапаможніка пры атрыманні	Адзнака навучэнцу за карыстанне вучэбным дапаможнікам
20 /			
20 /			
20 /			
20 /			
20 /			
20 /			

Вучэбнае выданне

Котаў Уладзімір Міхайлавіч
Лапо Анжаліка Іванаўна
Быкадораў Юрый Аляксандравіч
Вайцеховіч Алена Мікалаеўна

ІНФАРМАТЫКА

Вучэбны дапаможнік для 8 класа
ўстаноў агульнай сярэдняй адукацыі з беларускай мовай навучання

Заг. рэдакцыі *Г. А. Бабаева*. Рэдактар *К. І. Даніленка*. Вокладка *А. М. Багушэвіча*. Мастацкія рэдактары *А. А. Жданоўская, В. М. Карповіч*. Тэхнічнае рэдагаванне і камп'ютарная вёрстка *А. Ю. Агафонавай*. Карэктары *В. С. Казіцкая, А. П. Тхір, Г. В. Алешка*.

Падпісана да друку 30.08.2018. Фармат $70 \times 90^{1}/_{16}$. Папера афсетная. Гарнітура школьная. Друк афсетны. Умоўн. друк. арк. 12,29. Ул.-выд. арк. 10,0. Тыраж 16 000 экз. Заказ .

Выдавецкае рэспубліканскае ўнітарнае прадпрыемства «Народная асвета» Міністэрства інфармацыі Рэспублікі Беларусь. Пасведчанне аб дзяржаўнай рэгістрацыі выдаўца, вытворцы, распаўсюджвальніка друкаваных выданняў № 1/2 ад 08.07.2013. Пр. Пераможцаў, 11, 220004, Мінск, Рэспубліка Беларусь.

ААТ «Паліграфкамбінат імя Я. Коласа». Пасведчанне аб дзяржаўнай рэгістрацыі выдаўца, вытворцы, распаўсюджвальніка друкаваных выданняў № 2/3 ад 04.10.2013. Вул. Каржанеўскага, 20, 220024, Мінск, Рэспубліка Беларусь.