ОСТАПЕЦЬ В. С.

Комплексні Електронні Дидактичні Засоби *в системі* Інформаційних Технологій Навчання

методичний посібник для вчителів

2007 р.

Щасливський навчально-виховний комплекс

ОСТАПЕЦЬ ВОЛОДИМИР СТЕПАНОВИЧ

КОМПЛЕКСНІ ЕЛЕКТРОННІ ДИДАКТИЧНІ ЗАСОБИ В СИСТЕМІ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ НАВЧАННЯ

Методичний посібник для вчителів

2007 рік

Схвалено: науково-методичною радою
Київського обласного інституту
післядипломної освіти педагогічних кадрів
протокол №1 від 18.01.2007 р.

Рецензенти: **Валерій Анатолійович Федорчук**,
Завідуючий
центром інформаційних технологій навчання
Київського обласного інституту
післядипломної освіти педагогічних кадрів

Ірина Вільївна Скляр,
викладач УФМЛ КПІ ім. Т. Шевченка,
редактор газети ″Інформатика″,
Заслужений учитель України

Анотація: У посібнику розглядається питання про додаткові можливості інформаційно-комп'ютерних технологій навчання, які не пов’язані з використанням ліцензованих ППЗ. Ці можливості суттєво доповнюють традиційні форми ІКТ, передбачають творче застосування прикладного програмного забезпечення загального призначення, зокрема пакету Microsoft Office. Детально описано створення та використання електронних конспектів уроків, комплексних електронних дидактичних засобів та електронних сценаріїв уроків на прикладах динамічної електронної довідкової системи з алгоритмізації і програмування “Турбо Паскаль в таблицях“ та навчальної презентації з математики. Окремим розділом виділено опис технології створення складних електронних дидактичних засобів навчання доступними кожному користувачу персонального комп'ютера засобами, зокрема Microsoft Power Point. До посібника додається комплект з майже 70 таблиць та їх електронний варіант, виготовлений у Microsoft Power Point. Посібник може бути корисним як при викладанні алгоритмізації та програмування у загальноосвітніх школах так і при викладанні інших навальних предметів, особливо природничо-математичного циклу.

І. ПЕРЕДМОВА

Сьогодні надзвичайно актуальні питання впровадження інформаційно-комп'ютерних технологій навчання (ІКТН), які часто називають коротше - інформаційно-комп'ютерними технологіями (ІКТ). Серед учителів, робота яких не пов'язана безпосередньо з комп'ютерною технікою, побутує помилкове уявлення про функціональні можливості сучасного комп'ютера. З одного боку вони розуміють, що комп'ютер – це одне з найновіших досягнень науково-технічного прогресу, за яким майбутнє, з іншого, вважають, що використання комп'ютерної техніки у навчально-виховному процесі зводиться переважно до ролі технічного засобу навчання (ТЗН) у *традиційному розумінні*. Проте комп'ютер – це *універсальний* технічний засіб навчання, а демонстраційна функція, до якої зводилось використання традиційних ТЗН, є лише невеликою частиною його можливостей. Безперечно, при наявності сучасних ліцензованих програмних навчальних засобів (ПНЗ), виникають дуже широкі можливості ілюстрування та демонстрації історичних подій, фізичних, хімічних і біологічних явищ, суспільних процесів, технологій виробництва тощо. Але вони будуть ефективними лише на етапах накопичення та розуміння, для *аналізу, синтезу, використання* та *оцінки* отриманих знань, умінь і навичок необхідно використовувати потужніші та ефективніші можливості ІКТН, зокрема опрацювання великих обсягів інформації, організація складних обчислень і проведення чисельних експериментів, експертизи даних, пошук необхідної інформації тощо.

Ще в 60-80-і роки минулого сторіччя, коли в навчально-виховному процесі масово використовувались такі традиційні ТЗН, як кінопроектори, діаскопи, кодоскопи, програвачі, телевізори, було централізовано створено великі навчально-пізнавальні фільмотеки, фонотеки, діатеки тощо. Але дуже швидко стало зрозуміло, що все це малоефективне. Дійсно, навчальні кінофільми чи діафільми могли використовуватись лише епізодично, фрагментарно, на вступних чи підсумкових етапах вивчення теми, вимагали значного часу для монтажу і експлуатації навчальної техніки, мало відповідали конкретним умовам на уроці. Тому на практиці технічні засоби навчання застосовувались рідко. Природно, що при розгляді комп'ютера в якості демонструючого ТЗН, сучасні, навіть найкращі, ліцензовані ПНЗ успадкували три дуже суттєві хиби тих фільмо-, фоно- та діатек. Перша – *консервативність*, адже їх важко адаптувати до потреб учителя з конкретними педагогічними ситуаціями, навпаки, використання ПНЗ змушує учителя самому пристосовуватись до них. Це *нівелює творчий характер навчального процесу*, якщо не ставити за мету провести показовий, "відкритий" урок. Друга – *переважно інформаційний характер ПНЗ*. Учителю ж необхідні динамічні, із експериментальними, а не тільки демонстраційними та тестуючими можливостями програмні засоби. Нарешті, третя - *всеоб'ємність*, тобто поверхове охоплення всього курсу. Слід зауважити, що лише уроки інформатики по своїй сутності є ”живим” спілкуванням з комп’ютером.

В той же час унаочнення у процесі навчання також не можна ігнорувати. Вихід слід шукати у визначенні певної "золотої середини", з одного боку якої - громіздке, позбавлене динамічності ПНЗ, з іншого – безпосереднє використання стандартних додатків Windows та додатків пакету Microsoft Office. Ми бачимо таку "середину" в так званих *комплексних електронних дидактичних засобах* (КЕДЗ)[1], під якими розуміємо файли чи пакети файлів, що містять будь-яку інформацію навчального характеру, дидактичний або наочний засіб, що розсилається на робочі місця учнів (РМУ) через локальну мережу для використання учнями під час роботи з комп'ютером. КЕДЗ не слід плутати із ліцензованими програмними продуктами навчального призначення, бо він виготовляється учителем, гнучкий за характером будови та застосування, може оперативно змінюватись, включати гіперпосилання, анімаційні чи звукові ефекти, адаптований до конкретної ситуації, має, як правило, локальний зміст, володіє багатофункціональністю.

Прикладами КЕДЗ можуть бути таблиці та опорні схеми, електронні конспекти до уроків чи тем, презентації PowerPoint тощо. Перш за все це документи, які умовно можна назвати *електронними конспектами уроків*(ЕКУ). Вони можуть містити теоретичні відомості з необхідними питаннями, завданнями, тестами, бути оснащеними таблицями, схемами, малюнками. Завдяки системі гіперпосилань ці документи мають динамічний характер, розсилка на РМУ через локальну мережу дає змогу використовувати їх оперативно і комплексно з системами програмування та навчальними пакетами. Бажано, щоб кожен КЕДЗ охоплював вузьку тему, тоді створювати його легко, при потребі завжди можна доопрацювати і удосконалити. Крім того, робота з ним урізноманітнює урок, робить його більш динамічним, створює атмосферу позитивної мотивації вивчення навчального матеріалу, дає можливість відчути курс інформатики, як єдине ціле, учні майже безперервно працюють з комп'ютером, не відчуваючи нав'язливої ролі учителя. Такі форми роботи на уроці спонукають і вчителя підходити до більшості тем не формально, строго відповідно програмі, а із спрямуванням на наступне практичне застосування, все щойно вивчене при першій нагоді використовувати практично.

При постійному використанні ІКТН від електронних дидактичних засобів, які носять локальний за змістом характер, можна безболісно переходити до великих за обсягом і змістом КЕДЗ. При творчому підході вони будуть ефективнішими за ліцензовані ПНЗ, маючи надзвичайно важливі переваги перед останніми: можливості конструювання, доповнення та удосконалення на будь-якому етапі.

Ми розглянемо створення і застосування кількох окремих КЕДЗ з інформатики та математики. Але очевидно, що подібний підхід до використання ІКТН можна успішно застосовувати при вивченні практично всіх шкільних навчальних курсів.

[1] Назви "*комплексний електронний дидактичний засіб*", "електронний конспект уроку", "електронний сценарій уроку" та інші – це умовні робочі терміни, введені автором.

ІІ. ЕЛЕКТРОННІ КОНСПЕКТИ УРОКІВ

Спочатку зупинимось на понятті електроного конспекту уроку. Якщо не ототожнювати поняття “конспект уроку” з поняттям “поурочний план”, а мати на увазі достатньо деталізований опис відповідного навчального матеріалу з прикладами та методичними порадами (ремарками) щодо вивчення цього матеріалу, то такі конспекти уроків можна успішно використовувати і як фрагменти підручника для учнів, особливо в умовах недостатнього забезпечення учнів якісними підручниками. Необхідно лише “сховати” методичні ремарки. Кожен виготовлений засобами Microsoft Office друкований конспект уроку можна з допомогою локальної мережі розсилати на РМУ. Використання таких конспектів буде вирішувати одночасно кілька дидактичних завдань. По-перше, учні матимуть можливість читати, так би мовити, в оригіналі думки учителя, причому навіть під час підготовки домашнього завдання, якщо скопіювати конспект на ГМД. По-друге, працюючи над темою, кожен учень буде постійно комплексно застосовувати знання, уміння та практичні навички, одержані на уроках інформатики раніше, що буде для них додатковим чинником у формуванні сучасної інформаційної культури. Нарешті, по-третє, крім інформаційних технологій під час практичного заняття за комп’ютером не обійтись і без інших інтерактивних технологій навчання.

Як уже згадувалось вище, на прикладі електронного конспекту уроку ми вводимо якісно нове поняття *комплексного електронного дидактичного засобу*.

В електронних конспектах уроків ми широко застосовуємо не тільки графічні, а й анімаційні засоби та гіперпосилання. Це дає можливість роботу над навчальним текстом із статичного стану перевести в динамічний, що має дуже великі, раніше недоступні, дидактичні можливості. Наведемо приклад електронного конспекту уроку на тему:”Оператор вибору (Select Case)”, застосованого у 9-у спеціалізованому класі фізико-математичного профілю при вивченні візуального проектування та об’єктно-орієнтованого програмування на прикладі пакета Visual Basic. У цьому конспекті застосовано такі гіперпосилання: 🡻, 🡻🡻, 🡻🡻🡻, 🡹, 🡹🡹, 🡹🡹🡹. Вони дозволяють, при потребі, кілька разів переходячи з одного місця конспекту в інше, краще закріплювати навчальний матеріал. У поєднанні із графічними можливостями (блок-схема та малюнок) отримуємо згаданий вище унаочнений динамічний ефект.

Електронні конспекти уроків, комплекти таблиць та інші комплексні дидактичні матеріали можна створювати як з допомогою Microsoft Word, так і у вигляді презентацій та Web-сторінок, причому, створивши Word-варіант, перетворювати його у інший, перерахований вище, вигляд.

Пропонований конспект уроку з теми “Оператор вибору“ для 9-го спеціалізованого класу фізико-математичного профілю (У Щасливському НВК базовий курс інформатики у спеціалізованих та ліцейних класах фізико-математичного профілю викладається в 7-9 класах згідно програми “Інформатика. 7-9 класи. *Автори: Жалдак М.І., Морзе Н.В., Науменко Г.Г.* і програми для загальноосвітніх навчальних закладів фізико-математичного, природничого та технологічного профілів (Інформатика. 10-11 класи. *Автори: Жалдак М.І., Морзе Н.В., Мостіпан О.І., Науменко Г.Г.*) – зразок комплексного електронного дидактичного засобу (КЕДЗ), які вчитель масово використовує у своїй практиці

(детальніше див. Комп'ютер у школі та сім'ї, №7, 2006 р., стор.22), створений в електронному варіанті, розсилається на робочі місця учнів через локальну мережу. Конспект уроку оснащений гіперпосиланнями для оперативного пошуку потрібних місць із означеннями, поясненнями, завданнями тощо.

Тема уроку: **ОПЕРАТОР ВИБОРУ (SELECT CASE)**

Мета уроку:

- Розширити і узагальнити поняття про розгалужені процеси;
- Ознайомити учнів із командою вибору, вивчити її формат, блок-схему, правило виконання;
- На конкретних прикладах розібрати різні варіанти застосування команди вибору
- Порівняти команди вибору і розгалуження.

Обладнання:

- Таблиці для демонстрації формату команди вибору та її блок схеми[2];
- Таблиці для пояснення проекту з командою вибору на закріплення вивченого матеріалу;
- Електронний варіант конспекту уроку на кожному робочому місці учня;
- Проект у VB для закріплення оператора select Case на кожному робочому місці учня
- В.В. Федько, В.І. Плоткін, Основи алгоритмізації та програмування, 11клас , Харків, видавництво "Ранок", 2003р.

ХІД УРОКУ

І. **Повторення**.

а) У формі опитування перерахувати команди розгалуження і повторення, їх різновидності:

- Повторення (з передумовами while, until);
- Повторення (з післяумовами while, until);
- Повторення (з лічильником із step);
- Повторення (з лічильником без step);
- Розгалуження (однорядкове);
- Розгалуження (блочне);
- Розгалуження (повна форма);
- Розгалуження (скорочена форма).

б) У формі письмової самостійної роботи за варіантами перевірити знання команд розгалуження

ІІ. **Вивчення нового матеріалу**.

а) *Оголошення та актуалізація теми*.

Пригадати, що при вивченні команд розгалуження, згадувалась *команда вибору* (алгоритмічна мова, 8 клас) і повідомлялось, що блочна форма команди розгалуження нагадує команду вибору. Предметом сьогоднішнього вивчення є справжня команда вибору – оператор select case, тому слід весь час співставляти її із

[2] Одна з таблиць входить до конспекту уроку, інші надруковані на папері формату А2.

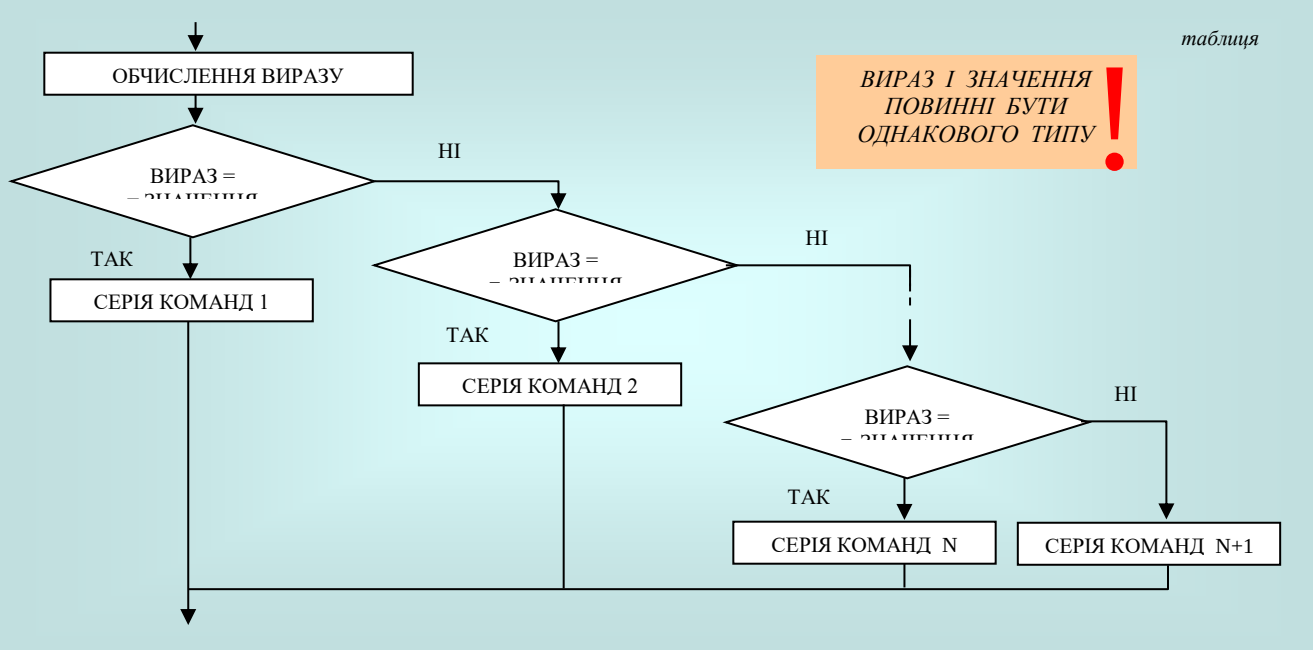
командою вибору (8 клас) та блочною формою команди розгалуження.

б) *Вивчення оператора* ***Select Case*** *(пояснення з допомогою таблиці).*

Оператор вибору у Visual Basic має вигляд (демонструється таблиця):

Select Case <Вираз>
[Case <Список_порівняння1>
<0ператори1>]
. . .
[Case <Список_порівняння N>
<0ператори N>]
[Case Else
<Оператори N+1>]
End Select

де *<Вираз>* може бути виразом будь-якого типу (цілого, дійсного, рядкового тощо).



<Список_порівняння> складається з елементів порівняння, розділених комою. Елемент порівняння можна записати в одному з трьох видів:

1) вираз, що дає одне значення для перевірки на збіг
(наприклад, 3, x+7);
2) конструкція <Вираз1> To <Вираз2>, що задає інтервал значень для перевірки влучення в діапазон
(наприклад, 1 To 5); (1)⬇
3) конструкція Is <Операція порівняння> <Вираз> задає значення, для яких виконується умова в порівнянні
(наприклад, Is > 32). (2)⬇⬇

Якщо серед умов вибору зустрічається математична подвійна нерівність вигляду

$$a < x < b,$$

то її можна реалізувати за допомогою оператора вибору в такий спосіб:

Select Case x
. . .
Is > a And (x < b) (3) ⬇⬇⬇

. . .
End Select

Правило (пояснення з допомогою блок-схеми, таблиця): Оператор вибору виконується так. Спочатку обчислюється вираз, що стоїть після ключових слів Select Case. Потім відшукується в списках перше порівняння, якому відповідає одержане значення. Якщо таке порівняння знайдене, виконуються оператори, які стоять у відповідному блоці Case, і керування передається на кінець оператора. Якщо ж значення виразу не задовольняється в жодному списку і є конструкція Case Else, виконуються оператори, які містяться в ній. У іншому випадку жоден із операторів, які містяться в операторі вибору, не виконується.

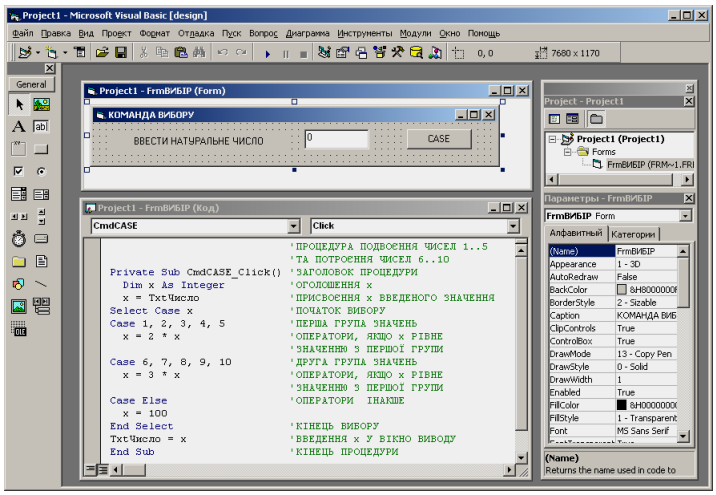
в) *Висновок*:

Таким чином, оператор вибору дуже близький за дією до умовного оператора. І, якщо вираз, який стоїть після слів Select Case - логічного типу, він цілком з ним збігається. Оператор вибору доцільно використовувати в тих випадках, коли обчислювальний процес складається з більш ніж двох гілок, тобто — в складних розгалужених процесах.

Це ще один приклад того, як один і той самий результат можна дістати різними способами. На практиці оператор If частіше використовується при обчисленнях, а оператор Select Case — для оброблення пунктів меню та груп перемикачів.

III. **Закріплення**.

Запустити у VB проект “Команда вибору”. Користуючись таблицями на дошці пояснити суть завдання (див. рис.):



Учні повинні виконати проект кілька разів, взявши різні діапазони, даних, наприклад, 1..5, 5..15.

Завдання 1. Використати для даного проекту (1) ↑;

Завдання 2. Використати для даного проекту (2) ↑↑

Завдання 3. Використати для даного проекту (3) ↑↑↑

IV. **Домашнє завдання**:

вивчити стор136-137, дати письмово відповіді на запитання 1-2 (стор.137).

Запитання

1. Яке призначення мають умовний оператор і оператор вибору?
2. Як задаються умови в операторі вибору?

III. ДИНАМІЧНА ДИДАКТИЧНА СИСТЕМА "ТУРБО ПАСКАЛЬ В ТАБЛИЦЯХ"

III.1 КОМПЛЕКТ ТАБЛИЦЬ ТА ЙОГО ДИДАКТИЧНІ ОСОБЛИВОСТІ

Спочатку зосередимось на комплексному електронному дидактичному засобі ″Турбо Паскаль в таблицях″. Він призначений для методичної та дидактичної підтримки розділу ″Алгоритмізація і програмування″, адже в шкільному навчальному плані з інформатики цей розділ, поза сумнівом, є провідним по змістовному навантаженню, незважаючи на те, що на його вивчення відведено часу порівняно небагато. Дійсно, без цього розділу середня освіта не була б повною, адже вміння логічно, алгоритмічно та структурно мислити сьогодні є дуже важливою характерною рисою.

Проте вивчення саме цього розділу, м'яко кажучи, "не престижне" серед учнів. Цьому є ряд причин, як об'єктивних, так і суб'єктивних. Однією з них може бути недостатній рівень наочності викладання, адже алгоритми і програми - це дуже абстрактні об'єкти. Сприймати пояснення учителя, маючи перед очима тільки екран комп'ютера з вікном Turbo Pascal, важко. Практика підтверджує, що поєднання роботи з комп'ютером із записами на дошці та демонстрацією таблиць, сприяє міцнішому засвоєнню методів програмування. Але використання дошки в комп'ютерному кабінеті з ряду відомих причин небажане, тому важливим, якщо не найважливішим, засобом унаочнення при вивченні програмування можуть бути таблиці. Їх успішно замінив би лише демонстраційний екран, але він сьогодні ще рідкісний і не дешевий. Таблиці ж особливо допомагають на початкових етапах вивчення алгоритмізації і програмування та при поясненні вузлових питань.

Але для вчителя інформатики сьогодні і таблиці малодоступні через їх практичну відсутність. З рекомендованих для використання Міністерством освіти і науки України сьогодні відомий єдиний посібник ″Основи інформатики у визначеннях, таблицях і схемах″, Малярчук С.В., Ранок, 1999 р. Із цього становища є простий і доступний вихід. У кожному комп'ютерному кабінеті можна створити будь-який комплект таблиць, причому сьогодні краще мати їх електронний варіант. Від паперового, який можна виготовити з допомогою принтера, він незрівнянно дешевший, може бути при потребі модернізований. Так, у Щасливському НВК Бориспільського району у 2002 році було створено паперовий варіант комплекту таблиць "TURBO PASCAL в таблицях". У 2004 році створено електронний варіант комплекту, в основу якого покладено ідею електронного конспекта уроку, яка на той час вже практикувалась на уроках інформатики у Щасливському НВК.

Повернемось до комплекту "TURBO PASCAL в таблицях", описавши його склад та пояснивши можливості використання. Він орієнтований на найпоширенішу в школах мову програмування Паскаль, може бути дуже корисним незалежно від виду техніки та версії мови програмування, що підтримується на цій техніці, але повністю відповідає Turbo Pascal 7.

Комплект складається із семи розділів (серій), які містять 66 таблиць:

1. серія "Інструментальне середовище TURBO PASCAL 7"

1.1 Використання функціональних клавіш та комбінацій клавіш у системі TURBO PASCAL 7
1.2 Вигляд вікна TURBO PASCAL 7
1.3 Вигляд вікна Change Directory

1.4 Вигляд вікна Open a File
1.5 Застосування вікна Watches
1.6 Робота з опцією головного меню File

2. серія “Величини та запис виразів у TURBO PASCAL 7”

2.1 Характеристики величин
2.2 Зміст поняття величини в інформатиці
2.3 Типи величин в TURBO PASCAL 7
2.4 Числові типи в TURBO PASCAL 7
2.5 Оголошення та опис величин
2.6 Алфавіт мови TURBO PASCAL 7
2.7 Зарезервовані слова TURBO PASCAL 7
2.8 Список математичних функцій TURBO PASCAL 7

3. серія “Програми, підпрограми та модулі в TURBO PASCAL 7”

3.1 Структура програми
3.2 Вигляд програми
3.3 Шаблон програми з використанням стандартного модуля
3.4 Опис процедури
3.5 Опис функції
3.6 Виклик процедури
3.7 Виклик функції
3.8 Структура модуля
3.9 Вигляд модуля

4. серія “Складені команди”

4.1 Команда розгалуження
4.2 Блок-схема команди розгалуження
4.3 Команда вибору
4.4 Блок-схема команди вибору
4.5 Цикл FOR .. TO
4.6 Блок-схема циклу FOR .. TO
4.7 Цикл FOR .. DOWNTO
4.8 Блок-схема циклу FOR .. DOWNTO
4.9 Цикл WHILE
4.10 Блок-схема циклу WHILE
4.11 Цикл REPEAT
4.12 Блок-схема циклу REPEAT

5. серія “Застосування складених команд ”

5.1 Програма визначення суми n введених чисел
5.2 Заміна циклу FOR циклом WHILE
5.3 Заміна циклу FOR циклом REPEAT
5.4 Програма визначення найбільшого спільного дільника двох чисел (алгоритм Евкліда)
5.5 Програма перевірки простоти натурального числа
5.6 Вкладені цикли FOR .. FOR
5.7 Програма виводу таблиці Піфагора
5.8 Вкладені цикли FOR .. WHILE
5.9 Вкладені цикли WHILE .. FOR
5.10 Вкладені цикли FOR .. REPEAT
5.11 Вкладені цикли REPEAT .. FOR
5.12 Вкладені цикли WHILE .. REPEAT
5.13 Вкладені цикли REPEAT .. WHILE

6. серія “Опрацювання масивів”

6.1 Оголошення, опис та введення масивів
6.2 Передача масивів у підпрограми
6.3 Програма визначення найбільшого числа в лінійному масиві

6.4 Пошук елемента в неупорядкованому масиві
6.5 Пошук елемента в упорядкованому масиві
6.6 Сортування масиву
6.7 Сортування лінійного масиву методом Шелла
6.8 Зміни в масиві під час сортування методом Шелла

7. серія “Додаткові таблиці”

7.1 Основні стандартні процедури і функції опрацювання рядків
7.2 Основні стандартні процедури і функції управління текстовим екраном
7.3 Основні стандартні процедури і функції управління графічним екраном
7.4 Програмне використання клавіш управління
7.5 Основні стандартні процедури і функції роботи з текстовими файлами
7.6 Програма створення текстового файлу
7.7 Програма доповнення текстового файлу
7.8 Програма читання текстового файлу
7.9 Додавання довгих чисел
7.10 Числа Фібоначчі

Серії не обов’язково відповідають темам розділу “Алгоритмізація і програмування” шкільного курсу інформатики, в основу структури комплекту покладено один із можливих варіантів викладання, але якщо ближче познайомитись і врахувати поради щодо використання окремих розділів чи таблиць, стане зрозуміло, що це достатньо універсальний комплект і він може задовольнити потреби широкого кола викладачів.

Більшість таблиць нетрадиційні на вигляд, багато з них доповнені опорними сигналами та вказівками, часто застосовується виділення курсивом, кольором і т.д. (якщо розглядати варіант презентації, то до опорних сигналів можна застосовувати засоби анімації). Все це дозволяє досягти більшої зрозумілості в поясненні.

В таблицях часто зустрічаються такі й інші знаки: ⇨ ⇧ ⇦ ⇩ ′ ☝ ☞ ☜, а також різноманітні виноски, заштриховки, виділення блоків кольоровим заповненням. Їх використання цілком зрозуміле. Але, використовуючи комплект слід дотримуватись правила: *при ознайомленні з таблицями на першому етапі треба звертати увагу лише на текст програми чи фрагменту, а при повторному, більш поширеному і уточненому поясненні, можна поетапно використовувати додаткові елементи таблиці, обов’язково акцентуючи увагу на кожному окремо*.

Багато таблиць включають конкретні програми чи фрагменти, тому їх слід використовувати по принципу: від конкретного до загального. Звичайно, учні на момент використання таких таблиць уже повинні бути знайомі з відповідними задачами, їх математичною постановкою і, можливо, алгоритмом. Інші таблиці, навпаки, передбачають використання по принципу: від загального до конкретного, що теж може дати досить великий ефект при поясненні. Звичайно, ситуацію, в якій слід використовувати ту чи іншу таблицю, можна легко зрозуміти, крім того кожен викладач має право на власне розуміння ситуації, тому відповідні рекомендації опускаються.

Важливо мати на увазі, що таблиці дійсно утворюють об’єднаний певними ідеями комплект, тому багато з них можна використовувати в комплексі.

Даний комплект таблиць можна упорядковувати (причому не єдиним способом) так, що він відповідатиме курсу алгоритмізації і програмування, як розгляду однієї глобальної задачі з цілком упорядкованими дозами теоретичного матеріалу. Звичайно, на кожному етапі вивчення курсу можна звертатись до вже знайомих таблиць циклічно. Перша серія призначена для унаочнення ознайомлення

з інструментальним середовищем Turbo Pascal7. Екранну демонстрацію цих таблиць слід робити паралельно з практичною роботою над конкретними програмами.

Серію таблиць “Величини та запис виразів у Turbo Pascal**7**” можна використовувати, як кожну окремо, так і комплексно на протязі вивчення всього курсу.

Таблиці серій № 3-6 слід розглядати разом із конкретними задачами, побудувавши задачі згідно схеми 3. Особливе місце тут мають таблиці серій “Складені команди”, “Застосування складених команд ” та “Опрацювання масивів”, вони добре узгоджені між собою, висвітлюють найважливіші теми розділу “Алгоритмізація і програмування”.

Серія “Додаткові таблиці” дозволить краще вивчити структуровані типи, зокрема рядки та файли. Останні дуже важливі, як засіб вводу інформації для опрацювання програмою. Ввод даних – це дуже специфічна частина роботи програміста, від способу вводу залежить швидкість відлагодження програми, діапазон задач і багато іншого. Можна поєднати в єдину серію таблиці №№ 2.5, 3.1, 6.1, 6.2, 7.5-7.7, які поступово прослідковують прийоми вводу даних у програму від найпростішого, найменш ефективного - з допомогою процедури Read, до дуже потужного методу вводу даних із текстового файла.

В сьому серію включені також дві довідкові таблиці по управлінню текстовим та графічним екраном. Інформацію такого характеру можна взяти в будь-якому підручнику чи посібнику, але інколи є потреба, щоб ця інформація завжди була перед очима. Це ж стосується і таблиць №№ 2.3, 24, 2.6, 2.7, 2.8.

III.2 СТРУКТУРА ДОВІДКОВОЇ ДИНАМІЧНОЇ СИСТЕМИ ″ ТУРБО ПАСКАЛЬ В ТАБЛИЦЯХ ″

Довідкова динамічна система ″ Турбо Паскаль в таблицях″ (ДДС ТП) виготовлена в середовищі Microsoft Power Point 2003. Вона повністю відповідає описанному вище комплекту таблиць, успадковує всі її дидактичні особливості, в тому числі містить необхідні елементи опорних схем, але має іншу, відповідну вимогам до комп'ютерних презентацій структуру. Перш за все – це реалізація опорних сигналів з допомогою анімаційних ефектів. Використано не тільки основні анімаційні ефекти: ВХІД, ВИДІЛЕННЯ, ВИХІД та ШЛЯХИ ПЕРЕМІЩЕННЯ, а й систему перемикачів (тригерів), які допомагають зосереджувати увагу учнів на суттєвих моментах під час пояснення. Застосована система кнопок переходу та меню, які надають ДДС ТП властивості гіпертекстових документів. Значна частина слайдів мають властивість динамічної модифікації під час демонстрації, іншими словами, трансформуються в кілька споріднених за змістом слайдів, тому, з метою уникнення громіздкості та забезпечення максимальної швидкодії, ДДС ТП складається з системи файлів, відповідних серіям таблиць (див. наступну таблицю):

Серія	*файл*
Титульний слайд	Index.ppt
Головне меню	MainList.ppt
Інструментальне середовище TURBO PASCAL 7	Chapter2.ppt
Величини та запис виразів у TURBO PASCAL 7	Chapter1.ppt
Програми, підпрограми та модулі в TURBO PASCAL 7	Chapter6.ppt
Складені команди	Chapter5.ppt
Застосування складених команд	Chapter3.ppt

Опрацювання масивів	Chapter4.ppt
Додаткові таблиці	Chapter7.ppt
Рекомендації по застосуванню ДДС ТП	Recommendation.ppt

Титульний слайд (слайд 1) має гіперпосилання на головне меню (слайд 2). Це гіперпосилання займає праву половину слайда і виявляється легко, тому нема потреби його виділяти окремо.

Коротко опишемо систему меню. Головне меню містить гіперпосилання на файли, описані в наведеній вище таблиці. Решта меню мають подібний вигляд з головним меню і реалізують гіперпосилання на слайди (закладки) всередині відповідних файлів. Нижче відобразимо послідовність слайдів, що входять у ДДС ТП, давши їм нумерацію та вказавши у правому верхньому кутку відповідні номери таблиць комплекту. По ходу ознайомлення із слайдами будемо описувати особливості їх демонстрації та методичні поради щодо використання.

слайд 2

Виберіть потрібний розділ

- 1. Величини та вирази
- 2. Інструментальне середовище
- 3. Циклічні структури та розгалуження
- 4. Масиви
- 5. Застосування складних команд
- 6. Програми, модулі
- 7. Додаткові таблиці
- 8. Рекомендації

Вихід

слайд 3

Величини та вирази

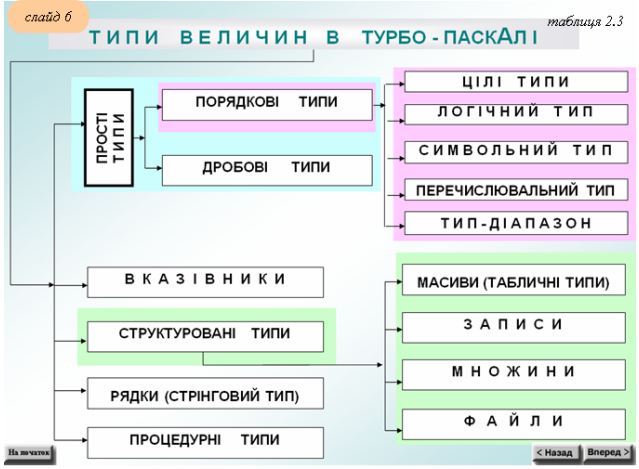
- Величини
- Типи величин
- Оголошення і опис величин
- Алфавіт мови програмування
- Зарезервовані слова ТП
- Список математичних функцій та процедур

< До змісту>

Слайд 4 — таблиця 2.1

слайд 5 — таблиця 2.2

слайд 6 — таблиця 2.3



слайд 7 — таблиця 2.4

ТИПИ ВЕЛИЧИН В ТУРБО - ПАСКАЛІ

тип	допустимі значення	формат
SHORTINT	-128 .. 127	1 байт зі знаком
INTEGER	-32768.. 32767	2 байти зі знаком
LONGINT	-2147483648 .. 2147483647	4 байти зі знаком
BYTE	0 .. 255	1 байт без знака
WORD	0 .. 65535	2 байти без знака

тип	допустимі значення	точність	формат
REAL	$2.9*10^{-39}$ .. $1.7*10^{+38}$	11 — 12 знаків	6 байт
SINGL	$1.5*10^{-45}$ .. $3.4*10^{+38}$	7 — 8 знаків	4 байти
DOUBLE	$5.0*10^{-324}$ .. $1.7*10^{-308}$	15 — 16 знаків	8 байт
EXTENDED	$3.4*10^{-4932}$ .. $1.1*10^{4932}$	19 — 20 знаків	10 байт
COMP	$-9.2*10^{+18}$ .. $9.2*10^{+18}$	19 — 20 знаків	8 байт

На початок < Назад Вперед >

слайд 8 — таблиця 2.5

```
const n=5;
```

ТИП ЕЛЕМЕНТІВ МАСИВА - ДІАПАЗОН

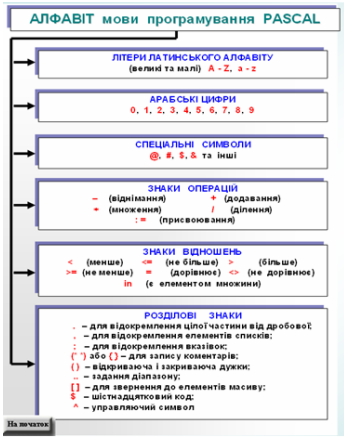
```
M : array[1..n] of 0..9 = (8,2,1,0,7);

type Mas = array[1..n,1..n] of real;

var a, b, c, max : real;
    st : string; d : byte;
    Mas1 : array[1..n,1..n] of integer;
    Mas2 : Mas;
```

ТАК ОГОЛОШУЮТЬСЯ КІЛЬКА ОДНОТИПНИХ ВЕЛИЧИН

ВЕЛИЧИНА Mas2 МАЄ ОГОЛОШЕНИЙ ТИП array[1..n,1..n] of real)

На початок < Назад Вперед >

слайд 9 — таблиця 2.6



Важливо вірно розуміти поняття "Алфавіт мови програмування Турбо Паскаль", його зв'язок із загальним поняттям "алфавіт" та роллю в цій мові програмування. До цього нас зобов'язує велика формалізація в описі мови програмування.

Алфавіт асоціюється із мовою. У будь-якій мові є свій алфавіт - це допустимий набір символів, з допомогою яких записуються слова та речення конкретної мови.

Під поняттям "український алфавіт" звичайно розуміється набір символів (літер): "А", "а", "Б", "б", ...,"Я", "я". Вже звідси очевидне протиріччя, адже вважається, що в українській мові 33 літери, насправді ж їх 66. Але в дійсності, як легко переконатись, написати довільний текст українською мовою неможливо без розділових знаків, арабських цифр, врешті пропусків між словами. Тому у повному розумінні поняття "український алфавіт" суттєво відрізняється від його побутового трактування, на що у середній школі не звертається увага.

Тому, при ознайомленні з поняттям "Алфавіт мови програмування Турбо Паскаль" слід наголосити на цих нюансах, звернути увагу на його формальність.

На початок < Назад Вперед >

слайд 8

таблиця 2.5

```
const n=5;

                    ТИП ЕЛЕМЕНТІВ МАСИВА - ДІАПАЗОН
M : array[1..n] of 0..9 = (8,2,1,0,7);

type Mas = array[1..n,1..n] of real;

var a, b, c, max : real;          ТАК ОГОЛОШУЮТЬСЯ КІЛЬКА ОДНОТИПНИХ ВЕЛИЧИН
    st : string; d : byte;
Mas1 : array[1..n,1..n] of integer;

Mas2 : Mas;        ВЕЛИЧИНА Mas2 МАЄ ОГОЛОШЕНИЙ ТИП array[1..n,1..n] of real)
```

На початок | < Назад | Вперед >

слайд 9

таблиця 2.6

АЛФАВІТ мови програмування PASCAL

ЛІТЕРИ ЛАТИНСЬКОГО АЛФАВІТУ
(великі та малі) A - Z, a - z

АРАБСЬКІ ЦИФРИ
0, 1, 2, 3, 4, 5, 6, 7, 8, 9

СПЕЦІАЛЬНІ СИМВОЛИ
@, #, $, & та інші

ЗНАКИ ОПЕРАЦІЙ
– (віднімання) + (додавання)
* (множення) / (ділення)
:= (присвоювання)

ЗНАКИ ВІДНОШЕНЬ
< (менше) <= (не більше) > (більше)
>= (не менше) = (дорівнює) <> (не дорівнює)
in (є елементом множини)

РОЗДІЛОВІ ЗНАКИ
. – для відокремлення цілої частини від дробової;
, – для відокремлення елементів списків;
; – для відокремлення вказівок;
(* *) або { } – для запису коментарів;
( ) – відкриваюча і закриваюча дужки;
.. – задання діапазону;
[ ] – для звернення до елементів масиву;
$ – шістнадцятковий код;
^ – управляючий символ

Важливо вірно розуміти поняття “Алфавіт мови програмування Турбо Паскаль”, його зв’язок із загальним поняттям “алфавіт” та роллю в цій мові програмування. До цього нас зобов’язує велика формалізація в описі мови програмування.

Алфавіт асоціюється із мовою. У будь-якій мові є свій алфавіт - це допустимий набір символів, з допомогою яких записуються слова та речення конкретної мови.

Під поняттям “український алфавіт” звичайно розуміється набір символів (літер): “А”, ”а”, ”Б”, ”б”, ...,”Я”, ”я”. Вже звідси очевидне протиріччя, адже вважається, що в українській мові 33 літери, насправді ж їх 66. Але в дійсності, як легко переконатись, написати довільний текст українською мовою неможливо без розділових знаків, арабських цифр, врешті пропусків між словами. Тому у повному розумінні поняття “український алфавіт” суттєво відрізняється від його побутового трактування, на що у середній школі не звертається увага.

Тому, при ознайомленні з поняттям “Алфавіт мови програмування Турбо Паскаль” слід наголосити на цих нюансах, звернути увагу на його формальність.

На початок | < Назад | Вперед >

слайд 10

таблиця 2.7

ЗАРЕЗЕРВОВАНІ СЛОВА В TURBO PASCAL 7

1	ABSOLUTE	21	FUNCTION	41	PROGRAM
2	AND	22	GOTO	42	PUBLIC
3	ARRAY	23	IF	43	RECORD
4	ASM	24	IMPLEMENTATION	44	REPEAT
5	ASSEMBLER	25	IN	45	SET
6	BEGIN	26	INHERITED	46	SHL
7	CASE	27	INLINE	47	SHR
8	CONST	28	INTERFACE	48	STRING
9	CONSTRUCTOR	29	INTERRUPT	49	THEN
10	DESTRUCTOR	30	LABEL	50	TO
11	DIV	31	MOD	51	TYPE
12	DO	32	NEAR	52	UNIT
13	DOWNTO	33	NIL	53	UNTIL
14	ELSE	34	NOT	54	USES
15	END	35	OBJECT	55	VAR
16	EXTERNAL	36	OF	56	VIRTUAL
17	FAR	37	OR	57	WHILE
18	FILE	38	PACKED	58	WITH
19	FOR	39	PRIVATE	59	XOR
20	FORWARD	40	PROCEDURE		

НЕ МОЖНА ВЖИВАТИ В ЯКОСТІ ІМЕН ВЕЛИЧИН, КОНСТАНТ, ПРОЦЕДУР, ФУНКЦІЙ

На початок < Назад Вперед >

слайд 11

таблиця 2.8

СПИСОК МАТЕМАТИЧНИХ ФУНКЦІЙ ТА ПРОЦЕДУР в Turbo Pascal 7

НАЗВА	*РЕЗУЛЬТАТ ЗАСТОСУВАННЯ*
ABS(X)	Модуль числа X
COS(X)	Косинус X (X – в радіанах)
SIN(X)	Синус X (X– в радіанах)
ARCTAN(X)	Арктангунс X (X – в радіанах)
EXP(X)	e^X
LN(X)	Натуральний логарифм X
PI	Число π
SQR(X)	Квадрат числа X
SQRT(X)	Арифметичний квадратний корінь з X

ФУНКЦІЇ

НАЗВА	*РЕЗУЛЬТАТ ЗАСТОСУВАННЯ*
TRUNC(X)	Число X, заокруглене, відкиданням цифр числа, що стоять після коми
FRAC(X)	Дробова частина X
INT(X)	Ціла частина X
ROUND(X)	Ціла частина X, одержана заокругленням за правилами математики
RANDOM(X)	Випадкове число з діапазону 0 . . X
ODD(X)	TRUE, якщо X непарне
INC(X,Y)	Збільшення X на Y
DEC(X,Y)	Зменшення X на Y
RANDOMIZE	Забезпечує неспівпадання послідовностей випадкових чисел, отриманих з допомогою RANDOM(X)

ПРОЦЕДУРИ

< Назад На початок

У слайді 3 зображено меню серії таблиць ″Величини та запис виразів у TURBO PASCAL7″. Кнопка <ДО ЗМІСТУ> є гіперпосиланням на головне меню (файл MainList.ppt). Така ж кнопка є в будь-якому іншому тематичному меню, тому далі їх згадувати не будемо. Головне меню оснащене аналогічним гіперпосиланням <ВИХІД> на титульну сторінку (файл Index.ppt). Кожен елемент списку будь-якого меню є гіперпосиланням на відповідний слайд чи таблицю. Для легшого пошуку всі меню виконані в однаковому стилі, відмінному від оформлення інших слайдів. Всі, слайди, що містять таблиці по оформленню ідентичні комплекту таблиць.

Слайди 4 - 11 містять таблиці 2.1 - 2.8 серії ″Величини та запис виразів у TURBO PASCAL7″. Таблиці серії ″Інструментальне середовище TURBO PASCAL7″, пронумеровані 1.1 – 1.6 будуть наведені і описані нижче у зв'язку з тим, що при вивченні програмування та мови Турбо Паскаль перед вивченням інструментального середовища бажано розібрати оголошення і опис величин, запис виразів, початкові відомості про мову Турбо Паскаль, зокрема алфавіт, зарезервовані слова, структуру типів величин, числові типи, стандартні математичні функції тощо.

Як уже зрозуміло читачеві, ДДС ТП дає можливості навігації, можна легко перейти від будь-якої таблиці до будь-якої іншої таблиці. Для цього призначена система меню і відповідні гіперпосилання між файлами та слайдами. Але звернемо також увагу на зручність розбивки пакету на окремі файли. Кожен файл Chapter№.ppt може бути завантажений у Microsoft Power Point автономно, тому при вивченні окремих тем алгоритмізації і програмування їх можна в довільному порядку розсилати на РМУ, або включати в описані вище КЕДЗ чи електронні конспекти уроків.

Коротко опишемо слайди 4 – 11. Слайд 4 крім традиційної загальної класифікації та характеристик величин має непрямий перехід на слайд 5 з таблицею 2.2 ″Зміст поняття величини в інформатиці″. Система виносок в анімаційній формі дає можливість акцентувати увагу на зв'язок між важливими поняттями ″доріжка″, ″сектор″, комірка пам'яті, ″адреса комірки″, ″розмір величини″ тощо. Анімаційні акценти реалізовано у формі трикратного мигання з допомогою перемикачів. Кнопка ВПЕРЕД слайду 4 здійснює перехід на слайд 6, де відображено система величин у Турбо Паскалі. З допомогою фіолетового та зеленого кольорів виділено блоки ″порядкові типи″ та ″структуровані типи″. У початковому вигляді слайду відсутні типи, що належать до вищевказаних, вони з'являються в результаті кликання мишею на стрілках, що виходять з блоків ″порядкові типи″ та ″структуровані типи″. Крім того, кликнувши по блоку, що з'явився, можна його видалити. Та й це ще не все. Кликання по блоках ″цілі типи″ та ″дробові типи″ поверх слайда відображає таблиці з переліком та характеристиками цих типів, включаючи їх формат і допустимі значення, тобто вміст таблиці 2.4.

Наступні слайди 9 – 11 дають довідку про алфавіт, зарезервовані слова та стандартні математичні функції мови Турбо Паскаль. Звернемо увагу на слайд 9 (таблиця 2.6). Текст справа, можливо, з оригінальним і дещо нетрадиційним уявленням про алфавіт з'являється у вигляді титрів після активації відповідного гіперпосилання. Така форма виводу вибрана не випадково. На наш погляд вона ефективно переключає увагу на акцент щодо *поняття алфавіту*. На думку автора у загальноосвітній школі необхідно звернути увагу як на розширення поняття

алфавіту, так і на узагальнення поняття МОВИ. Освіченій людині слід усвідомлювати, що будь-яка мова – це перш за все ЗАСІБ СПІЛКУВАННЯ.

Далі розглянемо блок слайдів 12 - 18 і таблиць 1.1 - 1.6 "Інструментальне середовище TURBO PASCAL7". Наведемо, не дублюючи опис меню цього блоку, а опис почнемо із слайда 13. На слайді 13 відображена таблиця 1.1 із основними функціональними клавішами та комбінаціями клавіш управління інструментальним середовищем Турбо Паскаль.

слайд 12

Інструментальне середовище

- Використання функціональних клавіш та комбінацій клавіш у ТП
- Вигляд вікна ТП 7
- Вигляд вікна *Change directory*
- Вигляд вікна *Open a file*
- Застосування вікна *Watches*
- Робота з опцією головного меню *File*

< До змісту>

слайд 13

таблиця 1.1

ВИКОРИСТАННЯ ФУНКЦІОНАЛЬНИХ КЛАВІШ ТА КОМБІНАЦІЙ КЛАВІШ У СИСТЕМІ TURBO PASCAL

Клавіша	Функція
F2	Запис на диск вмісту активного вікна
F3	Читання файлу з диска
F4	Виконання програми до місця розташування курсора
F6	Переключення вікна в активний стан
F7	Трасування в підпрограмах
F8	Трасування, не розкриваючи підпрограм
F10	Переключення між активним режимом і меню
ALT+ цифра	Перехід до вікна зі вказаним номером
Alt+F3	Закриття активного вікна
Alt+F5	Повертання до збереженого екрана
Alt+F9	Компіляція програм
Alt+X	Закінчення роботи
Ctrl+F2	Завершення сеансу трасування
Ctrl+F4	Обчислення чи модифікація змінної
Ctrl+F5	Перехід в режим управління активним вікном
←↑→↓	Переміщення активного вікна
Shift+ ←↑→↓	Зміна розмірів активного вікна
Shift+Ins	Вставка блоку з пам'яті
Shift+F6	Повернення до попереднього вікна
Shift+DeL	Стирання блоку
Ctrl+Ins	Запам'ятання блоку
Ctrl+F7	Додавання виразу у вікно перегляду
Ctrl+F8	Переключення точок зупину
Ctrl+F9	Запускання програми

На початок Вперед >

слайд 14

Таблиця 1.2

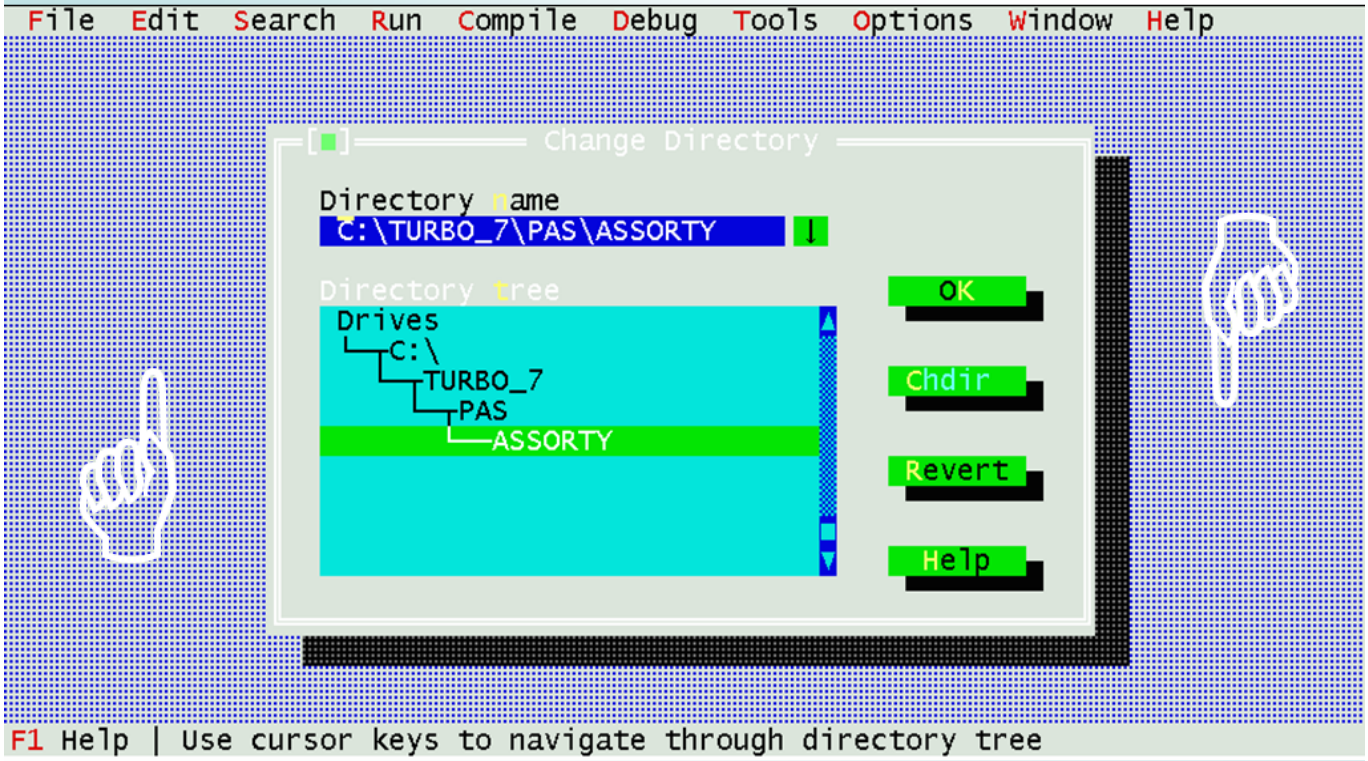
ВИГЛЯД ВІКНА TURBO PASCAL 7

На початок
< Назад Вперед >

слайд 15

Таблиця 1.3

ВИГЛЯД ВІКНА Change Directory

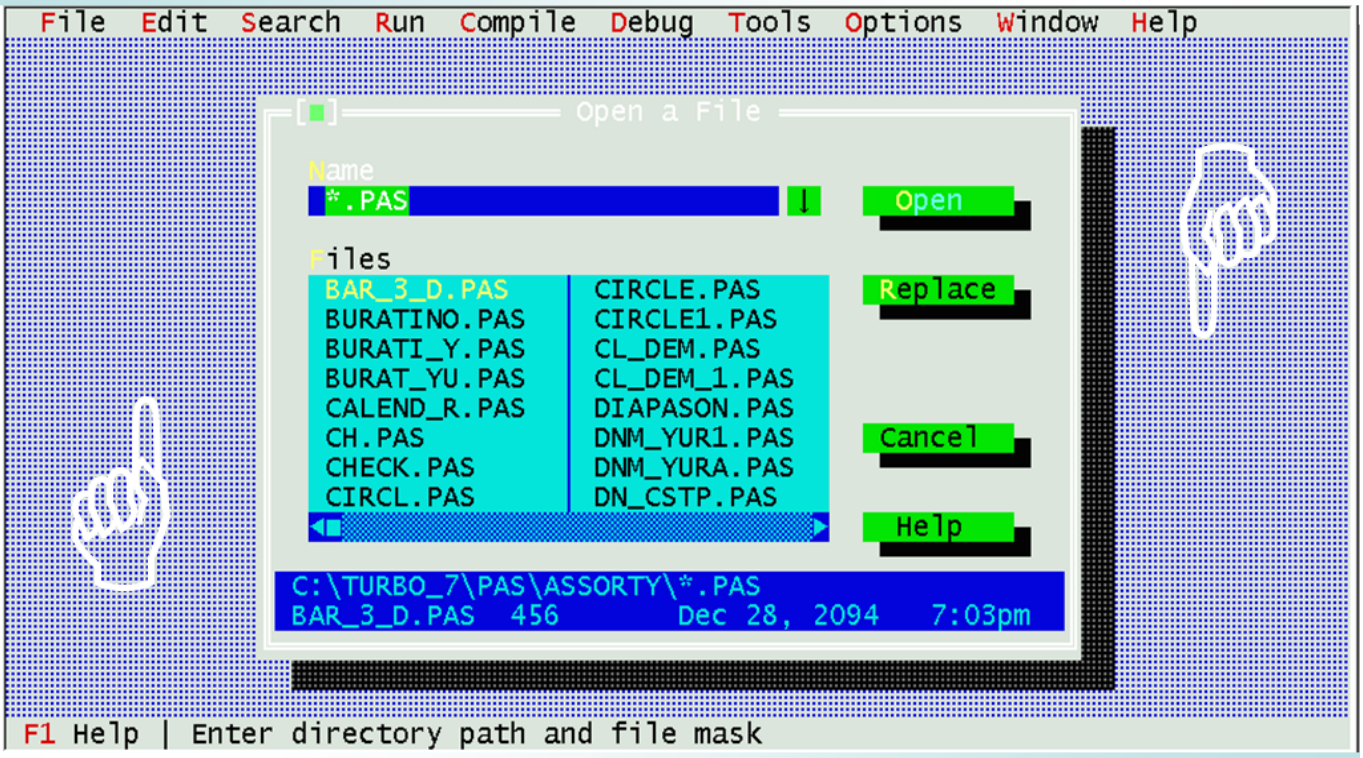


На початок
< Назад Вперед >

слайд 16

Таблиця 1.4

ВИГЛЯД ВІКНА Open a File



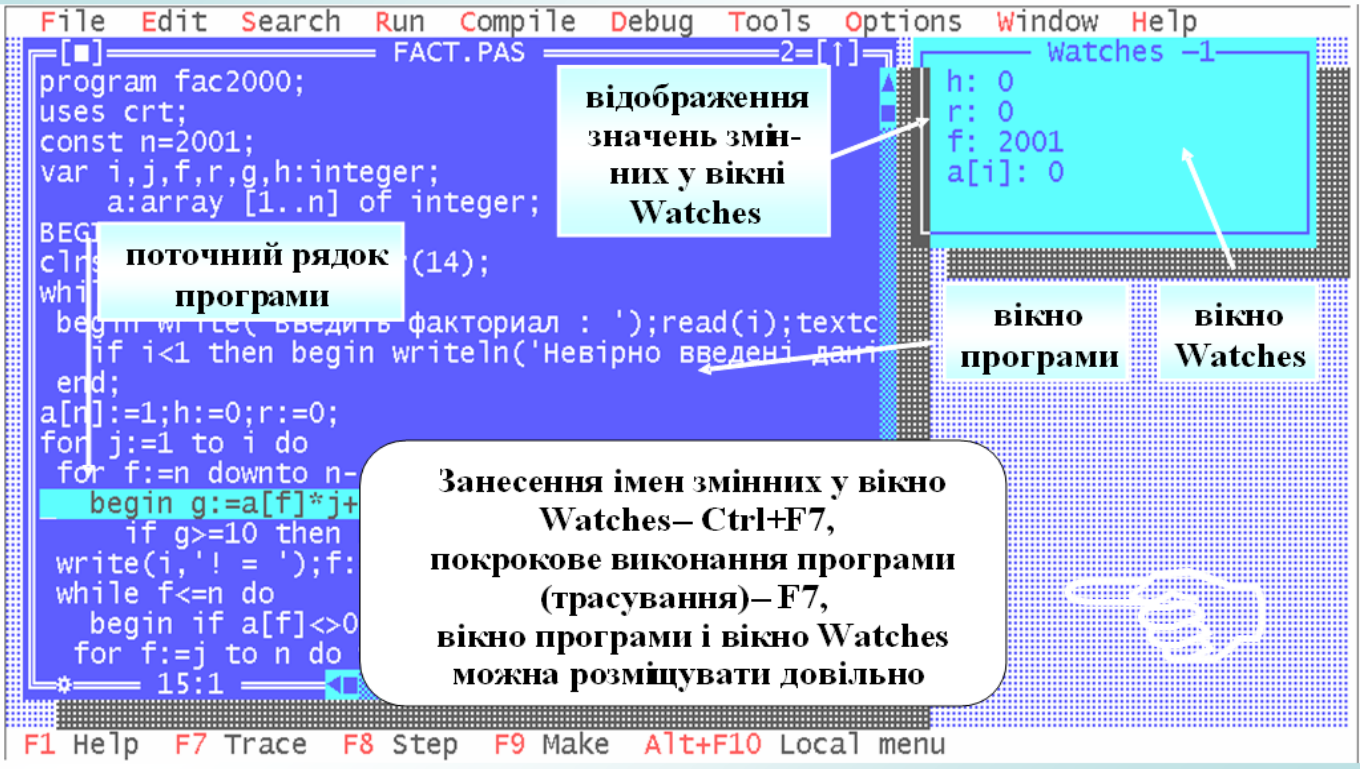
На початок

< Назад Вперед >

слайд 17

Таблиця 1.5

ЗАСТОСУВАННЯ ВІКНА Watches

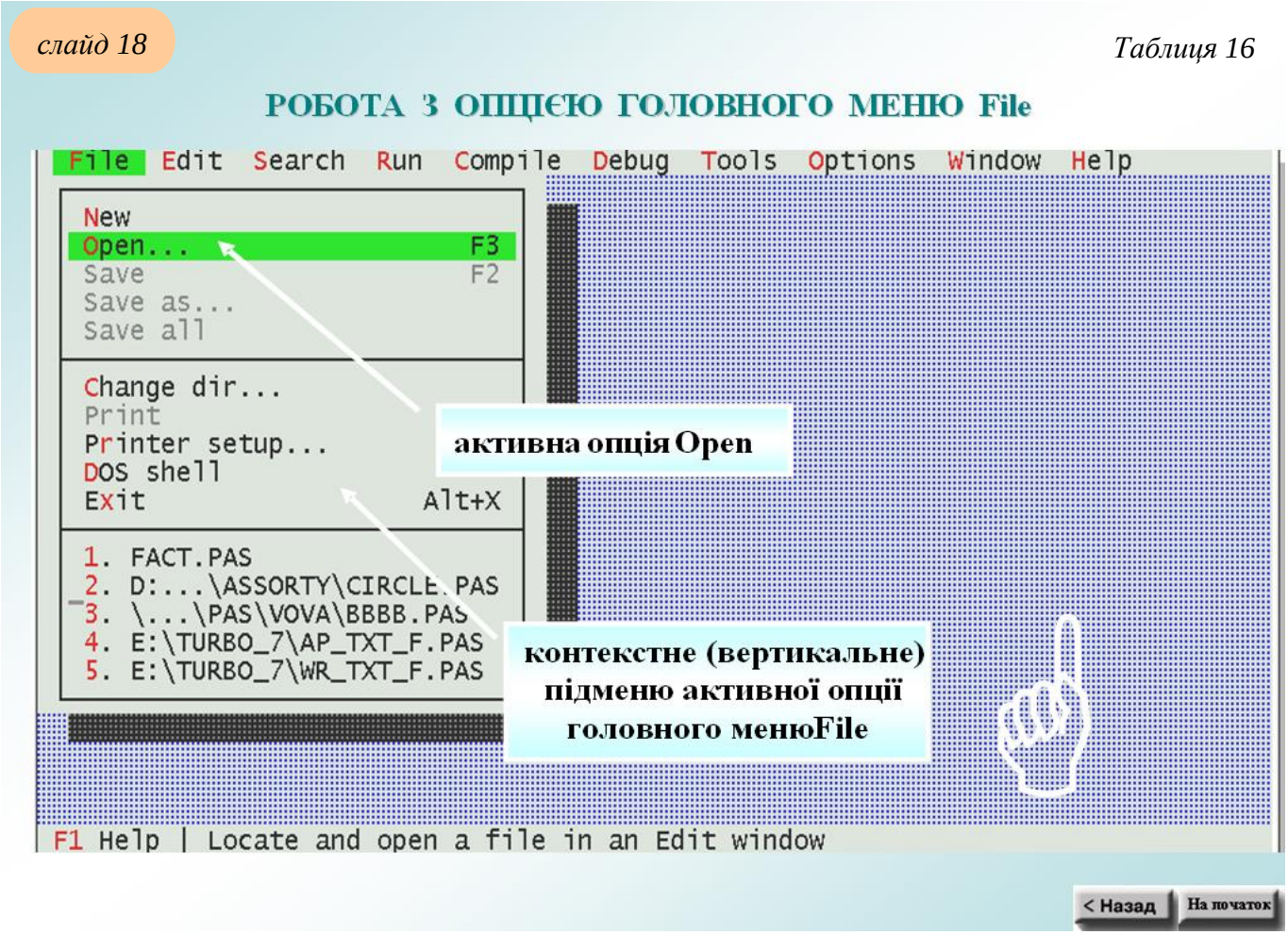


На початок

< Назад Вперед >

слайд 18

Таблиця 16



Слайди 14 – 18 призначені для пояснення системи вікон вмонтованого текстового редактора Турбо Паскаль, системи діалогових вікон інструментального середовища та їх взаємодію з режимами головного меню Турбо Паскаль.

Для появи контекстних пояснень слід використовувати кнопки зразка, ☝, ☞, ☞ та ☜ а для вказування на елементи управління вікнами застосовані графічні стрілки, які зміною кольору та трикратним миганням встановлюють візуальний зв'язок між цими елементами та відповідними їм текстовими поясненнями.

Третя група слайдів та відповідна їм серія таблиць мають дуже важливе значення для уміння правильно писати програми. Ця група допоможе уявити "архітектуру" програми на мові Турбо Паскаль, розрізняти її структурні особливості, привчить до пунктуальності і чіткості, допоможе уявити "правила хорошого тону" в програмуванні.

Поряд з описаними вище опорними анімаційними сигналами (мигаюча стрілка-вказівник, що активізується кликанням миші по коментарю та активізація коментарів кнопками-вказівниками) у третій цій групі застосовано анімаційне само виділення блоків з метою концентрації уваги на них (слайд 21), одночасна концентрація уваги на кількох блоках миганням потрійної стрілки (слайд 22).

Слайди 20-21 унаочнюють пояснення структури програми та блоків оголошення і опису величин, вводу/виводу і опрацювання інформації. Слайд 22 ознайомлює із застосуванням бібліотечних модулів на прикладі модуля CRT (управління текстовим екраном). Центральне місце в серії займають слайди 23-26,

які висвітлюють опис і виклик підпрограм. Слайди 27-28 присвячені модулям користувача.

слайд 19

Програми, модулі

- 1. Структура програми на ТП
- 2. Вигляд програми на ТП7
- 3. Шаблон програми з використанням модуля
- 4. Опис процедури
- 5. Опис функції
- 6. Виклик процедури
- 7. Виклик функції
- 8. Структура модуля
- 9. Вигляд модуля

< До змісту>

слайд 20

таблиця 3.1

Структура програми на Turbo Pascal

[program ***ім'я***;]

[uses ***імена модулів*** *(через кому)*;]

[const ***ім'я1 = значення; ім'я2 = значення; . .***]

[type ***опис нестандартних типів***]

[var ***ім'я1 : тип; ім'я1 : тип***;] *(однотипні – через кому)*

[***блок опису підпрограм***]

блок оголошень та описів

begin

[***тіло програми***]

end.

текст, взятий в квадратні дужки, може бути відсутній

На початок < Назад Вперед >

слайд 21

таблиця 3.2

Вигляд програми на Turbo Pascal 7

(з основними компонентами та коментарями)

```
program P_S; {визначення Pпр та Sпр за сторонами}
  var a,b,P,S: real;          {блок оголошення та опису}
begin
  WriteLn('ввести a ');ReadLn(a);   {блок вводу}
  WriteLn('ввести b ');ReadLn(b);
  P:=2*(a+b);S:=a*b;          {блок опрацювання}
  WriteLn('P=',P:8:2, 'S=', S:8:2);  {блок виводу}
end.
```

Синім кольором виділено формат виводу результатів: з фіксованою десятковою точкою, по 8 знаків, з них по 2 після десяткової точки

слайд 22

таблиця 3.3

Шаблон програми з використанням модуля

```
program  {…};
  uses CRT;
 {…}
begin
  ClrScr;
 {…}
  ReadKey
end.
```

{…} – майбутні фрагменти тексту програми

Синім кольором виділено оголошення модуля CRT (управління текстовим екраном), процедура ClrScr (очистки екрану), процедура ReadKey (введення символа з клавіатури, тут використано для блокування вікна програми до натискування клавіші).

ClrScr та ReadKey можна використовувати в тілі програми при потребі

слайд 23

таблиця 3.4

ОПИС ПРОЦЕДУРИ

```
procedure Max2( x, y:real ; var z:real );
```

ІМ'Я

ПАРАМЕТРИ-ЗНАЧЕННЯ

ПАРАМЕТРИ-ЗМІННІ

```
[блок опису локальних змінних]
begin
   if x >= y then z:=x
   else z:=y
end;
```

В описі процедури є заголовок, блок опису локальних змінних та тіло процедури.
В заголовку після *імені* в дужках описано ***параметри***. Опис ***параметрів -змінних*** починається словом *var*.

На початок

< Назад Вперед >

слайд 24

таблиця 3.5

ОПИС ФУНКЦІЇ

```
function Max2( x, y : real ) : real ;
```

ІМ'Я

ПАРАМЕТРИ

ТИП ФУНКЦІЇ

```
[блок опису локальних змінних]
begin
   if x >= y then
        Max2 :=x
   else Max2 :=y
end;
```

В описі функції є заголовок, блок опису локальних змінних та тіло функції.
В заголовку після *імені* в дужках описано ***параметри***, а за дужками описано тип результату. ***Ім'я результату співпадає з іменем функції***, тому в тілі функції воно (виділено синім кольором) використовується в якості результату.

На початок

< Назад Вперед >

слайд 25

таблиця 3.6

Виклик процедури

```
program Max3;
 var a, b, c, max : real;
procedure Max2(x, y:real;var z:real);
  begin
    if x >= y then z:=x else z:=y
  end;
begin
     Writeln('ввести a:'); Read(a);
   Writeln('ввести b:'); Read(b);
   Writeln('ввести c:'); Read(c);
   Max2 (a, b, max) ;
   Max2 (max, c, max);
    Writeln(max :6:2);
end.
```

Процедура – це підпрограма, допоміжний алгоритм.
Має декілька (або один) формальних параметрів та декілька результатів.
В даному випадку формальними є параметри *x, y;* а результатом є *z*.

Параметри, описані в заголовку – формальні. У команді виклику процедури їм відповідають фактичні параметри. Аргументи виділено синім, а результати – червоним кольорами

На початок | < Назад | Вперед >

слайд 26

таблиця 3.7

Виклик функції

```
program Max3;
 var a, b, c, max : real;
 function Max2( x, y : real) :  real;
begin
 if x >= y then Max2:=x else Max2 := y
end;
begin
   Writeln('ввести a,b,c:');
   Read(a); Read(b); Read(c);
   max : = Max2 (a, b) ;
   max : = Max2 (max, c);
    Writeln(max :6:2);
end.
```

Функція – це підпрограма, допоміжний алгоритм.
Має декілька (або один) формальних параметрів.
Суттєвою відмінністю її від процедури є те, що функція має лише один результат – *ім'я функції.*

Параметри, описані в заголовку – формальні. У команді виклику функції їм відповідають фактичні параметри. Аргументи виділено синім, а результати червоним кольорами. Інший варіант команди виклику:

max **:= Max2 (Max2 (*a, b*), *c*)**

На початок | < Назад | Вперед >

слайд 27

таблиця 3.8

СТРУКТУРА МОДУЛЯ

unit ім'я; {*заголовок модуля*}

interface {*інтерфейсна частина*}
***uses** заголовки доступних модулів*
оголошення доступних констант та змінних
заголовки доступних процедур та функцій

implementation {*реалізаційна частина*}
оголошення прихованих констант та змінних
тексти доступних і прихованих підпрограм

[begin
*блок ініціалізації модуля***]**

текст, взятий в квадратні дужки, може бути відсутній

end.

На початок
< Назад Вперед >

слайд 28

таблиця 3.9

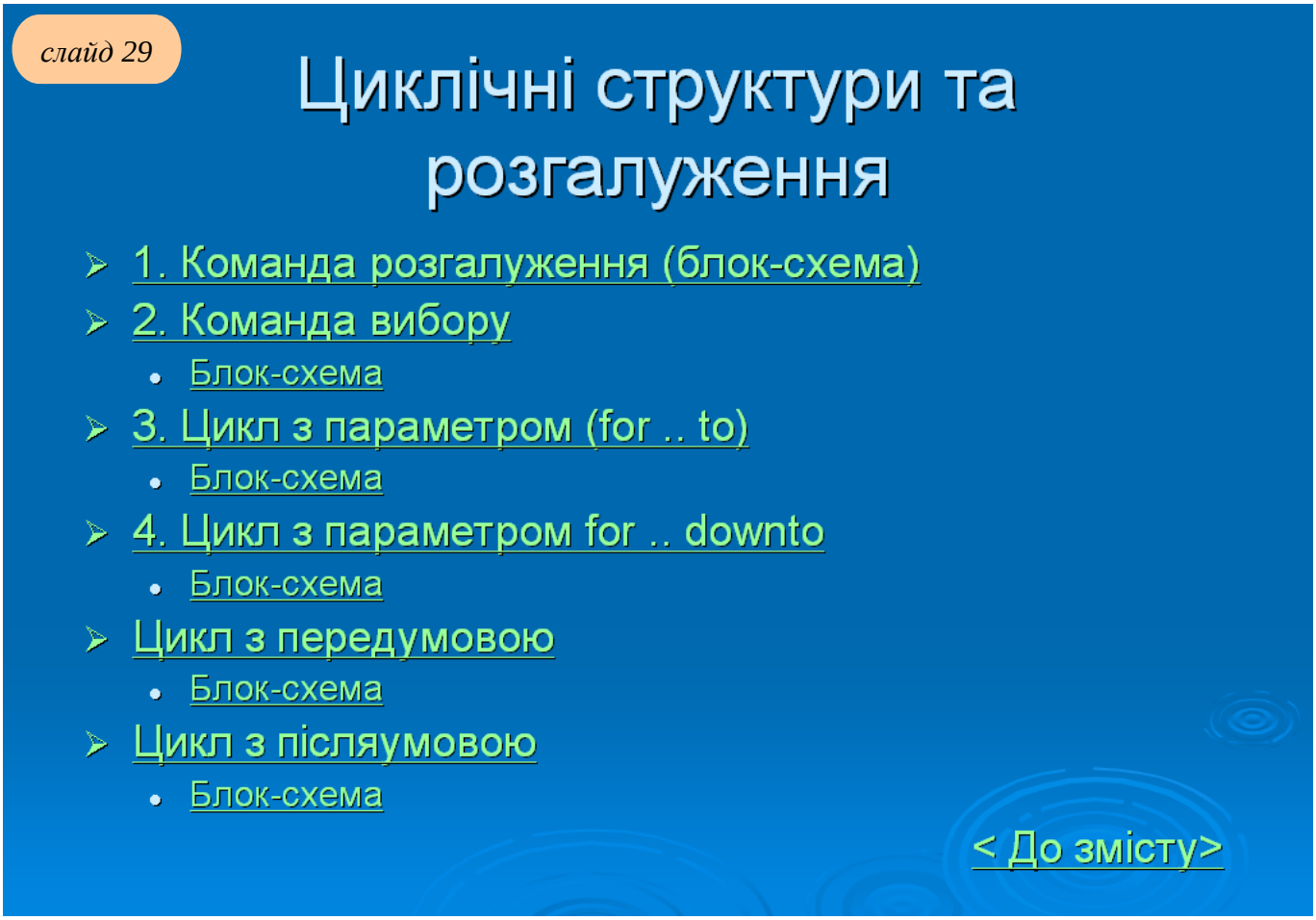
ВИГЛЯД МОДУЛЯ

```
unit MainMod;
interface
  function Max2( x, y : real) :   real;
implementation
 function Max2( x, y : real) :   real;
 begin if x >= y then Max2:=x else Max2 := y end;
end.
```

В заголовках процедур і функцій імплементаційної частини можуть не описуватись параметри (виділено сірим кольором), адже вони вже описані в інтерфейсній частині.
Блок ініціалізації модуля може бути відсутній.

< Назад На початок

Серія "Складені команди" (таблиці 4.1 – 4.11, слайди 29 – 40) займає особливе місце в ДДС ТП. Тут розглядаються структури розгалуження та цикли, які мають вирішальне значення в структурному програмуванні, тому слід забезпечити розуміння формату (загального запису), правила виконання цих команд, та їх блок-схеми, які допомагають розуміти правила виконання. В процесі вивчення команд розгалуження та повторення програмісти-початківці вперше стикаються з логічними виразами. Тому перед розробниками таблиць стоїть нелегке завдання вибрати ефективні і в той же час лаконічні засоби унаочнення команд. Представлення таблиць у вигляді слайдів презентації з анімаційними ефектами дещо полегшують це завдання. Ми намагались анімаційними засобами відобразити процес прийняття рішень залежно від умов.

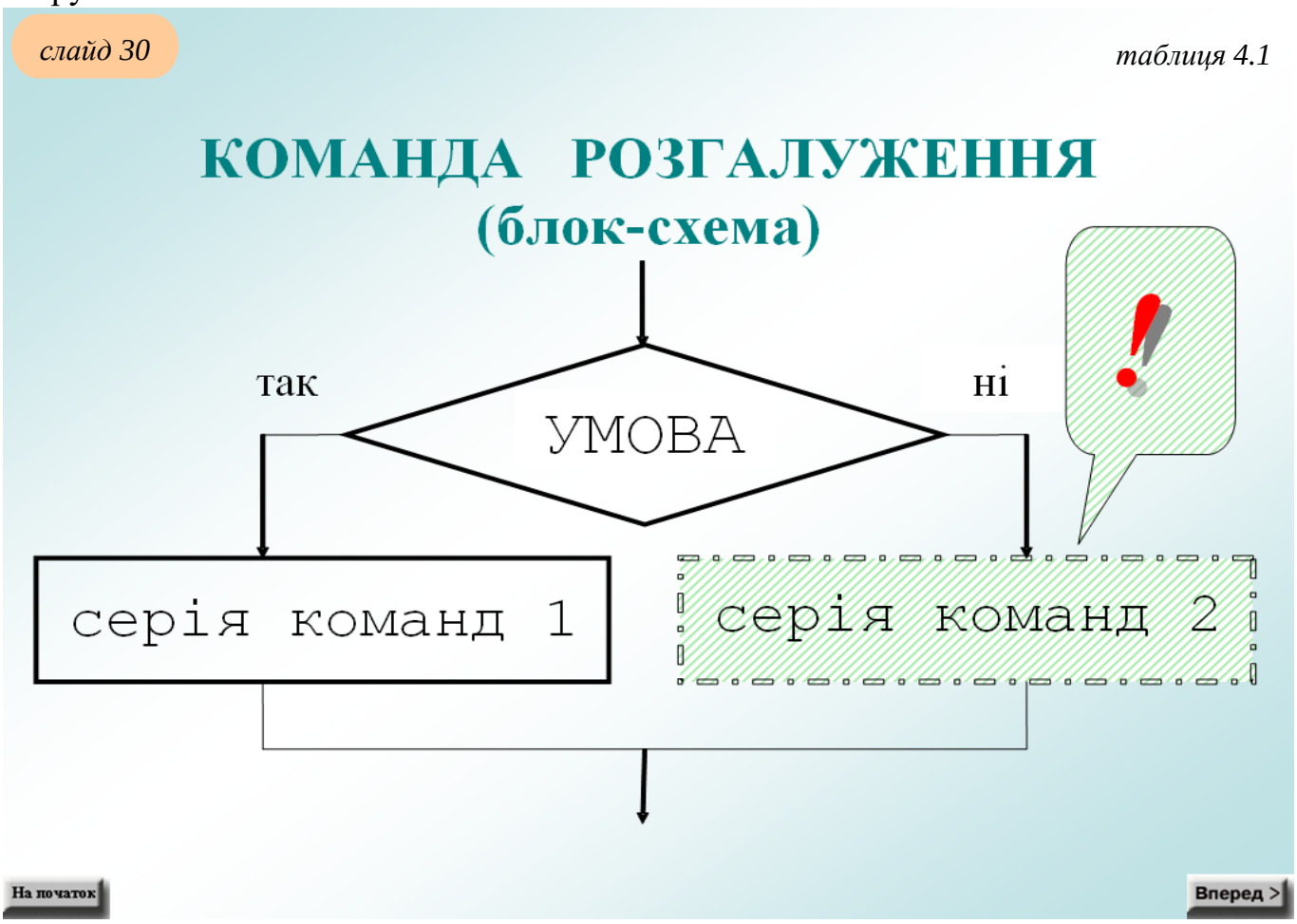


Як видно з меню, кожній команді відповідають два слайди, один із загальним виглядом команди, а інший із її блок-схемою. Основне смислове навантаження було покладено на слайди з блок-схемами. З допомогою перемикачів удалось досягти дослідження покрокового виконання команд, причому забезпечена можливість багаторазового повторення перевірки умов та прийняття рішень в залежності від їх істинності. Серія складається із 11 таблиць. Розглянуто повну і скорочену форми команди розгалуження, команда вибору та всі види циклів, включаючи цикли з параметром FOR … TO і FOR … DOWNTO. Таблиці із загальним записом команд розгалуження відсутні через їх короткість та подібність команді вибору. Ми

впевнені, що при використанні таблиць учитель легко може подолати таку незручність.

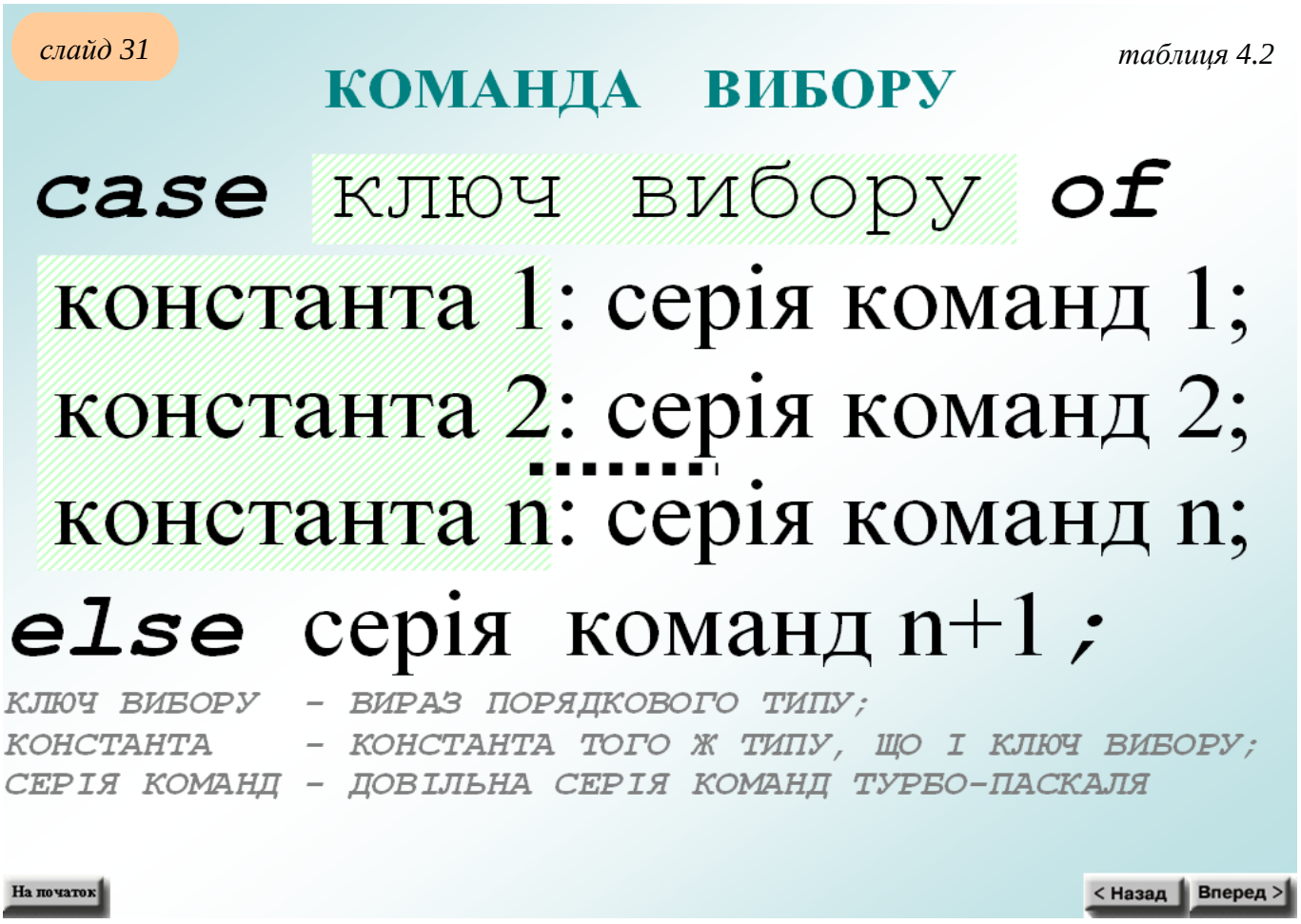
слайд 30

таблиця 4.1



слайд 31

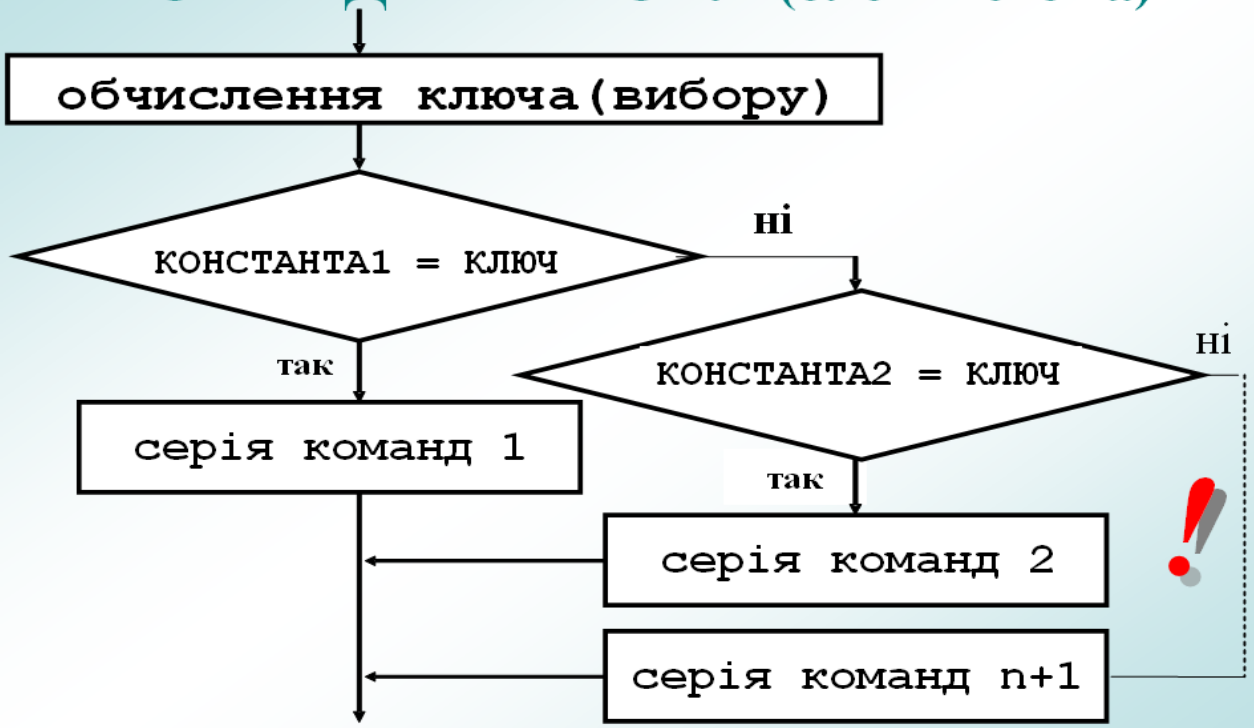
таблиця 4.2



слайд 32

таблиця 4.3

КОМАНДА ВИБОРУ (блок - схема)



На початок
< Назад
Вперед >

слайд 33

таблиця 4.4

ЦИКЛ З ПАРАМЕТРОМ (FOR .. TO)

for i:=i_{min} **to** i_{max} **do**

[**begin**]
тіло цикла (серія команд)
[**end**];

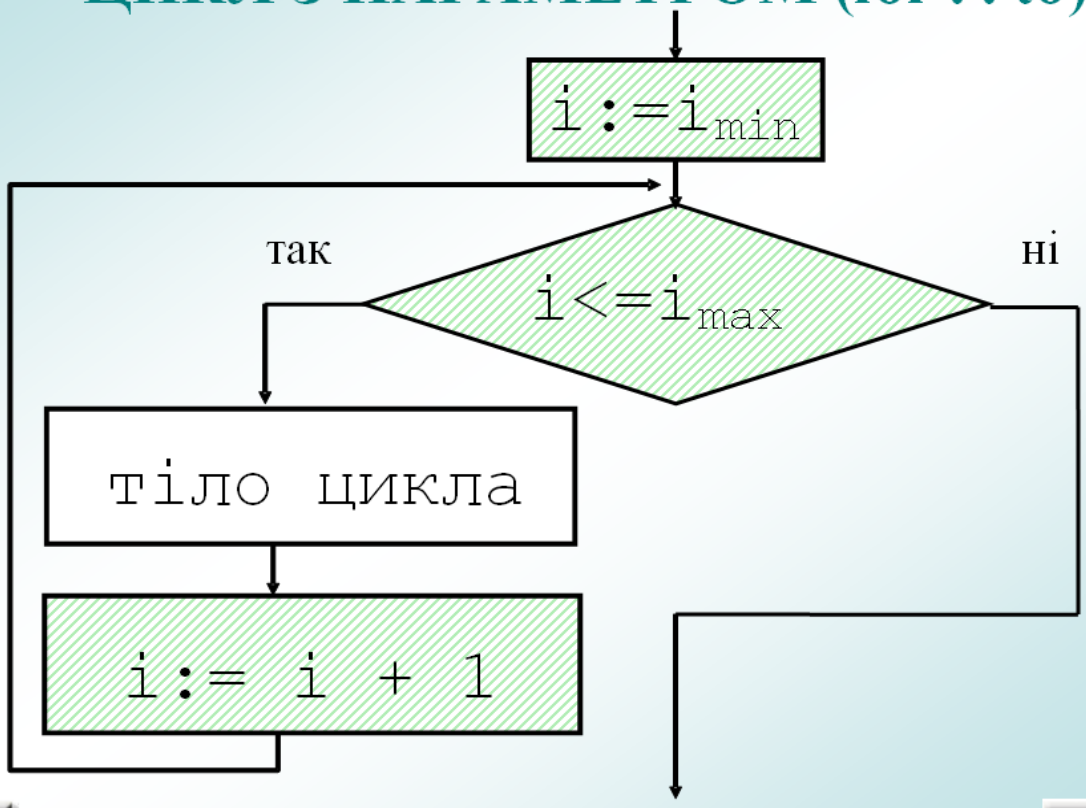
i - параметр цикла
i_{min} - початкове значення параметра
i_{max} - кінцеве значення параметра

На початок
< Назад
Вперед >

слайд 34

таблиця 4.5

ЦИКЛ З ПАРАМЕТРОМ (for . . to)



На початок

< Назад Вперед >

слайд 35

таблиця 4.6

ЦИКЛ З ПАРАМЕТРОМ (FOR .. DOWNTO)

for i:=i_{max} **downto** i_{min} **do**

! [**begin**]
тіло цикла (серія команд)
[**end**];

i - параметр цикла
i_{max} - початкове значення параметра
i_{min} - кінцеве значення параметра

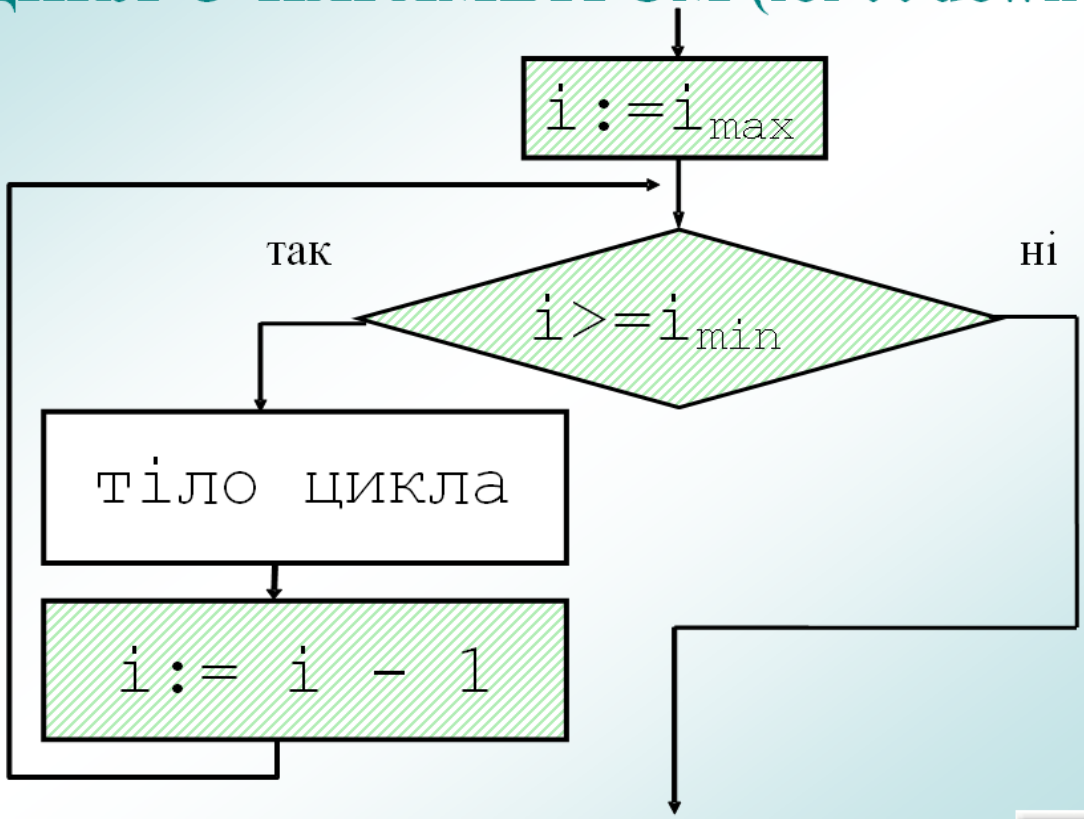
На початок

< Назад Вперед >

слайд 36

таблиця 4.7

ЦИКЛ З ПАРАМЕТРОМ (for . . downto)



На початок
< Назад
Вперед >

слайд 37

таблиця 4.8

ЦИКЛ З ПЕРЕДУМОВОЮ

```
while умова do
  [begin]
    тіло цикла
  [end];
```

умова - це твердження (логічний вираз), яке може приймати одне з двох значень:

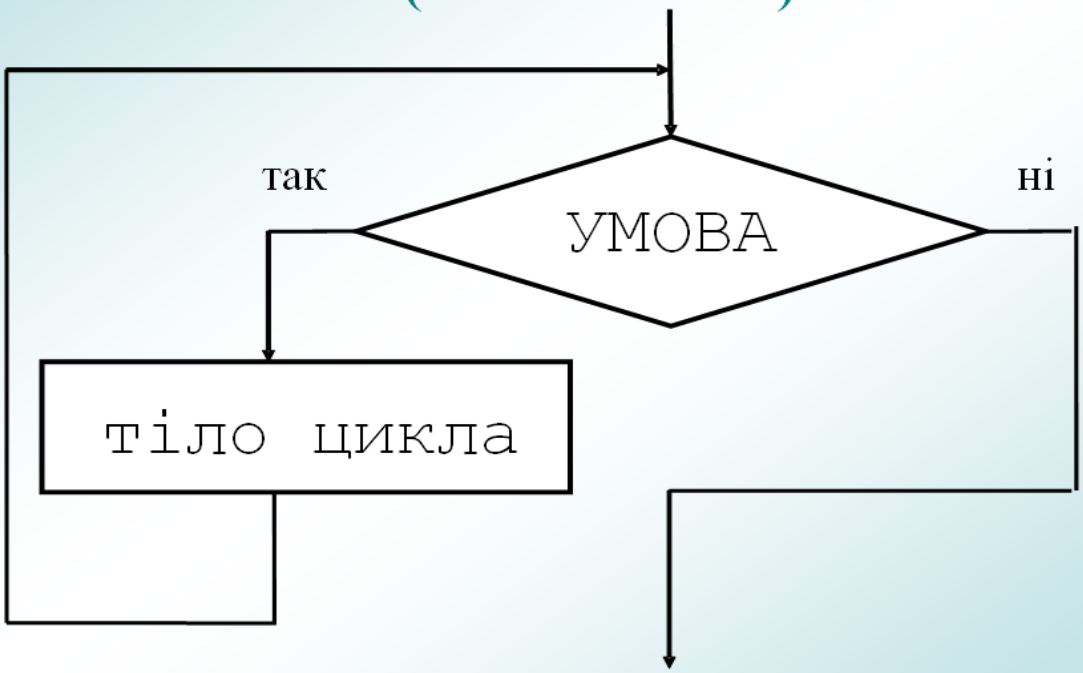
false та true

На початок
< Назад
Вперед >

слайд 38

таблиця 4.9

ЦИКЛ З ПЕРЕДУМОВОЮ
(блок - схема)



На початок
< Назад
Вперед >

слайд 39

таблиця 4.10

ЦИКЛ З ПІСЛЯУМОВОЮ

```
repeat
    тіло цикла
until умова;
```

умова - це твердження (логічний вираз), яке може приймати одне з двох значень:

false та true

На початок
< Назад
Вперед >

слайд 40

таблиця 4.11

П’ята серія команд тісно пов’язана зі складеними командами, до яких належать розгалуження, вибір та циклічні команди. Її місце у ДДС ТП центральне, адже ця серія допомагає зробити ″безболісний″ перехід до масивів та інших структурованих типів величин. Серія найбільша по кількості таблиць, їх 13.

Особливе місце належить циклу з параметром. Таблиця 5.1 присвячена застосуванню цього цикла на прикладі алгоритма визначення суми чисел.

Будь-яку команду можна розглядати, як автомат, що перетворює вхідні дані у результати.

малюнок 1

Циклічні команди можна розглядати у якості влучної та ефектної ілюстрації цього, адже в них не так прозоро, як у команді присвоєння видно процес перетворення даних у результати, а також циклічне виконання різних операцій (команд) принципово схоже на роботу верстата або автомата. На це бажано звертати увагу з дидактичних міркувань. Пропоновані нижче різноманітні цикли, їх взаємозаміна і вкладення проілюструють дію кількох видів таких інтелектуальних автоматів, що сприятиме розвитку в учнів не тільки навичок програмування, а й

підвищить загальну культуру абстрактного, логічного, алгоритмічного та структурного мислення.

слайд 41

Застосування складених команд

- 1. Програма визначення суми n чисел
- 2. Заміна циклу *For…* циклом *While*
- 3. Заміна циклу *For* циклом *repeat…until*
- 4. Програма визначення НСД
- 5. Програма перевірки простоти числа
- 6. Вкладені цикли з параметром (*For-For*)
- 7. Програма виводу таблиці Піфагора

Вкладені цикли:

- *For - While*
- *While - For*
- *For - Repeat*
- *Repeat - For*
- *While - Repeat*
- *Repeat - While*

< До змісту>

слайд 42

таблиця 5.1

Програма визначення суми *n* введених чисел

Визначення суми n чисел, що вводяться. Команди вводу - червоним кольором, визначення суми (s) - синім кольором

```
program Sum;
  uses Crt;
  var i, n : integer;
      x, s : real;
begin
  ClrScr;
  WriteLn('n ?');
  ReadLn(n);
  s := 0;
  for i:= 1 to n do          begin
    ReadLn(x); s := s + x    end;
  WriteLn(' s= ', s :8 :2);
  ReadKey
end.
```

записане таким кольором - НЕ ОБОВ'ЯЗКОВЕ !

На початок < Назад Вперед >

слайд 43

таблиця 5.2

ЗАМІНА ЦИКЛУ *FOR* ЦИКЛОМ З ПЕРЕДУМОВОЮ (*WHILE*)

```
i:=imin;
while i<=imax do
  begin
    тіло цикла; i:=i+1
  end;
```

На початок | < Назад | Вперед >

слайд 44

таблиця 5.3

ЗАМІНА ЦИКЛУ *FOR* ЦИКЛОМ З ПІСЛЯУМОВОЮ (*REPEAT ... UNTIL*)

```
i:=imin;
repeat
  тіло цикла; i:=i+1
until i>imax;
```

На початок | < Назад | Вперед >

слайд 45

таблиця 5.4

ПРОГРАМА ВИЗНАЧЕННЯ *НСД(a, b)*

записане таким кольором – НЕ ОБОВ'ЯЗКОВЕ !

```
program  EVC;
  uses Crt;
  var a, b, a1, b1, nsd : integer;
begin
  ClrScr;
  WriteLn('a, b?'); ReadLn(a, b);
  a1:=a; b1:=b;
  while a1 <> b1 do
    if  a1 > b1 then a1 := a1 - b1
    else             b1 := b1 - a1;
  nsd := a1;
  WriteLn('нсд(', a,',',b,'=',nsd);
  ReadKey
end.
```

алгоритм Евкліда

Прослідкуйте за вели чинами, виділеними синім та червоним кольором !

На початок | < Назад | Вперед >

слайд 46

таблиця 5.5

Програма перевірки простоти натурального числа *N*

```
program  PROST;
  var N, i, l : integer;
      x : boolean;
begin
  WriteLn('N?'); ReadLn(N);
  x := true; i := 2;
  l := trunc ( sqrt (N) );
  while ( i <= l ) and x do  begin
    if  N mod i = 0 then  begin
      x := false; i := l  end ;
  i := i + 1                  end;
  WriteLn(x);
end.
```

В умові **(i <= l) and x** використовується біжуче значення x = = true (видно із виділеного червоним кольором) !

На початок | < Назад | Вперед >

слайд 47

таблиця 5.6

ВКЛАДЕНІ ЦИКЛИ З ПАРАМЕТРОМ (FOR – FOR)

```
for i:=imin to imax do
  for j:=jmin to jmax do
      тіло цикла ;
```

На початок
< Назад Вперед >

слайд 48

таблиця 5.7

ПРОГРАМА ВИВОДУ ТАБЛИЦІ ПІФАГОРА

```
program PIFAGOR;
  uses Crt;
  var i, j : integer;
begin
  ClrScr;
  for i := 2 to 9 do
    for j := 2 to 9 do        begin
      GoToXY(i*3 + 10, j);
      Write( i*j : 2)         end;

  ReadKey
end.
```

форматований вивід елементів таблиці (кожен представляється, як двоцифрове число, вирівняне по правому краю)

Серед процедур модуля **Crt** *(виділено кольором) в програмі особливе місце має* **GoToXY (i*3 + 10, j)** *. Вона управляє виводом на екран елементів таблиці.*

На початок
< Назад Вперед >

слайд 49

таблиця 5.8

ВКЛАДЕНІ ЦИКЛИ (FOR – WHILE)

```
for i:=imin to imax do begin
  j:=jmin
  while j<=jmax do begin
        тіло цикла ;
        j:=j+1          end;
```

На початок

< Назад | Вперед >

слайд 50

таблиця 5.9

ВКЛАДЕНІ ЦИКЛИ (WHILE - FOR)

```
i:=imin
while i<=imax do begin
   for j:=jmin to jmax do
         тіло цикла ;
   i:=i+1
                    end;
```

На початок

< Назад | Вперед >

слайд 51

таблиця 5.10

ВКЛАДЕНІ ЦИКЛИ (FOR – REPEAT)

```
for i:=imin to imax do begin
    j:=jmin
    repeat
        тіло цикла ;
        j:=j+1
    until j>jmax
```

На початок
< Назад Вперед >

слайд 52

таблиця 5.11

ВКЛАДЕНІ ЦИКЛИ (REPEAT - FOR)

```
i:=imin;
repeat
    for j:=jmin to jmax
        тіло цикла ;
    i:=i+1
until i>imax;
```

На початок
< Назад Вперед >

слайд 53

таблиця 5.12

ВКЛАДЕНІ ЦИКЛИ *(WHILE - REPEAT)*

```
i:=imin
while i<=imax do begin
    j:=jmin
    repeat
      тіло цикла;j:=j+1
    until j>jmax
    i:=i+1
```

На початок < Назад Вперед >

слайд 54

таблиця 5.13

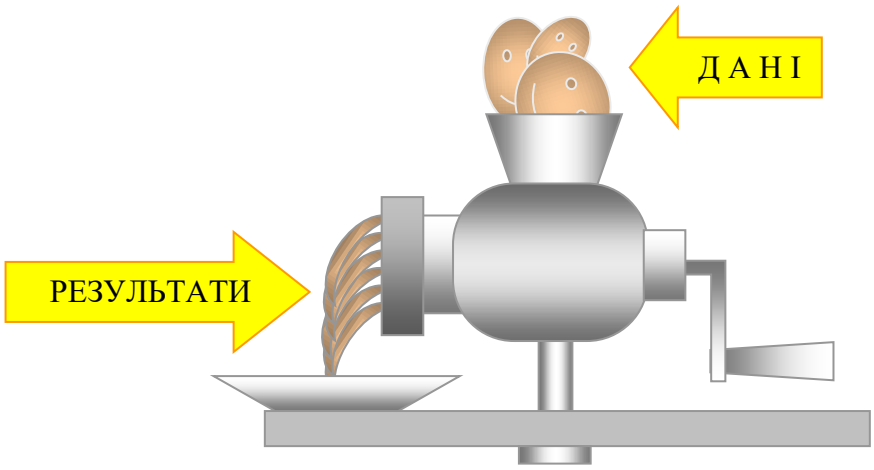
ВКЛАДЕНІ ЦИКЛИ (*REPEAT - WHILE*)

```
i:=imin;
repeat
  j:=jmin;
  while j<=jmax do    begin
    тіло цикла;j:=j+1end;
  i:=i+1
until i>imax;
```

На початок < Назад На початок

Логічно після серій таблиць про складені команди, їх застосування та підпрограми поставити серію "Опрацювання масивів". До серії входять 8 таблиць, перші дві з них 6.1 та 6.2 ілюструють нюанси оголошення, опису, введення масивів і передачу їх у якості параметрів у підпрограми. Наступну таблицю 6.3 про визначення найбільшого елемента лінійного масиву слід використати з метою формування уявлення про пошук, а таблиці 6.4 та 6.5 ілюструють пошук елемента за вказаною властивістю в неупорядкованому та упорядкованому лінійних масивах. Звернувши увагу на тому, що шукати потрібний елемент значно легше в упорядкованому масиві, доцільно розглянути проблему сортування лінійного масиву, для чого згодиться таблиця 6.6. Нарешті заключні таблиці серії 6.7 та 6.8 ілюструють один із ефективних методів сортування – метод Шелла. Звичайно, програми пошуку та сортування після розгляду питання про модулі користувача слід представляти у вигляді процедур, а при можливості, краще у вигляді функцій.

Це дасть можливість розвинути описану вище ідею аналітичного автомату (див. малюнок 1), який призначений для перетворення даних, що мають певні характеристики, в результати з потрібними характеристиками (малюнок 2):



малюнок 2

Зображена "м'ясорубка" є моделлю процедури або функції з одним аргументом і одним результатом. Можна підібрати приклад, де як у якості вхідних даних, так і у якості вихідних даних є по кілька інградієнтів. Тоді одержимо "чисту" модель процедури.

Очевидно, що наведені малюнок і приклад можна також представити окремою таблицею, яку буде доречно використати разом із таблицями 6.1, 6.2, 6.3, 6.6 в тому числі для пояснення понять параметрів-значень, параметрів-змінних, формальних та фактичних параметрів.

У таблицях серії 6 при потребі застосовано вже описані вище анімаційні ефекти та кнопки управління. Слід звернути увагу на використання фігурних стрілок у якості анімаційних вказівників.

При вивченні масивів ми звертаємо увагу на різні способи їх вводу: циклом з допомогою процедури READ, масиву-константи та з текстового файлу. Ілюстрації всього цього приділено чимало уваги і в пропонованих таблицях.

Пропоновані таблиці, як було зазначено вище, можна використовувати комплексно, у різних поєднаннях. Безперечно, за описаною технологією до комплекту можна додати інші таблиці, виготовлені користувачами цього комплекту.

слайд 55

Масиви

- Оголошення, опис та введення масивів
- Програма визначення найбільшого числа в лінійному масиві
- Передача масивів у підпрограми
- Пошук:
  - В неупорядкованому масиві
  - В упорядкованому масиві
- Програма сортування числового лінійного масиву
- Метод Шелла
  - Зміни в масиві під час сортування

< До змісту>

слайд 56

таблиця 6.1

Оголошення, опис та введення масивів

```
. . .
const  n=4;
  M1 : array[1..n] of 0..9 = (8,2,1,7);
   M2 : array[1..n,1..n] of 0..25 =(( 8,2,0,7 ),
                                    ( 8,2,1,7 ),
                                    ( 8,1,0,7 ),
                                    ( 2,1,0,7 ));
type M3 = array[1..n,1..n] of char;
var  M4 : array[1..n] of string;
     M5 : M3;
. . .
begin
    for i:=1 to n do                          begin
       WriteLn( 'ввести M4[' , i , ']:' );
       ReadLn(M4 [ i ])                       end ;
    for i:=1 to n do
       for  j:=1 to n do                      begin
          WriteLn( 'ввести M5[' , i , j , ']:' );
          ReadLn(M5 [ i, j])                  end;
. . .
```

На початок < Назад Вперед >

слайд 57

таблиця 6.2

Програма визначення найбільшого числа в лінійному масиві

```
program Max_n;
  type M = array[1..100] of real;
  var i,n:integer;max:real;Mas1:M;
  procedure R_M(N:byte;var Mas:M);
    var i:byte;
  begin
    for i:=1 to N do Read(Mas[i])
  end;
  function Max2(x,y:real):real;
  begin
    if x>=y then Max2:=x
    else Max2:=y
  end;
begin
  WriteLn('ввести n:');ReadLn(n);
  R_M(n,Mas1);
  max:=Mas1[1];
  for i:=2 to n do
    max := Max2(Mas1[i],max);
  WriteLn('max=',max:6:2);
```

Процедура зчитування елементів масиву з клавіатури.

Функція визначення найбільшого з двох чисел.

слайд 58

таблиця 6.3

ПЕРЕДАЧА МАСИВІВ У ПІДПРОГРАМИ

```
program Maxn;
  const n = 100;
  type Mas = array[1..n] of real;

  Var Mas1: Mas;
  Procedure Read_Mas (n:integer;var Mas2: Mas)
        var i: byte;
  Begin for i:=1 to n do Read(Mas2[i]); end;
begin
  . . .
  Read_Mas(n,Mas1);
  . . .
end.
```

Червоним кольором виділено правильну передачу масиву в процедуру ***Read_Mas****.*
Текст:
procedure ***Read_Mas(varMas2:array[1..n]of real);*** *– неправильний*

слайд 59

таблиця 6.4

ПОШУК В НЕУПОРЯДКОВАНОМУ МАСИВІ

```
program  Mas_Posh_Neupor;
  const n = 10;
  var i, x : integer; st : string;
      Mas : array [1..n] of integer;
begin
  st:='ні';
  for i := 1 to n do                    begin
      WriteLn ('Mas[',i,']:');
      Read ( Mas [i] )                    end;

  WriteLn ('ввести x:'); ReadLn(x);
  for i := 1 to n do
      if Mas[i] = x  then  st := 'так';
  WriteLn (st);
end.
```

Блок зчитування елементів масиву з клавіатури.

Блок перевірки на рівність кожного з елементів масиву з шуканим числом *x* (Пошук числа у масиві)

На початок < Назад Вперед >

слайд 60

таблиця 6.5

ПОШУК В УПОРЯДКОВАНОМУ МАСИВІ

```
Program Mas_Poshuk_Upor;
 const n = 10;
 M : array[1 .. n] of  0..100 =
        (3, 12, 18, 24, 29, 35, 39, 41, 49, 50);

 var i, i1, i2, i3, x : integer;
     str : string;
begin
 str := 'ні';
 Write ('ввести число');  Read (x);
 i1 := 1; i2 := n;
 while i2 - i1 > 0 do                    begin
    i3 := Trunc ( (i1+i2) / 2);
    if  x = M[i3] then        begin
       str := 'так';  i2 := i1 end
    else if x < M[i3] then i2 := i3
       else if (x > M[i3]) and (i1 <> i3) then
                  i1 := i3;
            else i1 := i2;
     if  x = M[i2] then str := 'так'          end;
 writeln ( str );end.
```

Масив, в якому здійснюватиметься пошук

Основний блок пошуку

На початок < Назад Вперед >

слайд 61

таблиця 6.6

Сортування числового лінійного масиву

```
program  Sort;
  const n = 10;
  type Mas = array[1 . .n] of integer;
  var i, j, k : integer;
     Mas1: Mas;
  procedure Read_Mas(var Mas2 : Mas);
  {...} end;
function Num_Min(var i1 : integer);
  {...} end;
begin
  Read_Mas(Mas1);
  for i  := 1 to n - 1 do                    begin
    k := Mas1[ i ];
    Mas1[ i ] := Mas1[ Num_Min(i + 1)];
    Mas1[ Num_Min ( i  + 1) ] :=k
                                             end;
  for i := 1 to n do WriteLn(Mas1[ i ]);
end.
```

тіло процедури вводу масиву

тіло функції визначення номера найменшого елемента "хвоста"

обмін

На початок
< Назад Вперед >

слайд 63

таблиця 6.7

Програма сортування числового масиву методом Шелла

```
program Sort_Shell;
   const n = 10;
         mas: array[1..n] of integer = (235, 430, 8, 2, 3, 4, 65, -12, 3,1);
   var f, i, j, p, m, d: integer;
begin
    m := n div 2;
   while m >= 1 do                                          begin
    for i: = m+1 to n do                                 begin
      p := mas[i]; f:=0; j:=i - m;
      while (j > = 1) and (f = 0) do
        if p <= mas[ j ] then                        begin
          mas[ j + m ] := mas[ j ]; j := j - m  end
        else f := 1; mas[ j + m ] := p                end;

    m := m div 2                                             end;
    for i := 1 to n do write ( mas[ i ]: 5 );
end.
```

На початок
< Назад Вперед >

слайд 64

таблиця 6.8

ЗМІНИ В МАСИВІ ПД ЧАС СОРТУВАННЯ ЧИСЛОВОГО МАСИВУ МЕТОДОМ ШЕЛЛА

	1	2	3	4	5	6	7	8	9	10
1	**33**	23	76	8	**15**	22	9	65	91	67
2	15	**23**	76	8	33	**22**	9	65	91	67
3	15	22	**76**	8	33	23	**9**	65	91	67
4	15	22	9	**8**	33	23	76	**65**	91	67
5	15	22	9	8	**33**	23	76	65	**91**	67
6	15	22	9	8	33	**23**	76	65	91	**67**
7	**15**	**22**	9	8	33	23	76	65	91	67
8	15	**22**	**9**	8	33	23	76	65	91	67
9	**15**	**9**	22	8	33	23	76	65	91	67
10	9	15	**22**	**8**	33	23	76	65	91	67
11	9	**15**	**8**	22	33	23	76	65	91	67
12	**9**	**8**	15	22	33	23	76	65	91	67
13	8	9	15	**22**	**33**	23	76	65	91	67
14	8	9	15	22	**33**	**23**	76	65	91	67
15	8	9	15	22	23	**33**	**76**	65	91	67
16	8	9	15	22	23	33	**76**	**65**	91	67
17	8	9	15	22	23	33	65	**76**	**91**	67
18	8	9	15	22	23	33	65	76	**91**	**67**
19	8	9	15	22	23	33	65	**76**	**67**	91
20	8	9	15	22	23	33	65	67	76	91

< Назад | На початок

Остання серія таблиць "Додаткові таблиці" не має свого меню. Таблиці охоплюють різні додаткові питання: від опрацювання рядків, управління текстовим та графічним екранами (таблиці 7.1 – 7.3) та програмне використання клавіш управління (на прикладі клавіш управління курсором - таблиця 7.4) до програм роботи із текстовими файлами (таблиці 7.5 – 7.8).

Окремо стоять ще дві додаткові таблиці. Це таблиці 7.9 – "Додавання довгих чисел" та 7.10 – "Числа Фібоначчі".

Дизайн майже всіх таблиць витримано в одному стилі, ліву половину слайду займає довідкова таблиця, в правій може з'являтися у вигляді титрів або блоків, що "виїжджають справа", пояснення. Якщо в правій частині слайда відсутній текст або пояснення, то вона видалена. Це дало змогу розмістити чотири слайди на одній сторінці.

Ця серія таблиць може служити своєрідною "скарбничкою", яка допоможе створювати програми, що виходять за межі діяльності учня на уроках інформатики. Ідеї та прийоми, реалізовані в таблицях 7.1 – 7.10, допоможуть учням, які додатково цікавляться програмуванням та хочуть брати участь у підготовці олімпіад з інформатики.

слайд 65

таблиця 7.1

ПРОЦЕДУРИ І ФУНКЦІЇ ОПРАЦЮВАННЯ РЯДКІВ

Delete(St, Poz, N) - *видаляє N символів рядка St, починаючи з позиції Poz.*

Insert(St1, St2, Poz) - *вставляє рядок St1 у рядок St2, починаючи з позиції Poz.*

Str(N, St) - *перетворює число N у рядок St.*

Copy(St, Poz, N) - *виділяє з St підрядок довжиною N символів, починаючи з позиції Poz. Poz, N - цілочисельні вирази.*

Concat(St1, St2, ..., St) - *виконує зчеплення рядків St1, ..., St у тому порядку, у якому вони зазначені в списку параметрів. Результуючий рядок не повинен перевищувати 255 символів.*

Length(St) – *обчислює кількість символів рядка St.*

Pos(St1, St2) - *виявляє першу появу в рядку St2 підрядка St1. Якщо такого рядка не виявлено, повертається 0.*

В ТП 7.0 існує тип даних **STRING** (рядок), спеціально призначений для обробки рядків (ланцюгів символів). Він не відноситься до простих типів даних і займає проміжне місце між простими та структурованими типами даних.

Змінна типу **STRING** складається з ланцюга сиволів, тобто елементів типу **CHAR.** Рядки можуть виводитися на екран монітора використанням стандартних процедур **Write** та **WriteLn** і вводитися з допомогою стандартної процедури **ReadLn** чи **Read.** В більшості випадків змінні типу **STRING** використовуються для зберігання слів та повідомлень, що складаються з декількох символів.

слайд 66

таблиця 7.2

СТАНДАРТНІ ПРОЦЕДУРИ І ФУНКЦІЇ УПРАВЛІННЯ ТЕКСТОВИМ ЕКРАНОМ

Uses CRT – *оголошення модуля управління текстовим екраном.*

ClrScr – *очистка екрану.*

ReadKey – *зчитування символу з клавіатури.*

GoToXY(X, Y) – *встановлення курсора у позицію з координатами X (номер стовпчика) та Y (номер рядка).*

TextColor (N) – *встановлення кольору символів за номером N (0≤N≤15) При N>15 здійснюється мигання символів.*

TextBackGround (N) – *встановлення кольору знакомісць символів за номером N (0≤N≤7).*

Delay (X) – *зупинка виконання програми на час X в мілісекундах.*

KeyPressed – *встановлення стану клавіші (натиснута - True, не натиснута - False)*

Модуль *Crt* містить ряд підпрограм, що надають можливість програмам, що працюють під DOS, ефективно керувати такими характеристиками ПК, як режими екрана, розширені коди клавіатури, кольори, вікна та звукові сигнали.

Однією з основних переваг використання модуля *Crt* є підвищення швидкості і гнучкості під час виконання операцій роботи з екраном. Програми, що не працюють з модулем *Crt*, виводять на екран інфорацію з допомогою засобів ОС DOS, що пов'язано з додатковими операціями, котрі уповільнюють вивід. При використанні модуля *Crt*, інформація, що виводиться, посилається до базової системи вводу-виводу (BIOS) або, для ще більшого прискорення операцій, безпосередньо у відеопам'ять.

слайд 67

таблиця 7.3

Стандартні процедури і функції управління графічним екраном

InitGraph(dr,reg,'f') – *ініціалізація графічного режиму екрану.*

CloseGraph – *вихід з графічного режиму екрану.*

PutPixel(X,Y,C) – *зображення точки на екрані з координатами (X,Y), кольору C.*

Line(X_1,Y_1,X_2,Y_2) – *зображення відрізка з координатами кінців (X_1,Y_1) та (X_2,Y_2).*

Rectandle(X_1,Y_1,X_2,Y_2) – *зображення прямокутника з координатами кінців діагоналі (X_1,Y_1) та (X_2,Y_2).*

DrawPoly(N,P) – *зображення многокутника з кількістю вершин кінців N та P – набором записів, де вказані координати вершин.*

Circle(X,Y,R) – *зображення кола з координатами кінців (X,Y) та радіусом R.*

MoveTo(X,Y) – *переміщення курсора в точку з координатами (X,Y).*

Усі стандартні процедури і функції управління графічним екраном містяться у модуль Graph. Для застосування графічних можливостей необхідно ініціювати графіку, за що відповідає процедура InitGraph(dr, reg, 'f'), де dr – графічний драйвер, reg – графічний режим, 'f' – шлях до модуля Garph.

слайд 68

таблиця 7.4

ПРОГРАМНЕ ВИКОРИСТАННЯ КЛАВІШ УПРАВЛІННЯ

```
...
  var kod : integer;
...
begin
 ...
  repeat
    kod := Ord (ReadKey);
    if kod = 0 then
      kod := Ord (ReadKey);
    case kod of
      77 : writeln('вправо');
      75 : writeln('вліво');
      80 : writeln('вниз');
      72 : writeln('вгору')   end
  until kod = 13 ;
end.
```

Зчитування клавіші та визначення її коду (для управляючих клавіш це робиться в два етапи)

коди клавіш :
enter - 13
← - 75; ↑ - 72;
→ - 77; ↓ - 80

слайд 69

таблиця 7.5

СТАНДАРТНІ ПРОЦЕДУРИ І ФУНКЦІЇ РОБОТИ З ТЕКСТОВИМИ ФАЙЛАМИ

Assign(varf,'file') – *зв'язування файлової змінної **varf** із файлом **file**.*

Close(varf) – *закриття файлу, зв'язаного із файловою змінною **varf**.*

Rewrite(varf) – *відкриття файлу, зв'язаного із змінною **varf** для запису.*

Reset(varf) – *відкриття файлу, зв'язаного із змінною **varf**, для читання даних з нього.*

Append(varf) – *відкриття файлу, зв'язаного із змінною **varf**, для дописування в його кінець.*

Read(varf,*a*) – *читання даних із файла, зв'язаного із змінною **varf**, у змінну **a**.*

Write(varf,*a*) – *запис даних із змінної **a**, у файл, зв'язаний із змінною **varf**.*

Eof(varf) – *функція аналізу визначення кінця файла.*

EoLn(varf) – *функція аналізу визначення кінця рядка.*

слайд

таблиця 7.6

ПРОГРАМА СТВОРЕННЯ ТЕКСТОВОГО ФАЙЛУ

```
program Rew_File;
var fp : text;
    i : integer;
    Mas : array [1..10] of integer;
begin
  Assign (fp, 'ft . txt');
  Rewrite ( fp );
  for i:=1 to 10 do begin
    Read(Mas[ i ]);
    Write(fp, Mas [ i ],' ') end;
  Close(fp);
end.
```

Файлова змінна повинна мати тип **text**.

Якщо в процедурі Assign не вказано шлях до файла ft.txt, то він повинен бути в поточному каталозі

Процедура **Write (fp,Mas[i],' ')** передає щойно введене i-те значення масива Mas[i] у файл ft . txt .

Закривати файл обов'язково

слайд

таблиця 7.7

ПРОГРАМА ДОПОВНЕННЯ ТЕКСТОВОГО ФАЙЛУ

```
program App_File;
  var fp : text;
      a : integer;
begin
  Assign (fp, 'ft . txt');
  Append( fp );
  Read( a );
  Write(fp,' ',a);
  Close(fp);
end.
```

Файлова змінна повинна мати тип **text**.

Якщо в процедурі Assign не вказано шлях до файла ft.txt, то він повинен бути в поточному каталозі

Процедура **Read (a)** - зчитує число **a**, яке процедура **Write (fp, ' ', a)** передає в кінець файлу **f.txt**, записавши перед числом **a** пропуск .

Закривати файл обов'язков

Слайд7

таблиця 7.8

ПРОГРАМА ЧИТАННЯ ТЕКСТОВОГО ФАЙЛУ

```
program Reset_File;
  var fp : text;
  i : integer;
  Mas : array[1..11] of integer;
begin
  Assign (fp, 'ft . txt');
  Reset ( fp );
  for i := 1 to 11 do        begin
    Read ( fp, Mas [ i ] );
    Write ( Mas [ i ],' ; ' ) end;
  Close(fp);
  end.
```

Файлова змінна повинна мати тип **text**.

Процедура **Read(fp, Mas[i])** - зчитує з файла **f. txt** символи і записує їх в масив **Mas**, процедура **Write(Mas[i],';')** виводить їх на екран, розділяючи знаком ";"

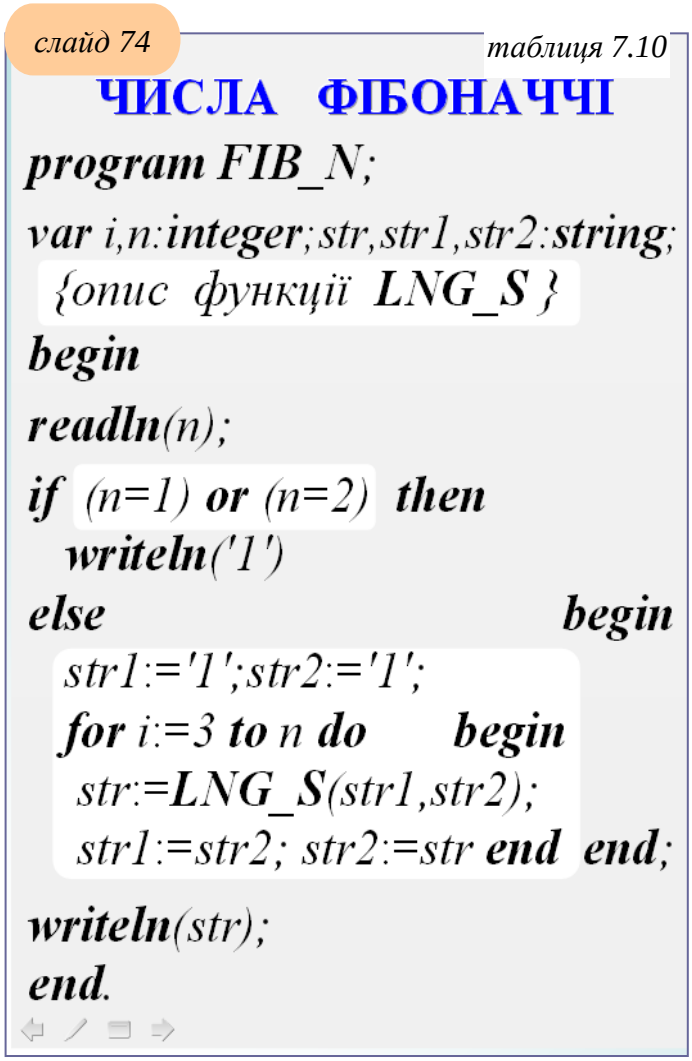
Закривати файл обов'язково

слайд

таблиця 7.9

ДОДАВАННЯ ДОВГИХ ЧИСЕЛ

```
function LNG_S(st1, st2: string):
                                  string;
var  i ,l1, l2, p, s1, s2, s:word;st:string;
begin
  if length(st1) < length(st2) then
  begin st := st1;st1 := st2;st2:=st  end;
  st1 := '0' + st1; p:= 0;
  l1 := length(st1); l2 := length(st2);
for i:=1 to l1- l2 do st2 := '0' + st2;
  for i := 1 to l1 do                begin
    s1 := (ord(st1[l1 – i + 1]) - 48);
    s2 := (ord(st2[l1 – i + 1]) - 48);
    s := s1 + s2 + p;
    st1[l1-i+1]:= char((s mod 10)+48);
    p := s div 10                    end;
if st1[1]='0' then delete(st1,1,1);
LNG_S:= st1     end;
```



III.3 ТЕХНОЛОГІЯ СТВОРЕННЯ СКЛАДНИХ ЕЛЕКТРОННИХ ДИДАКТИЧНИХ СИСТЕМ З ДОПОМОГОЮ MICROSOFT POWER POINT НА ПРИКЛАДІ ДОВІДКОВОЇ ДИНАМІЧНОЇ СИСТЕМИ "ТУРБО ПАСКАЛЬ В ТАБЛИЦЯХ "

Створення презентацій в середовищі Microsoft Power Point має властивості візуального проектування, яке в свою чергу забезпечується засобами об'єктно-орієнтованого програмування, тому слід розуміти поняття об'єкта. *Об'єктами* презентації будемо називати слайди та компоненти (елементи) слайдів, зокрема, тексти і графічні зображення. Для створення презентації із можливістю управляти об'єктами необхідно знати допустимі дії у Microsoft Power Point щодо об'єктів - *події*.

Ми використовували такі елементи слайдів: кнопки переходу між слайдами, *управляючі і підпорядковані* елементи (якщо, наприклад, подія полягає в тому, що натискування на один елемент, спричинює візуальний чи часовий ефект іншого елемента, то перший будемо називати управляючим елементом, або перемикачем чи тригером, а другий підпорядкованим елементом).

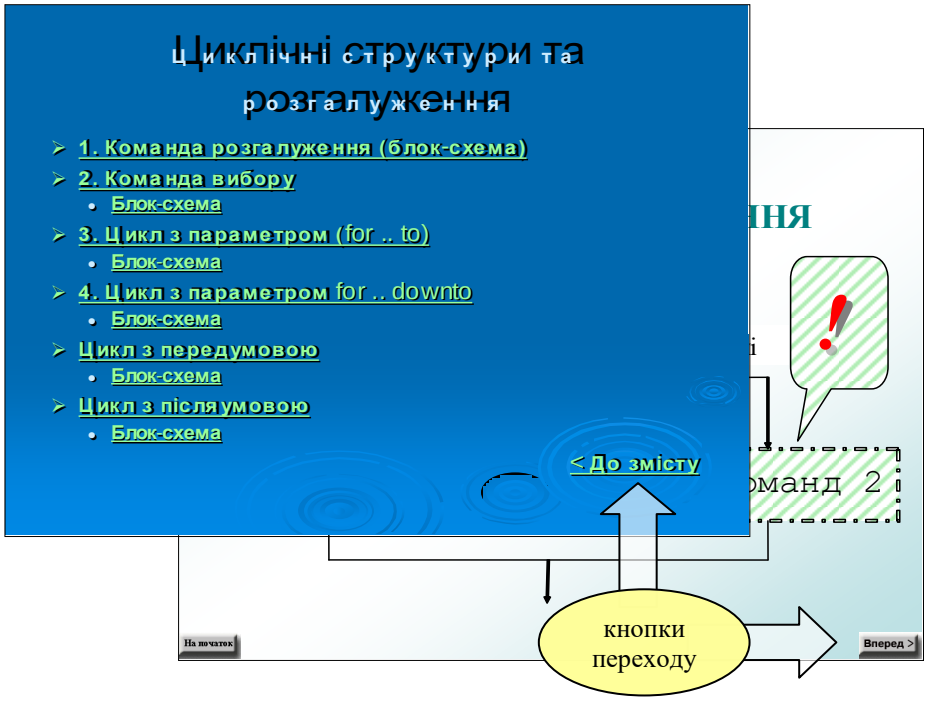
Події можна класифікувати на *управляючі* та *анімаційні*. Для повноцінної презентації необхідно використовувати такі основні події: *переходи між слайдами*, *вхід*, *вихід*, *виділення* і *переміщення* об'єктів, *взаємодія* об'єктів. При призначенні подій об'єкти слайдів називаються: *лінія*, *полілінія*, *прямокутник*, *група* та ін.

Опишемо докладніше вставку перерахованих подій.

а) **Переходи між слайдами** (мал. 3). Вони здійснюються за допомогою *гіперпосилань*. Гіперпосилання – це виділений в документі певним чином об'єкт, який містить інформацію про адресу іншого об'єкта. Вплив на гіперпосилання приводить до переходу на об'єкт, адреса якого вказана у гіперпосиланні. Як правило, гіперпосилання – це підкреслений текст, виділений іншим кольором, а наведення курсора перетворює його в знак "☝". Кликання мишею здійснює перехід за вказаною адресою.

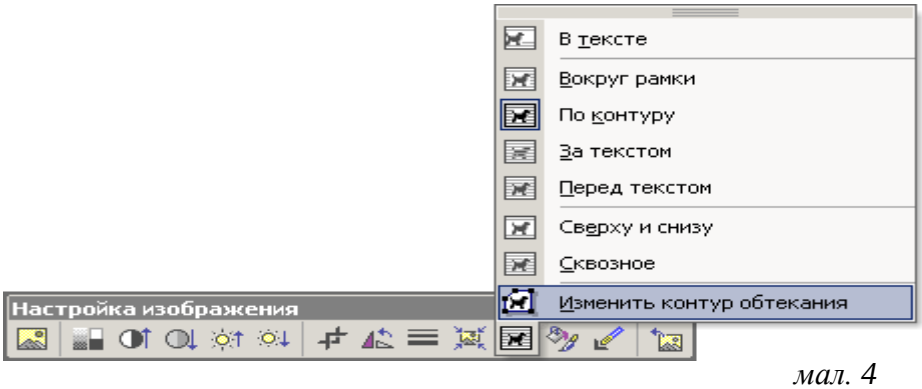
Перший варіант використано у слайді "Циклічні структури та розгалуження". В даному випадку кнопкою переходу є *текст* із гіперпосиланням „< До змісту>". Розглянемо процедуру вставки гіперпосилання у текст. Для цього слід виділити необхідний текст, у контекстному меню вибрати пункт „Гиперссылка...". На малюнку 4 зображено вікно вставки гіперпосилання *на файл* з коротким описом його головних полів. В адресному рядку пропонується вказати шлях до шуканого файла (точка посилання). Зміна чи видалення гіперпосиллання виконується з допомогою цього ж контекстного меню.

Звернемо увагу на слайд „Команда розгалуження (блок-схема)" (мал. 3). В



мал. 3

цьому випадку перехід між слайдами здійснюється з допомогою використання графічних об'єктів – кнопок, створених в середовищі Adobe Photoshop 7.0. Для того,



мал. 4

щоб додати малюнок до документа (чи презентації) можна піти через головне меню (Вставка/Рисунок/Из файла…) або простішим шляхом: просто перенести файл на документ.

Для зручності подальшого використання існує можливість змінити контур обтікання об’єкта текстом. Для цього у контекстному меню (мал. 4) слід вибрати пункт „Отобразить панель настройки изображения” (у випадку, якщо панель не відображується автоматично). Після вибору пункту меню „Отобразить контур обтекания” малюнок стане обмеженим червоною пунктирною лінією – контуром обтікання. Змінити контур можна за допомогою миші.

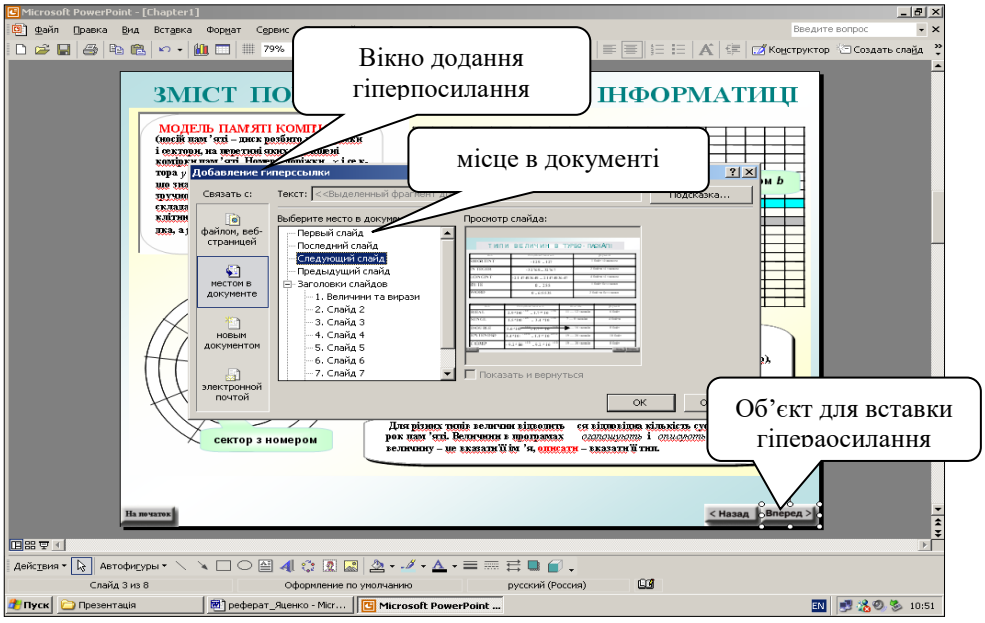
Вставка гіперпосилання в графічні об’єкти не має суттєвих відмінностей від вставки у текст. Для здійснення вставки достатньо в контекстному меню графічного об’єкта повторити вищеописані кроки для створення текстового гіперпосилання.

Дещо швидшим, щоправда, з меншим обсягом можливостей, є обрання пункту меню „Настройка действия…” контекстного меню. Перевага цього способу полягає в тому, що можна призначити дію не тільки при натисканні на об’єкт (як через вікно вставки гіперпосилання), а й при наведенні курсору миші на нього. Щоправда, буде відсутня можливість додати до об’єкту підказки, вибрати рамку, посилання на закладку або відправлення листа на електронну адресу тощо.

Гіперпосилання на місце в цьому ж документі створюється таким самим чином. Єдиною відмінністю є те, що посилання адресовані на сторінки, заголовки, слайди і т.д., що містяться лише на даному документі (малюнок 4).

б) **Засоби анімації та взаємодія між об’єктами**

Презентація буде вважатися більш повноцінною і „живою”, якщо в ній забезпечити взаємодію між об’єктами. Мається, наприклад, на увазі деяка подія при натисненні на окремий об’єкт чи наведенні курсора миші. Зручно розглядати взаємодію об’єктів у поєднанні із анімаційними ефектами, тому що значна їх кількість безпосередньо залежить від деяких конкретних об’єктів.



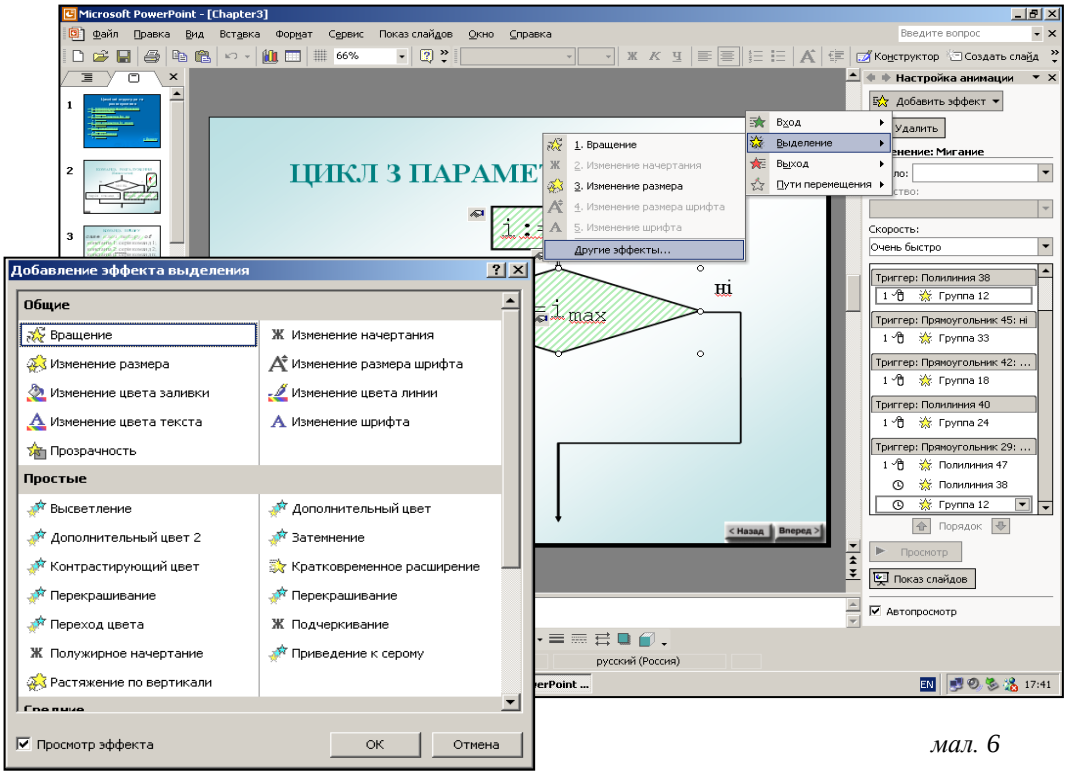
мал. 5

Спочатку звернемо увагу на ефекти анімації, а далі опишемо їхні параметри та властивості, включаючи можливість створення перемикачів (тригерів).

Під час імпортування у середовище Power Point малюнків, деяких об’єктів, автофігур (наприклад стрілок) Microsoft Word, Power Point перетворює їх у стандартні фігури власного призначення (наприклад звичайну стрілку у „полілінію”). Основні фігури Power Point, що масово використовуються в даному проекті, наступні: полілінія, лінія, прямокутник, округлі дужки, надписи. Решта елементів зустрічається вкрай рідко. Перераховані об’єкти часто об’єднуються у групи, що значною мірою спрощує застосування анімації, як до одного цілісного, а не до декількох об’єктів.

Отже, щоб додати анімаційний ефект до об’єкта (тексту, малюнка чи основної фігури Power Point) слід виконати такі дії. Виділити потрібний об’єкт, у контекстному меню обрати команду „Настройка анимации…” Праворуч з’явиться вікно налаштування анімації.

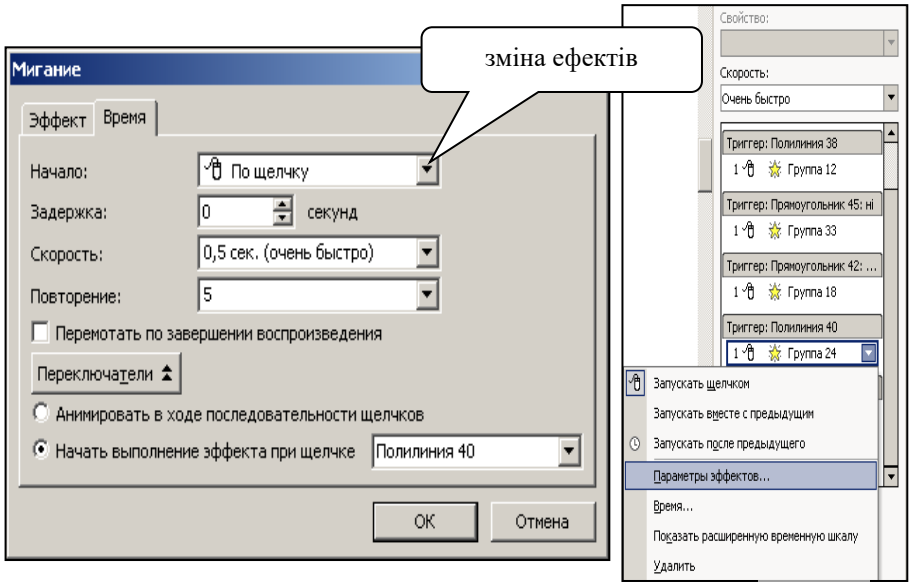
Щоб додати ефект слід кликнути на пункті меню „Добавить анимацию”. Наступний крок – із чотирьох запропонованих категорій (вход, выделение, выход, пути перемещения) обрати необхідну, а далі – потрібний ефект анімації. При потребі можна використати пункт „Другие эффекты…”. Буде запропоновано перелік усіх можливих ефектів у даній категорії (мал. 6).



мал. 6

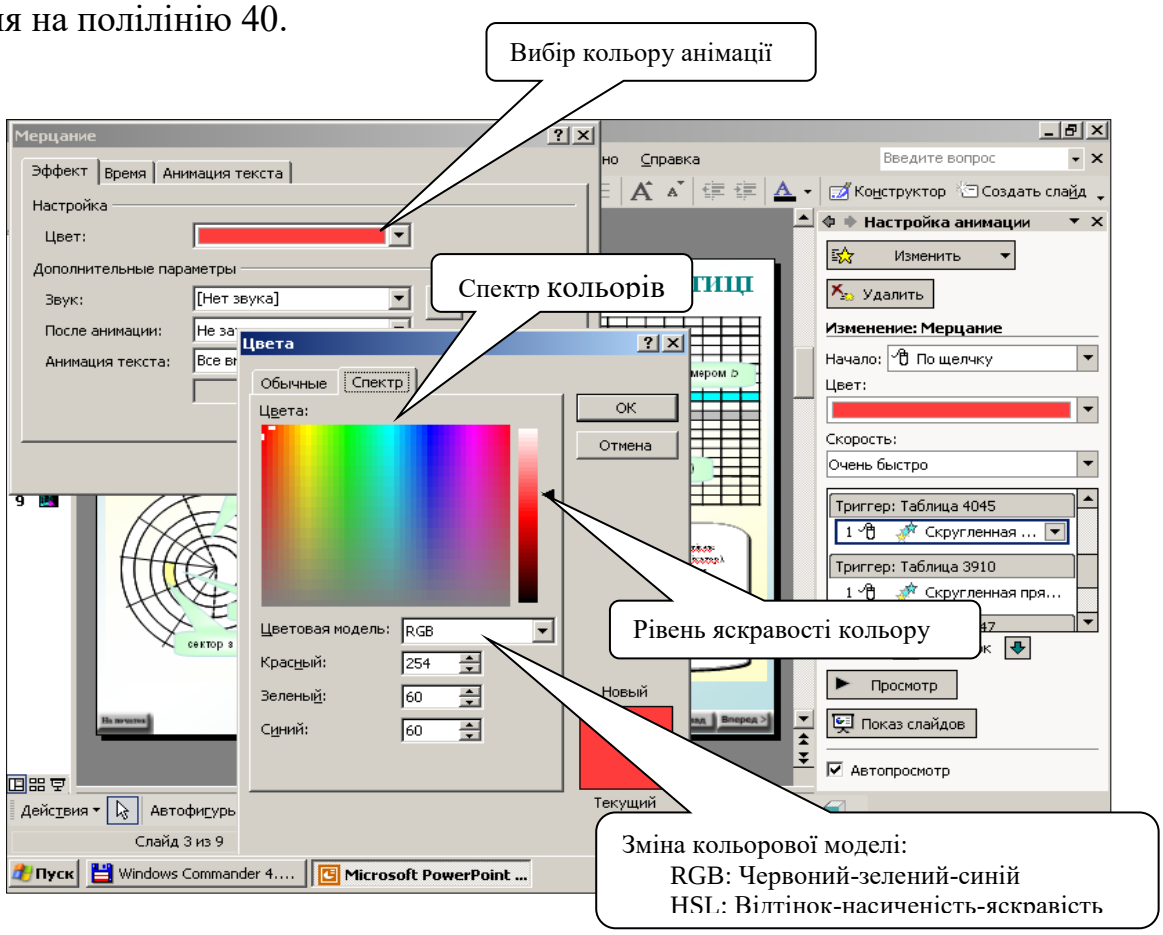
Розглянемо детальніше *параметри* ефектів, їхні властивості та можливості застосування. У контекстному меню одного із застосованих ефектів списку анімацій слід обрати пункт „Параметры эффектов…”. Тут знаходяться основні параметри, що стосуються, наприклад, кольору, звукового супроводу, анімації тексту. На вкладці „Время” міститься основна група параметрів, що стосується часових можливостей, а саме: порядку стартування ефектів, затримки, швидкості, а також перемикачів (мал. 7). Зміна порядку анімаційних ефектів (1 – при натисканні на об’єкт; 2 – старт одночасно із попереднім; 3 – старт після закінчення анімації попереднього) є однією із найчастіше застосовуваних елементів анімації.

Power Point є можливість керувати анімаційними ефектами одних об’єктів іншими. Ця взаємодія між об’єктами можлива завдяки використанню перемикачів (тригерів). При наявності тригера анімація залежного від нього об’єкта здійснюється



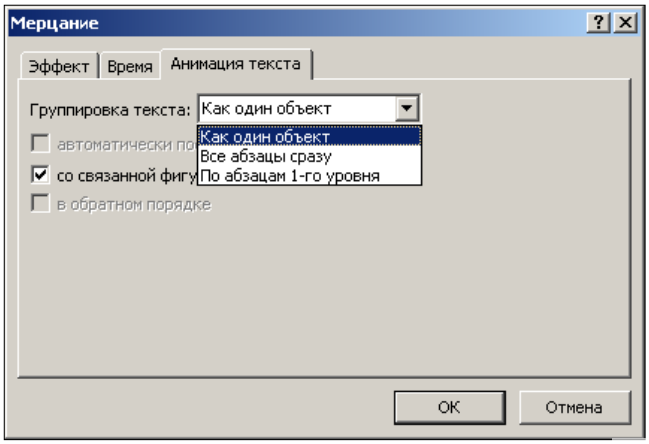
мал. 7

після натискання на тригер. Як видно з мал. 7, група 24 буде анімуватися лише після натискання на полілінію 40.



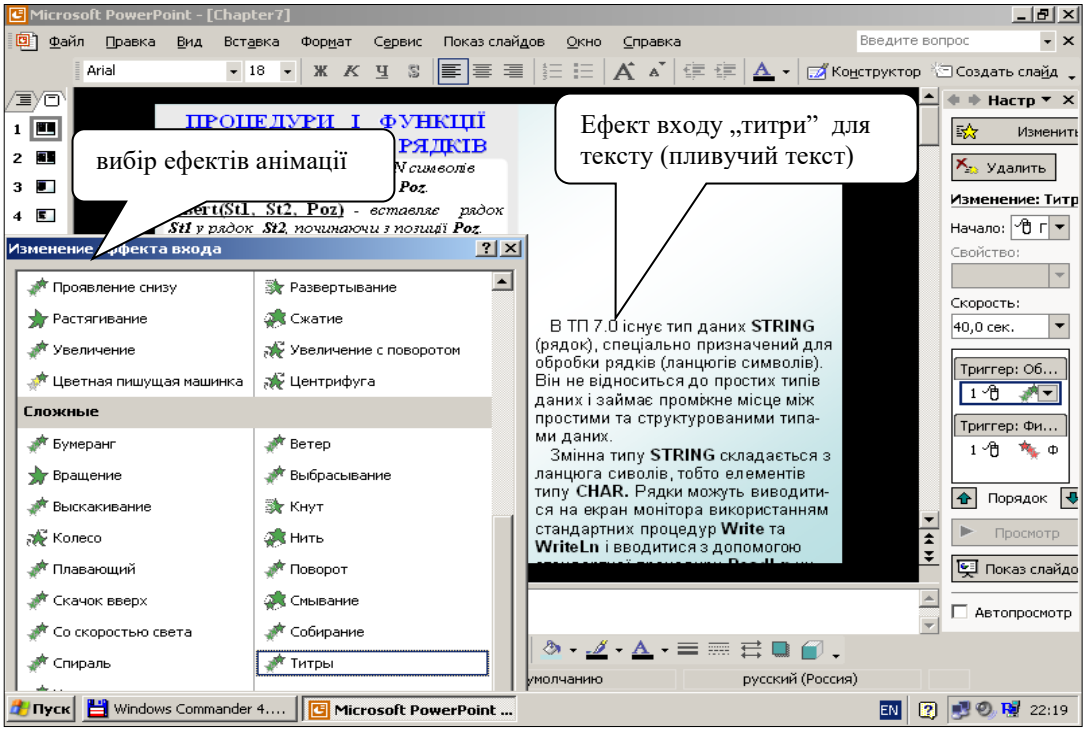
Дещо інші параметри має ефект блимання. На першій вкладці м *мал.8* ити колір блимання. Як видно з мал. 8, колір може бути довільним, а його відображення залежить від параметрів комп’ютера. Спектр дає змогу якнайточніше обрати колір у залежності від кольорової моделі. Зважаючи на потреби користувача, колір можна

обрати за вмістом трьох основних (червоний, зелений, синій – RGB-палітра), або ж за HSL (відтінок, насиченість, яскравість). Параметри часу відмінностей не мають – для всіх ефектів анімації параметри часу стандартні.



мал. 9

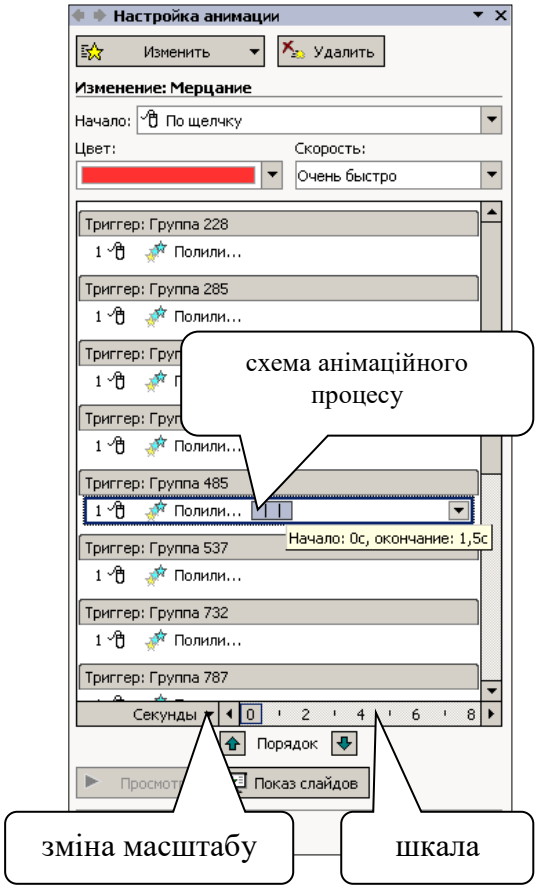
На вкладці „Анимация текста” (мал. 9) розміщено параметри анімації тексту. Групування тексту як „один об’єкт” зумовлює анімування тексту разом із фоном (текст із фоном зливаються в один зафарбований у вибраний колір прямокутник). „Все абзацы сразу” та „По абзацам 1-го уровня” – лише тексту.



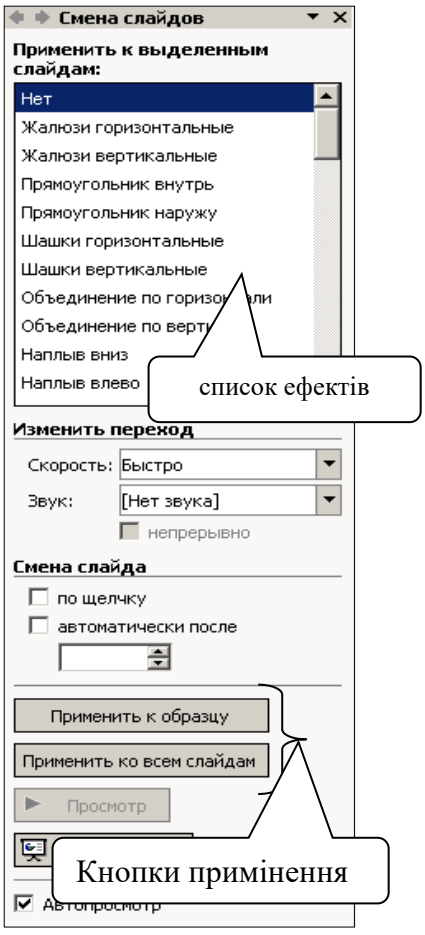
Мал. 10

Завдання створення ефекту анімації для текстів виконується подібно ефектів анімації графічних зображень. Так, наприклад, ефект „титри” з групи складних (категорія „Эффект входа”) виконує поступове переміщення тексту знизу із виходом за верхній край (мал. 10).

Якщо в контекстному меню списку анімаційних ефектів обрати пункт „Показать расширенную временную шкалу”, відкриється доступ до детального редагування тривалості ефекту поряд з іншими, що уможливлює точний підбір старту чи зупинки ефектів наочно (мал. 11). Зміна шкали анімації (від десятих секунди до десятків хвилин) дає змогу зручнішого та детальнішого редагування. Тривалість ефекту можна змінити як через вікно параметрів, так і з допомогою миші. Для цього необхідно лише змінити розмір схеми анімаційного процесу. Завдяки розширеній часовій шкалі існує можливість зручного встановлення часу стартування анімації. Для зміни часу стартування достатньо скористатися курсором миші – перемістити бігунок на відповідну відстань. Таким чином, послідовність та час виконання ефектів можуть редагуватися з максимальною точністю. Оскільки нерідко виникає потреба у додаткових часових можливостях анімації, дана шкала –



мал.11



мал. 12

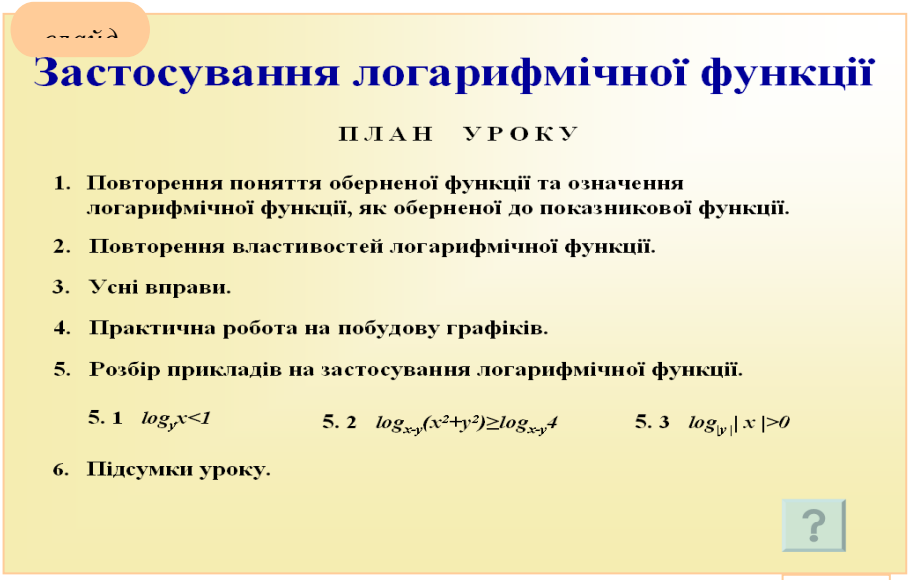
найкращий інструмент для забезпечення такого плану потреб.

Не залишається без анімації і перехід між слайдами. Для встановлення анімації під час зміни слайду слід обрати пункт „Смена слайдов...” контекстного меню. Праворуч з’явиться вікно на лаштування параметрів зміни слайдів, в т.ч. і анімацій (мал. 12). Обраний серед списку ефект та швидкість його анімації буде застосовано до даного слайду або до всіх слайдів у презентації – залежить від того, яку кнопку застосування буде натиснуто. Зміна слайдів може супроводжуватися звуковими ефектами, як і будь-яка анімація, не залежно від того, чи це анімація до окремого об’єкта, чи зміни слайду.

IV. СТВОРЕННЯ ЕЛЕКТРОННИХ СЦЕНАРІЇВ УРОКІВ ЗАСОБАМИ MICROSOFT POWER POINT

Розділ II присвячений електронним конспектам уроків (ЕКУ). Це документи, створені переважно засобами Microsoft Word, які виконують роль поурочного плану учителя та стислого конспекту для учнів із вмонтованими дидактичними засобами. Вивчення теми проходить у спілкуванні учнів із електронним уроком. Тут ми розглянемо поняття *електронного сценарію уроку* (ЕСУ). Для прикладу візьмемо ЕСУ з алгебри на тему "Застосування логарифмічної функції". ЕСУ – це презентація до вибраної теми уроку. Але традиційні презентації призначені для ілюстративного супроводу уроку. По суті – це пасивне навчання, адже учні, як правило, не втручаються в демонстрацію презентації, вони лише спостерігають її. Електронному сценарію уроку ми відведемо іншу, активну роль. По суті він близький до електронного конспекту уроку, але, на відміну від останнього, не містить блоків, призначених для учителя: запису мети, опису обладнання, теоретичних відомостей з методичними порадами, хронології дій учителя та учнів, які притаманні традиційним поурочним планам і їх електронним аналогам. ЕСУ – це *алгоритм дій учнів* по вивченню оголошеної теми із завданнями, запитаннями, ілюстраціями та прикладами. Він може використовуватись паралельно з ЕКУ, але не обов'язково. Роль учителя на уроці ЕСУ зводиться ролі арбітра, що займається хронометражем, оцінюванням, коментарями тощо. Невід'ємним є також комбіноване з ЕСУ використання учнями інших КЕДЗ і прикладних пакетів та їх робота з зошитами. ЕСУ – це сценарій, тому вся робота на уроці повинна бути вказана в ньому. В зв'язку з тим, що учні переважну частину такого уроку повинні індивідуально працювати з комп'ютером, остання особливість полягає в тому, що *урок із використанням ЕСУ повинен бути бінарним з інформатикою*.

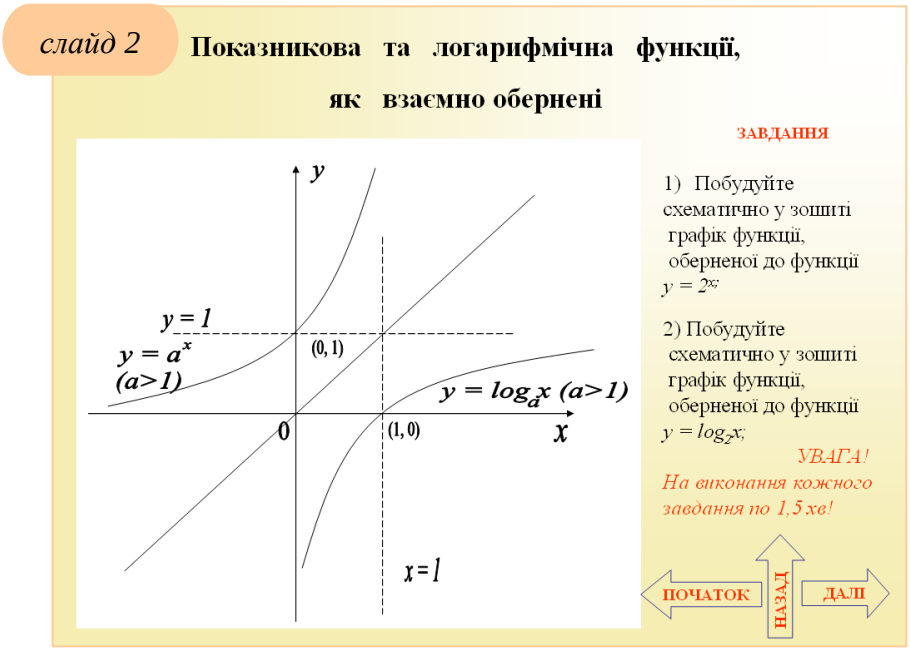
Розглянемо приклад. Стартовий слайд містить план дій учнів на уроці:



мал. 13

Всі пункти плану уроку, крім останнього, є гіперпосиланнями. Пункти 5.1-5.3 з дидактичною метою приховані, вони активізуються кликанням по п.5. В правому нижньому кутку міститься кнопка, яка активізує підказку. Переходи між слайдами довільні, вони організовані з допомогою згаданих гіперпосилань

та кнопок, що з'являються в наступних слайдах. Як буде видно далі, забезпечена можливість паралельної роботи з іншими додатками.



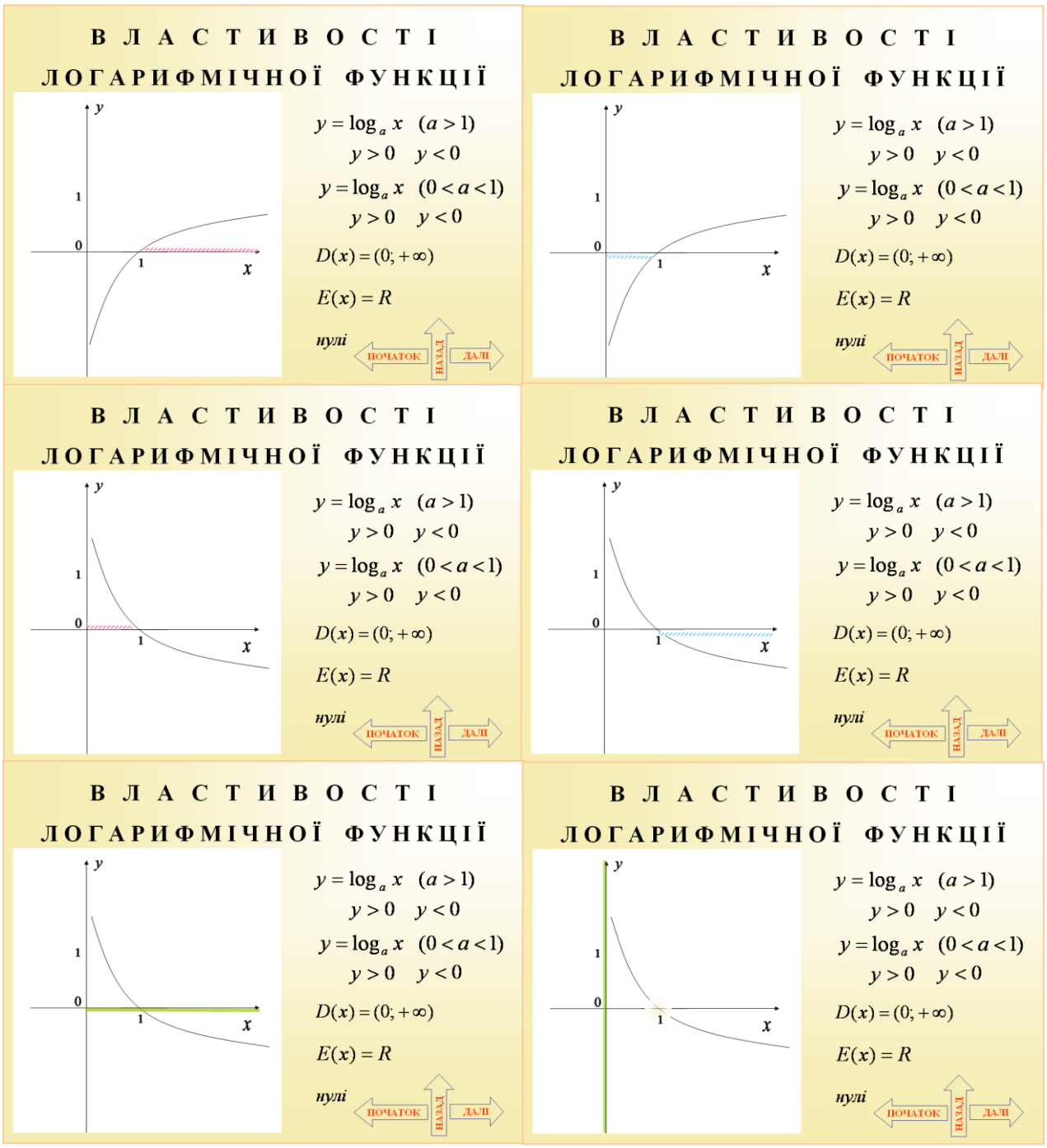
мал. 14

Другий слайд використовується для актуалізації опорних знань, підведення підсумків та узагальнення. З малюнка видно, що логарифмічна функція вводиться, як обернена до показникової. Завдання в правій частині слайда виникають при активізації гіперпосилання "Завдання", учитель може дати їх поваріантно. Повідомлення "Увага! На виконання кожного завдання по 1,5 хв!" задає темп уроку. На кожному з наступних слайдів також наявні кнопки переходу ПОЧАТОК, НАЗАД, ДАЛІ. Наступним є слайд 3 (малюнок 15).

мал. 15

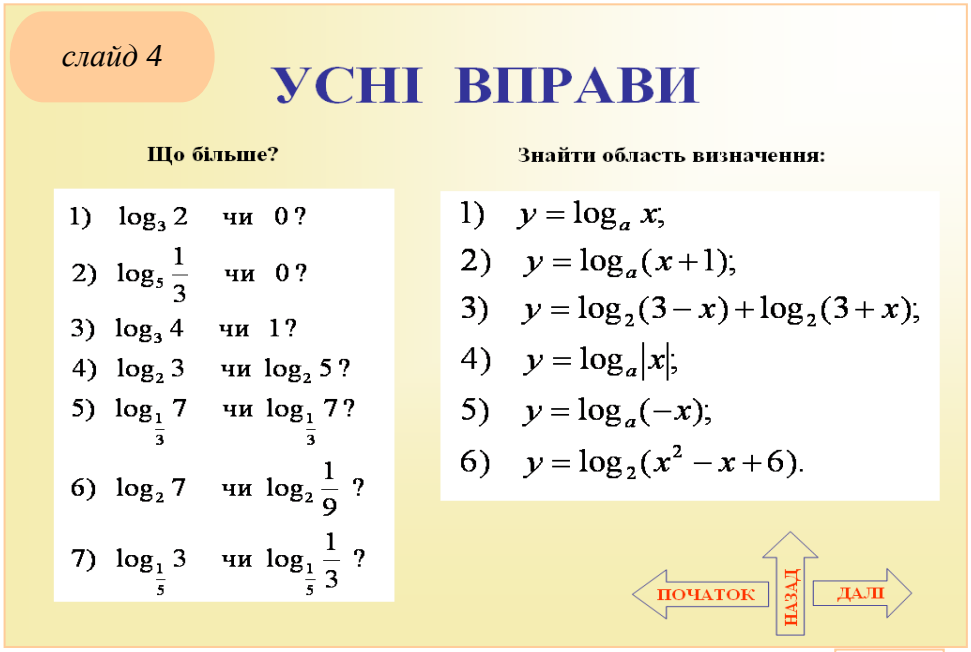
У його лівій частині накреслена прямокутна система координат, а в правій частині формули-гіперпосилання, активізація яких дає графічну ілюстрацію

різних властивостей логарифмічної функції. На прикладі цього слайду особливо добре видно, що при роботі з ЕСУ учні активно втручаються в процес демонстрації презентації, при цьому проводячи вивчення і закріплення навчального матеріалу та дослідження. Нижче наведено кілька варіантів слайду:

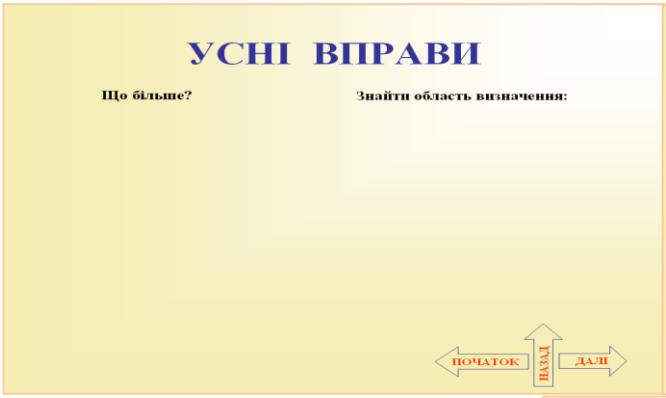


мал. 16

Наступний слайд 4 (малюнок 17) відповідає пункту 3 першого слайду і подібно до слайду 3 має кілька варіантів. Спочатку з'являється варіант, зображений на малюнку 17 *а*), потім, при потребі, можна одержати варіанти, зображені на малюнках 17 *б*), 17 *в*), 17 *г*). Різні варіанти слайду 4 можуть викликатись на екран в різних частинах уроку і демонструватись довільну кількість разів.

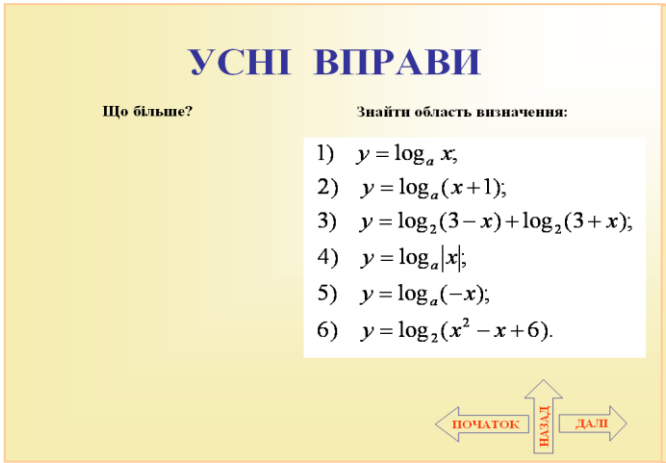


мал. 17

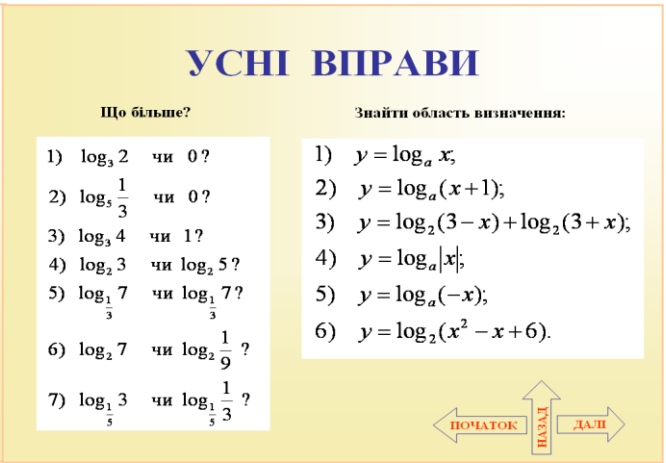


мал. 17а)

мал. 17б)

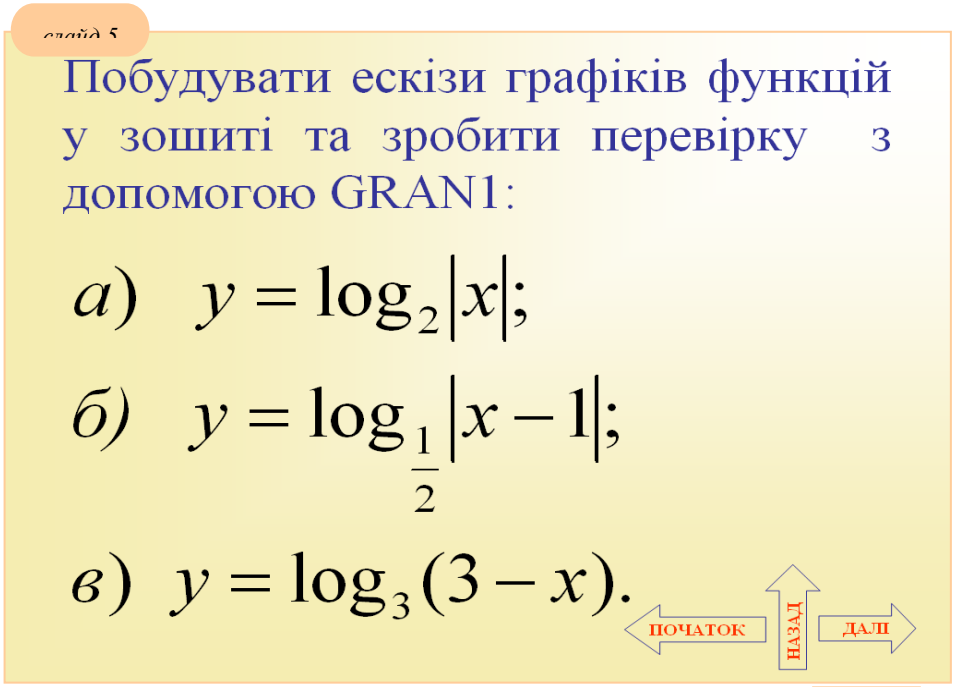


мал. 17в)



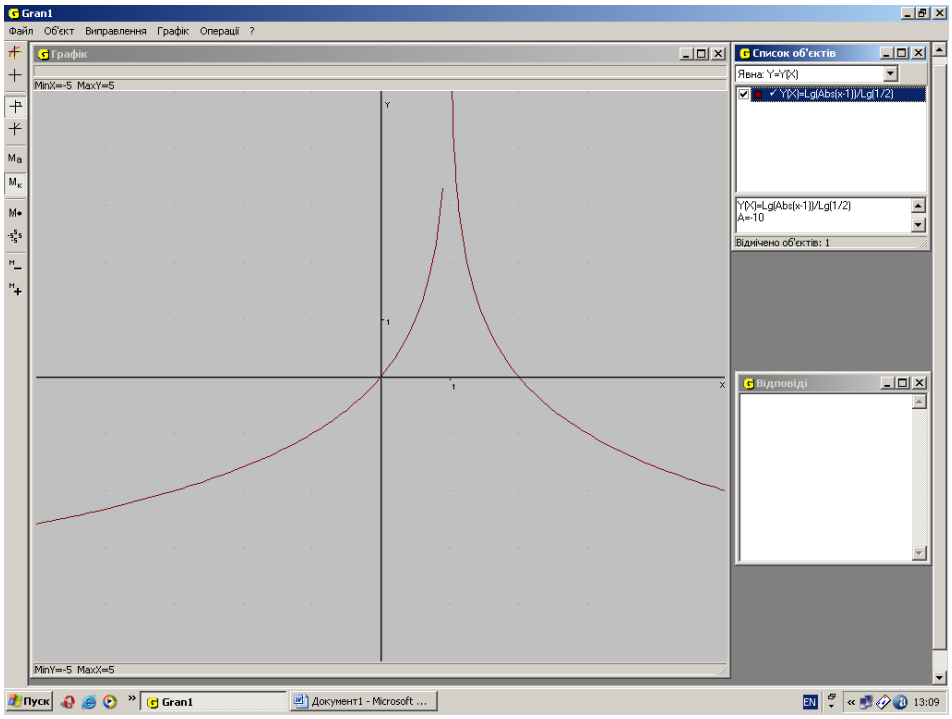
мал. 17г)

Слайд 5 дає вказівку про виконання в зошитах трьох завдань (можна давати поваріантно) та відкриття вікна навчального середовища Gran1 для самоперевірки, яку можна робити, помінявши зошити учнів між варіантами. Наголосимо, що для вставки цього виду роботи необхідні навички учнів виконання завдання в середовищі Gran1, що досягається на уроках інформатики. Крім того, як уже згадано вище, цей урок повинен бути бінарним з уроком інформатики.



мал. 18

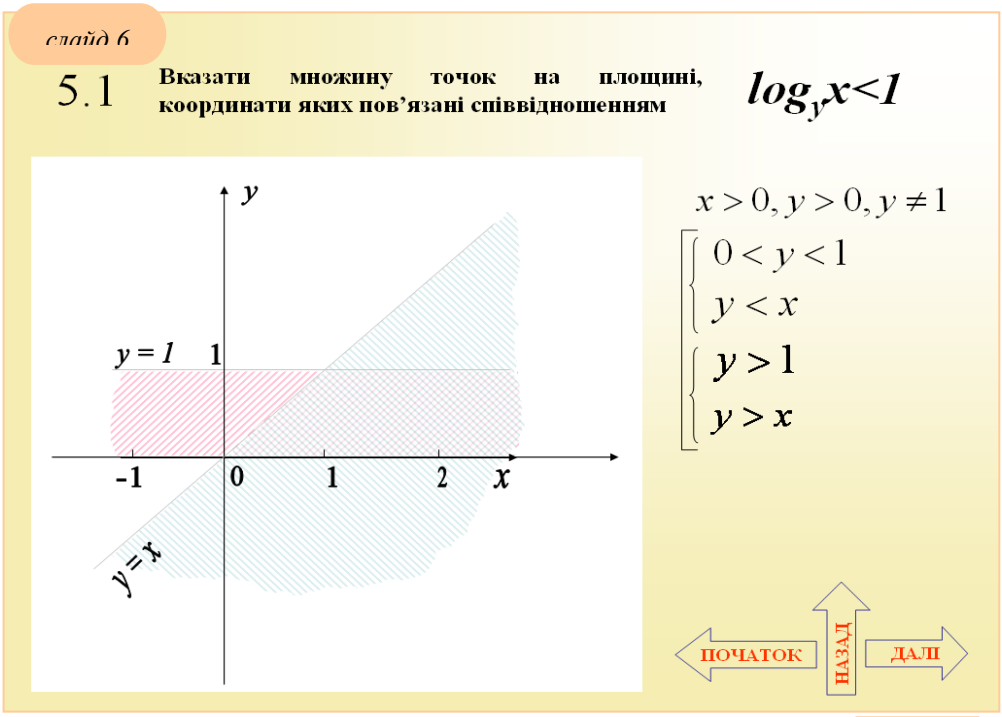
Малюнок 19 відображає одне із трьох завдань слайду 5 в середовищі GRAN1. Додамо, що вікно GRAN1 слід відкрити до початку уроку.



мал. 19

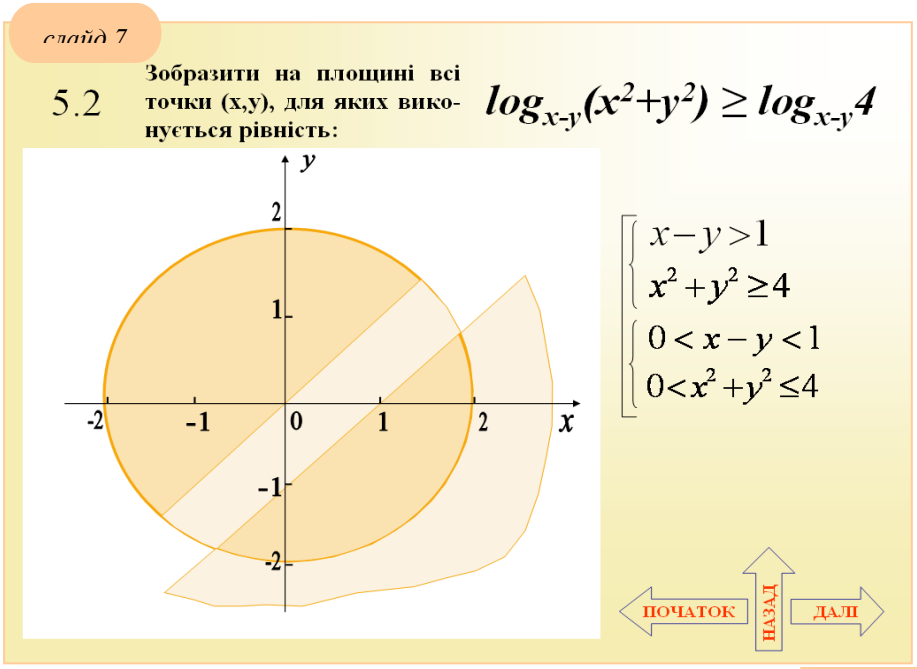
На малюнку 20 зображено один із варіантів слайда 6 (наводити різні варіанти не будемо, як структурно аналогічні слайдам 3 та 4). Вибрано графічний розв'язок першої системи. Останній варіант слайду демонструє графічну ілюстрацію розв'язку нерівності $\log_y x<1$.

Слайд 7, що відповідає завданню 5.2, також подамо в одному, остаточному варіанті (малюнок 21). Але ще раз звернемо увагу на структурну подібність слайдів.



мал. 20

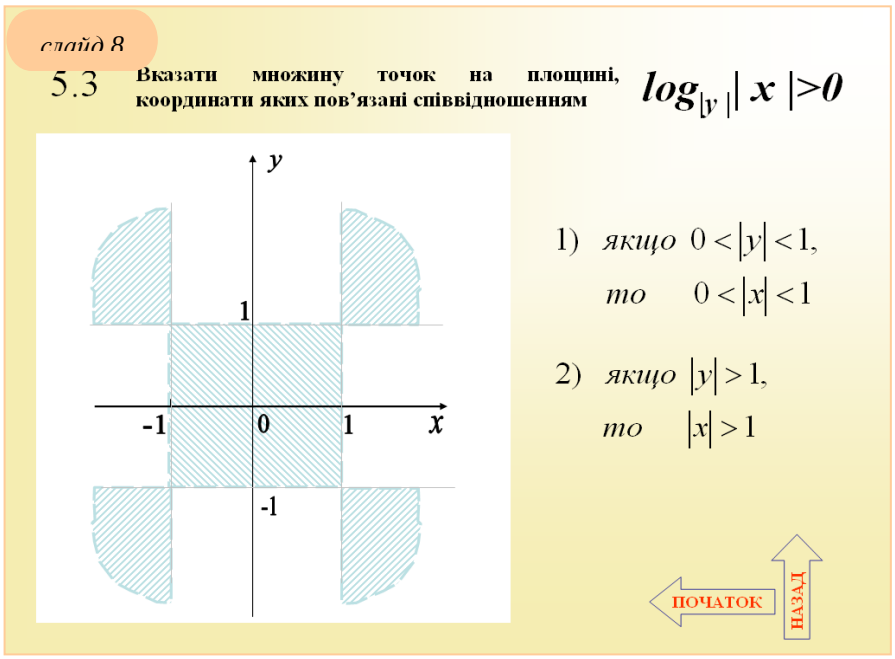
При використанні ЕСУ така подібність усіх слайдів дуже бажана. Справа в тому, що на відміну від електронних конспектів уроків, які створюються переважно в середовищі Microsoft Word, а учні працюють із екраном, як з підручником, при використанні ЕСУ учні повинні самостійно аналізувати складні ситуації при незначній кількості текстів (вимоги створення презентацій). Тому подібність слайдів допомагає уникнути додаткових психологічних труднощів, пов'язаних із необхідністю адаптації до екранного середовища.



мал. 21

Перед проведенням уроку згідно ЕСУ слід повторити роботу в середовищах, які будуть використовуватися, у данному випадку GRAN1 та Microsoft Power Point.

Нарешті представляємо останній слайд (малюнок 21):



мал. 22

Пункт 6 (Підсумки уроку), як зазначалось вище, учитель може проводити без екранного супроводу. Але при необхідності, підводячи підсумки уроку, можна використати оглядово всі вісім слайдів у довільному порядку необхідну кількість разів. Розглянуту презентацію можна також застосувати фрагментарно чи повністю на наступних уроках та при узагальненні по темі.

V. ПІСЛЯМОВА

Підсумовуючи, приходимо до висновку, що впровадження ІКТ сьогодні ще не стало повноцінним. Воно переважно зводиться до використання демонстраційних форм. Тому сьогодні слід створювати ширші можливості для мислительної діяльності учнів безпосередньо за дисплеєм, тобто частіше практикувати чисельний експеримент, дослідницькі форми роботи.

мал. 23

Можна говорити, що комп'ютер сьогодні став одним із основних інструментів розвитку інтелектуальних здібностей дитини. Але він, як інструмент для інтенсифікації та автоматизації розумової діяльності, крім стандартних функцій, пов’язаних із використанням ППЗ загального призначення, володіє важливою властивістю – *здатністю до розширення обсягу власних можливостей завдяки програмуванню*. В досягненні цього вирішальна роль належить шкільному курсу інформатики. Він покликаний забезпечувати реалізацію дуже важливої функції цементування в єдине ціле всієї сукупності сучасних базових знань, яку можна уявити у вигляді діаграм або кругів Ейлера, де інформатика відіграє роль і окремого компоненту, і пов’язуючого матеріалу (див. мал. 23). Крім того, інформатика допомагає формувати вузлові компоненти загальної освіти - логічне, алгоритмічне і структурне мислення, необхідні тепер будь-якій пересічній людині.

Міркування про роль шкільного предмету інформатики були б не повними без окреслення ще й так би мовити “технологічної“ її ролі у системі середньої освіти. Нам вона бачиться так: *першою і обов’язковою складовою інформаційних технологій навчання є повноцінний курс шкільної інформатики*. Якщо ця складова не присутня у повному обсязі, то всі інші, разом узяті, проблеми не вирішать. Учням необхідно добре оволодіти засобами навігації у потоках інформації, а комп’ютер, подібно компасу у морській справі, - можна розглядати тут як головний інструмент.

При застосуванні інформаційно-комп'ютерних технологій навчання слід дотримуватись схеми: *використання на уроках ліцензованих ПНЗ з допомогою проектуючих засобів ⇒ курс інформатики у вищеописаному розумінні ⇒ повноцінне використання можливостей систем програмування та офісних пакетів для моделювання навчальних ситуацій і чисельних експериментів на уроках з інших предметів ⇒ дослідницька позаурочна робота з допомогою комп’ютера*.

Деякі аспекти такого підходу до застосування ІКТ у навчально-виховному процесі порушено в цьому матеріалі.

ЛІТЕРАТУРА:

1. Я.М. Глинський, Практикум з інформатики, навчальний посібник, Львів, 2003 р.
2. В.В. Федько, В.І. Плоткін, Основи алгоритмізації та програмування, 11 клас, Харків, видавництво ″Ранок″, 2003 р.
3. М.І. Жалдак, Комп’ютер на уроках математики, посібник для вчителів, Київ, 2003 р.
4. Методичні рекомендації щодо організації та змісту навчально-виховного процесу в закладах освіти Київщини в 2003-2004 н.р.

ЗМІСТ