Міністерство освіти і науки України
Вінницький національний технічний університет

В.І. Савуляк, Т.Ф. Архіпова, А.В. Губанов

ІНФОРМАТИКА

Частина 1

Затверджено Вченою радою Вінницького національного технічного університету як навчальний посібник для студентів спеціальностей “Технологія та устаткування відновлення та підвищення зносостійкості машин і конструкцій“ та “Автомобілі та автомобільне господарство”. Протокол № 12 від 29 червня 2006 р.

Вінниця ВНТУ 2007

УДК 621(075)
С 13

Рецензенти:
***Ю.А.Бурєнніков*,** кандидат технічних наук, професср ВНТУ
***В.А.Лужецький*,** доктор технічних наук, професор ВНТУ
***П.С.Берник*,** доктор технічних наук, професор ВДАУ

Рекомендовано до видання Вченою радою Вінницького національного технічного університету Міністерства освіти і науки України

Савуляк В.І., **Архіпова Т.Ф, Губанов А.В.**
С13 **Інформатика. Частина 1.** Навчальний посібник. – Вінниця: ВНТУ, 2007. – 144 с.

Навчальний посібник є базовим для підготовки студентів машинобудівних спеціальностей з інформаційних технологій, при вивченні відповідних дисциплін, виконанні розрахунково-графічних, курсових та дипломних робіт і проектів.

УДК 621(075)

Зміст

В С Т У П

На сучасному етапі розвитку ринкової економіки науково-технічний прогрес полягає у якісних змінах знарядь праці, технологічних та управлінських процесів. Одним з основних чинників впливу науково-технічного прогресу на всі сфери діяльності людини є широке застосування нових інформаційних технологій, тобто сукупність методів та засобів отримання та використання інформації на базі обчислювальної та комунікаційної техніки і широкого застосування математичних методів. Під впливом нових інформаційних технологій здійснюється перехід від екстенсивного зростання об'ємів виробництва до інтенсивного, відбуваються докорінні зміни у суспільному розподілі праці, істотні зміни зазнає технологія управління (процеси обгрунтування та прийняття рішень, а також організація їх виконання). Нові інформаційні технології сприяли появі наукового та прикладного напрямку, що іменується інформатикою.

З сучасної точки зору інформатику можна розглядати і як науку, і як прикладну дисципліну, і як галузь народного господарства.

Широке застосування комп'ютерів в усіх галузях народного господарства вимагає оволодіння комп'ютерною грамотою усіма верствами нашого суспільства. Комп'ютерна грамотність є складовою частиною загальнолюдської культури. Провідником цієї грамотності є наука – інформатика.

Інформатика – це фундаментальна природнича наука, що вивчає структуру та загальні властивості інформації, а також питання, пов'язані з процесами збирання, зберігання, пошуку, передачі, переробки, перетворення та використання інформації в різних сферах людської діяльності.

Інформація – це деяка сукупність відомостей, які деяка система сприймає від навколишнього світу, зберігає в собі або видає в зовнішнє середовище.

Сучасна інформатика є результатом інтенсивного розвитку науки та техніки за останні десятиріччя в напрямках:

- розробки методів автоматизованого збирання, зберігання, пошуку та передачі інформації;

- розробки методів обробки та перетворення інформації;

- розробки технологій та комп'ютерної техніки для розвитку перших двох напрямків.

Інформатика може бути віднесена до класу природничих наук, оскільки закони обробки інформації в суспільних, біологічних та інженерних

системах єдині. Інформатика може бути віднесена до класу фундаментальних наук, оскільки поняття інформації та процеси її переробки носять загальнонауковий характер. Інформаційні моделі використовуються у різних областях науки.

Інформатика історично виникла та розвивалась як природнича дисципліна, до складу якої входило розроблення:

- методів та правил раціонального проектування пристроїв та систем обробки інформації;

- технології використання цих пристроїв та систем для розв’язання наукових та практичних задач;

- методів взаємодії людини з цими пристроями та системами.

Розглядаючи інформатику як галузь народного господарства, можна виділити в ній три відносно автономні складові частини:

- виробництво технічних засобів обробки та передачі інформації;

- обробку інформації;

- виробництво та реалізацію програмних засобів та систем.

Курс "Інформатика " використовує також поняття "комп'ютерна техніка".

Комп'ютер – це електронний пристрій, здатний автоматично сприймати, перетворювати, зберігати, накопичувати, поновлювати та видавати інформацію. Він має широкий набір пристроїв, які забезпечують його зв'язок із зовнішнім світом та програмне забезпечення, яке дозволяє працювати з числовою, текстовою та графічною інформацією, з базами даних, електронними таблицями та іншими спеціальними предметними системами. Комп'ютер можна також сприймати як пристрій, що виконує обчислення деяких величин за заданим правилом – алгоритмом.

Програмування – це дисципліна, що служить для вивчення методів формулювання та розв'язання задач за допомогою комп'ютерів. Умовою застосування комп'ютерів для розв'язання задач є побудова відповідного алгоритма, що містить в собі правила отримання результуючої інформації із заданої інформації. Правила запису алгоритму для виконання його комп'ютером є досить жорсткими, тому що він не може нічого домислювати за людину.

Мовою програмування називається сукупність засобів та правил подання алгоритму у вигляді, що сприймається комп'ютером.

Програмою називається алгоритм, записаний мовою програмування.

ЗМІСТ ТА МЕТА КУРСУ

Особливе значення набуває інформатика у підготовці інженерів. Це пов'язано з тим, що в умовах розвитку інформаційних технологій випускникові технічного вузу потрібно:

- працювати користувачем персонального комп'ютера (ПК) (автоматизованого робочого місця – АРМ, робочої станції тощо) в умовах "електронного офісу", інтегрованної інформаційної системи, електронної пошти, в локальних та глобальних телекомунікаційних мережах;

- удосконалювати технологічні та управлінські процеси на своєму робочому місці (автоматизацію управлінських задач) з використанням найновітніших технічних та програмних засобів.

В курсі "Інформатика" вивчаються основи інформатики, розглядаються технічні засоби, базові та прикладні програмні засоби у інформаційних системах. З системних позицій розглядається класифікація комп'ютерів та тенденції їх розвитку; організаційно-технічні та периферійні засоби обробки інформації; локальні та глобальні обчислювальні мережі; базові програмні засоби персональних комп'ютерів, у тому числі сучасні операційні системи та тенденції їх розвитку; методи та засоби захисту інформації, у тому числі від комп'ютерних вірусів. Розглядаються також сучасні методи та засоби роботи інженера в умовах використання нових інформаційних технологій, першорядна увага приділяється особливостям створення інформаційних систем та автоматизації задач проектування та управління, починаючи з обстеження предметної області, постановки та алгоритмизації задачі, проектування бази даних, закінчуючи характеристиками та тенденціями розвитку світового та вітчизняного ринку прикладних програмних продуктів для електронного офісу.

Навчальний курс "Інформатика " орієнтований на те, щоб у результаті його опанування студент:

- сформував фундамент сучасної інформаційної культури;

- забезпечив стійкі навики роботи з комп'ютером в умовах локальних та глобальних обчислювальних мереж та систем телекомунікацій, нових інформаційних технологій в інженерній діяльності;

- ознайомився з архітектурою, техніко-експлуатаційними характеристиками та програмним забезпеченням комп'ютерної техніки;

- опанував основи сучасної методології алгоритмізації інженерних задач та практичної їх реалізації з використанням комп'ютерів та типових програмних продуктів.

В результаті вивчення дисципліни студент повинен:

- знати основи нових інформаційних технологій;

- знати сучасний стан та напрямок розвитку комп'ютерної техніки та програмних засобів;

- володіти основами автоматизації розв’язування інженерних задач;

- впевнено працювати користувачем комп'ютера;

- знати основи створення інформаційних систем та використання нових інформаційних технологій переробки інформації, у тому числі вміти працювати з сучасними програмними засобами;

- мати уявлення про роботу в локальних та глобальних комп'ютерних мережах, мати навики використання електронної пошти, телеконференцій, електронних дошок оголошень, засобів електронного офісу.

Значна роль у курсі "Інформатика" належить комплексу лабораторних робіт, головною задачею якого є набуття студентами навиків використання сучасних програмних продуктів у процесі роботи з комп'ютером.

Етапом, що завершує підготовку студента з основ інформатики має бути курсова робота, у межах якої синтезуються раніше отримані знання та навики. При виконанні курсової роботи студент оволодіває навиками алгоритмізації розв’язування інженерних задач, починаючи з аналізу предметної області, побудови моделі, блок-схеми алгоритму та програми на алгоритмічних мовах і закінчуючи її реалізацією на комп'ютері з використанням електронних таблиць, систем управління базами даних та пакетів прикладних програм з подальшим аналізом отриманих результатів.

1 ЕЛЕКТРОННО-ОБЧИСЛЮВАЛЬНІ МАШИНИ (КОМП'ЮТЕРИ)

Історія розвитку комп'ютерної техніки

Комп'ютер – це універсальний інструмент для розв'язання різноманітних інформаційних задач.

Історія розвитку обчислювальної техніки починається з давніх часів, коли виникла необхідність розрахунків. Перша "обчислювальна машина" – ***Абак*** – була подібна до сучасної рахівниці.

В XVI ст. шотландський математик Джон Непер вперше використав для складних розрахунків таблицю логарифмів, що призвело до винайдення логарифмічної лінійки. На початку XVII ст. французський філософ та математик Блез Паскаль сконструював перший механічний калькулятор.

В 30-х роках XIX ст. британський винахідник Чарльз Беббідж висунув ідею "аналітичної машини", яка за принципами своєї організації нагадує сучасні комп'ютери. Вона повинна була виконувати розрахунки будь-якої складності за заданою програмою. Перші програми для цієї машини склала дочка видатного англійського поета лорда Байрона – Аделіна Ловлейс, яку можна вважати першим програмістом в історії людства. В 1890 році в США був сконструйований удосконалений варіант такого калькулятора, який використовувався для перепису населення. Поява на початку 30-х років XX ст. електромеханічних реле послужило поштовхом для створення електромеханічних калькуляторів. Значних успіхів в цьому досягнули молодий німецький інженер Конрад Цузе (1941 рік, машина «Енігма») та американський вчений Говард Ейкен, який для фірми IBM у 1943 році побудував машину «Марк-1».

В 1936 році 24-річний математик Алан Тьюрінг дав чіткий логічний доказ можливості створення універсальної програмно-керованої обчислювальної машини. Така машина на базі електромеханічних реле була створена групою А.Тьюрінга в 1942 році для Британської розвідки, де і використовувалась для дешифровки таємних повідомлень німецьких спецслужб, що шифрувались машиною «Енігма».

Перший в світі комп'ютер на основі електронних ламп ЕНІАК був створений за принципами Тьюрінга в 1945 році в США. Цей комп'ютер був в 1000 разів продуктивніший ніж «Марк-1», але програмувався він шляхом комутації провідників і не міг запам'ятовувати раніше введені програми. Тому незабаром в США розпочались роботи з проектування нового комп'ютера, основні принципи функціонування якого виклав відомий ма-

тематик Джон фон Нейман. Перший такий комп'ютер був побудований в 1949 році і з тих пір всі сучасні комп'ютери створюються у відповідності з принципами Дж. фон Неймана.

В СРСР проект першого комп'ютера був створений групою під керівництвом академіка С.А.Лебедєва ще напередодні війни, але реалізований він був тільки в 1951 році. Ця машина отримала назву МЕСМ. Незабаром також під керівництвом С.А.Лебедєва в Києві були створені унікальні для свого часу комп'ютери БЕСМ-1 та М-20, які пішли в серійне виробництво. Це були в той час найпотужніші комп'ютери в Європі, які виконували 10 000 операцій за секунду. Для порівняння: сучасні комп'ютери виконують за одну секунду десятки і сотні мільйонів операцій.

Постійне удосконалення комп'ютерів докорінно змінює організаційні форми їх використання.

Перші комп'ютери використовувались лише в деяких провідних наукових установах для потреб фізики, авіації, оборонної та космічної промисловості. Це були дуже громіздкі та дорогі машини, що вимагали спеціальних приміщень, багато енергії, витратних матеріалів та великого штату обслуговуючого персоналу.

З розвитком обчислювальної техніки вартість комп'ютерів поступово знижувалась, збільшувалось їх виробництво. Комп'ютери почали застосовуватись в економіці, управлінні виробництвом та в інших галузях народного господарства. На цьому етапі створювались регіональні обчислювальні центри колективного користування, на яких встановлювались комп'ютери, вартість яких була ще досить високою. Це повністю виключало можливість придбання таких комп'ютерів підприємствами та господарствами, які змушені були опрацьовувати свою інформацію на відстані, для переважної більшості середніх та дрібних підприємств комп'ютери залишались недоступними.

Докорінно ситуація змінилась тільки з появою персональних комп'ютерів, які мають габарити звичайного телевізора і вартість, посильну навіть для невеликих господарств. Персональні комп'ютери прийшли безпосередньо на робочі місця спеціалістів різного профілю, дозволивши цим створити автоматизовані робочі місця (АРМи). Це дозволяє спеціалісту будь-якого профілю після невеликої попередньої підготовки ефективно використовувати їх для розв'язання різних задач.

Архітектура комп'ютера

Згідно з принципами Дж. фон Неймана комп'ютер виконує операції введення, зберігання, перетворення та виведення інформації, тобто перетворює введену інформацію деякого типу в результуючу інформацію того ж або іншого типу.

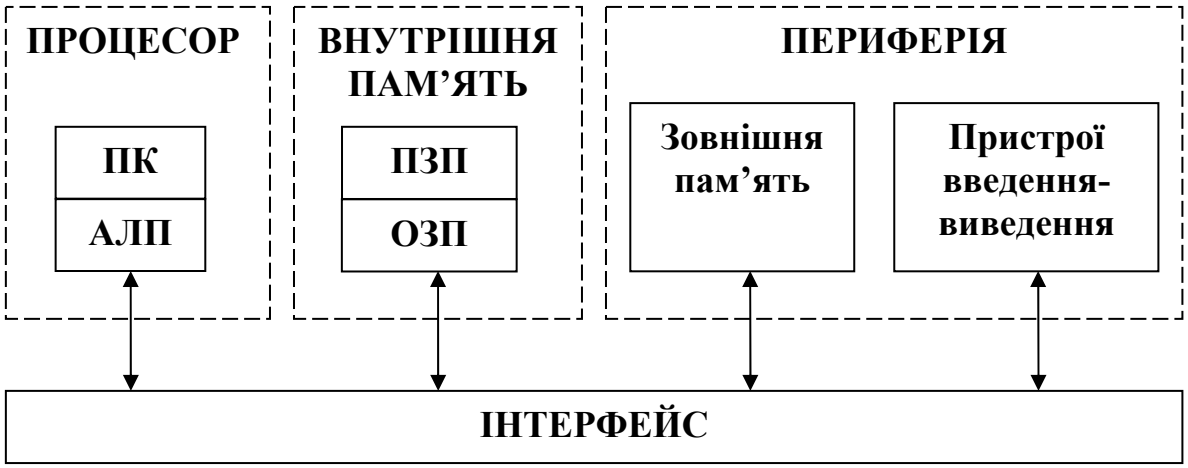


Рисунок 1.1 – Архітектура комп'ютера

Комп'ютер (рис.1.1) в своєму складі повинен мати такі пристрої:

- арифметико-логічний пристрій (АЛП);
- пристрій керування (ПК);
- запам'ятовувальні пристрої (пам'ять) для зберігання програм та даних;
- пристрої введення інформації;
- пристрої виведення інформації.

Комп'ютер автоматично виконує набір інструкцій, що називається комп'ютерною програмою.

За допомогою програми людина дає комп'ютеру наказ, які операції і в якому порядку слід виконувати, щоб отримати необхідний результат. Комп'ютер зберігає в пам'яті введену програму і у відповідності з нею виконує такі операції:

- із заданих областей пам'яті вибирає інформацію;
- за допомогою арифметичного блоку виконує над нею деякі дії;
- результати розміщує в назначену область пам'яті.

В сучасних комп'ютерах арифметико-логічний пристрій та пристрій керування об'єднані в єдиний блок – центральний процесор.

Центральний процесор (CPU) – це основна частина комп'ютера, яка керує усіма пристроями системи та безпосередньо виконує сам процес обробки інформації. Процесор призначений для:

- керування обміном інформацією між пристроями;
- обчислень за заданою програмою;
- задання режимів роботи пристроїв;
- організації введення-виведення інформації та ін.

Конструктивно всі пристрої процесора об'єднуються в один блок у вигляді однієї або декількох інтегральних схем.

Запам'ятовувальні пристрої (ЗП) – це технічні засоби, що забезпечують зберігання та видачу інформації. В цих пристроях зберігаються дані, які необхідно опрацювати в комп'ютері, програми обробки даних та результати роботи.

Пристрої введення-виведення інформації призначені для зв'язку з комп'ютером. Ці пристрої дозволяють вводити в комп'ютер дані та отримувати від комп'ютера результати його роботи, діалогові повідомлення та іншу інформацію.

Основні відомості про будову комп'ютера

Електронною обчислювальною машиною (ЕОМ) називається пристрій, що виконує такі операції:

- введення інформації;
- обробку інформації за закладеною програмою;
- виведення результатів обробки.

За кожну з названих дій відповідає спеціальний блок комп'ютера, відповідно: пристрій введення, центральний процесор (ЦП), пристрій виведення. Всі вони досить складні і, в свою чергу, складаються з окремих більш дрібних пристроїв. Зокрема, в центральний процесор входять арифметико-логічний пристрій та керуючий пристрій. Пристрій введення має також декілька конструктивних одиниць. Види інформації, що вводяться – різноманітні, джерел теж може бути декілька, так що термін "пристрій введення" слід розуміти у широкому смислі.

Те саме стосується і пристрою виведення.

Роботою пристроїв введення-виведення необхідно керувати. Раніше ці обов'язки покладались на ЦП, але вони віднімали у нього чимало часу. Архітектура сучасних комп'ютерів передбачає передачу значної частини

функцій керування периферійними пристроями спеціалізованим процесорам, які забезпечують тим самим розвантаження ЦП та підвищення його продуктивності. Отже реальна структура комп'ютера значно складніша, ніж зображена на схемі, за рахунок компонентів, призначених для підвищення продуктивності та наближення функціональних можливостей комп'ютера до потреб користувачів. Проте в цілому структура комп'ютера зберігається. Розглянемо її більш детально.

Центральний процесор комп'ютера

До складу центрального процесора входять пристрой керування (ПК) та арифметико-логічний пристрій (АЛП).

Пристрій керування (ПК) здійснює координацію роботи всіх компонентів комп'ютера. Він відбирає з пам'яті у визначеній послідовності команду за командою. Кожна команда:

- декодується;
- з вказаних в ній ділянок пам'яті передаються в АЛП (або навпаки) елементи даних;
- АЛП налаштовується на виконання дії, передбаченої командою (у цій дії, можуть брати участь і пристрої введення-виведення);
- дається команда на виконання цієї дії.

Цей процес буде продовжуватись до тих пір, доки не складеться одна з таких ситуацій:

- вичерпані вхідні дані;
- від одного з пристроїв введення надійшла команда на припинення роботи;
- вимкнуто живлення комп'ютера.

Арифметико-логічний пристрій – це саме те місце, де виконується обробка даних, визначена командами програми:

- арифметичні дії над числами;
- перетворення кодів;
- порівняння слів тощо.

Якість комп'ютера характеризується багатьма показниками. Це і набір інструкцій (команд), які комп'ютер здатний розуміти та виконувати, і швидкість роботи (швидкодія) центрального процесора, і кількість пристроїв введення-виведення (периферійних пристроїв), які можна приєднати до нього одночасно, і споживання електроенергії, і багато іншого. Та голов-

ною, як правило, характеристикою є швидкодія, тобто кількість операцій, які центральний процесор здатний виконати за одиницю часу.

Швидкість роботи елементів неможливо збільшувати безмежно (існують сучасні технологічні обмеження та обмеження, зумовлені фізичними законами). Тому розробники комп'ютерів шукають вирішення цієї проблеми шляхом удосконалення схемних рішень або удосконалення архітектури ЕОМ. Так з'явились мультипроцесорні комп'ютери, в яких декілька процесорів працюють одночасно, і, значить, продуктивність машини дорівнює сумі продуктивностей процесорів. У цьому випадку говорять про мультипроцесорну архітектуру. В особливо потужних комп'ютерах (такі машини можуть, наприклад, моделювати ядерні реакції з швидкістю природного процесу) число процесорів досягає декількох десятків.

Запам’ятовувальні пристрої комп’ютера

Класифікація запам'ятовувальних пристроїв (ЗП) може бути проведена за багатьма ознаками. Зокрема за напрямками спрямування інформаційних потоків запам'ятовувальні пристрої можуть поділятись на три класи:

- двонаправлені, що допускають читання та запис даних;
- умовно-постійні, призначені для зберігання інформації, яка не часто поновлюється;
- постійні, що допускають тільки читання інформації.

До першого класу відносяться такі види:

- зовнішній (ЗЗП);
- оперативний (ОЗП);
- надоперативний (НОЗП);
- пам'ять пристроїв введення-виведення інформації.

До другого класу відносяться умовно-постійні запам'ятовувальні пристрої (**УПЗП**).

До третього класу відносяться постійні запам'ятовувальні пристрої **(ПЗП).**

Основні характеристики ЗП – об'єм та швидкість обміну інформацією.

Зовнійшній запам'ятовувальнимй пристрій (ЗЗП) приназначений для довготермінового зберігання інформації у великих об'ємах. ЗЗП є

електромеханічним накопичувачем, в якому інформація зберігається на технічних носіях. Мета цих пристроїв – забезпечити швидкий пошук потрібної адреси на технічному носієві та обмін інформацією з цією ділянкою.

Технічними носіями інформації для ЗЗП є магнітні диски та плівки, а також оптичні та магніто-оптичні диски.

Магнітний диск (МД) – це диск, обидві сторони якого покриті феромагнітною речовиною. Записування і зчитування інформації здійснюється за допомогою електромагнітних головок. Інформація на диск наноситься у вигляді концентричних кіл – доріжок. Швидкість обміну інформацією залежить від швидкості обертання диска і досягається за рахунок можливості прямого переміщення головки на потрібну доріжку. Магнітні диски бувають двох видів.

1. **Дискети** – магнітні диски, виготовлені на гнучкій основі. Швидкість обміну інформацією з комп'ютером невисока, але за рахунок мініатюрності і простоти мають широке застосування.

2. Касетні магнітні диски (**вінчестери**) – диски, які разом із зовнішнім пристроєм становлять єдиний блок, що не дозволяє, як правило, замінювати один диск іншим. За рахунок герметичності цього блоку, точності та мініатюрності елементів досягається велика щільність запису і швидкість обміну з комп'ютером.

Магнітна плівка (МП) – це стрічка, покрита ферромагнетиком. Обмін інформацією також здійснюється за допомогою електромагнітних головок. Швидкість обміну інформацією МП з ЕОМ набагато нижча, ніж у МД через відсутність прямого доступу до інформації (на сьогодні майже вийшли із вжитку).

Використовується МП в *стримерах* – пристроях для швидкого збереження всієї інформації, що знаходиться на вінчестері. Вони записують інформацію на касети, що схожі на касети до побутових магнітофонів.

Оперативний запам'ятовувальний пристрій (**ОЗП**, **RAM, оперативна пам'ять**)

Ємність **ОЗП** досягає сотень, а інколи і тисяч Мбайт. **ОЗП** використовується для зберігання інформації та програм, що безпосередньо беруть участь в роботі в даний момент, а також для тимчасового зберігання якихось частин вхідних даних та проміжних результатів.

Йому властиві:

- здатність записувати (або зчитувати) елементи програм та даних у довільне місце пам'яті (або з довільного місця пам'яті);

- висока швидкодія.

Термін "довільне місце" означає можливість звернутись до заданої адреси пам'яті без необхідності перегляду всіх попередніх.

При відімкненні комп'ютера від електромережі інформація, що зберігається в оперативній пам'яті, втрачається. Тому інформацію, яку необхідно зберегти для подальшого використання, слід переносити із ОЗП на зовнішній запам'ятовувальний пристрій.

Швидкість роботи комп'ютерів істотно залежить від швидкості роботи ОЗП. Тому постійно ведуться пошуки елементів для ОЗП, які вимагали б якомога менше часу на операції зчитування-записування. Проте виявилось, що разом з швидкодією росте (та дуже швидко) вартість елементів пам'яті, так що побудова ОЗП необхідної ємності на швидких елементах неприйнятна економічно. Ця проблема вирішена шляхом побудови багаторівневої пам'яті. ОЗП складається з двох-трьох частин: основна частина великого об'єму будується на відносно повільних (але зате більш дешевих) елементах, а додаткова частина (її називають кеш-пам'ять) складається з швидкодійних елементів. Ті дані, до яких АЛП звертається найбільш часто, містяться у кеш-пам'яті; більший об'єм оперативної інформації зберігається у основній пам'яті. Розподілом інформації між складовими частинами ОЗП керує спеціальний блок центрального процесора (ЦП). Об'єм ОЗП та кеш-пам'яті належить до числа найважливіших характеристик комп'ютера.

Надоперативний запам'ятовувальний пристрій (НОЗП, кеш, Cach memory), як зазначалось вище, призначений для підвищення швидкості роботи комп'ютера шляхом усунення диспропорції між швидкостями роботи процесора, основної частини ОЗП та ЗЗП. Деякі зовнішні пристрої (принтери, дисплеї тощо) мають свою власну оперативну пам'ять, яка входить або безпосередньо до їх складу, або до складу електронних схем комп'ютера, що керують роботою цих пристроїв.

Умовно-постійний запам'ятовувальний пристрій (УПЗП, CMOS) використовується для зберігання параметрів конфігурації комп'ютера. Зміст **УПЗП** задається спеціальною програмою Setup і не змінюється при відімкненні електроживлення комп'ютера.

Постійний запам'ятовувальний пристрій (ПЗП, ROM) – це ЗП, призначений для зберігання та зчитування інформації. Запис інформації в них здійснюється, як правило, один раз на підприємстві-виробнику. **ПЗП** в процесі обчислень використовуються тільки для зчитування інформації. Існують ПЗП у вигляді мікросхем, які монтуються на основній (материнській) платі комп'ютера, та оптичних дисків, інформація з яких читається за допомогою спеціального технічного пристрою (**CD ROM**).

Пристрої введення та виведення інформації

Пристрої введення за принципом роботи поділяються на три види: 1) ручного введення; 2) з технічних носіїв; 3) безпосереднього введення.

Для ручного введення інформації служить клавіатура.

До пристроїв введення з технічних носіїв відносяться зонішні запам'ятовувальні пристрої та телекомунікаційні пристрої.

Пристрої безпосереднього введення інформації дозволяють комп'ютеру зчитувати дані з друкованого тексту, з графічних зображень (сканери) та сприймати інформацію з голосу людини. Вони дозволяють швидко вводити інформацію, обминаючи попередню підготовку даних перед введенням їх в комп'ютер.

Пристрої виведення бувають двох видів:

- пристрої, що виводять інформацію для сприйняття людиною;
- пристрої, що виводять інформацію на технічний носій.

Перший вид *пристроїв виведення* подає інформацію на екран, на папір, або у вигляді звуку.

Для виведення інформації на екран використовуються дисплеї. Вони бувають кольоровими та монохромними і можуть працювати в одному з двох режимів: в текстовому або графічному. В текстовому режимі екран поділяється на окремі позиції, що складають 25 рядків по 80 позицій у кожному. Графічний режим дисплея призначений для виведення графіків, рисунків і т.д. В цьому режимі можна виводити також і текстову інформацію у вигляді букв різних шрифтів. В графічному режимі екран складається із окремих точок, кількість яких по вертикалі та горизонталі визначає якість зображення та тип дисплея: **CGA, EGA, VGA**, **SVGA** або **XGA**. Дисплеї **CGA** та **EGA** практично не використовуються, дисплеї **VGA** застосовуються все рідше, а дисплеї **SVGA** є сьогодні найпоширенішими. Дисплеї **SVGA** та **XGA** мають дещо більшу вартість, але вони за-

безпечують найвищу якість зображення за рахунок того, що відстані між сусідніми точками зображення по вертикалі та горизонталі є однаковими. Найвищу якість забезпечують дисплеї, які мають на екрані сітку розміром 1024x768 точок. Перспективними на сьогодні є дисплеї, які мають плоский екран та побудовані з використанням рідких кристалів. Пристрої виведення на папір поділяються на графічні, псевдографічні (матричні), струминні, лазерні, термічні та символьні.

Графічні пристрої виводять інформацію у вигляді графіків, креслень або схем. Вони називаються графопобудовниками або плотерами. Зображення формується шляхом переміщення спеціального фарбувального "олівця". Графопобудовники дають можливість виводити інформацію різними кольорами.

Псевдографічні пристрої виводять інформацію за допомогою друкувальної матричної головки, яка містить вертикальний ряд тонких металевих голок. Головка рухається вздовж рядка, а голки в потрібний момент ударяють папір через фарбувальну стрічку, що забезпечує формування на ньому символів та зображень у вигляді сукупності дрібних точок. Ці пристрої називаються матричними принтерами.

В струминних пристроях зображення формується мікрокраплями різнокольорових чорнил, що видуваються на папір за допомогою сопел.

Лазерні пристрої використовують принцип ксерографії. Зображення переноситься на папір зі спеціального барабана, до якого електричним способом притягуються частинки фарби. Друкувальний барабан електризується лазером, який керується комп'ютером.

Термічні пристрої побудовані на основі термоелементів, що нагрівають фарбувальні стрічки або спеціальний термопапір. Така технологія є досить дорогою, але вона дозволяє отримувати високоякісні зображення.

Символьні пристрої виводять інформацію звичайним друкованим способом і називаються алфавітно-цифровими пристроями друку. В цих пристроях рядок знаків друкується одночасно по всій довжині. Вони мають складну схему, великі габаритні розміри та високу швидкість друку.

Виведення інформації у вигляді звуку здійснюється за допомогою спеціальних електронних схем (звукових адаптерів) та аудіосистем.

Розглянемо тепер технічні носії інформації, які використовуються пристроями виведення.

Технічні носії інформації

Призначення будь-якого ***технічного носія інформації*** – зберігання даних, що записані на ньому. В комп'ютерах дані передаються, зберігаються та перетворюються в цифровій формі, де кожне повідомлення (символ, число, звук) подаються у вигляді деякого числа. Тому всі сучасні комп'ютери називаються цифровими. Цифри, що описують повідомлення в комп'ютерах, повинні мати деяку фізичну реалізацію. В комп'ютерах використовуються елементи, які мають два стани. Наприклад:

- є напруга – немає напруги;
- намагнічена ділянка – розмагнічена ділянка;
- відбивається промінь лазера від поверхні – не відбивається.

Тому робота комп'ютерів основана на застосуванні бінарного (двійкового) коду – мови, в алфавіті якої є тільки два символи "1" і "0", що можна відповідно розглядати як "так" і "ні". Бінарний код зручний для реалізації, оскільки відповідає основному принципу дії перемикачів.

Одиниця інформації, яка описує два стани "так" і "ні", прийнята за мінімальну одиницю інформації – 1 біт. Основною ж одиницею інформації в комп'ютері є 1 байт, який складається із восьми бітів. Байт дає можливість описувати різні числа та символи шляхом формування різних комбінацій 0 і 1.

Технічні носії поділяються на магнітні, оптичні, магнітооптичні та паперові.

В магнітних носіях записування і зчитування інформації здійснюється електромагнітними сигналами. До магнітних носіїв відносяться магнітні плівки та магнітні диски.

На оптичний носій (компакт-диск) інформація записується тільки один раз при його виготовленні. Компакт-диск виготовляється із алюмінію тому запис даних досить легко здійснюється за допомогою преса, який наносить на поверхню диска доріжки з інформацією. Існують також спеціальні компакт-диски, на які інформація може записуватись променем лазера. Зчитується інформація з компакт-дисків променем лазера. Ємність сучасних компакт-дисків до 8,7 Гбайт.

Магнітооптичні носії поєднують в собі переваги магнітної та оптичної технології. Запис інформації на диск здійснюється променем лазера в магнітному полі. За рахунок цього досягається висока

продуктивність та надійність інформаційного обміну. Ємність сучасних магніто-оптичних дисків від 128 Мбайт до 69 Гбайт.

На паперові носії інформація заноситься у вигляді пробивок у визначених місцях і читається за допомогою фотоелементів. Швидкість обміну інформацією у паперових носіїв з комп'ютером невелика. До паперових носіїв відносяться перфострічки та перфокарти. Паперові носії більш надійні, але вони мають два суттєвих недоліки:

- неможливість стирання старої і запису на них нової інформації;
- малу швидкість введення даних в комп'ютер.

Ці недоліки призвели до того, що в наш час паперові носії витіснені іншими.

Контрольні питання

1. В чому Ви бачите протиріччя та спільність в розвитку програмного та апаратного забезпечення?
2. Що є спільного і в чому Ви бачите різницю між поняттями *програмне забезпечення* та *інформаційне забезпечення* засобів електронно-обчислювальної техніки?
3. Які види робіт, що є характерними для великого промислового підприємства (наприклад, машинобудівного), можуть бути автоматизовані за допомогою електронно-обчислювальних машин? Які категорії програмних засобів для цього є необхідними?
4. Які категорії програмного забезпечення можуть бути використані в роботі малого підприємства і з якою метою?
5. Основні поняття інформатики. Інформація.
6. ЕОМ та історія їхнього розвитку. Роль вчених в розвитку інформатики та комп’ютерної техніки.
7. Архітектура ЕОМ.
8. Центральний процесор та його склад. Основні функції та властивості його складових.
9. Запам’ятовувальні пристрої комп’ютера та їх призначення.
10. Оперативний ЗП.
11. НЗП (надоперативний), умовно постійний ЗП.
12. Пристрої введення- виведення інформації та їх характеристики.
13. Технічні носії інформації та їх характеристики (магнітні, магнітооптичні, паперові).

2 ПЕРСОНАЛЬНИЙ КОМП'ЮТЕР

Перші моделі комп'ютерів фірми IBM поділялися на три покоління: IBM PC; IBM PC/2; IBM PC/3. Цей розподіл визначався типом основного мікропроцесора (CPU). До першого покоління відносили моделі IBM PC XT та IBM PC AT, що побудовані на основі мікропроцесорів відповідно Intel-8088 та Intel-80286, до другого – моделі, що побудовані на основі мікропроцесорів Intel-80386, Intel-80486, Intel-5x86, а до третього – Pentium.

Основний мікропроцесор визначає продуктивність комп'ютера та його функціональні можливості. Він монтується на материнській платі комп'ютера (MB).

Продуктивність основного мікропроцесора визначає область застосування комп'ютера. В наш час комп'ютери IBM PC не використовуються. Найбільше поширення набули комп'ютери побудовані на основі процесорів фірм Intel та AMD моделей Intel Celeron, Intel Pentium, AMD Athlon, AMD Sempron.

Склад персонального комп'ютера

Персональний комп'ютер (ПК) складається з декількох агрегатів (блоків), з'єднаних з'єднувальними кабелями. Номенклатура блоків може змінюватись, але в мінімальний комплект входять: системний блок, клавіатура, монітор та, як правило, маніпулятор.

Як додаткові пристрої можуть бути: принтер, додатковий накопичувач тощо. Докладніше про це буде сказано далі. В наступних підрозділах розглядаються пристрої агрегатів персонального комп'ютера, включаючи і найбільш поширені зовнішні пристрої.

Системний блок стаціонарного ПК це – прямокутний каркас, в якому розміщені всі основні вузли комп'ютера:

- материнська плата;
- адаптери;
- блок живлення;
- 1-2 дисководи для гнучких магнітних дисків (НГМД);
- один (значно рідше – два) дисковод на жорстких магнітних дисках (НЖМД) – вінчестер;
- дисковод для компакт-дисків (CD ROM);
- динамік;
- органи керування (електроживлення, кнопка перезавантаження, перемикач тактової частоти, індикатори живлення та режимів роботи).

Материнська (системна) плата. Так називають найбільшу плату, на якій розміщуються найголовніші компоненти комп'ютерної системи: центральний мікропроцесор, оперативна пам'ять, мікросхеми підтримки, центральна магістраль або шина, контролер шини та декілька гнізд. Останні (часто їх називають слотами, від англійського slot – щілина) служать для під'єднання до материнської плати інших плат (контролери, плати розширення та ін.). Частина слотів в стандартній комплектації ПК залишаються вільними. Окрім того, на материнській платі знаходяться мініатюрні перемикачі, за допомогою яких здійснюється налаштування електричної схеми плати. На системній платі розташовані також порти, до яких за допомогою спеціальних кабелів під'єднуються додаткові пристрої.

Мікропроцесор є мініатюрною обчислювальною машиною, що розміщена в одній надвеликій інтегральній схемі (НВІС). На одному кристалі надчистого кремнію (сіталу) за допомогою складного, багатоступеневого та високоточного технологічного процесу створено декілька мільйонів транзисторів та інших схемних елементів, з'єднувальні провідники та точки під'єднання зовнішніх виведень. В сукупності вони утворюють всі логічні блоки, тобто арифметичний пристрій, пристрій керування, регістри і т.ін.

Світовою промисловістю випускаються різні типи центральних мікропроцесорів. В таблиці показані деякі з процесорів, що випускаються та випускались в недавньому минулому компаніями Intel та AMD – лідерами в цій галузі.

Основними параметрами мікропроцесорів є:

- набір команд, що виконуються;
- тактова частота;
- розрядність.

Тактова частота вказує, скільки елементарних операцій (тактів) мікропроцесор виконує за одну секунду. Тактова частота вимірюється у гігагерцах (1 ГГц = 1 000 000 000 Гц).

Розрядність показує, скільки двійкових розрядів (бітів) інформації обробляється (або передається) за один такт, а також – скільки біт може бути використано в процесорі для адресації оперативної пам'яті.

Тактова частота служить тільки відносним показником продуктивності процесора, оскільки схемні розбіжності процесорів призводять до того, що в деяких з них за один такт виконується робота, на яку інші витрачають декілька тактів.

Таблиця 2.1 – Моделі центральних мікропроцесорів

Позначення	Розрядність шини даних	Тактова частота, МГц	Адресовна пам’ять
I8008	8	0.8	16 Кбайт
I80486DX	32	25 … 66	64 Кбайт
Pentium	32	200	256 Кбайт
Celeron D345	32	3000	256 Кбайт
Sempron 2800	32	1600	256 Кбайт
Athlon 64 3000	64	1800	512 Кбайт
Pentium IV-630	32	3000	1 Гбайт

Зовні мікропроцесор виглядає як прямокутна пластмасова пластина розмірами 5×5×0,5 см з численними виведеннями (до 240). На сучасні високошвидкісні мікропроцесори установлюється вентилятор, необхідний для охолодження.

Оперативний запам'ятовувальний пристрій (ОЗП, RAM) також реалізований на НВІС. Швидкість доступу, тобто час, необхідний для зчитування даних із ОЗП або запису їх туди, в сучасних ОЗП складає близько 60 нс (60×10^{-9} с). Існують два типи НВІС пам'яті: а) статична; б) динамічна.

Статична пам'ять будується на так званих тригерних схемах. Будучи встановлена вхідним імпульсом у один з двох можливих станів (0 або 1), така схема зберігає його до наступного імпульсу або до вимикання живлення. При зчитуванні записаного в пам'ять значення її стан теж не змінюється. Інакше працює динамічна пам'ять: вона складається з мікроскопічних конденсаторів, кожен з яких може перебувати у стані "заряджений" (що символізує двійкову одиничку) або "незаряджений" (двійковий нуль). Щоб зберігати дані в такій пам'яті, заряджені конденсатори необхідно періодично "підживлювати". Тому динамічний ОЗП при інших рівних умовах істотно повільніший статичного. Але зате він менш енерговитратний. Слід підкреслити, що обидва види пам'яті зберігають дані тільки при постійному електроживленні, тобто є енергозалежними.

Дані в цій пам'яті стираються після вимикання або перезавантаження комп'ютера.

Конструктивно сучасна НВІС ОЗП (наприклад, SIMM, single in-line memory module) є невеликою платою з розміщеними на ній мікросхемами. Останнім часом у основному застосовуються 168-контактні (168-pin). Окрім ОЗП, в сучасних ПК обов'язково є надоперативна (кеш, cache) пам'ять. Вона призначена для узгодження швидкості роботи повільних пристроїв з більш швидкими, наприклад мікропроцесорами та динамічною пам'яттю.

Кеш-пам'ять буває двох рівнів:

а) кеш-пам'ять першого рівня розміром до 1024 кбайт вбудовується безпосередньо в мікропроцесор;

б) кеш-пам'ять другого рівня розміром до 1024 Мбайт (та більше) розміщується (як правило) на системній платі.

Системна магістраль даних (системна шина) – це група електричних з'єднань (провідників) для передавання даних, адрес та сигналів між різними компонентами комп'ютера. Для забезпечення взаємозамінності пристроїв, що виготовляються різними виробниками, кількість, призначення та розміщення цих провідників стандартизовані.

Ще один важливий елемент з числа встановлюваних на материнській платі – це мікросхема BIOS (Basic Input-Output System, базова система введення-виведення). Вона є енергонезалежним постійним запам'ятовувальним пристроєм, в який записані програми, що реалізують функції введення-виведення, а також програма тестування комп'ютера в момент ввімкнення живлення (POST, Power-OnSelf-Test) та ряд інших спеціальних програм.

В своїй роботі BIOS користується даними про апаратну конфігурацію комп'ютера, які зберігає ще одна мікросхема – CMOS RAM (Complementary Metal-OxideSemiconductor RAM). Ця енергозалежна пам'ять постійно підживлюється від батарейки, яка теж знаходиться на материнській платі. Та ж батарейка живить і схему кварцового годинника, що безперервно відліковує час та поточну дату.

Блок живлення (БЖ) перетворює змінний струм мережі електроживлення в постійний струм низької напруги. БЖ має декілька виходів на різні напруги (3.5, 5 та 12 В), які забезпечують живленням відповідні пристрої комп'ютера. Електронні схеми блока живлення підтримують ці

напруги сталими незалежно від коливань мережевої напруги в досить широких межах (від 180 до 250 В).

Накопичувачі **–** це запам'ятовувальні пристрої, призначені для тривалого (тобто енергонезалежного) зберігання великих об'ємів інформації. Розрізняють накопичувачі із змінними та незмінними носіями. Дисковий носій може мати жорстку або гнучку основу. Гнучкі носії для магнітних накопичувачів у наш час випускаються у вигляді дискет. Власне носій є плоским диском із спеціальної плівки (майлара), з достатньою міцністю та сталістю розмірів. Він покритий феромагнітним шаром та розміщений у захисному конверті (оболонка дискети). В таблиці наведені характеристики дискет.

Таблиця 2.2 – Дискети

Розмір дискети	Тип оболонки	Об'єм, Мбайт	Ціна, $	Примітка
3,50 дюйма	Пластмаса	до 2,88	0,2..1,0	Виходять із використання

Накопичувач на жорстких магнітних дисках (НЖМД) – це пристрій з незмінним носієм. Його конструктивна схема схожа із схемою НГМД, але реалізація істотно відрізняється. Від НЖМД вимагається в сотні разів більші об'єми та швидкість обміну даними. Тому інформація записується не на один диск, а на пакет дисків, що складається з декількох пластин, ідеально плоских та з відшліфованим феромагнітним шаром. При цьому запис здійснюється на обидві поверхні кожної пластини (окрім крайніх).

Отже, працює не одна, а група магнітних головок, зібраних у єдиний блок. Доки комп'ютер увімкнений, пакет дисків обертається безперервно з великою швидкістю (до 7200, а у окремих моделях до 10000 об/хв), і тому механічний контакт головок та дисків неприпустимий. Кожна головка "плаває" над поверхнею диска на відстані 0,5…0,13 мікрометра. Проникнення в такий механізм найдрібнішого пилу вивело б його з ладу, тому електромеханічна частина накопичувача розміщується в герметичному корпусі. Технічні характеристики деяких сучасних та перспективних НЖМД наведені в таблиці.

Таблиця 2.3 – Накопичувачі на жорстких магнітних дисках

Модель	Інтерфейс	Ємність, Гбайт	Швидкість обертання, об/хв	Час доступу, мсек
Quantum Bigfoot 1280	IDE	1,3	3600	15.5
Seagate Medalist 2140	IDE	2,1	5400	10
Seagate Barracuda 4LP	SCSI	2,2	7200	8
Samsung C100	SCSI	40,9	5400	8
Seagate Barracuda 200NI	SCSI	40,9	7200	8
WD 80FG	SCSI	80	7200	8
Samsung D600	SCSI	120	7200	8
Seagate Barracuda 350BG	SCSI	160	7200	8
Maxtor	SCSI	200	7200	8
Hitachi	SCSI	250	7200	8
Seagate Serial ATA	SCSI	300	7200	8
WD Serial ATA	SCSI	300	7200	8

Накопичувачі продукуються десятками фірм-виробників. Щоб забезпечити взаємозамінність пристроїв, розроблені стандарти на їх габаритні та електричні характеристики. Останні визначають, наприклад, номенклатуру з'єднувальних провідників, їх розміщення в перехідних гніздах, електричні параметри сигналів (тобто те, що називають інтерфейсом). На сьогодні найбільш вживані стандарти IDE (Integrated Drive Electronics) та більш продуктивні EIDE (Enhanced IDE) та SCSI (Small Computer System Interface). EIDE відрізняється від свого попередника (IDE) тим, що допускає під'єднання до одного адаптера до п’ятнадцяти пристроїв, підтримує накопичувачі з величезним об'ємом, обслуговує і немагнітні накопичувачі, допускає підвищені швидкості обміну даними.

Адаптери. Форми подання даних та керуючих сигналів, що використовуються в різних пристроях ПК, досить різні. Для підтримки взаємодії пристроїв необхідно виконувати перетворення форм подання інформації. Цю задачу вирішують спеціальні пристрої, що називаються адаптерами.

Конструктивно вони оформляються у вигляді плат, які мають стандартне з'єднання з шиною та специфічне з'єднання з відповідним пристроєм.

Сьогодні в номенклатурі адаптерів стійко фігурують:

- відеоадаптери (вони ж – відеоплати, відеокарти);
- адаптери (контролери) дисководів;
- адаптери портів введення-виведення;
- звукові плати (аудіокарти);
- адаптери мережі (плати мережі);
- модеми.

Відеоадаптер – це пристрій, що перетворює набір даних у відеосигнал, який надходить до монітора по кабелю для відображення на екрані. Зображення на екрані монітора складається з окремих точок, що формуються електронним променем, траєкторія якого на екрані утворює декілька сот горизонтальних ліній. Керуючи інтенсивністю електронного променя, можна керувати яскравістю точки. Таким чином, для створення на екрані потрібного зображення (наприклад, літери) необхідно відповідним чином модулювати електронний промінь, а для цього коди символів, що зберігаються в ОЗП потрібно заздалегідь перетворювати у відеосигнал.

Зображення на екрані монітора потрібно регенерувати (відтворювати) як мінімум 25…30 разів у секунду, кожного разу формуючи заново відеосигнал, звертаючись до оперативної пам'яті за вихідними даними. В дійсності частоту регенерації роблять ще більш високою (до 100 Гц), з тим щоб ослабити мерехтіння та знизити втому очей оператора. Щоб цей процес не заважав роботі центрального процесора, дані для виведення зберігаються в спеціально виділеній для цього відеопам'яті. Конструктивно це може бути реалізовано у формі виділення ділянки основного ОЗП або (як правило) включенням спеціальної мікросхеми пам'яті до складу відеоадаптера. Вимоги до об'єму відеопам'яті зростають з підвищенням якості зображення та кількості відтворюваних кольорів. Видеоадаптер, звичайно, розміщується в системному блоці комп'ютера.

Контролер дисків призначений для керування роботою механічних рухомих частин дисководів та формування електричних імпульсів при операціях запису та читання.

Контролер введення-виведення або адаптер портів, є пристроєм, що обслуговує різні зовнішні пристрої такі, як принтери, маніпулятори тощо. Приєднання їх до процесорного блока здійснюється через спеціальні

схемні елементи, що називаються портами. Розрізняють паралельні та послідовні порти.

Паралельний порт дозволяє передати за один такт принаймні один байт, оскільки кожному біту виділений один провідник (один контакт) і, таким чином, всі складові байта передаються одночасно, паралельно.

Послідовний порт отримує для передавання даних тільки одну пару провідників, і тому біти, що утворюють сигнал, проходять через порт послідовно. Послідовний порт використовують не тільки для цифрового передавання, але і для передавання некодованих сигналів (наприклад, від маніпулятора).

Звукова плата (аудіокарта) дає можливість за допомогою додаткових аудіосистем комп'ютера відтворювати з високою якістю музичні твори та різноманітні звукові ефекти. Без цього адаптера комп'ютер може виводити в кожний момент звук тільки одного тону.

На платах деяких аудіокарт іноді розміщують додаткову електронну схему (радіотюнер), яка дозволяє приймати на аудіосистеми радіопередачі. Телевізійний тюнер дає можливість не тільки відтворювати сигнал з радіоефіру, але також записувати його на магнітний диск та модифікувати.

Плата мережі. Цей адаптер призначений для з’єднання персонального комп'ютера з фізичним каналом передавання даних, наприклад, з коаксіальним кабелем. Він дає можливість під'єднати комп'ютер до локальної мережі. При цьому користувач може отримати доступ до даних, що знаходяться в інших комп'ютерах.

Модем – пристрій, що дозволяє використовувати звичайну телефонну мережу для передавання інформації між комп'ютерами, які можуть бути віддалені один від одного на тисячі кілометрів.

Монітор (дисплей) комп'ютера призначений для відображення текстової та графічної інформації. Монітори бувають кольоровими та монохромними. Вони можуть працювати у одному з двох режимів: текстовому або графічному. Сучасні монітори працюють в графічному режимі.

В текстовому режимі екран монітора умовно поділяється на окремі ділянки – позиції (частіше за все на 25 рядків по 80 символів). На кожній позиції може бути відображений один з 256 зазделегідь визначених символів. В число цих символів входять великі та малі літери, цифри і т.ін.

Загальна номенклатура символів, що відтворюються на екрані (вона залежить від системи кодування символів), не обмежена числом 256. Од-

ному і тому ж коду залежно від режиму (латинський алфавіт, кирилиця, псевдографіка) на екрані монітора можуть відповідати різні символи.

В графічному режимі екран монітора є растром, що складається з точок (пікселів). Кількість точок по горизонталі та вертикалі, які монітор здатний відтворити чітко та роздільно, називається роздільною здатністю монітора. Вираз "роздільна здатність 800 × 600“ означає, що монітор може виводити 600 горизонтальних рядків по 800 точок у кожному рядку. Ця властивість монітора визначається, зокрема такою його характеристикою, як розмір точки (зерна) екрана. Не слід думати, що потенційна роздільна здатність монітора реалізується автоматично. Реальна роздільна здатність відеотракту залежить ще і від відеокарти.

Існують два основних типи моніторів:

а) рідкокристалічні;

б) з електронно-променевою трубкою.

Монітори на рідких кристалах мають (при інших рівних умовах) на порядок меншу вагу та геометричний об'єм, споживають на два порядка менше енергії, але зате вони приблизно у 3 рази дорожчі.

В сучасних ПК використовуються спеціальні апаратно-програмні способи для продовження строку служби екрана монітора та заощадження електроенерії. Якщо протягом визначеного часу користувач не здійснює ніяких дій, то спочатку операційна система видає команду на виведення на екран спеціальної картинки, а ще пізніше – на переведення електроживлення монітора у "сплячий" режим. Живлення зберігається тільки для логічних схем, для того щоб монітор міг увімкнутись знову.

Клавіатура призначена для введення в комп'ютер інформації та команд.

Маніпулятори. Під час роботи комп'ютера на екран монітора виводиться курсор (пульсуючі риска, прямокутник, стрілка та ін.), який відіграє важливу роль в організації діалога користувача та комп'ютера: курсор відзначає місце на екрані, куди потрапить черговий введений символ, вказує на програмне вікно, яке треба активізувати і т.д.

Користувач може переміщувати курсор в потрібне місце, використовуючи клавіші керування курсором, зокрема клавіші із стрілками. Проте це не завжди зручно, а якщо задіяне графічне операційне середовище (Windows) – то незручно зовсім. Значно ефективніше йде робота при використанні маніпуляторів, тобто спеціальних пристроїв для керування курсо-

ром та подачі деяких команд. Найбільш вдалими виявились маніпулятори "**миша**", "**трекбол**" та "**джойстик**" (перший поширений значно більше).

"Миша" – маніпулятор для керування курсором. "Миша" є невелика "коробочка" із пластмаси. Щоб змінити положення курсора на екрані необхідно перемістити "мишу" по плоскій поверхні стола. При цьому кулька, що розташована знизу усередині корпусу "миші", обертається і механічно пов'язані з нею мініатюрні датчики генерують дві серії електричних імпульсів. Число імпульсів в одній серії відзначає шлях "миші" по осі X, інша серія описує шлях, пройдений по осі Y. На підставі цих імпульсних послідовностей визначається положення курсора на екрані. Окрім того, "миша" має дві (або три, або замість третьої кнопки встановлено до трьох коліщаток для “прокручування” інформаціїї на екрані монітора) кнопки. Натискуючи їх користувач може подавати команди. Смисл команд визначається операційною системою, що використовується, та програмним продуктом. Широке розповсюдження мають сьогодні оптичні бездротові “миші”.

Плати розширення. Так прийнято називати додаткові електронні пристрої, які не входять в комплект поставки ПК і купуються власником ПК пізніше з метою розширення функціональних можливостей машини. Конструктивно такий пристрій є платою стандартної форми із стандартним гніздом; на платі встановлені необхідні мікросхеми та інші електронні компоненти. Плата вставляється у вільний слот материнської плати і після необхідного налагодження використовується для роботи. Платами розширення можуть бути: модем, звукова плата та ін.

Додатковими називають ***пристрої***, які розміщені поза системним блоком. Насамперед це пристрої фіксації результатів: принтери, плотери, графопобудовники, а також модеми, стримери, сканери, проекційні панелі та ін.

Термін "додаткові пристрої" досить умовний. До них можна віднести, наприклад, накопичувач на компакт-дисках, якщо він виконаний у самостійному корпусі і приєднується спеціальним кабелем до зовнішнього гнізда системного блока. І навпаки, модем може бути конструктивно оформлений як плата розширення, і тоді немає підстав вважати його додатковим пристроєм.

Принтерами називають пристрої, призначені для виведення на "тверді" носії (головим чином – на папір) результатів роботи програм. Існує велика кількість різних моделей принтерів, що відрізняються прин-

ципом дії, інтерфейсом, продуктивністю, функціональними можливостями. Виробництвом принтерів зайняті десятки найвідоміших фірм світу.

Основу принтера складає електромеханічний агрегат, що забезпечує формування зображення, переміщення паперу, подачу фарбника та ін. Окрім того, до складу принтера входить ще і електронна частина, що складається із схеми керування (СК) та буферного запам'ятовувального пристрою (БЗП).

СК здійснює інтерпретацію команд, що надходять від комп'ютера, і керування локальними операціями (наприклад, заправкою паперу в друкарський тракт), а БЗП зберігає чергову порцію інформації для виведення.

За способом формування зображень принтери поділяються на контурні (ударні) та растрові.

В контурних принтерах зображення символа формується шляхом удару деталі з рельєфом відповідної форми об папір через фарбувальну стрічку (як це відбувається, наприклад, у друкарських машинках).

В растрових принтерах зображення утворюється множиною дрібних (0,1...0,3 мм) точок, нанесених на папір у необхідному порядку. Перехід до растрового принципу друку має фундаментальне значення, оскільки він дав можливість принтеру виводити на папір не тільки текст, але і графіку. І саме текстове виведення збагатилось різноманітними можливостями, які дозволяють комбінувати в одному документі шрифти різних розмірів, типів і т.ін.

В сучасних ПК застосовуються тільки растрові друкарські пристрої. За способом нанесення фарбувальних точок їх можна поділити на три основних види:

- матричні;
- струминні;
- лазерні.

Матричні принтери. Основним вузлом матричного принтера є друкувальна головка, у вигляді сукупності тонких металевих "голок", які розміщені вертикально і перпендикулярно до паперу. Головка рухається вздовж рядка, а голки в потрібний момент ударяють папір через фарбувальну стрічку. Це і забезпечує формування на папері символів та інших зображень.

В дешевих моделях принтерів використовується головка, що друкує дев’ятьма голками. Якість друку у таких принтерів посередня. Більш якісний та швидкий друк забезпечується принтерами з 24 та 48 голками.

Швидкість друку матричних принтерів – від 60 до 200 секунд на сторінку. З експлуатаційної точки зору матричні принтери характеризуються невибагливістю до якості паперу та можливістю відразу одержати декілька копій документа (через копіювальний папір). Але у них найбільший рівень шуму.

Струминні принтери. В цих принтерах зображення формується мікрокраплями спеціальних чорнил, що викидаються на папір через мініатюрні "сопла". Цей спосіб друку забезпечує більш високу якість в порівнянні з матричними принтерами, у тому числі дозволяє простіше реалізувати кольоровий друк.

Струминні принтери практично не створюють шуму. Проте вони дорожчі матричних і вимагають більш ретельного догляду та обслуговування, більш вимогливі до якості паперу.

Швидкість друку струминних принтерів дещо більша, ніж у матричних, – від 4 до 60 секунд на сторінку.

Лазерні принтери забезпечують на сьогодні найкращу (близьку до типографської) якість друку. В цих принтерах для друку використовується принцип ксерографії: зображення спочатку формується на спеціальному барабані у вигляді сукупності електричних зарядів. До заряджених ділянок поверхні барабана прилипає тонкодисперсний фарбник, утворюючи зображення. Потім воно відбитком переноситься на папір та закріпляється на ньому потужним, але короткочасним прогріванням. Відмінність від звичайного ксерокопіювального апарату полягає в тому, що електричний рельєф на друкувальному барабані формується за допомогою лазера, промінь якого модулюється командами з комп'ютера.

Роздільна здатність лазерних принтерів – від 300 точок на дюйм (тобто розмір точки ~0,08 мм) до 600 та більше точок на дюйм. Швидкість друку лазерних принтерів – від 3 до 15 секунд на сторінку при виведенні текстів. Сторінки з рисунками можуть виводитись значно довше: на виведення великих рисунків може витрачатись декілька хвилин.

Модем – це пристрій з’єднання комп'ютера із звичайною телефонною лінією. Комп'ютер продукує дискретні електричні сигнали (тобто послідовності двійкових 0 та 1) , а телефонними лініями інформація передається в аналоговій формі (тобто у вигляді сигналу, рівень якого змінюється безперервно, а не дискретно).

Модеми виконують цифроаналогове (та зворотне) перетворення. При передаванні інформації модеми накладають цифрові сигнали комп'ютера

на безперервну частоту телефонної лінії (модулюють її), а при одержанні виділяють (демодулюють) інформацію та передають її в цифровій формі в комп'ютер.

Модеми передають дані звичайним телефонним каналом із швидкостями від 300 до 56000 бод (1 бод = 1 біт за секунду), а орендованим (виділеним) каналом – з швидкістю 64000 бод та вище. Складні модеми, окрім передавання та одержання сигналу, мають додаткові функції, наприклад, автоматичний набір номера, відповідь та повторний набір і т.ін. Деякі модеми конструктивно з'єднані з телефаксами (факсмодемами).

Без відповідного комунікаційного програмного забезпечення модеми не можуть виконувати будь-яку корисну роботу.

Конструктивно модеми бувають вбудованими (розміщуються в системному блоці комп'ютера у вигляді плати розширення в слоті материнської плати) та зовнішніми (виконані в окремому корпусі і під'єднуються до комп'ютера через комунікаційний порт).

Нові пристрої на компакт-дисках. Найбільш розповсюджені пристрої на компакт-дисках (CD-ROM) характерні тим, що здатні тільки читати дані, один раз занесені на диск. Маючи більший об'єм (близько 700 Мбайт) та високу швидкість зчитування, вони дуже зручні для зберігання та розповсюдження великих об'ємів інформації (великі програмні комплекси, довідники тощо).

В 1995-1996 роках на ринку з'явились нові пристрої, що використовують компакт-диски:

- *CD-R (CD-Recordable);*
- *CD-E (CD-Erasable);*
- *DVD (Digital Video Disk).*

CD-R дозволяють користувачу виконувати одноразове записування на диск і потім проводити читання. CD-RW дозволяє багаторазовий перезапис даних на дискові.

DVD – пристрій для читання цифрових відеодисків. Зовні DVD-диск схожий на звичайний CD-ROM, проте, відрізняється він тим, що на одній стороні DVD-диска може бути записано до 4.7 Гбайт інформації, а на обох – 9.4 Гбайт. При використанні двошарової схеми записування на одній стороні можна розмістити вже до 8.5 Гбайт, відповідно на обох – близько 17 Гбайт. DVD-диски допускають і перезаписування інформації.

Сканери. Нерідко виникає необхідність ввести в ЕОМ з друкарського оригіналу текст або графічне зображення для його подальшої обробки

(редагування та ін.). Введення цього тексту за допомогою клавіатури (на сьогодні – найпростіший спосіб) вимагає багато часу та праці. Але існують спеціальні пристрої, сканери, які здатні читати текст та перетворювати його в електронну картинку. Потім спеціальна програма (наприклад, FineReader) розшифрує цю картинку та перетворить її у текстовий файл, де кожен байт відповідає певному символу.

Існує чимало моделей сканерів, що відрізняються методом оптичного "зчитування" зображення, допустимими розмірами оригіналу, якістю оптичної системи. За способом організації переміщення вузла читання відносно оригіналу сканери поділяються на планшетні, барабанні та ручні.

В планшетних сканерах оригінал кладуть на скло, під яким рухається оптоелектронний пристрій читання.

В барабанних сканерах оригінал через вхідну щілину втягується барабаном в транспортний тракт і пропускається повз нерухомий пристрій читання.

Ручний сканер необхідно (вручну) плавно переміщувати по поверхні оригіналу.

Результат роботи сканера (точніше – системи сканер-дешифратор) дуже сильно залежить від якості оригіналу (чіткість рисунків, символів, сталість розмірів символів в рядку, насиченість кольорів). Тому зчитування з неякісного оригіналу дасть файл з великою кількістю нерозпізнаних або помилкових символів. Їх прийдеться корегувати вручну, звичайними засобами редагування файлів.

До числа додаткових пристроїв можна віднести і ***джерела безперебійного живлення*** (англійська абревіатура UPS – Uninterruptable Power SUPPLY). Їх призначення – оберігати обчислювальні пристрої, зокрема ПК, від різних недоліків електропостачання (підвищена або надмірно низька напруга, високовольтні імпульси, раптове відімкнення). До складу UPS обов'язково входить акумулятор, який зберігає заряд, достатній для живлення ПК протягом деякого часу у випадку відмови електромережі. За цей час користувач ПК може нормально завершити роботу програм та вимкнути ПК. UPS розраховуються на різні потужності, з різною тривалістю автономного живлення і з різним ступенем автоматизації (деякі UPS можуть видавати комп'ютеру сигнал про перерву в електропостачанні, у відповідь на який комп'ютер активізує спеціальну програму, яка зупинить роботу і вимкне комп'ютер автоматично, без участі користувача).

Контрольні питання

1. Назвіть основні складові частини персонального комп'ютера.
2. Від чого залежить продуктивність персонального комп'ютера?
3. Що таке порт? Які бувають порти?
4. В чому різниця між послідовним та паралельним передаванням даних?
5. Що таке адресний простір? Чим він визначається?
6. Для чого на материнській платі встановлена батарейка?
7. Чи можна працювати на ПК, що не має НГМД? Якщо так, то які переваги це дає та які незручності може викликати?
8. Чим визначаються вимоги до розміру ОЗП при виборі ПК?
9. Порівняйте матричний та струминний принтери. В чому полягає їх взаємна перевага?
10. Що таке плати розширення? Що вони розширюють та куди їх для цього слід помістити?
11. Чи можна купивши ПК з пам'яттю 4 Мбайт, згодом наростити її до 8 Мбайт та більше?
12. Опишіть процес перетворення тексту, що зберігається в ОЗП монітора, в зображення на екрані.
13. Що таке шрифти True Type? В чому їх головна особливість?
14. Чи можна побачити носій інформації, що входить до складу НЖМД?
15. Чи можна редагувати інформацію на лазерних дисках?
16. Прослідкуйте перетворення способу подання даних при їх передаванні з ОЗП на НЖМД. Що таке адаптер?
17. Які бувають накопичувачі? Які з них є сьогодні обов'язковими складовими частинами ПК?
18. Що таке роздільна здатність монітора? Якщо ви знаєте, що будете використовувати ПК виняткова для роботи з текстовими документами, то який монітор виберете?

3 КЛАСИФІКАЦІЯ ТА ТЕНДЕНЦІЇ РОЗВИТКУ ЕОМ

Номенклатура видів ЕОМ у наш час величезна: машини розрізняються за призначенням, потужністю, розмірами, елементною базою, стійкістю до впливу несприятливих умов і т.ін.

Таблиця 3.1 – Класи сучасних ЕОМ

Клас ЕОМ	Основне призначення	Основні технічні дані	Деякі моделі або виробники
Супер-ЕОМ	Складні наукові розрахунки	Інтегральна швидкодія до десятків мільярдів операцій за секунду; число паралельно працюючих процесорів до 100	CRAY, VAX-1000, MULTICON
Великі ЕОМ, мейнфрейми	Обробка великих об'ємів інформації банків, великих підприємств	Мультипроцесорна архітектура; під'єднання до 200 робочих місць	Tandem Computer
Супер міні-ЕОМ	Системи керування підприємствами; багатопультові обчислювальні системи	Мультипроцесорна архітектура; під'єднання до 200 терміналів: дискові запам'ятовувальні пристрої, нарощувані до десятків Гбайтів	Сімейство VAX (Digital Equipment); SPARC (SUN Microsystems); AS /400 (IBM)
Міні-ЕОМ	Системи керування підприємствами середнього розміру; багатопультові обчислювальні системи	Однопроцесорна архітектура, розгалужена периферія	ES/9000; ES/9370¦(IBM); серії А та 2200 (Unisys);
Робочі станції	Системи автоматизованого проектування, системи автоматизації експериментів	Однопроцесорна архітектура, висока швидкодія процесора; ОЗП 256-1024 Мбайт; спеціалізована периферія	MERVA-2 (IBM RS-6000)
Мікро-ЕОМ	Індивідуальне обслуговування користувача: робота в локальних автоматизованих системах керування	Однопроцесорна архітектура, гнучкість конфігурації – можливість під'єднання різних зовнішніх пристроїв	Найширший перелік моделей та виробників

Тому класифікувати ЕОМ можна було б з різних точок зору, за різними класифікаційними ознаками. Але для наших цілей найбільш цікаво згрупувати ЕОМ за продуктивністю та за габаритними характеристиками (розміри, вага).Прийнята на сьогодні класифікація ЕОМ показана в таблиці, де за класифікаційні ознаки взяті потужність та габаритні дані. Віднесення машин до того чи іншого підкласу є дещо умовним через:

- стрімкий розвиток цієї галузі науки та техніки, при якому, наприклад, сьогоднішня мікро-ЕОМ не поступається потужністю міні-ЕОМ п'ятирічної давності;

- розмитість меж груп;

- широке впровадження в життя практики замовленого складання машин, коли номенклатура вузлів ПК та навіть конкретні моделі складаються за вимогою замовника.

Підводячи підсумки, можна сказати, що сучасні тенденції розвитку ЕОМ виражаються у такому:

а) продовжується зростання обчислювальної потужності мікропроцесорів. Щільність розміщення елементів переходить через значення 3 млн на 1 $см^2$ (тобто майже досягає меж, зумовлених фізичними законами), тактова частота доходить до 3 ГГц. Подальше нарощування потужності стає можливим тільки шляхом нових схемних рішень;

б) підвищення потужності мікропроцесорів дозволяє поєднувати в одному елементі (на одному кристалі) більше пристроїв. Це, в свою чергу, дає можливість реалізувати на одній платі більшу кількість функцій і за рахунок цього скоротити число окремих пристроїв;

в) розширюється набір функцій, що реалізується одним ПК, він стає різнобічним апаратом. Особливо наочно це проявляється в мультимедійному комп'ютері, який є функціональним комбайном. Окрім своїх "прямих обов'язків" – обробки алфавітно-цифрової інформації – він здатний:

- працювати із звуком (відтворення, запис, редагування, створення спеціальних ефектів та ін);

- відтворювати відеосигнал (приймання телепередач; запис кадрів та їх обробка; відтворення аналогових та цифрових відеозаписів та ін).

Різноманітність можливостей вимагає розширення номенклатури компонентів та істотного підвищення потужності базових блоків.

4 ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ТА АЛГОРИТМІЗАЦІЯ ІНЖЕНЕРНИХ ЗАДАЧ

Тенденції розвитку ПЗ вказують на те, що динаміка витрат має стійку тенденцію до зростання, приблизно 20% на рік.

Під програмним забезпеченням інформаційних систем розуміють сукупність програмних та документальних засобів для створення та експлуатації систем обробки даних засобами обчислювальної техніки.

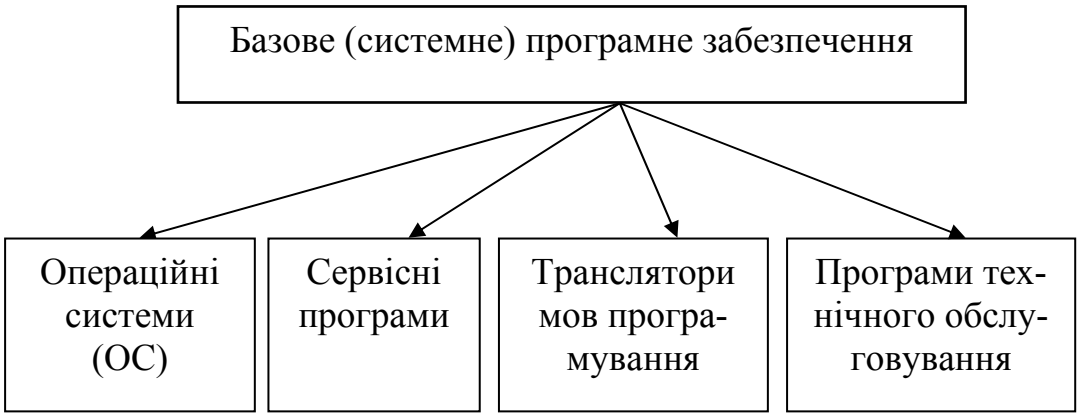
В комп'ютері при розв'язанні задачі задіяні два режими:

- керування;
- виконання програм користувачів.

Керуючий режим здійснює підготовку до режиму виконання програм користувачів, при необхідності перериває його та забезпечує його завершення.

Залежно від функцій, які виконуються програмним забезпеченням, його можна поділити на 2 групи: ***базове*** (загальносистемне) програмне забезпечення та ***прикладне*** (предметне) програмне забезпечення.

4.1 Базове (загальносистемне) програмне забезпечення



Базове (загальносистемне) ПЗ забезпечує режим керування, тобто, організовує процес обробки інформації в комп'ютері та забезпечує нормальне робоче середовище для прикладних програм. Базове ПЗ настільки тісно пов'язане з апаратними засобами, що його іноді вважають частиною комп'ютера.

Прикладне ПЗ призначене для забезпечення режиму виконання програм користувачів, а саме: вирішення конкретних задач користувача та організації обчислювального процесу інформаційної системи в цілому.

Операційна система **(ОС) –** це базова програма, яка при вмиканні комп'ютера автоматично викликається в його оперативну пам'ять і знаходиться там постійно (резидентно).

ОС – це набір програм загальносистемного призначення, що використовується для деякого класу комп'ютерів, забезпечуючи керування процесом обробки інформації та взаємодію між пристроями комп'ютера. Вона здійснює керування комп'ютером, активізує інші програми, підтримує діалог з користувачем.

Однією з найважливіших функцій ОС є автоматизація процесів введення-виведення інформації, керування виконанням прикладних задач користувача. ОС завантажує потрібну програму в пам'ять ЕОМ і слідкує за ходом її виконання; аналізує ситуації, що перешкоджають нормальним обчисленням, та дає вказівки про те, що необхідно зробити, якщо виникли ускладнення.

Виходячи з цього функції ОС можна розбити на три групи:

а) однозадачні;

б) багатозадачні;

в) функції мереж.

Однозадачні ОС призначені для роботи одного користувача в кожен конкретний момент з однією конкретною задачею. Типовим представником таких операційних систем є MS-DOS (Microsoft).

Багатозадачні ОС забезпечують колективне використання ЕОМ в мультипрограмному режимі розподілу часу (в пам'яті ЕОМ знаходиться декілька програм-задач і процесор розподіляє ресурси комп'ютера між задачами). Типовими представниками подібного класу ОС є: Unix, OS/2 корпорації IBM, Microsoft Windows 98, Microsoft Windows NT та деякі інші.

Операційні системи мереж пов'язані з появою локальних та глобальних мереж і призначені для забезпечення доступу користувача до всіх ресурсів обчислювальної мережі. Типовими представниками ОС мереж є: Novell NetWare, Microsoft Windows NT, Unix, Solaris.

Основні тенденції розвитку ОС:

- зниження цін на операційні системи;

- перехід багатьох функцій ОС, що реалізовувались у вигляді програм, до реалізації у вигляді мікропрограм, "зашитих" в апаратну частину комп'ютера;

- забезпечення роботи багатопроцесорних комп'ютерів;

- забезпечення сумісності програм для різних типів (поколінь) комп'ютерів;

- забезпечення виконання паралельних програм;

- створення ОС, в яких окремі функції реалізуються в процесорах різних комп'ютерів, що створюють розподілену обчислювальну мережу.

Сервісне програмне забезпечення – це сукупність програмних продуктів, що надають користувачеві додаткові послуги в роботі з комп'ютером та розширюють можливості операційних систем. За функціональними можливостями сервісні засоби поділяються на такі, що:

- поліпшують інтерфейс користувача;
- захищають дані від руйнування та несанкціонованого доступу;
- відновлюють дані;
- прискорюють обмін даними між диском та ОЗП;
- виконують операції архівації-розархівації;
- здійснюють антивірусний захист.

За способом організації та реалізації сервісні засоби поділяються на:

- оболонки;
- утиліти;
- антивірусні програми.

Різниця між оболонками та утилітами часто полягає тільки в універсальності перших та спеціалізації других.

Оболонки є немовби надбудовами над операційними системами або групою утиліт. Оболонки, що є надбудовами над ОС, називаються операційними оболонками. Утиліти та автономні програми мають вузькоспеціалізоване призначення і виконують кожна свою функцію. Але утиліти, на відміну від автономних програм, виконуються в середовищі відповідних оболонок. При цьому вони конкурують в своїх функціях з програмами ОС та іншими утилітами. Тому класифікація сервісних засобів за їх функціями та способами організації є досить умовною.

Оболонки надають користувачу якісно новий інтерфейс та звільняють його від детального знання операцій та команд ОС. Функції більшості оболонок, наприклад, сімейства MS DOS, направлені на роботу з файлами та каталогами і забезпечують швидкий пошук файлів: створення, перегляд та редагування текстових файлів; видачу повідомлень про розміщення файлів на дисках, про ступінь зайнятості дискового простору та ОЗП. Всі оболонки забезпечують деякий рівень захисту від помилок користувача, що зменшує ймовірність випадкового знищення файлів.

Серед існуючих оболонок для сімейства MS DOS найбільш популярною є оболонка **Total Commander**.

Утиліти надають користувачу додаткові послуги (що не вимагають розробки спеціальних програм) в основному з обслуговування дисків та файлової системи. Утиліти частіше за все дозволяють виконувати такі операції:

- обслуговування дисків (форматування, забезпечення збереження інформації, можливості її відновлення у випадку пошкодження і т.ін.);
- обслуговування файлів та каталогів (аналогічно оболонкам);
- створення та поновлення архівів;
- надання інформації про ресурси комп'ютера, про дисковий простір, про розподіл ОЗП між програмами;
- друк текстових та інших файлів в різних режимах та форматах;
- захист від комп'ютерних вірусів.

Серед існуючих систем утиліт найбільш популярним є багатофункціональний комплекс **Norton Utilites**.

Програмні засоби ***антивірусного захисту*** забезпечують діагностику (виявлення) та лікування (нейтралізацію) вірусів. Терміном "вірус" позначається програма, здатна розмножуватись, проникати в інші програми, здійснюючи при цьому різні небажані дії.

Транслятори. Переважна більшість мов програмування використовується винятково для зручності людини при складанні програм. Кожний пристрій комп'ютера реагує на команди процесора, що подаються його внутрішньою машинною мовою. Переклад програм, створених мовами високого рівня, в програми машинної мови (трансляція) виконується комп'ютером автоматично за допомогою спеціальних програм – трансляторів. Залежно від способу перекладу з мови програмування транслятори поділяються на компілятори та інтерпретатори.

Інтерпретатори виконують трансляцію та виконання програм покомандно. Кожна команда мови програмування (що не містить помилок) транслюється в одну або декілька команд машинної мови, які тут же виконуються без збереження на дискові. У випадку знаходження помилки комп'ютер призупиняє свою роботу з повним збереженням отриманих на даний момент результатів. Це дає можливість після виправлення помилки поновити процес розв'язання задачі з місця зупинки.

Отже, при інтерпретації програма у вигляді машинних кодів не зберігається і тому при кожному запуску вихідної програми на виконання її треба транслювати заново, що знижує швидкість виконання програм.

Головною перевагою інтерпретатора в порівнянні з компілятором є простота. Транслятор такого типу використовується мовою BASIC.

Компілятори виконують трансляцію всієї програми в цілому. Це означає, що транслятор спочатку переглядає початковий текст програми і, якщо він не містить помилок, формує проміжний (об'єктний) модуль машинною мовою. Потім до цього модуля під'єднуються необхідні для його виконання підпрограми із бібліотеки операційної системи, що приводить до створення так званого активного модуля, готового до виконання та зберігання у вигляді файла на магнітному дискові. Саме цей модуль і виконується комп'ютером. Він може бути виконаний багаторазово без повторної трансляції. При знаходженні помилок компілятор видасть їх перелік в кінці процесу трансляції. В цьому випадку необхідно виправити текст програми і направити його на повторну трансляцію.

Контрольні питання

1. Що розуміють під програмним забезпеченням?
2. Які програмні засоби відносяться до базового ПЗ?
3. Яка основна функція виконується базовим ПЗ?

Операційні системи

Операційна система (ОС) – це сукупність програмних засобів, що здійснюють:

- керування ресурсами ЕОМ;
- активізацію прикладних програм та їх взаємодію із зовнішніми пристроями та іншими програмами;
- діалог користувача з комп'ютером.

Ресурсом є будь-який компонент ЕОМ та його потенційні можливості, а саме:

- центральний процесор;
- оперативна або зовнішня пам'ять;
- зовнішній пристрій;
- програма і т.ін.

Кожен комп'ютер обов'язково комплектується операційною системою, яка автоматично активізується при його увімкненні. Необхідність ОС полягає в тому, що операції з керування комп'ютером та його пристроями

складаються із сотень або тисяч елементарних дій. ОС, приховуючи від користувача ці непотрібні подробиці, надає йому зручний спосіб спілкування (інтерфейс) з обчислювальною системою. Відомі два види інтерфейсів:

- програмний інтерфейс;
- інтерфейс користувача.

Програмний інтерфейс – це сукупність засобів, що забезпечують взаємодію пристроїв та програм в межах обчислювальної системи.

Інтерфейс користувача – це програмні та апаратні засоби взаємодії користувача з програмою або ЕОМ. В свою чергу, інтерфейс користувача може бути: а) командним; б) об'єктно-орієнтованим.

Командний інтерфейс передбачає введення користувачем команд з клавіатури при виконанні дій з керування ресурсами комп'ютера.

Об'єктно-орієнтований інтерфейс – це керування ресурсами обчислювальної системи шляхом здійснення операцій над об'єктами, що подають файли, каталоги (теки), дисководи, програми, документи і т.ін.

Характерною особливістю ОС є модульність, тобто побудова їх на основі набору самостійних програм. В результаті конкретні ОС можуть спрощуватись, якщо деякі модулі не потрібні, або ускладнюватись, якщо необхідні додаткові модулі.

Головний модуль (ядро) постійно знаходиться в оперативній пам'яті комп'ютера, забезпечуючи безперервну координацію обчислювального процесу. Він утворений із основних керуючих програм, які реалізують окремі команди, керують ходом розв'язання задач, забезпечують багатопрограмний режим, здійснюють розподіл ресурсів між програмами, керують введенням-виведенням даних.

Операційні системи класифікуються за:

- кількістю одночасно працюючих користувачів: однотермінальні, багатотермінальні;
- числом процесів, що одночасно виконуються під керуванням системи: однозадачні, багатозадачні;
- кількістю процесорів, що підтримуються: однопроцесорні, багатопроцесорні;
- розрядністю коду ОС: 8-розрядні, 16-розрядні, 32-розрядні, 64-розрядні;
- типом інтерфейсу: командні (текстові) та об'єктно-орієнтовані (графічні);

- типом доступу користувача до ЕОМ: з пакетною обробкою, з розподілом часу, реального часу;

- типом використання ресурсів: мережні, локальні.

На сьогодні розповсюджені такі сімейства операційних систем: DOS; OS/2; Unix; Windows; Linux, ОС реального часу.

ОС сімейства DOS

Перший представник цього сімейства – система MS-DOS (Microsoft Disk Operating System – дискова операційна система фірми Microsoft) була випущена в 1981 році в зв'язку з появою IBM PC.

Операційні системи сімейства DOS є однозадачними і мають такі характерні риси та особливості:

- інтерфейс з ЕОМ здійснюється за допомогою команд, що вводяться користувачем;

- модульність структури, що спрощує перенесення системи на інші типи ЕОМ;

– невеликий об'єм доступної оперативної пам'яті (640 Кбайт).

Істотним недоліком операційних систем сімейства DOS є відсутність засобів захисту від несанкціонованого доступу до ресурсів ПК та ОС. На сьогодні широке розповсюдження одержала ОС MS-DOS версії 6.22.

Операційні системи сімейства **UNIX** – це 32-розрядні багатозадачні багатотермінальні операційні системи. Сильна сторона Unix полягає в тому, що вона використовується на різних комп'ютерах – від суперкомп'ютера до ПК, що дає можливість перенесення системи з однієї машинної архітектури на іншу з мінімальними витратами.

Unix об'єднує в собі:

- доступ до розподілених баз даних;

- локальні мережі;

- віддалений дистанційний зв'язок та можливість виходу в глобальні мережі, використовуючи звичайний модем.

Поштова служба в Unix – одна з найважливіших її компонентів. На сьогодні існує велика кількість програм для Unix. Більшість популярних програм для DOS та Windows можуть експлуатуватись в Unix.

Існує декілька ОС сімейства Unix. Різні версії цього сімейства мають свої назви, але в загальних рисах повторюють особливості базової ОС Unix. Файлова система ОС Unix забезпечує захист файлів від несанкціонованого доступу на рівнях користувача та групи користувачів. На сьо-

годні з ОС мереж сімейства Unix широке розповсюдження одержала ОС для мереж підприємств UnixWare 2.0. Це 32-розрядна багатотермінальна багатозадачна ОС, що підтримує програми реального часу.

***ОС сімейства* WINDOWS** розроблені фірмою Microsoft. Вони є багатозадачними операційними системами, що надають зручний графічний інтерфейс. Основними представниками даного сімейства є ОС Windows 98 та ОС Windows NT.

Windows 95 розроблена на базі ОС MS-DOS та операційних оболонок Windows 3.x. Windows 98 є 32-розрядною операційною системою.

Операційна система Windows NT – одна з найбільш розповсюджених 32-розрядних ОС мереж. Windows NT випускається в двох модифікаціях:

– Windows NT Server;

– Windows NT Workstation.

Сімейство ОС реального часу. Термін «реальний час» (РЧ) застосовують до системи обробки інформації в тих випадках, коли вимагається, щоб затримка відповіді системи не перевищувала визначеного часу. Операційна система реального часу (ОС РЧ) гарантує визначений час реакції системи. Як правило, цей час коливається від декількох мікросекунд до декількох десятих секунди.

ОС РЧ в основному застосовується для автоматизації технологічних процесів.

***Операційна система* MS-DOS** є комплексом програм з функціями:

- керування виконанням програм;
- керування ресурсами ПК;
- організації обміну інформацією між процесором та зовнішніми пристроями ПК;
- зберігання інформації та обслуговуванню технічних носіїв інформації.

Ця ОС зберігається на дискові, тому вона одержала назву дискової операційної системи – ДОС або англійською DOS – Disk Operating System.

Програми DOS завантажуються в оперативну пам'ять з магнітного диска при необхідності. Сьогодні фактично стали стандартними декілька сімейств ОС для ПК: MS-DOS, Windows 95, PC DOS, DR DOS, Unix та OS/2. На IBM PC-сумісних машинах найчастіше використовуються MS-DOS та її аналоги DR DOS та PC DOS.

MS-DOS була розроблена фірмою Microsoft, DR DOS – фірмою Digital Research, а PC DOS– фірмою IBM. Для користувача істотної різниці між

цими ОС немає (якщо робота здійснюється винятково в середовищі DOS). На сьогодні найбільш розповсюджена версія MS-DOS 6.22.

MS-DOS складається з таких функціональних частин:

- файлова система;
- драйвери зовнішніх пристроїв;
- командний процесор;
- утиліти MS-DOS;
- команди MS-DOS;
- пакетні командні файли.

Файлова система

Файл – це іменована сукупність даних, що знаходиться на технічному носієві інформації. Поняття файла застосовується в основному до даних, що зберігаються на дисках, і тому файли, звичайно, ототожнюють з ділянками пам'яті на цих носіях інформації. Дані, що зберігаються в файлах, – це програми алгоритмічною або машинною мовою, вихідні дані для роботи програм або результати виконання програм, літературні та технічні тексти, закодовані зображення і т.ін.

Поняття файла в MS-DOS поширюється на принтер, клавіатуру, дисплей та ін.

Файлова система містить в собі, окрім самих файлів, правила створення імен файлів та способів роботи з ними, ієрархічну систему заголовків файлів та структуру зберігання файлів на дисках.

Файл має ім'я та атрибути.

Файл характеризується: а) розміром в байтах; б) датою та часом його створення або останнього редагування.

Ім'я файла може бути повним та неповним. Повне (складове) ім'я файла в MS DOS складається з двох частин: а) імені файла; б) розширення.

Розширення файла може бути відсутнім; в цьому випадку ім'я файла є неповним. Ім'я файла відокремлюється від розширення точкою. Символи, що використовуються в імені файла та його розширенні, беруться з такої множини:

- великі та маленькі літери латинського алфавіту;
- цифри;
- символи: -, _, $, #, &, @, !, %, (,),}, {, ', ~, ^.

В імені файла може бути від одного до восьми символів, а в розширенні – від нуля до трьох. Розширення вказує тип файла, причому деякі з них є стандартними:

- *.COM – готовий до виконання файл (1-й різновид);
- *.EXE – готовий до виконання файл (2-й різновид);
- *.BAT – командний пакетний файл;
- *.TXT – текстовий файл довільного типу;
- *.MDB – файл СУБД Access;
- *.XLS – електронна таблиця Excel;
- *.DOC – текстовий файл, що містить документацію щодо деякого програмного продукту або файл редактора Microsoft Word;
- *.ARJ – упакований файл;
- *.ZIP – упакований файл.

Застосування стандартних розширень дозволяє не вказувати їх при виконанні системних програм та пакетів прикладних програм, при цьому використовується принцип замовчування.

Наприклад: COMMAND.COM, PERTE1.ZIP, TABLHZ.XLS.

Для позначення групи файлів в MS-DOS можна використовувати шаблони. ***Шаблоном*** або ***зразком*** (від англійського слова pattern зразок, шаблон), називається ім'я файла або розширення, в яких використовуються глобальні символи (символи шаблону). Глобальними є символи * (зірочка) та ? (знак питання).

Зірочка в імені (розширенні) файла означає, що на її місці, починаючи з цієї позиції і до кінця імені (розширення) можуть стояти будь-які припустимі знаки. Наприклад, *.XLS – всі файли з розширенням XLS, FORMA.* – всі файли з іменем FORMA, ABS*.EXE – всі файли з розширенням EXE, імена яких починаються з ABS, *.* – будь-які файли з будь-якими розширеннями.

Каталоги MS-DOS створюють ієрархічну структуру подібну перевернутому дереву, в якому головний каталог утворює "корінь" дерева (друга назва головного каталогу – "кореневий"), а інші каталоги подібні гілкам.

Якщо деякі файли об'єднані в каталог, то говорять, що вони входять в цей каталог. Проте об'єднання файлів в каталоги не означає, що вони деяким чином згруповані в одному місці на дискові. Більш того, фрагменти одного файла можуть бути "розкидані" (фрагментовані) по всьому дискові. Дані про місцезнаходження окремих частин файла (кла-

стерів), зберігаються в таблиці розміщення файлів (FAT, File Allocation Table), що знаходиться на цьому ж дискові. Номер першого кластера файла міститься також в каталозі, в який входить файл, завдяки чому прискорюється пошук файла на дискові.

***Утилітами* MS-DOC** називають її зовнішні команди. Вони поділяються на діалогові та недіалогові. При використанні недіалогових утиліт вся необхідна інформація вводиться в командному рядку. До недіалогових утиліт відноситься більшість зовнішніх команд MS-DOS. При використанні діалогових утиліт користувач може обирати параметри в процесі їх роботи за допомогою меню та діалогових вікон.

Як правило, окрім утиліт MS-DOS, в програмному забезпеченні ПК існує велика кількість сервісних програм-утиліт (utilities) незалежних виробників програмного забезпечення, наприклад, популярний пакет утиліт Norton Utilities фірми Symantec. Деякі з утиліт MS-DOS є спрощеними варіантами утиліт незалежних виробників і використовуються фірмою Microsoft за ліцензійною угодою.

Команди MS-DOS

***Команди* MS-DOS** бувають внутрішніми та зовнішніми. Внутрішня команда є складовою частиною (модулем) командного процесора. Вона автоматично завантажується з транзитної частини КП при введенні її з клавіатури або з командного файла.

Команди MS-DOS вводяться з клавіатури у відповідь на запрошення MS-DOS в командних рядках. Командний рядок складається з імені команди або специфікації файла, що виконується і, при необхідності, аргументів, розділених пропусками, а також ключів, що починаються символом "/" (прямий слеш). Сигналом про закінчення командного рядка служить натискання клавіші **Enter**. Програма після її виконання може залишатись резидентною в ОЗП. Резидентна програма, на відміну від нерезидентних програм, після закінчення її роботи залишається в ОЗП в робочому стані і тому може активізуватись без додаткового завантаження за командами користувача або MS-DOS. Звичайно, резидентними є сервісні програми, наприклад, Norton Commander, а також додаткові драйвери пристроїв.

Всього в MS-DOS 6.22 існує більше сотні команд. Повний їх перелік з вказівкою призначення команд можна одержати, подавши команду FASTHELP.

Форматування дискети може бути виконане в одному з трьох режимів.

Стандартне форматування. В цьому режимі здійснюється розмітка диска на доріжки та сектори, тестування (перевірка) поверхні диска та формування його системної області. При цьому вся інформація, що була на дискеті, втрачається. Даний режим називається також низькорівневим форматуванням.

Безпечне форматування. У випадку безпечного (safe) форматування дискети виконуються такі дії:

– тестується поверхня дискети без руйнування інформації;

– на дискеті зберігається її системна область (BR, FAT та кореневий каталог), що полегшує відновлення, при необхідності, файлової структури, зруйнованої внаслідок помилкового виконання форматування;

– очищається зміст истемного аталогу, здійснюється ініціалізація FAT нулями, і в ній реєструються дефектні кластери. Якщо програма форматування визначить, що диск ніколи ще не ініціалізувався згідно з потрібним форматом, то буде здійснений перехід в режим стандартного форматування.

Швидке форматування. У випадку швидкого (quick) форматування очищається тільки системна область диска з попереднім її збереженням, як і у випадку безпечного форматування. Перевірка поверхні диска на наявність дефектних секторів не проводиться, внаслідок чого форматування здійснюється з максимальною швидкістю. Режими безпечного та швидкого форматування дозволяють при деяких умовах відновити системну область дискети і тим самим здійснити доступ до файлів, пошкоджених при форматуванні дискети.

Тестування дисків полягає у виявленні логічних та фізичних дефектів на них. Програмні засоби перевірки дисків можуть виконувати також автоматичне відновлення інформації на дисках, зруйнованої у випадку виникнення дефектів. В процесі експлуатації дисків періодично виникають різні дефекти, які можна поділити на:

– фізичні дефекти, пов'язані з механічними пошкодженнями або старінням магнітного покриття (поява дефектних секторів);

– логічні дефекти, викликані пошкодженнями файлової структури.

До останніх відносяться:

– поява загублених кластерів (lost clusters), які зафіксовані у FAT як такі, що використовуються файлами, але доступ до них через жоден каталог диска неможливий;

– поява порожніх файлів та повністю порожніх каталогів;

– поява файлів, що пересікаються (cross-linked), тобто файлів, що мають спільні кластери;

– руйнування інформації в каталогах, FAT та стартовому секторі;

– неідентичність копій FAT.

Логічні дефекти виникають внаслідок поломки обладнання, раптового вимкнення живлення, виконання некоректних програм та дії комп'ютерних вірусів. Наприклад, загублені кластери з'являються тоді, коли оформлення файла на дискові виявилось незавершеним через раптове вимкнення живлення ПК або закінчення виконання програми без явного закриття файла, в який здійснювався запис. З цих же причин на дискові створюються порожні файли та каталоги.

Перевірка дисків та відновлення на них інформації, зруйнованої внаслідок фізичних та логічних дефектів, здійснюється командою MS-DOS **CHKDSK**, утилітою MS-DOS **SCANDISK** або утилітою **NDD** з пакета Norton Utilities. Утиліти SCANDISK та NDD мають широкі можливості, що перевершують можливості команди CHKDSK.

В результаті модифікації файлової структури диска файли та каталоги стають фрагментованими. Це означає, що файли та каталоги розміщуються в несуміжних кластерах. Фрагментація збільшує час доступу до файлів і тим самим знижує реальну швидкодію ПК.

Для усунення фрагментації файлів та каталогів на дискеті досить скопіювати файлову структуру дискети на жорсткий диск командою XCOPY, виконати швидке форматування дискети та відновити на ній початкову файлову структуру з жорсткого диска. Виконувати дефрагментацію файлів та каталогів на жорсткому диску таким способом недоцільно, оскільки для цього потрібно дуже багато часу та дискет. Для дефрагментації жорстких дисків доцільно використовувати утиліту MSDOS **Microsoft Defragmenter** або утиліту **Speed Disk** пакета **Norton Utilities**. Утиліта **Microsoft Defragmenter** є спрощеним варіантом утиліти **Speed Disk**. Вона активізується командою **DEFRAG.EXE**.

Архіватори

Збереження інформації на дискетах забезпечує її цілісність при пошкодженні жорстких дисків комп'ютера внаслідок аварій або дії вірусів. Дискети зручні також і при передаванні інформації між комп'ютерами, не об'єднаними в мережу. Але великі об'єми файлів потребують багато дискет, що незручно і багато коштує. Архіватори дозволяють попередньо стиснути (упакувати) інформацію, записати на носій для збереження, а потім розархівувати при необхідності використання.

До числа найпоширеніших архіваторів (розархіваторів) відносяться: PKZIP.EXE, PKUNZIP.EXE, ARJ.EXE, PKPAK.EXE, PKUNPAK.EXE, LHA.EXE.

Усі ці програми запускаються з командного рядка DOS. Для прикладу наведемо порядок запуску архіватора ARJ. Командний рядок в цьому випадку має такий формат:

ARJ<параметр>[-(sw)[-(sw)…]]<archive-name>[(file-names)…], де:

- параметр визначає напрям дії програми. Наприклад, a – архівація, e –розархівація, x – розархівація з новим ім'ям тощо;
- sw – ключ, який задає додаткову функцію. Ключів може не бути, або бути декілька;
- archive-name – ім'я архівного файла;
- file-names – список файлів, які архівуються (розархівуються). Коли цей параметр відсутній, при розархівовуванні з архіву розкриваються всі файли, а при архівуванні обробляються всі файли поточного каталогу. При необхідності включити каталог застосовується ключ –r (перед ключем завжди ставиться знак "-").

Приклади.

1. ARJ a d:\DOC\proba*.pas

Всі файли поточного каталогу з розширенням pas архівуються і створюється архівний файл з іменем proba.arj, який розташовується у каталозі DOC диска d. Якщо в імені архіва не вказують шлях d:\DOC, то архів розташовується в поточному каталозі.

2. ARJ e jessi.arj

Архів jessi.arj розархівується в поточному каталозі, де він сам і знаходиться.

Для полегшення роботи з архіваторами широко використовуються оболонки архіватарів RAR, NARC, SHEZ, WINZIP та ін.

У Windows-системах працюють програми WINZIP. Використовуючи меню та піктограми можна розархівувати (архівувати) файли, каталоги та виконувати інші операції з файлами.

Лабораторна робота №1

ОПЕРАЦІЙНІ СИСТЕМИ КОМП'ЮТЕРІВ

Мета роботи: вивчення основних можливостей та команд поширених операційних систем та їх практичне використання.

Порядок виконання роботи

1.Вімкнути комп'ютер.

2.Ввійти в операційну систему (MS DOS, RSX або ін. вказує викладач).

3.Викликати і прочитати каталог. Знайти вказаний викладачем файл. Визначити його тип та властивості.

4.Ввійти в середовище програмування (запустити інтерпретатор Basic або компілятор Pascal).

5.Набрати на клавіатурі програму за індивідуальним домашнім завданням.

6.Запам'ятати програму в каталозі з іменем Вашої групи під повним іменем.

7.Покинути операційне середовище. Переконатись в наявності запам'ятованого файла в поточному каталозі.

8.Знову ввійти в операційне середовище, викликати Ваш файл з зовнішньої пам'яті, прочитати його та запустити на виконання. Виправити при необхідності помилки. Скопіювати Ваш файл у каталог INFORM.

9. Віддрукувати файл.

10. Видалити Ваш файл з усіх каталогів.

11. Оформити звіт.

Питання для самоконтролю

1.Назвіть поширені операційні системи, дайте їх характеристику.

2.Основні команди операційних систем.

3.Імена файлів, їх компоненти.

4.Основні повідомлення операційної системи.

5.Як виконати перезавантаження системи?

Лабораторна робота №2

ОПЕРАЦІЙНА ОБОЛОНКА “TOTAL COMANDER” ТА ЇЇ ВИКОРИСТАННЯ

Мета роботи: вивчення операційної оболонки “Total Comander” та оволодіння навичками практичної роботи в ній.

Апаратне забезпечення: персональний комп’ютер.

Порядок виконання роботи

1.Ввімкнути комп’ютер.

2.Запустити **“Total Comander”.**

3. Вивчити інформацію на екрані.

4. Вибрати вказаний диск .

5. За допомогою клавіші **TAB** перейти на іншу панель (з правої на ліву або навпаки). При цьому виділена (бод) стрічка повинна переміститися.

6. Вибрати (виділити) вказаний файл за допомогою клавіш керування курсором (↑, ↓, →, ←).

7. За допомогою функціональної клавіші F1 ввійти в підсистему Help, знайти вказану інформацію про ***“Total Comander”***.

8. Вийти з підсистеми Help **(Esc**).

9. Викликати вказаний файл для читання (**F3**).

10. Вийти з режиму читання файла (**Esc** або **F10**).

11. Створити новий каталог з іменем Inform.

12.Скопіювати вказаний файл в каталог Inform.

13. Виділити каталог Inform і натиснути Enter. Каталог Inform відкриється. В ньому повинен бути скопійований файл.

14.Вибрати скопійований файл. Видалити його, натиснувши F8 та Enter.

15. Видалити каталог Inform, використовуючи маски.

16.Відсортувати каталоги і файли за вказаною ознакою (**Ctrl + F3** – за іменем по алфавіту, **Ctrl + F4** – за розширенням, **Ctrl + F5** – за розміром).

17. Скласти звіт. В звіт включити назву і мету роботи та описати основні команди ***Total Comander***.

Питання для самоконтролю

1. Яка послідовність дій необхідна для виконання пунктів 1-15?

2. Призначення операційних оболонок.

3. Функції операційних оболонок.

4. Назвіть поширені операційні оболонки та дайте їх порівняльну характеристику.

Лабораторна робота №3

ОПЕРАЦІЙНА СИСТЕМА “WINDOWS” ТА ЇЇ ВИКОРИСТАННЯ

Мета роботи: вивчення операційної оболонки “Windows” та оволодіння навичками практичної роботи в ній.

Апаратне забезпечення: персональний комп’ютер.

Порядок виконання роботи

1. Ввімкнути комп’ютер.

2. Запустити “Windows” .

3. Вивчити інформацію на екрані.

4. Використовуючи клавішу **Пуск,** вибрати вказаний пакет.

5. Вибрати вказаний диск (**Alt + F1** або **Alt + F2**).

6. За допомогою клавіші **TAB** перейти на іншу панель (з правої на ліву або навпаки). При цьому виділена (бод) стрічка повинна переміститися.

7. Вибрати (виділити) вказаний файл за допомогою клавіш керування курсором (↑, ↓, →, ←).

8. За допомогою функціональної клавіші F1 ввійти в підсистему Help, знайти вказану інформацію про **“Total Commander”**.

9. Вийти з підсистеми Help **(Esc**).

10. Викликати вказаний файл для читання (**F3**).

11. Вийти з режиму читання файла (**Esc** або **F10**).

12. Створити новий каталог з іменем Inform.

13.Скопіювати вказаний файл в каталог Inform.

14. Виділити каталог Inform і натиснути Enter. Каталог Inform відкриється. В ньому повинен бути скопійований файл.

15. Вибрати скопійований файл. Видалити його, натиснувши F8 та Enter.

16. Видалити каталог Inform, використовуючи маски.

17. Відсортувати каталоги і файли за вказаною ознакою (**Ctrl + F3** – за іменем по алфавіту, **Ctrl + F4** – за розширенням, **Ctrl + F5** – за розміром).

18. Скласти звіт. В звіт включити назву і мету роботи та описати основні команди ТС.

Питання для самоконтролю

1. Яка послідовність дій необхідна для виконання пунктів 1-15?
2. Призначення операційних оболонок.
3. Функції операційних оболонок.
4. Назвіть поширені операційні оболонки та дайте їх порівняльну характеристику.

4.2 Прикладне програмне забезпечення

Алгоритми

Алгоритмом називається чітко і однозначно визначена послідовність дій, необхідна для досягнення поставленої мети.

Властивості алгоритмів:

а) детермінованість – при багаторазовому виконанні алгоритм забезпечує отримання одного і того ж результату при одному і тому ж наборі вихідних даних;

б) масовість – можливість використання алгоритму при різних наборах вихідних даних для деякого класу задач;

в) результативність – при будь-якому наборі вихідних даних алгоритм забезпечує або отримання шуканого результату, або повідомлення про те, що результат не можна отримати;

г) дискретність – алгоритм забезпечує досягнення мети з допомогою кінцевого числа окремих елементарних дій.

Форми запису алгоритмів:

- вербальна (словесна);
- алгебраїчна (з допомогою літерно-цифрових позначень виконуваних дій);
- графічна;
- з допомогою алгоритмічних мов програмування.

Словесна форма запису алгоритмів використовується в різних інструкціях, призначених для виконання їх людиною.

Алгебраїчна форма найчастіше використовується у теоретичних дослідженнях фундаментальних властивостей алгоритмів.

Графічна форма у відповідності з державними стандартами (ДСТ) на оформлення документації прийнята як основна для опису алгоритмів. У таблиці 1.1 наведено найчастіше використовувані графічні позначення дій алгоритму.

Правила графічного запису алгоритмів.

1. Блоки алгоритмів з'єднуються стрілками (лініями потоків інформації).

2. Лінії потоків не повинні перетинатися.

3. Будь-який алгоритм може мати лише один блок початку і один блок кінця.

4. Для полегшення аналізу і опису алгоритмів блоки можуть нумеруватися. Номери проставляються у розриві контуру блоку у верхній лівій частині.

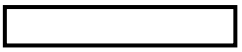
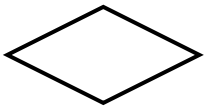
Алгоритм, записаний з допомогою алгоритмічної мови програмування, називається програмою. Алгоритм у такій формі може бути введений в ЕОМ і після відповідної обробки виконаний з метою отримання шуканого результату.

Будь-який, навіть найскладніший, алгоритм може бути поданий у вигляді комбінацій кількох елементарних фрагментів, які називаються базовими структурами. До них належать:

- послідовне проходження (рис. 4.1);
- розгалуження "якщо – то" (рис. 4.2);
- розгалуження "якщо – то – інакше" (рис. 4.3);
- обрання варіанта за ключем (рис. 4.4);
- цикл за параметром (рис. 4.5);
- цикл з передумовою (рис. 4.6);
- цикл з післяумовою (рис. 4.7).

У більшості алгоритмічних мов програмування існують оператори, які відповідають базовим структурам алгоритмів. Умовні позначення, що використовуються в блок-схемах алгоритмів подані в таблиці 4.1

Таблиця 4.1- Умовні позначення на блок-схемах алгоритмів

Назва символа	Позначення	Зміст позначення
ПРОЦЕС		Виконання операцій, в результаті яких змінюються значення або подання даних
РОЗВ'ЯЗАННЯ		Перевірка умови та обрання напрямку виконання циклічного алгоритму
МОДИФІКАЦІЯ		Зміна значення змінної, яка керує виконанням циклічного алгоритму
НАПЕРЕД ВИЗНАЧЕНИЙ ПРОЦЕС		Виконання операцій у відповідності до окремо описаного алгоритму
ВВЕДЕННЯ - ВИВЕДЕННЯ		Введення або виведення за допомогою стандартного пристою введення-виведення
ПУСК-ЗУПИНКА		Початок та кінець алгоритму
З'ЄДНУВАЧ		Позначення зв'язків між перерваними лініями потоку
ЛІНІЯ ПОТОКУ		Позначення послідовності виконання дій

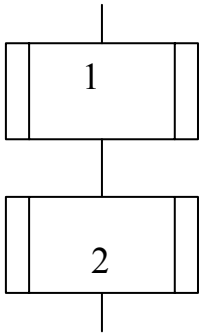


Рисунок 4.1 - Блок-схема лінійного алгоритму

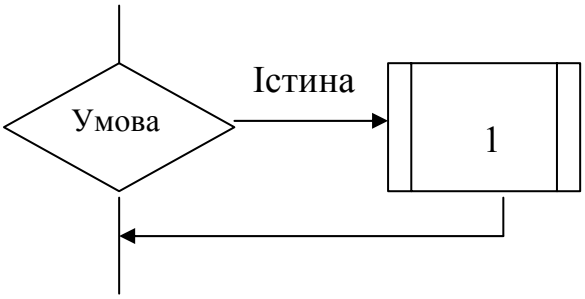


Рисунок 4.2 – Блок-схема алгоритму з розгалуженням

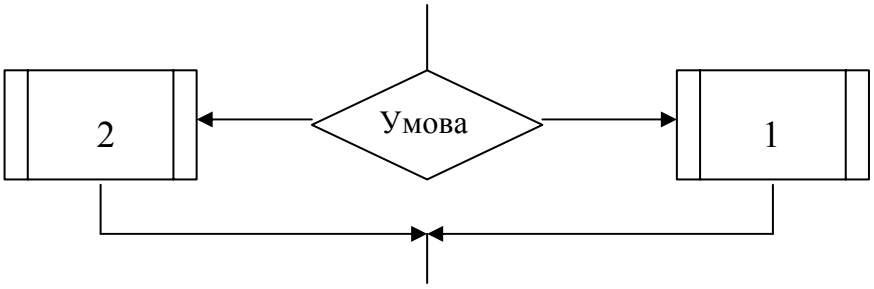


Рисунок 4.3 – Блок-схема алгоритму з розгалуженням за двома умовами

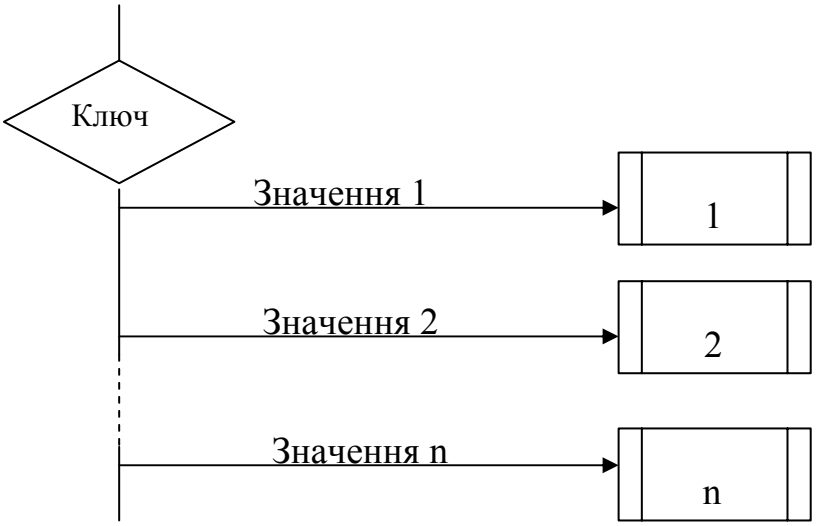


Рисунок 4.4 – Блок-схема алгоритму вибору (меню)

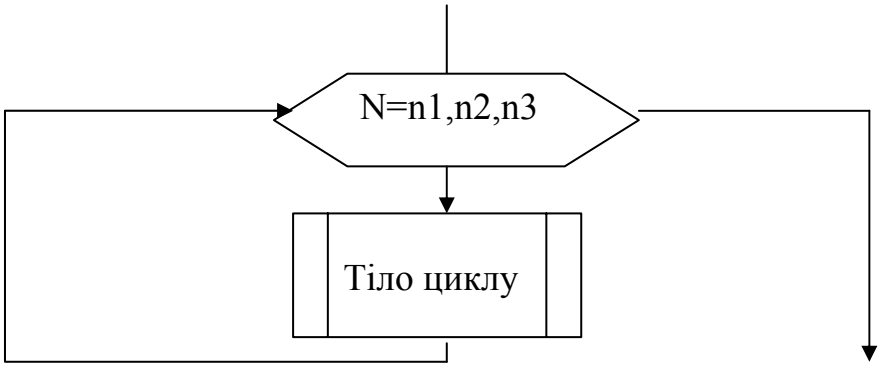


Рисунок 4.5 – Блок-схема циклічного алгоритму

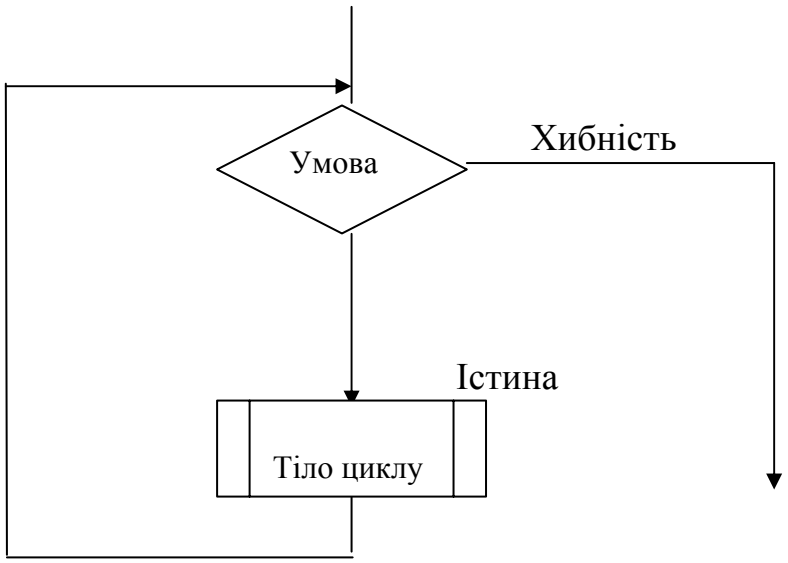


Рисунок 4.6 – Блок-схема алгоритму циклу з передумовою

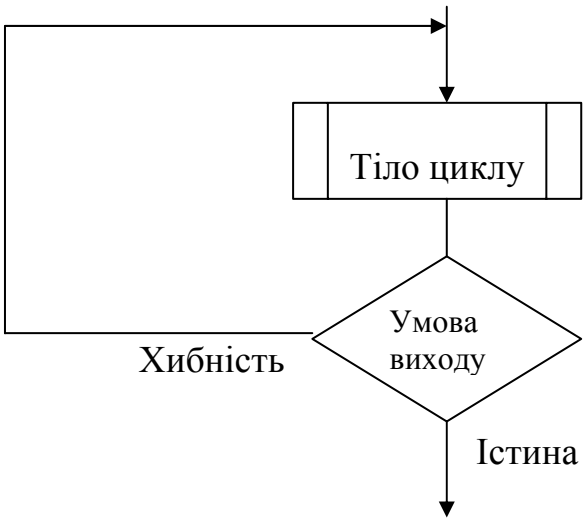


Рисунок 4.7 – Блок-схема алгоритму з післяумовою

5 ПРОГРАМУВАННЯ МОВОЮ TURBO PASCAL

5.1 Загальна характеристика

Безумовною перевагою мови Turbo Pascal є наочність та читабельність її програм.

Розробник системи Turbo Pascal — фірма Borland International виникла в 1984 році і за порівняно короткий час неодноразово дивувала користувачів персональних ЕОМ своїми Turbo системами. Було випущено декілька версій Turbo Pascal: 3.0, 4.0, 5.0, 5.5, 6.0, 7.0, Pascal for Windows, Borland Pascal.

Можна рекомендувати читачу декілька книг для вивчення цієї мови [3,5,6].

В цьому посібнику викладання матеріалу буде базуватися на версії **6.0,** що буде достатньо для початківця.

Головні особливості мови Turbo Pascal:

- широкий спектр типів даних;
- можливість обробки рядкових та структурних даних;
- достатній набір операторів керування розгалуженнями та циклами;
- відносно слабкі можливості введення-виведення даних (порівняно з мовами Fortran та PL/I);
- добре розвинутий апарат підпрограм;
- зручні конструкції роботи з файлами;
- великі можливості керування усіма ресурсами ПЕОМ;
- різні варіанти стикування з мовою Асемблера;
- використання інтегрованого середовища, яке значно підвищує продуктивність праці користувача;
- підтримка ідей об'єктно-орієнтованого програмування **(О.О.П.).**

```
Program                          {приклад вельми простої програми}
Uses crt, printer;               {підключення модулів з бібліотеки }
Const A=6;
Var X,Y: real;                   {це розділ описування даних       }
Begin                            {початок виконуваних операторів }
Writeln (‘ введіть значення X’); { виведення повідомлення на екран }
readIn(X);                       {введення числа      }
```

```
     Y=A *X;
Writeln ('X=',X, 'Y=',Y);          {виведення результатів на екран }
writeln(lst,'X=',X,'Y=',Y);        {виведення результатів на принтер}
  End.                             {кінець програми }
```

Рисунок 5.1 - Приклад простої програми

На початку програми підключається бібліотека стандартних програм, описуються всі дані.

Виконавчі оператори замикаються командами **Begin ... End.**

Введення даних виконується оператором **Readln,** а виведення результатів — **Writeln.** Пояснення записуються між фігурними дужками.

Таку програму можна підготувати та виконати в турбосередовищі.

5.2 Постійні та змінні величини

Перше знайомство з мовою починається з алфавіту та даних.

Алфавіт – це сукупність символів, які можна використовувати в мові. Для мови **Turbo Pascal** це такі символи: **A, B, ..., Z, a, Ь, ... , z, _** (символ підкреслення), **0, 1, 2, ... , 9.**

Спеціальні символи: **+ – * / = < > . , : ; @ ' ( ) [ ] { } № $ ".** Символ "проміжок" ніяк не визначається, це «пусте місце» між якими-небудь конструкціями.

В персональній ЕОМ вживається таблиця кодів ASCII, в якій 258 символів. Більшість з них не входить до алфавіту, але їх можна застосувати в текстових константах та коментаріях. Це відноситься також до букв кирилиці та символів псевдографіки (таких, як ®, ¶ , тощо).

За допомогою символів алфавіту можна складати різні конструкції: постійні, змінні, оператори тощо.

Постійна величина (константа) не змінюється в процесі роботи програми. Є декілька видів констант.

Числові константи — цілі та дійсні.

Приклади цілих констант:

815, -53, 56384

Дійсні константи можна записувати одним з двох способів:

1) **204. 586**
   - 0.67

2) 0.204586E+3

-0.67E0

Зверніть увагу, що в цілій константі немає ні крапки, ні коми, а в дійсній константі записується так звана десяткова крапка (а не кома).

В мові Turbo Pascal можна використовувати також шістнадцяткові константи. Вони починаються знаком **$** і складаються з цифр **0 1... 9 A B C D E F.**

Приклади:

$24

$3ABF

Символьна константа — це один символ в лапках.

Приклади: ‘d’, ‘+’

Рядкова константа — це послідовність символів в лапках, *наприклад:* **‘Вивчайте рідну мову’.**

Логічні константи визначаються словами **True та False,** відповідно «істина» та «хибність».

Ці константи без імені. Їх не описують, а можна вживати в різних конструкціях: в виразах, операторах **If, Case** тощо. Крім таких констант є так звані іменовані константи, їх треба описати на початку програми після слова **Const.** *Приклади:*

```
Const F=80;
Den=3;
Zhoda='J';
Pusk='Натисніть клавішу Enter';
```

Такі константи можна використовувати в аналогічних випадках, як і константи без імені. Крім того, цілі константи вживають для описування масивів, в заголовках циклів тощо.

Приклади:

```
Var X: array[l..M] of integer;
  For I:=1 to M do ...
```

Іменовані константи підтримують наочність програм.

В розглянутих константах не вказується їх тип, транслятор визначає його в залежності від значення константи. Так, константа F має цілий тип, тому що її значення (80) — ціле число. Ще один вид, який називається «типовані константи», буде розглянуто пізніше.

Важливим моментом кожної програми є змінні. Змінна визначається ім'ям, яке починається з букви, може мати в середині цифри та знак підкреслення. Відмітимо, що великі та малі букви в іменах не розрізняються.

Приклади:

B

XZm

Stanok

Rikl998

MashFak

Довжина імені може бути яка завгодно, але **транслятор** «розуміє» тільки перші 63 символи.

Букви кирилиці в іменах використовувати не можна.

Змінна одержує та змінює (може і не змінювати) своє значення в процесі роботи програми.

Кожна змінна повинна бути описана на початку програми після слова **Var.** При цьому вказується тип змінної. Turbo Pascal має справжній «букет» типів.

Таблиця 5.1 – Типи цілих змінних

Тип	Діапазон	Розмір в байтах
Byte (коротка ціла без знаку)	0..255	1
Shortint (коротка ціла із знаком)	-128..127	1
Word (ціла без знаку)	0..65535	2
Integer (ціла із знаком)	-32768..32767	2
Longint (довга ціла із знаком)	-2147483648.. 2147483647	4

Програміст сам вибирає потрібний йому тип в залежності від можливих значень змінної (з урахуванням діапазону даних та розміру пам'яті, яка виділяється транслятором під окрему змінну). Всі можливі типи дійсної змінної наведені в табл. 5.2.

Стандартний тип для дійсної змінної — **Real.** Його розуміє будь-який транслятор на машині з будь-якою конфігурацією. Інші типи рекомендується вживати, якщо в ПЕОМ є математичний сопроцесор Intel 80x87 (наприклад, 80387), який дозволяє швидко обробити такі дані. Якщо такого пристрою нема, то його можна емулювати за допомогою спеціальної ди-

рективи транслятору {$E+}. При цьому дійсні змінні будуть оброблятися програмним шляхом, але це суттєво зменшить швидкість виконання програми.

Таблиця 5.2 – Типи змінних

Тип	Диапазон	Кількість цифр	Розмір в байтах
Real (дійсний)	$10^{-39} – 10^{38}$	11—12	6
Single (одинарна точність)	$10^{-45} – 10^{38}$	7—8	4
Double (подвійна точність)	$10^{-394} –10^{308}$	15—16	8
Extended (розширена точність)	$10^{-4954} – 10^{4992}$	19—20	10

На відміну від інших мов програмування в системі Turbo Pascal треба обов'язково описувати всі змінні на початку програми.

Приклад:

Var A,B : Real;

X : Double;

I, J, K : Byte;

F : Integer;

Для описування символьних змінних вживають слово **Char**, рядкових — **String**, логічних **— Boolean.** *Приклад:*

Var Z, Buk, Sim : Char;

Str : String [7];

S : String;

Q,R : Boolean;

Найбільша довжина рядку – 255 символів, в квадратних дужках вказують довжину для конкретної змінної. Якщо довжина не вказана, то автоматично береться найбільша – 255.

Ці ж зарезервовані слова вживають при описуванні так званих типованих констант. *Приклад:*

***Const* N :Integer=40;**

Tabl :String [20] =’Табуляція функції’;

Prizn : Boolean=True;

На відміну від простих констант, типовані константи можна змінювати в програмі, але їх не можна використовувати при описуванні масивів.

Всі приведені вище типи даних застосовують для обробки числових, логічних та текстових даних.

В мові Pascal велику увагу приділяють забезпеченню наочності програми. Це досягається декількоми шляхами, один з них – використання перелічуваних даних.

Описування таких даних робиться з уживанням стандартного слова **Type** та слів, придуманих програмістом і записаних латинськими літерами. *Приклад:*

Type Transport = (Tramvay, Mashina, Potyag, Metro, Wiz);

Тепер треба описати змінну, яка буде мати тип **Transport.**

Var Pasagir: Transport ;

Якщо на початку програми зустрінеться така конструкція, становиться зрозумілою "тема" програми.

Слід зауважити, що слова в круглих дужках можна писати в якому завгодно порядку, але для цих слів автоматично підтримується властивість впорядкованості. Для нашого прикладу маємо:

- Tramvay < Mashina < Potyag...

Взагалі, для елементів списку можна вживати операції відношення **=, <>, <, < =, >, > =.**

Інша важлива властивість програм на мові Turbo Pascal – їх надійність. Є чимало способів для її забезпечення, один з них – використання даних обмеженого типу (інші назви – діапазон, "відрізок").

Базовими для обмеженого типу є цілі, символьні, логічні та перечислювані типи (зверніть увагу – нема дійсних даних).

Початкове та кінцеве значення обмеженого типу розділяються двома послідовними точками.

Приклад:

Var Grupa 1 .. 30;
Class 'A' .. 'H';
Wicladach Shkola ..Wuz;

Тепер видно, що змінна, наприклад, **Class**, повинна змінюватися тільки в межах букв від **'A'** до **'H'**.

Підтримка надійності програми в даному випадку залежить від того, вживається чи ні відповідна директива компілятора.

Директив взагалі багато. Кожна з них записується в окремому рядку програми. Знак *«+»* визначає, що директива активна **(ON),** знак **«-»** — пасивна **(OFF)**. Для контролю обмеженого типу вживають директиву **{$R}.** Якщо директива активна і робиться спроба вийти за описані межі, з'явиться повідомлення про помилку. Якщо директива не активна, то контроль не ведеться.

Треба відмітити, що багато директив компілятору продубльовано в турбосередовищі.

Якщо в програмі нема директиви **{$R},** то все залежить від результатів попереднього налагодження режиму ***Options /Compiler/Range checking***. Якщо цей режим має значення **ON,** це відповідає директиві **{$R+},** якщо — **OFF**, то контроль не ведеться.

Дані обмеженого типу вживають при описуванні масивів. Це означає, що є можливість контролювати вихід індексу масиву за його межі. Відомо, що така помилка зустрічається в практиці програмування досить часто.

Досвідчені програмісти роблять так: перед небезпечним місцем ставлять директиву **{$R+}** (це, наприклад, таке місце, де обчислюється індекс і важко визначити, якого значення він може набути). Після проходження небезпечного місця програми ставлять директиву **{$R-},** яка знімає контроль. Такий прийом дозволяє зоощаджувати час, який витрачається на контролювання даних обмеженого типу.

При розв’язуванні задач з допомогою ЕОМ, в основному використовують два види даних: прості змінні та масив.

Всі приклади змінних, наведені в цьому розділі, відносяться до простої змінної.

Тепер розглянемо особливості описування масивів. Для цього є декілька способів.

Приклад безпосереднього описування одновимірного масиву та матриці:

```
Const N=5; M=6;
Var X : array[l..N] of Integer;
      R : array[l..4,1..M] of Real;
```

Ім'я масиву вибирається так само, як ім'я змінної. В квадратних дужках записують межі для індексу (індексів), після слова **of** вказують тип елементів масиву.

Іноді масив описують в два етапи. *Приклад:*

```
Const K=25;
Type Clas=array[1..K] of Char;
Var C1,C2,C3 : Clas;
```

В мові Turbo Pascal немає очевидних обмежень на розмірність масивів та кількість елементів по кожному виміру. Але має місце значне обмеження на загальний обсяг пам'яті, яка виділяється під всі дані програми – **64 Кб**.

Незалежно від способу описання масиву компілятор виділяє для нього необхідну пам'ять. Зверніть увагу на те, що довжина масиву всюди, навіть в підпрограмах, має конкретне значення (в одній із старших версій цей недолік ліквідовано). Елементами масива можуть бути будь-які дані, в тому числі є поняття «масив масивів». *Приклад:*

```
Const k=8;
Type Massiv=array[l..k] of real;
Var X:array[1..6] of Massiv;
```

В цьому випадку **X** — це матриця, в якій 6 рядків та k стовпців.

Нарешті ще одна новинка: константа-масив. *Приклад:*

```
Type Transport = ( Tramvay, Mashina, Potyag, Metro, Wiz);
  Const A :array[Transport] of String[10]=('Трамвай', 'Машина',
'Потяг', 'Метро', 'Віз' );
 Var Pasagir: Transport ;
```

Така конструкція дозволяє об'єднати в одній програмі перечислюваний тип з англомовними словами з масивом рядків, які мають значення в вигляді, наприклад, слів на українській мові. Потреба в цьому виникає, наприклад, в задачі будування **меню**.

Звертатись до елементів масивів можна за допомогою індексованих змінних. *Приклади:*

```
X[3]
X[j]
X[3*K-2]
R[1,3]
```

R[i,j]
R[5,M]

Для масиву-константи індексована константа матиме такий вигляд: **A[Transport].**

Якщо **Transport = Metro**, то оператор **Writeln(A[Transport]);** виведе на екран слово **Metro**.

5.3 Вирази, оператор присвоєння

Обробка даних виконується в виразах та операторах присвоєння.

В мові Pascal вирази не поділяють на арифметичні та логічні (як, наприклад, в Фортрані).

Це потребує від програміста певної обережності при конструюванні виразів та аналізі можливого результату.

Вираз може складатися з констант, змінних, знаків операцій, круглих дужок та функцій.

Нижче приведені знаки операцій у порядку зниження старшинства їх виконання:

@,Not

***, /, Div, Mod, And, Shl, Shr**

+, -, Or, Xor

=, <>, <, >, <=, >=, In

Операція **@** вживається частіш за все під час роботи з динамічною пам'яттю. При роботі з логічними даними виникає потреба брати протилежне (негативне) значення логічної величини. Для цього застосовують операцію **Not.** Так, якщо вираз

A>B має значення **"True"** то вираз **Not(A>B)** при тих же **A** і **B**, матиме значення **False.**

Знак ***** визначає операцію множення, а **/** — операцію ділення. Ділене та дільник можуть бути цілими та дійсними, а частка — завжди дійсне число. На відміну від цього операції **Div та Mod** мають справу тільки з цілими операндами. Операція **Div** обчислює цілу частину частки, а **Mod** — залишок.

Приклад:

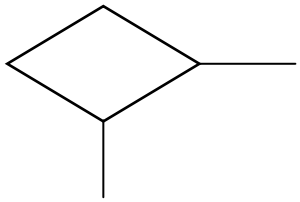
Результат операції 10 **Div** 3 дорівнює 3, а результат 10 **Mod** 3 буде рівним 1.

Зверніть увагу на те, що перед та після слів **Div** та **Mod** ставлять проміжок.

Типове використання операції **and** —при роботі з логічними даними. Результат операції обчислюється за відомою таблицею істинності: він буде **True** тільки в тому випадку, коли обидва операнди мають таке ж значення.

Приклад:

В схемі алгоритму є блок:



Такий блок можна реалізувати за допомогою одного оператора **If**

If (X>3) and (X<A) then ... *else*

Обчислимо результат для двох варіантів даних:

1. X=5, A=8.

Обидві операції відношення **X>3** та **X<A** дадуть результат **True,** тому і після операції **And** результат буде **True,** що відповідає виходу **Then** оператора **If**.

2. X=5, A=4.

Тепер операція відношення **X<A** буде **False,** в операції **And** один з операндів **False,** що відповідає виходу **Else.**

Операндами операції **Shl** та **Shr** можуть бути цілі числа. Ці операції забезпечують циклічні зсуви відповідно ліворуч та праворуч двійкових розрядів чисел.

Приклад: ***2*** **Shi 3**

Обчислимо результат, для чого спочатку число 2 запишемо за допомогою 8 двійкових розрядів: 00000010. Тепер зсунемо ліворуч на 3 позиції всі розряди, будемо мати 00010000, а це число 16. Значить: 2 **Shi** 3= 16.

Такі операції корисні для грамотних програмістів.

Операція **Or** реалізує логічну функцію диз'юнкція, її результат **True,** якщо принаймні один з операндів буде **True.**

Результатом операції **Xor** буде **True,** якщо значення операндів не збігаються, в протилежному випадку — **False.**

Як не дивно, операції **And, Or, Not**, **Xor** можуть бути корисними при роботі з графікою.

Ще одне незвичне використання мають ці операції, якщо їх операнди — цілі числа. Тоді вони оперують з двійковими розрядами цих чисел. Це вживається, наприклад, для накладання маски при роботі з перериваннями.

Дійсно унікальною операцією є операція **In**. Її операндами є: з одного (правого) боку — множина деяких елементів, а з другого – елемент, який може входити або не входити в множину. *Приклад:*

***U* Bukva in ['a', 'e', 'и', 'o', 'y', 'ю', 'я'] then ...**

Тут **Bukva** — змінна типу **Char,** а в квадратних дужках записана множина голосних букв. Якщо значення змінної **Bukva** збігається з однією з букв множини, результат операції буде **True,** якщо ні – **False.** Це зручно під час обробки текстів.Взагалі дані типу «множина» використовуються рідко. Тепер розглянемо останній елемент виразу — функції. В мові Turbo Pascal є своя бібліотека функцій (файл **Turbo.Tpl),** в якій вони згруповані в окремі модулі **(Unit).** Це означає, що для використання тієї чи іншої функції треба оператором **Uses** підключити до програми відповідний модуль, наприклад: **Uses ert**. Ця вимога не відноситься до модуля **System**, який підключається до будь-якої програми автоматично. В цьому модулі розміщені в основному всі математичні функції, *наприклад:***Abs(x) — x Arctan(x) — arctg(x) Cos(x) — cos(x) Sin(x) — sin(x) Exp(x) — e^{x} Ln(x) — ln(x) Sqr(x) — x^2 Sqrt(x) — √x, Round(x)** — функція заокруглення дійсного числа до цілого. **Odd(x)** – перевіряє парність аргументу. Якщо це парне число, результат функції **False,** якщо непарне – **True**.

Random(x) — виробляє випадкове число від О до х. *Приклади виразів:*

Математика	**Turbo Pascal**
$ax^2 + b$	**a*Sqr(x)+b**
$\frac{(a+b)}{\lvert c-d \rvert}$	**(a+b)/Abs(c-d)**
$e^x \ln(x+2)$	**Exp(x)*Ln(x+2)**
$\sqrt{x^2+1}$	**Sqrt(Sqr(x)+l)**
a^b	**Exp(b*Ln(a))**

Останній приклад потребує невеликих пояснень. В мові Pascal відсутня операція піднесення до степеня, тому її замінюють поєднанням функцій **Exp** та **Ln** (треба спочатку прологарифмувати вираз a^b) .

Розглянемо особливості обчислень виразів з цілими та дійсними даними.

Якщо обидва операнди в операціях +, -, * однакового типу, то результат буде такий самий.

Якщо в операції / обидва операнди дійсні, то результат теж дійсний, а якщо в цій операції операнди цілі, то результат буде все одно дійсним.

Якщо обидва операнди цілого типу, але різного діапазону, то спочатку вони перетворюються на загальний тип, а вже потім виконується операція. Загальним типом є цілий тип з найменшим діапазоном, який включає всі можливі значення обох типів. Так, загальним типом для цілого **(Integer)** та цілого **(Byte)** буде **Integer**, а загальним типом для **Integer** та **Word** є **Longint**. Типом результату операції буде загальний тип.

В тому випадку, коли один або обидва операнди в операції мають дійсний тип, то тип результату буде дійсним, якщо використана директива компілятора **{$N-},** або типом з подвійною точністю при використанні директиви **{$N+}**. В останньому випадку всі обчислення будуть виконуватись з допомогою числового співпроцесора.

Вираз обчислюється відповідно до старшинства операцій. В сумнівних випадках рекомендується вживати круглі дужки.

Оператор присвоєння має загальний вигляд:

a:=b;

де **a**—проста або індексована змінна, а **b** —вираз.

При виконанні оператора спочатку обчислюється вираз, а потім результат присвоюється змінній **a**. При цьому повинна бути сумісність за присвоюванням. Тут багато випадків, головні з них такі:

- **a** та **b** є цілими типами та значення **b** попадає в діапазон значень **a;**
- **a та b –** дійсні типи та значення **b** попадає в діапазон допустимих значень **a**;
- **a –** дійсний тип, а **b** – цілий (але не навпаки);
- **a** та **b** – рядкові дані;
- **a -** рядок, а **b** – символ.

Приклади операторів присвоєння:

```
Var A,B,C : real;
    I, J, K : integer;
    R1,R2 : string[20];
       S1,S3 : char;
       K:=X div J;
   C:=A+B/2;
    Rl :=’текст’;
    R2 :=R1;
    SI :=′*′;
    S3:=S1;
    A :=K; {ціле число буде перетворене в дійсне).
```

Якщо треба цілій змінній присвоїти дійсне число можна використати функцію округлення, *наприклад:*

```
I:=Round(B);
```

Початківцям особливу увагу треба звернути на оператор такого вигляду:

```
k:=k+l;
a:=a - b;
```

Це звичайний оператор присвоєння, його вживають для змінювання поточного значення змінної (тобто, спочатку змінна k буде якимось чином визначена, Наприклад, **k:=l**;).

Оператори керування

Програма — це послідовність операторів, які виконуються один за одним. В багатьох випадках виникає потреба в зміні такого порядку, що стає можливим завдяки операторам керування. До них в першу чергу відносяться **Goto, If, Case, Exit, Halt** та оператори циклу. Зважаючи на важливість питання організації циклів, цю тему винесено в окремий розділ (див. 5.4).

Оператор **Goto** реалізує операцію безумовного переходу з одного місця програми в інше. Загальний вигляд:

```
Goto m;
```

де **m** — мітка того оператора, якому передається керування.

Мітка в мові Turbo Pascal може починатися з букви (як ім'я змінної) або визначатися цифрами від 0 до 9999. Незалежно від вигляду кожна мітка повинна бути описана на початку програми після слова **Label. Наприклад:**

Label Nl, 63, VSE;

Можна помічати оператори виконання та оператор **End.** Мітка ставиться перед оператором і відокремлюється від нього двокрапкою. Перед **Begin** мітку не ставлять. **Наприклад:**

```
63:  A:=sin(x);
  X=A+0.25;
  Goto VSE;
  Nl:  Readln(C);
  ……..
    VSE: writeln('Результати роботи ');
```

Є деякі обмеження на вживання **Goto.** Не можна передавати керування на оператор всередині підпрограми (її треба викликати). Треба також ухилятися від переходів всередину складеного оператора **Begin ... End.**

Деякі автори рекомендують зовсім відмовитись від оператора **Goto.** Більшість сходиться на тому, що вживати його можна, а от зловживати ним не треба, бо це приводить до погіршення розуміння програми.

Оператор **If** реалізує операцію умовного переходу (операцію розгалуження на два напрямки).

Загальний вигляд:

If *умовний вираз* **then ... else ...;**

В умовному виразі задається умова розгалуження. При виконанні оператора **If** цей вираз обчислюється з отриманням логічного результату.

Якщо результат **True,** то виконується простий або складений оператор після слова **Then.** Якщо результат **False,** то виконується оператор після **Else.**

Наприклад:

```
If  A< 3.2 then Y:=5*A else goto Ml;
```

Частину оператора **Else** ... можна не вживати.

```
If X[i]>0 then K:=K+l;
```

Це означає, що у випадку, коли число **X[i]** додатне, буде виконано оператор **К:=К+1**. Якщо така умова для конкретного числа хибна, то **К** не змінюється, а керування передається на оператор, який в програмі записано після **If.**

Немає обмежень на складність умовного виразу.

If(Q='Y') or (Q ='y') then...

Така конструкція буває корисною в діалогових програмах, коли користувач дає згоду (**Y** — перша літера слова **Yes**) великою або малою буквою **Y**.

Складений оператор **Begin ... End** суттєво розширює можливості **If:**

```
If A[i,j]<=0 then begin
                  k:=k+l
                  R[k]:=a[i,j];
                 End
Else begin
                 l:=l+1;
                 Q[l]=A[i,j];
                 End;
```

В складеному операторі записують будь-яку кількість операторів. Вони виконуються «як одне ціле». Тут можуть бути «свої» **If,** цикли тощо. Можна вийти з групи **Begin ... End** оператором **Goto.**

Зверніть увагу на те, що після оператора, який стоїть перед **Else,** не ставиться крапка з комою.

Формально оператори **If** можуть бути вкладеними. Це погіршує наочність програми, тому для полегшення її аналізу треба вкладений **If** записати як складений оператор.

```
If умова1  then Begin {зовнішній If}

 If умова2 then ...
              Else ...
          End
        Else .... ; {зовнішній If}
```

В таких випадках краще «пари» **Then ...Else** внутрішнього та зовнішнього **If** писати одне під одним.

Оператор **Case** забезпечує розгалуження на декілька напрямків.

Загальний вигляд:

Case *індекс вибору* **of** *список вибору;*

Else ... ; end;

де *індекс вибору* — проста змінна цілого, символьного, перечислюваного або логічного типу;

список вибору — сукупність простих або складених операторів, перед кожним з яких стоїть константа вибору, тип якої збігається з типом індексу вибору.

Після слова **Else** може стояти простий чи складений оператор (ця конструкція може бути відсутня).

Приклад:

```
Case Note of
1: Y:=sin(x);
   2: Y:=cos(x);
3: Y:==sin(x)+cos(x);
Else Y:=0;
End; {case}
```

Змінна Note (цілого типу) повинна бути визначеною до виконання оператора **Case**. Якщо Note дорівнює 1, обчислюється функція У:=sin(x), якщо вона дорівнює 2, то Y:=cos(x). В тому випадку, коли Note відрізняється від 1, 2 або 3 буде виконано оператор Y:=0.

Не треба змішувати мітки операторів з **константами вибору.** Останні описувати немає **потреби.**

Приклад:

```
Case T of
    M .. 20: begin
    ……
            End;
    21 .. 30: Oznaka:=l;
    End;
```

Якщо ціла змінна Т має значення з діапазону М..20, виконується складений оператор, а якщо Т в діапазоні 21..30, то оператор Oznaka:=1.

Оператор в списку вибору може мати декілька констант вибору.

Приклад:

```
Var Znak: char;
```

```
……..
Case Znak of
 ‘a’,’o’,’e’,’u’: begin {перший складений оператор}
                  ……..
                  End;
 ‘x’,’z’: begin     {другий складений оператор}
          ……..
            End;
End; {Case}
```

В цьому випадку аналізується символьна змінна **Znak** і якщо вона збігається з однією з букв ‘a’, ‘o’, ‘e’, ‘u’, то виконується перший складений оператор, якщо ні – другий.

Приклад вживання індексу вибору логічного типу:

```
Var A: array[1..20] of integer:
……..
Case Odd (A[i]) of
   False: begin
    k:=k+l;
    Y[k]:=A[i];
      End;
      True: begin
    J:=J+1;
    R(j):=A(j);
            End;
End; {Case}
```

Це фрагмент програми, в якій створюється масив Y з парних чисел масиву A та масив R з непарних чисел того ж масиву.

Нарешті *приклад* на використання змінної перечислюваного типу.

```
Type Transport = ( Tramvay, Mashina, Potyag, Metro, Wiz);
Var Pasagir: Transport;
Case Pasagir of
```

Tramvay: writeln(‘це пасажир трамваю’);
Mashina: ;
Potyag: ;
Metro: ;
Wiz: begin;
Writeln(‘це пасажир воза’);
End;
End; {Case}

Зверніть увагу на «пусті» оператори в деяких рядках програми. Так на перших етапах розробки програми організують так звані програмні заглушки, щоб колись вставити на це місце потрібні оператори.

5.4 Організація циклів

Відомо, що ***цикл*** — це група операцій, які виконуються багаторазово.

В мові Pascal є три спеціальні оператори організації циклів: **For, While та Repeat.**

Загальний вигляд циклу **For:**

For *параметр:=вираз1* **{to**
downto} *вираз2* **do** *{оператор}**;***

де *параметр* — змінна цілого, символьного, логічного або перечислюваного типу (нема дійсного типу!);

вираз1, вираз2 — відповідно початкове та кінцеве значення параметра (константа, змінна або вираз);

оператор — простий або складений оператор (це, так зване, тіло циклу). *Приклад:*

For i:=l to 25 do A[i]:=0;

В циклі **For** параметр завжди змінюється з кроком “один”. Якщо **i** — ціла змінна, то вона збільшується від 1 до 25. В загальному випадку параметр цілого типу може бути від'ємним, *наприклад:*

For K:= -5 to 2 do ... ;

Якщо треба зменшувати параметр, вживають слово **downto.** *Приклад:*

For i:=30 downto 5 do begin
…..
End;

В цьому випадку параметр **i** буде зменшуватись з кроком 1, при кожному значенні **i** виконується складений оператор. З циклу можна вийти достроково.

Вважається, що параметр циклу після його завершення дорівнює кінцевому значенню.

Якщо кінцеве значення дорівнює початковому, то **цикл** виконується один раз.

Якщо кінцеве значення недосяжне, то цикл не виконується зовсім.

Не рекомендується змінювати в середині циклу **For** *параметр* та *вираз1* або *вираз3.*

Важливо, щоб типи виразів та параметра збігались.

Якщо в тілі циклу треба виконати декілька операторів (більше одного), то їх треба оформити як складений оператор **Begin...End.** В ньому можна вживати нові оператори виконання, в тому числі інші цикли.

Приклад:

```
   For i:=l to N do begin
                Y:=X[i]*Sin(X[i]);
                Writeln(X[i],Y);
           End;
……….……….……
For i:=l to N do
      For j:=l to M do C[i,j]:= -l;
```

В цьому випадку тілом внутрішнього циклу є цикл з параметром **j**, його не треба оформляти як складений оператор, тому що він один.

Приклад:

```
For i:=1 to N do begin

                  Sum:=0;
                For j:=l to M do Sum:=Sum+D[i,j];
               R[i]:=Sum;
         End;
```

Така група операторів обчислює суму елементів кожного рядка матриці **D** та заносить її до масиву **R.**

В тілі зовнішнього циклу три оператори, тому вжиті операторні дужки **Begin ... End.**

Цікавим є використання параметрів інших типів.

Приклад:

Var Znak: Char;

For Znak:=’A’ to ‘Z’ do ... ;

В цьому випадку змінна **Znak** послідовно приймає значення всіх великих букв латинської абетки, тобто, в цьому разі крок “один” означає перехід до наступної букви.

Приклад:

Type Transport = (Tramvay, Mashina, Potyag, Metro, Wiz);

Var Pasagir: Transport;

For Pasagir:= Tramvay to Wiz do …:

Зверніть увагу, що параметр **Pasagir** спочатку прийме значення **Tramvay**, потім **Mashina** і т.д. *Приклад:*

Var Q: boolean;

……

For Q:=False to True do ... ;

Такий цикл виконається усього два рази, до того як **False < True.**

Цикл **While** називають циклом з передумовою. Загальний вигляд:

While *умова* **do** *оператор;*

де *умова* – це вираз, який після обчислення приймає значення **True** або **False;**

оператор — простий або складений оператор.

Типовим є таке вживання цього циклу, коли в його тілі змінюється хоча б одна змінна, яка входить до “умови”.

Зручніше розлядати цикл **While** за допомогою такої структури:

N:=0;

While N<=52 do begin

N:=N+1;

C[N]:=Cos(X[N]);

End;

Ціла змінна **N** приймає значення перед циклом. При такому **N** умова **N<=52** приймає значення **True,** значить тіло циклу буде виконуватись. В складеному операторі **begin ... end** окрім інших є оператор **N:=N+1**;, який змінює параметр **N.** Після того, як **N** дорівнюватиме 52, тіло циклу вико-

нається в останній раз **(N:=52+1, ...),** тобто буде сформовано 53 елементів масиву **C.**

Якщо значення "умова" з самого початку **False,** цикл не виконується жодного разу.

Такий цикл зручно використовувати для зміни дійсного параметра з будь-яким кроком.

```
X:=1; {X –  Real}

  While X<=35 do begin
  Y:=Sqr(X);
  Writeln(X.Y);
 X:=X+2;
End;
```

Зверніть увагу на те, що в цьому випадку тіло циклу виконається останній раз при **X=35.**

Взагалі цикл **While** вживають у різних випадках. *Приклад:*

```
Var Litera: char;
……
While Litera <>'*' do ....
```

Такий цикл буде працювати до тих пір, поки змінна **Litera** не прийме значення '*'. *Приклад:*

```
While not Eof(f)  do ...
```

Відомо , що функція **Eof(f)** дорівнює **True,** якщо при роботі з файлом **f** досягнуто спеціальної позначки "кінець файла". Операція **Not** змінює на протилежне значення **Eof(f)**, тобто такий цикл виконується, доки не зустрінеться "кінець файла".

Цикл **Repeat** називають циклом з постумовою. Для його розуміння можна застосувати, наприклад, таку структуру:

```
X:=1;       {X має тип Real }

  Repeat      {початок циклу }
Y:=Sqr(X); {початок тіла циклу }
Writeln(X,Y);
X:=X+2;   {кінець тіла циклу }
Until X>35;   {кінець циклу }
```

Тіло циклу завжди виконується хоча б один раз, тому, що умова продовження циклу стоїть у кінцевому операторі **Until.**

Результат обчислення умови завжди логічна величина, якщо він **False** цикл продовжується, а якщо **True** — перестає робити. В наведеному .прикладі цикл буде працювати до тих пір, доки **X** не перевершить число 35.

Інші приклади:

```
I:=1;
Repeat
……
I:=I+1;  {цикл завершиться, як тільки }
Until l=d;  {змінна I дорівнюватиме d}

Var ch: char;

……

  Repeat      (цикл завершиться, як тільки }
Until ch='*';   {змінна ch збіжиться з '*' }
Repeat  {тут можуть бути будь-які оператори }
Until KeyPressed,
```

В мові Turbo Pascal логічна *функція* **KeyPressed** контролює натискання клавіш. Якщо жодна з клавіш не натиснута, ця функція виробляє значення **False, в** протилежному випадку – **True.**

Конструкція **Repeat Until KeyPressed** вживається для організації паузи в процесі виконання програми (до натискання будь-якої клавіші).

```
Repeat
……
Until False;
```

Це приклад нескінченного циклу. Вихід з такого циклу треба здійснити будь-яким чином.

5.5 Введення-виведення даних

Під час роботи на персональному комп'ютері е декілька можливостей для введення-виведення даних. Тут будуть розглянуті типові операції введення даних з клавіатури в оперативну пам'ять та виведення даних з пам'яті на екран дисплея або принтер.

Введення даних з клавіатури підтримується стандартним текстовим файлом **Input,** а виведення на екран – файлом **OutPut.** В мові Turbo Pascal ці файли можна ніде не описувати, їх не треба відкривати або закривати, як це робиться з іншими типами файлів .

Якщо в операторі **Read** або **Readln** не вказано ім'я файла, значить вживається файл **Input.**

В операторах **Write** або **Writeln** можна не писати ім'я **OutPut,** якщо інформація виводиться на екран дисплея. При необхідності віддрукувати результати на папері принтера в операторах виведення треба вказати ім'я файла **Lst**, а оператором **Uses** підключити модуль **Printer,** який "підтримує" цей файл.

Загальний вигляд операторів введення даних з клавіатури:

Read(список);
Readln(список); {список може бути відсутнім}

В списку перелічуються через кому імена змінних, значення яких вводиться з клавіатури. Тут можна вказати просту чи індексовану змінну або складене ім'я поля запису **Record**.

Приклад:

```
Read(A,B,C);
Read(X[i],Y[i]);
Readln(K,J);
```

Якщо вводяться значення числової змінної, то ці оператори читають послідовність літер, формат якої погоджено з форматом числової константи відповідного типу. Якщо така послідовність не відповідає формату, то виникає помилка.

Звичайно, після числа вводиться один чи декілька проміжків або натискується клавіша ***Enter.***

Введена послідовність сприймається як число і записується в відповідну комірку пам'яті. Проміжки перед числом та після нього ігноруються.

Якщо послідовність пуста (зразу було натиснуто ***Enter*****),** то змінна зберігає попереднє значення.

При введенні змінної **Char** читається одна літера.

Якщо вводиться рядкова змінна **String,** то читається стільки символів, скільки вказано в довжині змінної або менше (при натисканні клавіші ***Enter)***.

Між операторами **Read та Readln** є деяка різниця. Її можна продемонструвати прикладами.

```
Read(x.y);   {x,y — Real}

Read(a,b,c); {a,b,c — Integer}
```

При виконанні таких операторів дані можна вводити з одного рядка, *наприклад:*

```
4.5  6.8  15  -4  10
```

Якщо замість першого оператора поставити **Readln(x,y),** а числа вводити як і раніше, виникає така ситуація: після читання двох перших чисел буде зроблено перехід на наступний рядок (якщо він є), тобто останні три числа першого рядка будуть пропущені. Взагалі, після виконання **Readln,** завжди відбувається перехід до наступного рядка, незалежно від того є в операторі список даних чи нема.

При введенні масиву треба здійснити відповідний цикл для зміні індексу.

Приклад:

```
Var X:array[1..10] of real;
      i: integer;
      ……….

 For i:=l to 10 do read(X[i]); readln;
```

Якщо треба ввести елементи матриці, вживають вкладені цикли. *Приклад:*

```
Const M=6; N=4;
Var y: array [1..M,1..N] of integer;
   I,J: integer;

 For I:=1 to M do begin
```

```
For J:=1 to N do read(y[I,J]);
                Readln;
              End;
```

Зверніть увагу на складений оператор **Begin ... End.**

Саме так його треба записувати, це дасть змогу вводити матрицю в "природному" вигляді — рядок за рядком. Якщо вилучити слова **Begin** та **End,** то матрицю доведеться вводити у вигляді одного рядка, а якщо замість **Read** записати **Readln,** то матриця витягнеться у стовпець.

Масив можна вводити в будь-якому циклі, а не тільки в циклі **For.**

Звичайно, при введенні даних заздалегідь відома їх кількість.

Інколи виникає потреба ввести невідому кількість даних. В наведеному прикладі забезпечується таке введення.

```
Begin
CheckEof:= True;
Sum:=0;
Kol:=0;
While not Eof do begin
 Readln(X);
 Sum:=Sum+X;
 Kol:=Kol+l;
          End;
WriteIn('введено',Kol,'даних',' їх сума=',Sum);
End;
```

В прикладі використано функцію **Eof** , без імені файла, значить припускається стандартний файл **Input.** Такий цикл буде працювати, доки не буде натиснуто одночасно дві клавіші **CTRL** та **Z** (це ознака "кінець файла" **Input).** Результат залишиться незмінним, якщо записати **EofInput**).

Треба зауважити, що ознака "кінець файла" при натисканні **CTRI/Z** буде вироблятися тільки в тому випадку, коли спеціальна змінна **CheckEof** із модуля **Crt** матиме значення **True.** Якщо не виконати оператор **CheckEof:=True,** то ця змінна матиме значення **False.**

Виведення даних виконується операторами **Write та Writeln.** Загальний вигляд цих операторів:

Write(список);

Writeln(список); {список може бути відсутнім)

В списку перелічуються через кому імена змінних, вирази та текстові константи. Можна вживати прості, індексовані або складені імена. Текстова константа записується у лапках. Крім того, в список можна включати специфікацію формату для керування виведенням даних. Якщо в списку є вираз, то він попередньо обчислюється, а вже потім його результат виводиться на екран.

Приклади:

Write(X,Y);

Write(a,b,a+b);

Writeln('X=',X,'Y=',Y);

В останньому прикладі буде виведено назву кожної змінної та її значення.

Writeln(‘Введіть початкові дані’);

Такий оператор розміщують перед операторами введення даних, на екрані з'явиться своєрідна підказка користувачу.

Після виконання **Writeln** забезпечується перехід на наступний рядок. Так трійка операторів

Write(K,N);
WriteIn(A,B);
Write(t);

виведе в першому рядку значення змінних **K, N, A** та **B**, а в наступному – значення **t**.

В тих випадках, коли в операторах виведення не регулюється розмір поля, під значення вживаються такі розміри: для цілих та логічних змінних 15 позицій, для дійсних значень — 18. Значення змінної розміщується з правого боку виділеного **поля.** Дійсне значення має нормалізовану форму з таким шаблоном:

x.xxxxxxxxxxE ***знак*** **xx**

Тобто всього в числі 11 цифр, в цілій частині – одна, значення числа зберігається автоматично за рахунок правильного вибору порядку.

Приклад виведення дійсної змінної:

Writeln('A=',A);

Результат:

A=-2.8300000000E+00

Для регулювання розміру поля треба через двокрапку після імені змінної записати число, а для дійсної змінної – два числа. Перше (або єдине) число вказує на загальний розмір поля, а друге – на кількість цифр після крапки (замість чисел можна писати цілі змінні). *Приклад:*

```
Writeln('X=',X:7);      {X—Integer}
Writeln(S:60);          {S—String }
Writeln(K:10:2);        {K — дійсна змінна}
```

В тому випадку, коли в дробовій частині **числа** більше цифр, ніж треба вивести, проводиться округлення, *наприклад:*

```
N:=10; M:=2; A:=25.979;
Writeln('A=',A:N:M);
```

Результат матиме вигляд: A=25.97. Якщо під дробову частину виділити 0 позицій, число буде виведено без неї і без крапки. Виведення масивів виконується в циклі. *Приклад:*

```
Var A: array[1..5,1..6] of integer;
……….
For i:=l to 5 do begin
For j:=l to 6 do write(A[i,j]:7);writeln;
                End;
```

В цьому випадку матриця буде виведена відповідно з тим, як вона описана (5 рядків по 6 чисел).

Чимале значення має наочне оформлення результатів роботи програми. В мові Turbo Pascal є декілька можливостей для цього. За допомогою операторів виведення можна забезпечити подання результатів у вигляді таблиць, рамок тощо.

Приклад виведення "шапки" таблиці:

```
WriteIn('----------------------------------------');
WriteIn(' I          I прізвище I оцінки I ');
WriteIn(' I номер  I              I          I ');
Writeln(' I          I студента  I  МФП  I ');
WriteIn('————————————————————————————————');
```

Такий спосіб зручний тим, що відразу видно, який вигляд матиме таблиця.

Щоб вивести результати в таку таблицю в операторі **Writeln,** треба поєднати текстові константи та імена відповідних змінних.

Приклад.

Writeln('I ',N,' I ',Pr,' I ',mat,' I ',Fiz,' I ',Prog,' I ');

Таблиця буде акуратною, якщо прізвища всіх студентів будуть однакової довжини (проміжки добавляти з правого боку).

В мові Turbo Pascal є можливість контролювати операції введення-виведення даних. Для цього вживають директиву компілятора **{$I}** або відповідний підрежим **I/O** *Checking (Options / Compiler)* турбосередовища. Якщо директива **{$I}** записана із знаком + (тобто **{$I+}**), то в процесі роботи програми здійснюється контроль за операціями введення/виведення і, якщо виникає помилка, виконання програми припиняється, на екрані дисплея з'являється повідомлення про помилку.

Якщо директива **{$I}** не активна, тобто **{$I-},** контроль здійснюється, але при виникненні помилки виконання програми не припиняється, а спеціальна функція **IoRezult** приймає значення, яке відрізняється від нуля. Треба звернутися до цієї функції (опитати), після чого стан помилки буде "скинуто" і роботу програми буде продовжено.

Такий режим зручно використовувати тоді, коли зупин програми при виникненні помилки введення/виведення є небажаним.

Приклад:

```
{$I-}
Writeln ('Введіть дійсне число');
 M: ReadIn (A);
If IoRezult <>0 then begin
Writeln('помилка, введіть число знову');
Goto M;
End;
{продовження програми }
```

Якщо введене число правильне (з точки зору синтаксису), то функція **IoRezult** буде дорівнювати нулю, а програма буде нормально виконуватись.

В тому разі**,** коли введене число помилкове, наприклад, замість крапки в числі буде кома, виникне помилка, функція **IoRezult** стане відмінною від нуля, з'явиться повідомлення “помилка, введіть число знову” і керування буде передано на оператор **Readln(A).**

Якби директива **{$1}** була активна, то в останньому випадку з'явилось би повідомлення **Invalid numerical format** (неправильний числовий формат) і виконання програми було б припинено.

В тих випадках, коли в програмі немає директиви **{$1}**, її функцію підтримує підрежим ***I/O Checking***. Він може бути **ON** або **OFF**, що відповідає директивам **{$I+}** та **{$I-}.**

Друкування результатів за допомогою принтера в цілому збігається з виведенням даних на екран, тільки в операторах **Write** або **Writeln** треба писати **ім'я** файла **Lst** і в операторі **Uses** записати ім'я модуля **Printer.**

Приклад:

```
Writeln(Lst,'X=',X:7);
```

Частину результатів у програмі можна виводити иа екран, а інші на принтер.

Якщо треба вивести тіж самі дані і на екран, і на принтер, то для таких операцій записують окремі оператори.

5.6 Структура програми. Розробка програм з масивами чисел

Структуру типової програми на мові **Turbo Pascal** можна подати таким чином:

```
Uses crt;
Const .....
Label .....
Type ......
   Var.....
Begin {початок операторів виконання}
……..
End {кінець програми}
```

В операторі **Uses** вказують ті модулі із системної бібліотеки, які потрібні для роботи програми.

Всі дані повинні бути описаними на початку програми. Деякі з приведених рядків в конкретній програмі можуть бути відсутніми.

Треба завжди пам'ятати, що загальний обсяг всіх статичних даних не повинен перебільшувати число 64Кб. При звичайному описуванні даних вони є статичними, пам'ять для них виділяється на етапі компіляції, на відміну від динамічних даних.

В розділі операторов виконання розміщують таку послідовність операторів, яка реалізує алгоритм розв'язування задачі. Немає обмежень на кількість рядків в програмі, але загальний її обсяг не повинен перебільшувати 64Кб.

Характерною ознакою програм є їх наочність та читабельність. Для досягнення цієї мети вживають деякі заходи. Один з них — коментар. Він може вміщувати будь-які літери з таблиці кодів ASCII, обмежуючись фігурними дужками або парами символів (* та *).

В деяких випадках (наприклад, при відлагоджуванні програми) виникає потреба вживати вкладені коментарі. Тоді використовують різні визначення коментарів, *наприклад:*

{це зовнішній (* а це внутрішній *) коментар}

Коментар може займати окремий рядок або бути його частиною. В останньому випадку його ставлять на місці проміжка між окремими конструкціями оператора або в його кінці.

Приклад:

For I:=1 to M {ця величина вводиться} **do ...**

Коментувати треба важкі для розуміння фрагменти, якісь особливості тощо. Не жалкуйте коментарів, але знайте міру. Не залишайте програму “голою” , але і не засмічуйте її.

Якщо є можливість, не вживайте складних конструкцій (наприклад, вкладених **If).** Не зловживайте оператором **Goto.**

Текст програми записуйте з використанням зміщення праворуч вкладених циклів, складених операторів **Begin ... End** тощо.

Взагалі є одна порада: пам'ятайте про людину, яка буде аналізувати вашу програму, і все буде гаразд.

Розглянемо приклади програм з масивами чисел.

Нехай треба переставити елементи масиву з цілих чисел у зворотному порядку.

```
Uses crt;
    Const n=10;
    Var x : array[1..n] of integer;
    i,j,m,a : Integer;
    Begin
Writeln('Введіть початкові дані');
For i:=1 to n do read(x[i]); readln;
For i:=1to n do write(x[i]:5); writeln;
m:=n div 2; {число перестановок}
 i:=n;
   For i:=1 to m do begin
            a:=x[i],
           x[i]:=x[j];
        x[j]: =a;
        i:=i-1;
            end;
        Writeln{ ' резул ьтат');
For i:=1 to n do write(x[i]:5); writeln;
        End.
```

Рисунок 5.2 - Програма перестановки чисел в масиві

Зверніть увагу на визначення числа перестановок (якщо кількість чисел в масиві буде непарною, то середнє число залишиться на місці). Крім того, складається враження, що оператор **i:=n** зайвий і можна замість **j** писати **n**. Але це неправильно: не можна змінювати іменовану константу **n!**

Приклад обробки матриці.

В квадратній матриці з правого боку від побічної діагоналі знайти найбільший елемент і відповідний номер рядка.

```
Uses crt;
Const M=5; N=5;
Type Matr =array[1..M,1..N] of integer;
Var a: Matr;
i,j,maxzn,nom : integer;
{знайти max елемент справа від побічної діагоналі)
 Begin
Writeln ('введіть початкові дані');
```

```
For i:=1 to M do begin
 For j:=1 toN do read(a[i,j]); readln;
              end;
maxzn:= – 32000; nom:=0;
For i:=1 to M do
For j:=N-i+1 to N do
 If maxzn<a[i,j] then begin
maxzn:=a[i,j];
nom:=i;
End;
{виведення результатів}
 Writeln('початкова матриця');
For i:=1 to M do begin
 For j:=1 to N do write(a[i,j]:5);
          Writeln;
          End;
  Writeln(‘max елемент =’,maxzn,' номер рядка=',nom);
  End.
```

Рисунок 5.3 - Програма обробки матриці

Важливе місце в цій програмі займає організація внутрішнього циклу. При **i=1** змінна **j** приймає значення **i=***5,* при **i=2** індекс j буде змінюватись від 4 до 5 і т. д.

При **i=5** параметр **j** змінюється від 1 до 5. Такі значення індексів забезпечують перегляд всіх елементів, які розміщені на побічній діагоналі та з правого боку від неї.

Нижченаведені варіанти внутрішніх циклів для аналізу інших частин матриці.

For j:=l to i do ... {зліва від головної діагоналі}

For j:=i to n do ... {справа від головної діагоналі}

For j:=1 to n+l-i do ... {зліва від побічної діагоналі}

5.7 Типові режими підготовки та виконання програм

На відміну від інших систем програмування Turbo Pascal має дуже зручне середовище, яке об'єднує у собі текстовий редактор, компілятор, укладач, налагоджувач та систему підказки. Користувач має можливість підготувати текст програми в редакторі, потім виконати його компіляцію

(переклад на мову машинних команд) та підготувати до виконання (цей етап називається компоновкою програми — при цьому до основної частини програми додаються різні стандартні підпрограми, виконується об'єднання окремих частин програми тощо). Після цього програму можна запустити на виконання. На різних етапах підготовки та виконання програми можуть виникати помилки, які зручно виділити та виправити, не виходячи з середовища.

Система Turbo Pascal має в своєму складі багато файлів, головні з них — **Turbo.exe, Turbo.tpl та Turbo.hlp.** Перший з них підтримує роботу в турбосередовищі, другий вміщує бібліотеку підпрограм, а файл **Turbo.hlp** забезпечує виведення на екран підказки для окремих режимів роботи, конструкцій програми тощо.

Щоб ввійти в середовище, треба виконати файл **Turbo.exe**. На екрані з'явиться головне меню системи.

File Edit Search Run Compile Debug Options Window Help

Fl Help F2 Save F3 Open Alt-F9 Compile F9 Make F10 Menu

Рисунок 5.4 – Головне меню системи

У верхньому рядку подані всі режими головного меню, в нижньому – рядок статусу, в якому перелічені імена функціональних клавіш, призначених для виконання деяких операцій в конкретній ситуації.

Щоб перейти в головне меню, можна натиснути клавішу **F10,** або виставити курсор маніпулятора “миша” на позначку **F10 Menu** та натиснути його ліву кнопку. Після такої операції смугою буде виділено один із режимів.

Вибрати та активізувати потрібний режим роботи можна з допомогою клавіш керування курсором і **Enter** або скориставшись маніпулятором.

Є ще третій спосіб вибору потрібного режиму: одночасно натиснути клавішу **ALT** та букву, яка виділена в назві режиму.

Кожен режим головного меню має один або декілька (вкладених одне в одне) підменю.

Типовий режим створення нового файла починається з входу в текстовий редактор: активізувати спочатку режим **File***,* а потім підрежим **New**. На екрані з'явиться пусте вікно редактора з умовною назвою файла **Noname.pas** та відповідним порядковим номером (в правому верхньому кутку). Курсор в цей час блимає в лівому верхньому кутку. Тепер можна набирати текст програми рядок за рядком. В кінці кожного рядка натискують клавішу **Enter.** Для введення великих букв натискують клавішу **Shift***,* щоб ввести букви кирилиці, підключають відповідний драйвер, наприклад, клавішами **Shift/ Shift** (одночасно натиснути ліву та праву **Shift*).***

В процесі підготовки тексту виникають помилки, їх можна виправляти за допомогою команд редактора. Щоб вилучити символ зліва від курсора, натискують клавішу **Backspace.** Клавіша **Del** вилучає символ в позиції курсора. Одночасне натискання клавіш **CTRL/Y** вилучає рядок з курсором, а клавіші **CTRL/N** вставляють пустий рядок.

Клавіша **Insert** включає або виключає режим вставки. Якщо режим включено **(Insert***),* то при введенні символу інша частина рядка (якщо вона є) зсувається праворуч. Якщо режим виключено (це значить, що діє режим перезапису — **Overwrite**), то під час введення символу він заміняє той символ, на який показує курсор — зсув не відбувається.

Після підготовки тексту треба натиснути клавішу **F2** (це відповідає підменю **Save** (зберегти) меню **File**). На екрані з'явиться вікно з пропозицією ввести ім'я файла, куди буде записано підготовлену в редакторі програму. Після введення імені, наприклад, **Peter**, натискують клавішу **Enter.** В результаті цих дій в поточному каталозі буде записано файл **Peter.pas**. Якщо створений файл треба записати в інше місце, вводять повний шлях та ім'я файла, *наприклад.* **A:\ Peter** — файл **Peter.pas** буде розміщено на гнучкому диску.

В звичайних умовах роботи зразу після створення файла виконують його компіляцію. Для цього достатньо натиснути клавіші **CTRL/F9 –** “гарячі клавіші” підменю **Run** меню **Run** (виконати). В процесі компіляції виявляється перша синтаксична помилка, на екрані з'явиться текст програми, курсор показує місце помилки, а у *верхній* частині екрана – коротке повідомлення про помилку, *наприклад:*

; **Expected (чекаємо ;).**

Після виправлення помилки знову натискують **F2** (щоб записати новий варіант), а потім **CTRL/F9** — виявляється наступна помилка і т.п.

Якщо всі помилки виправлені, то програму після компіляції буде

відкомпоновано (автоматично) і виконано. Такий режим називають “виконання програми з пам'яті”, розуміючи під цим те, що на диск не записується результат компоновки.

На етапі виконання програми теж можуть з'явитися помилки, наприклад, при спробі обчислення логарифму від'ємного числа, діленні на нуль тощо. В таких випадках робота припиняється, на екрані знову з'явиться програма, курсор буде показувати місце помилки.

Причини таких помилок в загальному випадку більш складні для розуміння, тому нерідко доводиться звертатися до послуг налагоджувача програм (режим **Debug**).

Якщо в програмі явних помилок не виявлено, вона буде виконана. На екрані на незначний час з'являться результати, які потім будуть закриті вікном редактора турбосередовища. Для їх спостереження треба натиснути **ALT/F5**. Після детального аналізу результатів роблять висновок про правильність програми. Якщо виникли сумніви у деяких результатах треба ретельно перевірити відповідні місця програми, для чого у пригоді стане налагоджувач.

Режим “виконання, програми з пам'яті” використовується в тих випадках, коли немає впевненості в її правильності або вона більше не буде виконуватись (наприклад, навчальна програма).

Якщо програма налагоджена і буде експлуатуватись, треба створити відповідний .exe-файл. Для цього замість **CTRL/F9** налагоджують .режим **Compile/Destination** так, щоб праворуч від слова **Destination** стояло слово **Disk.** Якщо там записано ***Memory,*** виставляють інверсну стрічку на цей рядок і натискують ***Enter*** — з'являється слово **Disk.** Тепер достатньо натиснути **ALT+F9** (режим **Compile/Compile***),* програма буде відкомпільована, в поточний каталог диску запишеться .exe-файл (його ім'я збігається з ім'ям відповідного **.pas**-файла).

Існує можливість таким чином сформований файл виконання записати до будь-якого місця файлової системи. Для цього попередньо треба налагодити режим **Options/Directories /Exe&Tpu directory**, виставити інверсну стрічку, натиснути ***Enter*** і набрати шлях до каталогу, куди записати створені **.exe** -файли, *наприклад,* **D:\TP6\GRUP1**.

Після створення **.exe** - файла можна вийти з середовища *(***ALT/X** або **File/Quit***),* знайти у відповідному каталозі .exe-файл і виконати його.

Наступний типовий режим — коригування існуючого **.pas** файла, *наприклад,* **Peter.pas**.

Спочатку його треба завантажити в текстовий редактор. Для цього у командному рядку слід набрати **Turbo Peter** *і* натиснути **Enter,** відбудеться вхід в турбосередовище і у вікні редактора з'явиться потрібний файл.

Якщо середовище вже завантажене і виникла потреба відкоригувати файл, треба натиснути **F3** (режим **File/Open***),* з'явиться діалогове вікно **Open a File***,* в якому перечислені існуючі **.pas**-файли поточного каталогу. Курсор миготить в рядку **Name***,* тут слід набрати ім'я файла і після **Enter** його буде завантажено в редактор (так можна завантажити файл з будь-якого каталогу, для цього достатньо вказати повний шлях і ім'я файла), *наприклад,* **A: Peter.pas.**

Можна не набирати ім'я файла, а клавішею ***Tab*** перевести курсор в список файлів, звичайним способом вибрати там потрібний файл та натиснути **Enter.**

Текстовий редактор версії Turbo Pascal 6.0 дозволяє обробляти файл довжиною до 1 Мбайт.

Тепер важливе зауваження: починаючи з версії 6.0 є можливість виконувати різні операції з декількома файлами. Кожен з них розміщується у своєму вікні редактора. В верхній частині вікна записано ім'я файла, справа від нього – : номер вікна.

При завантаженні в редактор (при відкритті) наступного файла він попадає у нове вікно.

Є декілька варіантів взаємного розміщення вікон, вони вибираються в режимі **Window** головного меню **(Tile, Cascade, Size/Move, Zoom** тощо).

Зручним є варіант **Window\Cascade***,* коли на екрані показані заголовки (верхній рядок) всіх вікон. Тільки одне з вікон активне (воно займає більшу частину екрана). Щоб змінити активність вікон натискують клавішу **ALT** та клавішу номера вікна. Це можна також зробити, якщо підвести курсор миші до видимої частини потрібного вікна і натиснути короткочасно на ліву кнопку маніпулятора.

Після того, як вікно з потрібним файлом активізовано, можна приступати до його коригування. Щоб закрити активне вікно достатньо натиснути **ALT/F3.**

Існує ще багато можливостей відносно керування вікнами: зміна розмірів, місця розміщення, створення скролінгу тексту тощо. Наприклад, щоб вікно закрити, треба виставити курсор миші на позначку [_] зліва від імені файла та натиснути ліву клавішу.

В деяких випадках виникає потреба в копіюванні частини тексту одного файла в інший. Для цього застосовують вікно проміжного буфера **(Clipboard***)* і команди роботи з блоками тексту (режим **Edit***)*.

Блок – це група рядків, які йдуть один за одним. Блок можна помітити: перевести курсор на початок групи рядків, натиснути одночасно **CTRL/K/B,** перевести курсор на кінець групи — **CTRL/K/K**. Блок буде виділено іншим кольором. Тепер з блоком можна виконувати різні операції".

Нехай в вікні 3 розміщується файл **Petrl.pas**, а в вікні 5 — **Petr2.pas**. Треба скопіювати блок з файла **Petrl.pas** до файла **Petr2.pas.**

Спочатку активізуємо вікно **S(ALT/3***),* виділемо блок, запишемо його до буфера, для чого натиснемо **CTRL/Ins.** Тепер клавішами **Alt/5** активізуємо вікно 5, висунемо курсор в потрібне місце файла і натиснемо ***Shift/Ins*** для копіювання блоку з буфера **Clipboard** в місце розміщення курсора.

Якщо є потреба, можна коригувати текст блоку в буфері. Для цього активізувати режим **Edit** головного меню, а потім підрежим **Show clipboard** (показати **clipboard**). На екрані з'явиться вікно з назвою **Clipboard** з текстом. Після коригування тексту можна натиснути **F3,** завантажити потрібний файл, виставити курсор в потрібне місце і натиснути ***Shift/Ins*** — текст з **clipboard** буде скопійовано саме в цей файл.

В підменю режиму **File** є команда **Print**. Вона забезпечує друкування тексту програми із активного вікна. Для цього треба підготувати принтер і активізувати вказану команду. Якщо виникає потреба в друкуванні частини тексту, її треба помітити як блок і натиснути одночасно **CTRL/K/P.**

Нарешті, програму можна віддрукувати звичайним способом — як текстовий файл. Для цього треба вийти з середовища, виставити інверсну стрічку на ім'я файла, натиснути ***F5,*** набрати **PRN** та ***Enter.***

Лабораторна робота №4

Робота в середовищі програмування Turbo Pascal.

Запуск програм на виконання

Мета роботи: ознайомитись з середовищем програмування Паскаль. Навчитись зчитувати, редагувати, записувати та виконувати програми.

Система програмування Turbo Pascal об'єднує в собі текстовий редактор, компілятор, укладач, налагоджувач та систему підказки.

Запуск програм на виконання

1. Ввійти в каталог системи програмування. (Наприклад: TP7.0);

2. Виконати командний файл turbo.exe.

3. Перейти в головне меню (F**10).** Щоб вийти з головного меню і повернутись в редактор текста — натисніть клавішу **Esc**.

Виконати команду означає: з допомогою клавіш керування курсором виставити курсор на потрібну команду і натиснути клавішу **Enter.** (Виконати команду можна ще іншими способами: з допомогою маніпулятора "миша" або натисканням комбінацій клавіш **Alt** + та буква, яка виділена в назві команди).

Кожна команда головного меню має один або декілька (вкладених одне в одне) підменю.

Створити новий файл

- Ввійти в головне меню.
- Виконати команду **File.**
- Рядок за рядком набирати текст програми. В кінці кожного рядка натискують клавішу **Enter.**
- Виконати програму. Натиснути комбінацію клавіш **CTRL+F9.** Можна використати ще 2-ий спосіб. Ввійти в головне меню і виконати команду **Run.**
- Виправити при необхідності помилки.
- Після виправлення помилок знову виконати програму.
- Переглянути отримані результати, натиснувши комбінацію клавіш **ALT + F5.**
- Записати текст програми в файл на диск. Ввійти в головне меню. Виконати команду **Save з** ім’ям файла (наприклад **Proba.pas**).
- Щоб прискорити процес записування програми в файл на диск можна натиснути клавішу **F2** і ввести ім’я файла. Натиснути **Enter.**

Порядок редагування існуючого pas-файла:

- Ввійти в головне меню.
- Вибрати і виконати команду **File**.

- Вибрати і виконати команду **Open** (альтернатива – натиснути клавішу **F3**).
- Натиснути клавішу **TAB,** вибрати з таблиці ім'я потрібного файла і натиснути **Enter**. Можна ще в рядку **Name** набрати ім'я необхідного файла і натиснути **Enter.**
- Внести необхідні доповнення до програми, виконати її і виправити помилки.
- Записати програму в файл на диск, використавши команду **Save**. Якщо необхідно записати програму в файл на диск під новим іменем, то замість команди **Save** портрібно вибрати команду **Save as.**

Створення виконуваного exe-файла

- Ввійти в головне меню, вибрати і виконати команду **Compile.**
- Вибрати і виконати команду **Destination – Memory,** яка після натискання на клавішу **Enter** зміниться на **Destination – Disk.**
- Натиснути комбінацію клавіш **ALT + F9.**

Щоб вийти з середовища Turbo Pascal необхідно натиснути комбінацію клавіш **ALT+X.**

Завдання для практичної роботи

1. Ввійти в середовище програмування.
2. Познайомитись з середовищем програмування.
3. Вивчити редактор тексту програмного середовища.
4. Зчитати з диска програму, яка записана в файлі proba.pas.
5. Виконати дану програму.
6. Перевірити отримані результати.
7. Записати дану програму в файл на диск під іншим іменем.
8. Створити виконуваний exe-файл.
9. Вийти з середовища програмування.
10. Виконати в системі DOS записаний exe-файл.
11. Оформити звіт з лабораторної роботи.

У звіті з лабораторної роботи описати призначення і порядок виконання всіх вказівок, які використовувалися в роботі.

Питання для самоконтролю

1. Для чого призначена мова програмування Паскаль ?

2. Що таке транслятор?

3. Склад та функції системи програмування.

4. Як ввійти в середовище програмування Turbo Pascal?

5. Як ввійти і вийти з головного меню середовища програмування Паскаль?

6. Що означає «виконати команду»?

7. Який порядок створення програми і запису її в файл?

8. Як відредагувати програму, яка записана в файл на диск?

9. Як вийти з середовища програмування?

Завдання для самостійного виконання

1. В текстовому редакторі середовища програмування Turbo Pascal ввести програму за індивідуальним завданням у вигляді:

```
Program PR1;
Const A= 4;
      B=7;
 Var x,y: integer;
begin
y:=(A+B)*x;
writeln('y=',y);
end.
```

2. Виконати програму.

3. Проглянути результати.

4. Записати програму в файл на диск під іменем Fpr1.

5. Вийти з середовища програмування Turbo Pascal.

6. Переконатись у наявності на диску файла Fpr1.pas.

Лабораторна робота №5

Тема: Проектування та реалізація алгоритмів розгалуження

1. Нарисувати блок-схему та скласти програму згідно з номером завдання на мові Pascal, яка забезпечує розрахунки з використанням алгоритмів розгалуження.

2. Ввести програму в ЕОМ.

3. Виконати розрахунки та отримати результат.

Питання для самоконтролю

1. Яку структуру має програма, написана мовою програмування Паскаль ?

2. Що таке вираз?

3. Якими правилами визначається послідовність виконання операцій у виразах ?

4. Які дії виконує оператор присвоєння в середовище програмування Turbo Pascal?

5. У чому полягає відмінність між процедурами *read* та *readln* ?

6. Які аргументи не можна використовувати у викликах процедур читання?

7. Що таке сумісність типів?

8. Що визначає тип даних? Охарактеризуйте порядкові типи.

9. Що являють собою типізовані константи?

10. Який тип має результат операції відношення?

Лабораторна робота №6

Тема: Проектування та реалізація циклічних алгоритмів.

1. Нарисувати блок-схему та скласти програму згідно з номером завдання на мові Pascal, яка забезпечує циклічні розрахунки та виведення результату у вигляді таблиці.

2. Ввести програму в ЕОМ.

3. Виконати розрахунки та отримати результати.

Питання для самоконтролю

1. У чому полягає відмінність між циклами з передумовою та циклами з постумовою ?

2. Якому типу даних може належати лічильник у циклі *for*?

3. Яке значення має лічильник після завершення циклу *for*?

4. Що може спричинити “зациклення” програми?

5. За яких умов цикли *while* та *for* не виконуються жодного разу?

6. У чому полігає відмінність між такими операторами циклів, як *for ...to ...do* та *for ...downto ...do*?

7. Яка структура працює ефективніше: вкладений оператор *if...then ... else* чи серія операторів *if ...then* ? Відповідь обгрунтуйте?

8. Чи можна перервати роботу циклу, не використовуючи процедуру *break* ?

6 ТЕКСТОВИЙ РЕДАКТОР MICROSOFT WORD

Microsoft Word 2000 – текстовий редактор, програма для створення й редагування текстових документів. Подання **WYSIWIG** (від англійського “What You See Is What You Get”) дозволяє переглядати на дисплеї готовий до друку документ без необхідності витрачати папір на пробний друк. Відформатовані символи відображаються на екрані так, як вони будуть виглядати при друку.

Для запуску Microsoft Word необхідно виконати подвійне натискання на його ярлику .

Робота з вікнами

Багатовіконна організація Microsoft Word дозволяє одночасно працювати з декількома документами, кожний з яких розташований у своєму вікні. При введенні й редагуванні тексту користувач працює з активним документом в активному вікні. Для переходу до іншого вікна документа необхідно натиснути на його імені на панелі задач або в меню **Окно,** що містить перелік усіх відкритих документів.

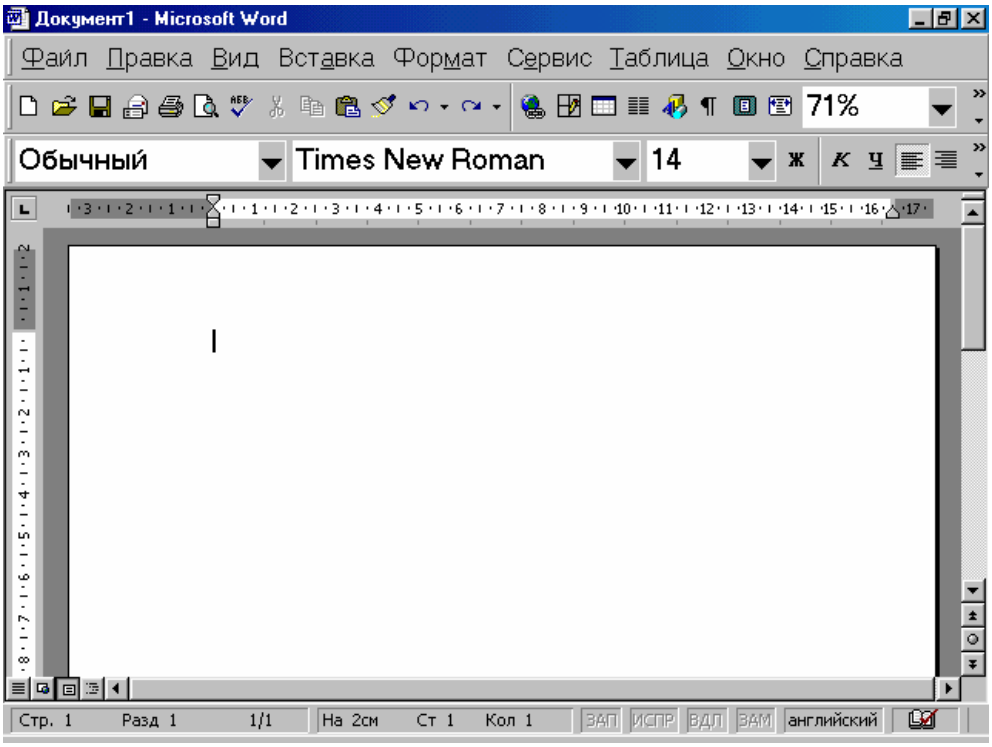


Рисунок 6.1 - Вікно програми Microsoft Word

Курсор введення

Існують два поняття – курсор введення і покажчик миші. **Курсор введення** являє собою миготливу вертикальну смужку **|**. Він вказує місце,

в яке буде вводитися текст. Для його переміщення використовуються клавіші керування курсором або миша. Для переміщення курсора введення за допомогою миші слід установити покажчик миші в потрібну позицію й натиснути клавішею миші.

Таблиця 6.1 - Переміщення курсора введення за допомогою клавіатури

Клавіша	Переміщення
↑	На один рядок угору
↓	На один рядок униз
←	На одну позицію ліворуч
→	На одну позицію праворуч
Ctrl+↑	На один абзац угору
Ctrl+↓	На один абзац униз
Ctrl+←	На одне слово ліворуч
Ctrl+→	На одне слово праворуч
PgUp	На один екран угору
PgDn	На один екран униз
End	У кінець рядку
Home	У початок рядку
Ctrl+**Home**	У початок документа
Ctrl+**End**	У кінець документа

Меню

Під заголовком вікна знаходиться рядок меню, через який можна викликати будь-яку команду Microsoft Word. Для відкриття меню необхідно клацнути мишею на його імені. Після чого з'являться ті команди цього меню, які вживаються найчастіше (рис. 6.2).

Якщо клацнути на кнопці  , то у нижній частині меню то з'являться усі команди цього меню (рис. 6.3).

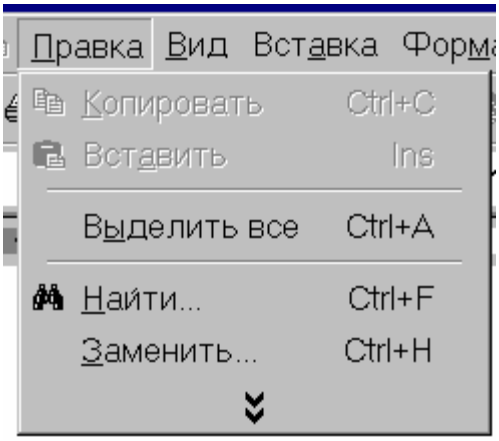


Рисунок 6.2

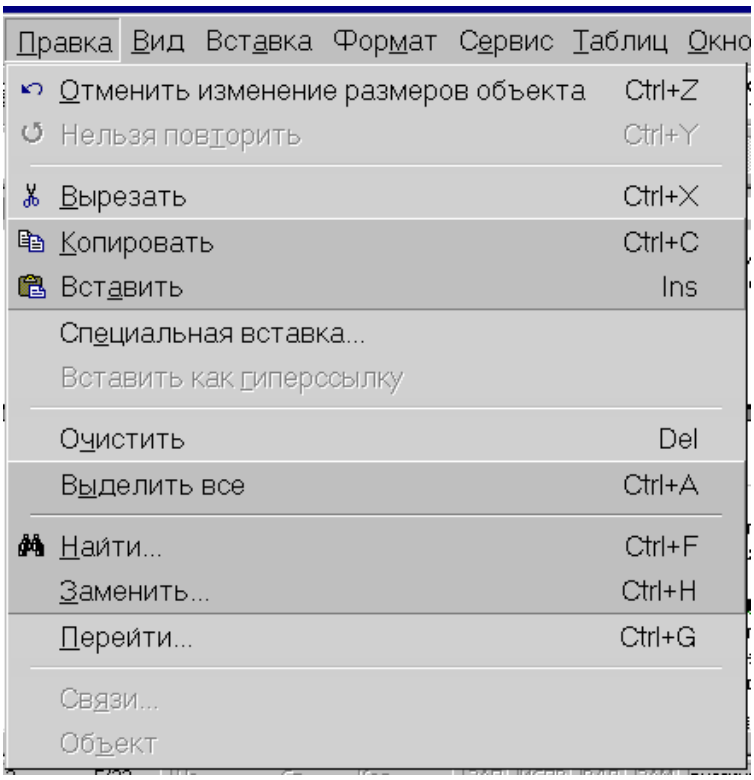


Рисунок 6.3

Панелі інструментів

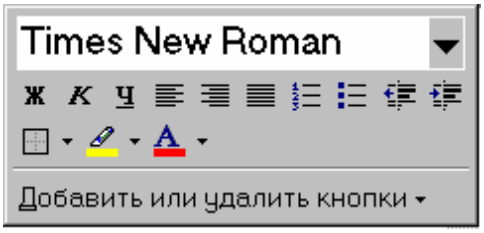


Рисунок 6.4

Під рядком меню розташовані панелі інструментів, що складаються з кнопок з рисунками (рис. 6.4). Кожній кнопці відповідає команда, а рисунок на цій кнопці передає значення команди. Більшість кнопок дублюють найбільш часто вживані команди, доступні в меню. Для виклику команди, зв'язаної з кнопкою, необхідно натиснути мишею на цій кнопці. Якщо навести покажчик миші на кнопку й трохи почекати, поруч з'явиться рамка з назвою команди.

Зазвичай, під рядком меню знаходяться дві панелі інструментів – **Стандартная** і **Форматирование**. Щоб вивести або забрати панель з екрану слід вибрати в меню **Вид** пункт **Панели инструментов,** а потім натиснути на ім'я потрібної панелі**.** Якщо панель присутня на екрані, то навпроти її імені буде стояти позначка ✔ .

Якщо для виведення усіх кнопок на панелі недостатньо місця, то виводяться кнопки, які були вжиті останніми. Якщо натиснути на кнопку » у кінці панелі, то з'являться інші кнопки (рис. 6.5). При натисканні на кнопку **Добавить или удалить кнопки** з'явиться меню (рис. 6.6), в якому можна вивести або забрати кнопку з панелі.

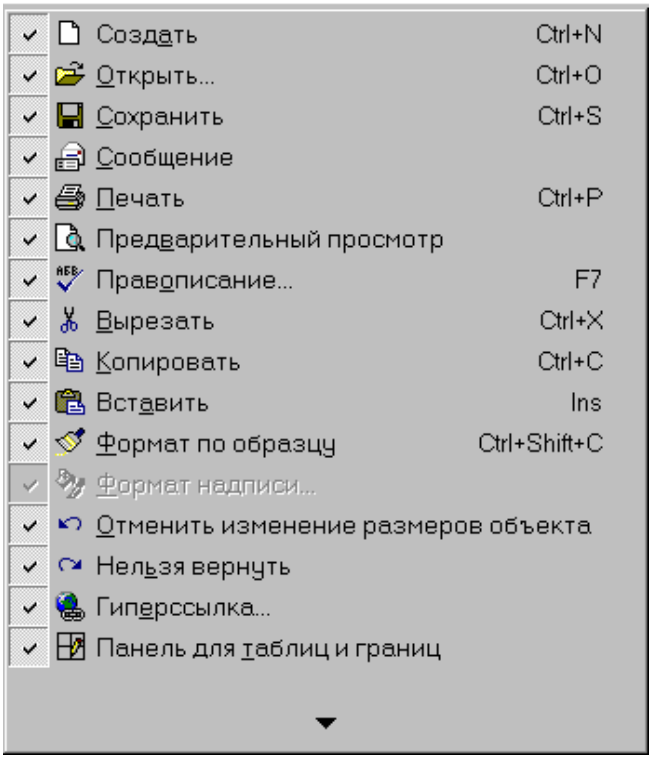


Рисунок 6.5

Також для зміни складу панелі інструментів можна у меню **Сервис** вибрати пункт **Настройка.** У діалоговому вікні необхідно вибрати

вкладку **Команды**. У переліку **Категории** необхідно вибрати групу кнопок, після чого у переліку **Команды** з'являються кнопки цієї групи. Щоб додати кнопку до панелі інструментів слід пересунути її з діалогового вікна в потрібну позицію меню. Процес встановлення кнопки завершується натисканням кнопки **Закрыть.** Для видалення кнопки з панелі інструментів необхідно пересунути її в діалогове вікно **Настройка**.

Керувати панелями інструментів зручно за допомогою контекстного меню (рис. 6.6), яке викликається натисканням правої клавіші миші на будь-яку кнопку.

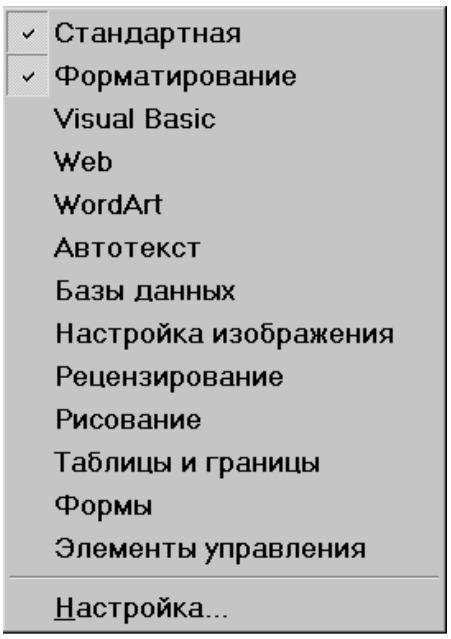


Рисунок 6.6

Координатні лінійки

Горизонтальна координатна лінійка розташована над робочим полем, **вертикальна** – ліворуч від робочого поля. З їхньою допомогою можна установлювати поля сторінок, абзацні відступи, змінювати ширину шпальт і установлювати позиції табуляції. За замовчуванням координатна лінійка градуйована в сантиметрах. Виводяться/ забираються лінійки за допомогою команди **Линейка** в меню **Вид**.

Рядок стану

Рядок стану (рис. 6.7) розташований у нижній частині вікна Microsoft Word. У ньому виводяться різні повідомлення й довідкова інформація.

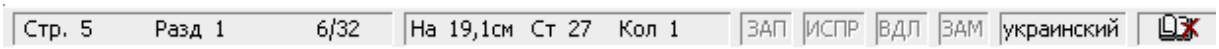


Рисунок 6.7 - Рядок стану

Таблиця 6.2 - Інформація в рядку стану

Індикатор	Значення
Стр 1	Порядковий номер видимої у вікні сторінки документа
Разд 1	Номер розділу, в якому знаходиться видима сторінка
3/38	Номер видимої сторінки/ загальне число сторінок у документі
На 16,9см	Відстань від курсора введення до верхнього краю сторінки
Ст 28	Номер рядка, в якому знаходиться курсор
Кол 36	Номер позиції курсора в рядку
ЗАП	Індикатор режиму запису макрокоманди
ИСПР	Індикатор режиму редакторської правки
ВДЛ	Індикатор режиму розширення маркірування
ЗАМ	Індикатор режиму заміни
украинский	Індикатор мови

Режими відображення документа

Редактор Microsoft Word дозволяє переглядати документ у різних режимах:

- Обычный – найбільш зручний для виконання більшості операцій;

- Web-документ - відображає документ у вигляді Web-сторінки;

- Разметка страниц – відображає документ у точній відповідності з тим, як він буде виведений на друк; у цьому режимі зручно працювати з колонтитулами, фреймами й багатоколонною верстkою документа; тільки у цьому режимі відображається вертикальна координатна лінійка;

- Структура – призначений для роботи зі структурою документа, дозволяє показувати й приховувати текст і заголовки визначеної глибини укладення, створювати та працювати з піддокументами.

Перехід між режимами здійснюється за допомогою відповідних команд меню **Вид** або кнопок, розташованих ліворуч від горизонтальної смуги прокручування (рис. 6.8).

Рисунок 6.8

Смуги прокручування (скролінг)

Смуги прокручування (вертикальна і горизонтальна) розташовані в правій і нижній частині вікна програми. Вони призначені для переміщення тексту у вікні редактора по вертикалі та по горизонталі. Переміщення по документу з використанням лінійок прокручування здійснюється за допомогою миші.

 Переміщення вікна на один рядок угору

 Переміщення вікна на один рядок униз

 Переміщення вікна ліворуч

 Переміщення вікна праворуч

 Переміщення вікна в напрямку зсуву прямокутника

 Переміщення вікна на один об’єкт (сторінку, рисунок, таблицю та ін.) угору

 Переміщення вікна на один об’єкт униз

 Вибір об’єкта пересування

Вихід із Microsoft Word

Для завершення роботи з Microsoft Word необхідно закрити вікно програми (кнопка закриття вікна ✕, або комбінація клавіш **Alt + F4**).

Операції з документами

Створення нового документа

Для створення нового документа в MS Word (див.ярлик на рис. 6.9) слід у меню **Файл** вибрати команду **Создать.** У діалоговому вікні, що відкрилося, (рис. 6.10) вибрати спочатку вкладку, а потім шаблон, на основі якого буде створено документ; після чого натиснути на кнопку **OK**. Шаблони документів Microsoft Word мають розширення **dot** і значки на рис. 6.11.

Рисунок 6.9

Звичайні документи створюються на основі шаблону **Новый документ**. Для створення документа на основі шаблону **Новый документ** можна натиснути на кнопку .

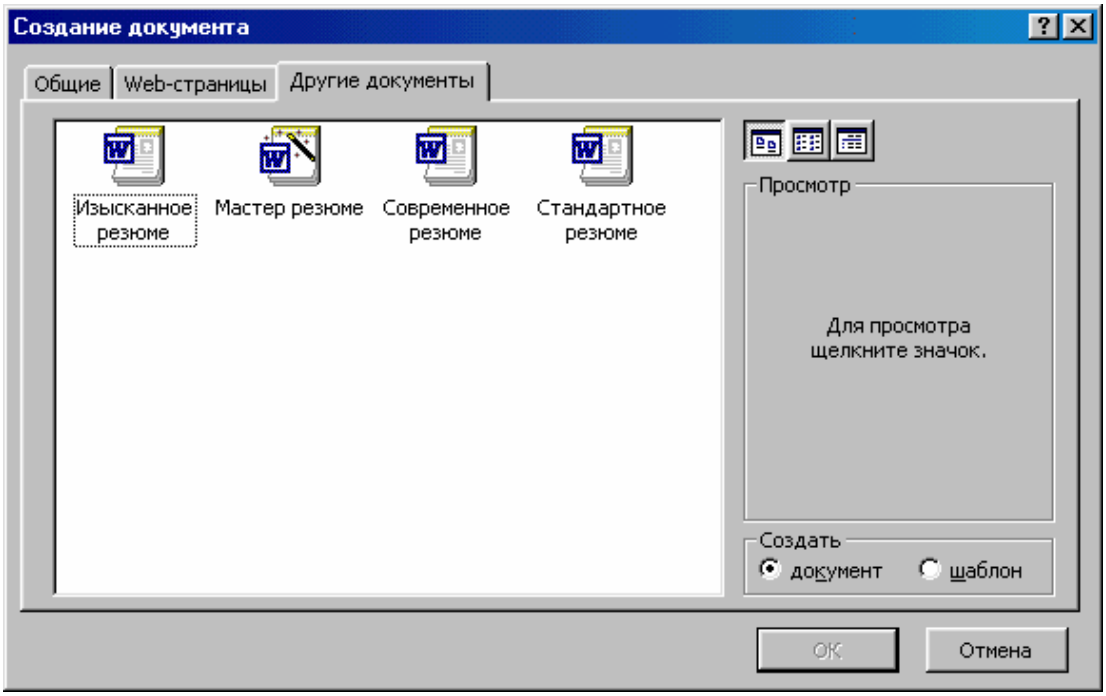


Рисунок 6.10

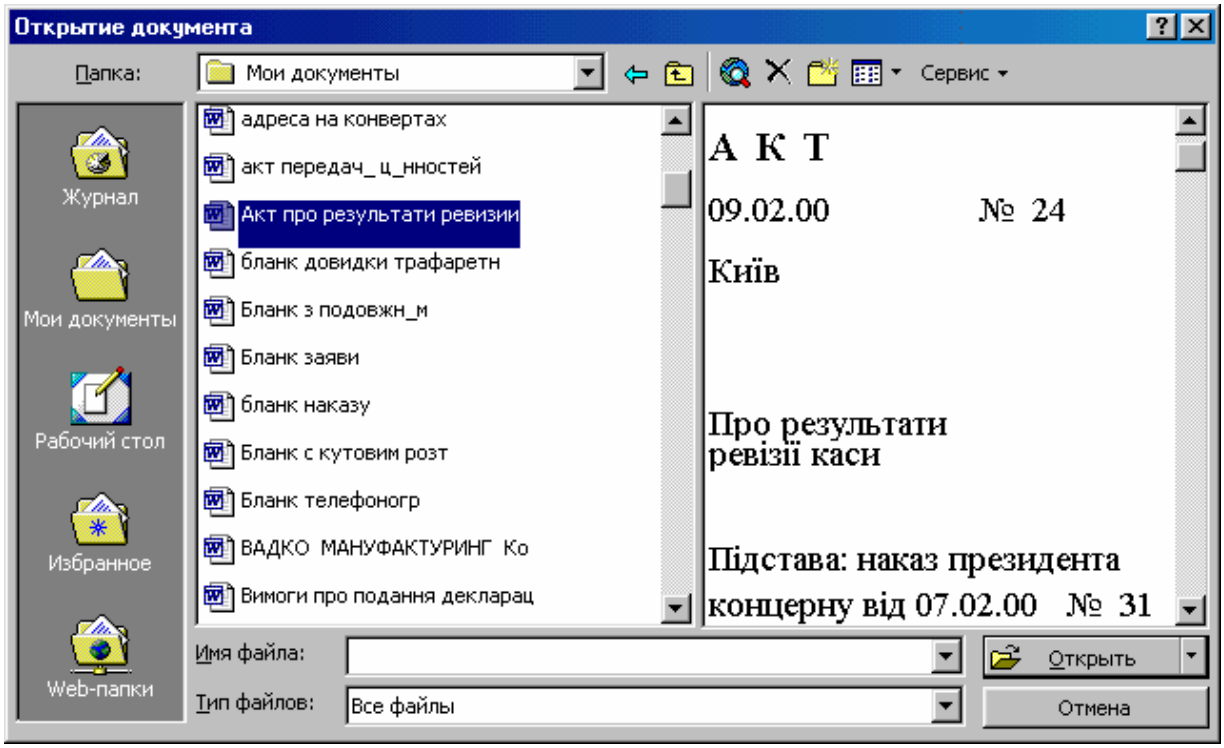


Рисунок 6.11

Відкриття документа

Для відкриття існуючого документа необхідно в меню **Файл** вибрати команду **Открыть** або натиснути на кнопку  , після чого відкривається діалогове вікно **Открытие документа** (рис. 6.11). У прихованому переліку **Папка** слід вибрати диск, на якому знаходиться потрібний документ. У переліку, що розташований нижче вибрати (подвійним натисканням) папку з документом і сам документ. Документи Microsoft Word мають розширення **doc** і значки на рис. 6.12.

Рисунок 6.12

У верхньому рядку вікна знаходяться 4 кнопки, які дозволяють подати вміст відкритої папки у 4-х виглядах:

 – у вигляді переліку файлів і папок;

 – у вигляді таблиці з інформацією про файли та папки;

 – праворуч буде подано властивості файла, на який наведено курсор;

 – праворуч буде подано фрагмент файла, на який наведено курсор.

За замовчуванням у переліку виводяться тільки файли з документами Microsoft Word**.** Для виведення інших типів файлів або усіх файлів необхідно вибрати відповідний тип у полі прихованого переліку **Тип файлов.**

Збереження документа

Для збереження документа необхідно викликати команду **Сохранить** меню **Файл** або натиснути на кнопку  .

При першому збереженні з'являється діалогове вікно **Сохранение документа** (рис. 6.13). У прихованому переліку **Папка** слід вибрати диск, у переліку, що розташований нижче, папку, в якій необхідно зберегти документ. У полі прихованого переліку **Тип файла** – формат, в якому буде збережено документ. У полі **Имя файла** – ввести ім'я файла документа й натиснути кнопку **Сохранить**.

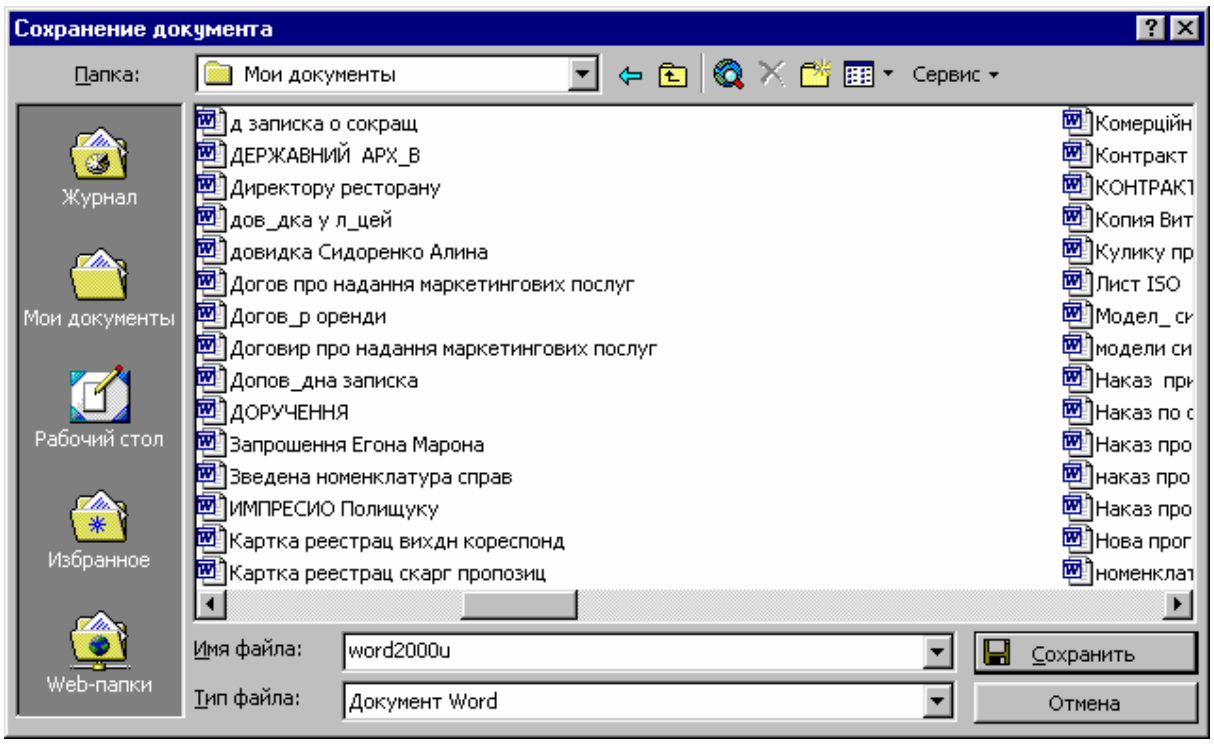


Рисунок 6.13

При повторному збереженні діалогове вікно **Сохранение документа** не виводиться, документ автоматично зберігається в тому ж файлі. Щоб зберегти документ під іншим ім'ям або в іншій папці, слід у меню **Файл** вибрати команду **Сохранить как,** після чого з'являється вікно **Сохранение документа.**

Закриття документа

Для закриття документа необхідно вибрати в меню **Файл** команду **Закрыть** або натиснути на кнопку ✕ вікна документа.

Робота з текстом

Введення тексту

Курсор вказує місце, в який буде вводитися текст. Після досягнення краю сторінки курсор автоматично переходить на початок наступного рядка. Для переходу в початок наступного абзацу слід натиснути **Enter**.
Існує два режими введення тексту – встановлення й заміни. **У режимі встановлення** при введенні нового тексту, текст, що міститься в документі, посувається праворуч від місця введення. **У режимі заміни** старий текст замінюється новим. Перемикання між режимами

здійснюється клавішею **Insert** або подвійним натисканням на індикаторі **ЗАМ** у рядку стану.

Виділення фрагмента тексту

Перед тим, як виконати будь-яку операцію над фрагментом тексту, його необхідно виділити одним із таких способів:

- установити покажчик миші в ліве поле (він перетвориться в стрілку, спрямовану праворуч), при натисканні лівої клавіші миші виділиться один рядок, при подвійному натисканні – абзац, при потрійному – увесь документ;
- установити покажчик миші в ліве поле навпроти першого рядка фрагмента, натиснути ліву клавішу миші і, не відпускаючи її, розширити виділення на весь фрагмент;
- установити покажчик миші на початок фрагмента, натиснути ліву клавішу миші та, не відпускаючи її, розширити виділення на весь фрагмент;
- для виділення одного слова досить двічі натиснути на ньому;
- для виділення одного абзацу можна зробити в ньому потрійне натискання;
- для виділення одного речення слід натиснути клавішу **Ctrl** і натиснути клавішу миші на будь-яке слово;
- для виділення всього тексту слід натиснути клавішу **Ctrl** і натиснути клавішу миші в лівому полі;
- щоб виділити фрагмент тексту за допомогою клавіатури, необхідно встановити курсор введення в початок фрагмента та, натиснувши клавішу **Shift**, розширити виділення на весь фрагмент.

Зняти виділення можна натисканням клавіші миші в будь-якому місці тексту. При виділенні нового фрагмента попереднє виділення знімається.

Редагування тексту

Символ праворуч від курсора видаляється клавішею **Delete**, а символ ліворуч від курсора – клавішею **Backspace.** Для видалення фрагмента тексту слід виділити його й натиснути клавішу **Delete**. Якщо виділити фрагмент тексту й набрати на клавіатурі новий текст, він вставиться замість виділеного фрагмента.

Щоб розділити абзац на два, необхідно встановити курсор у передбачуваний кінець першого абзацу й натиснути клавішу **Enter**.

З'єднати два абзаци в один можна двома способами:

- установити курсор за останнім символом першого абзацу й натиснути **Delete**;
- установити курсор перед першим символом другого абзацу й натиснути **Backspace**.

При натисканні клавіші **Enter** у поточну позицію курсора вставляється невидимий символ ¶. Для перегляду невидимих символів (спецсимволів, недрукованих символів) слід натиснути на кнопку ¶.

Скасування операцій над текстом

Для скасування останньої операції редагування необхідно в меню **Правка** вибрати команду **Отменить …** або натиснути на кнопку . Якщо натиснути на стрілці поруч із цією кнопкою, то розкриється перелік операцій, виконаних у поточному сеансі. Натиснувши на імені однієї операції, можна скасувати її та усі операції виконані після неї.

Щоб повернути останню скасовану операцію, слід у меню **Правка** вибрати команду **Повторить …** або натиснути на кнопку . Для перегляду переліку скасованих операцій слід натиснути на стрілці поруч із цією кнопкою.

Копіювання тексту

Для копіювання фрагмента тексту необхідно:

- виділити фрагмент тексту;
- натиснути кнопку або вибрати в меню **Правка** команду **Копировать**;
- установити курсор на те місце, куди слід вставити фрагмент;
- натиснути на кнопку або вибрати в меню **Правка** команду **Вставить**.

У процесі цієї операції копія виділеного фрагмента тексту розташовується в буфері проміжного збереження **Clipboard**, а потім потрапляє в документ. Вставляти фрагмент із буфера можна скільки завгодно разів, але після копіювання в буфер нового фрагмента тексту, попередній фрагмент видаляється.

Переміщення тексту

Для переміщення фрагмента тексту необхідно:

- виділити фрагмент тексту;
- натиснути кнопку або вибрати в меню **Правка** команду **Вырезать**;
- установити курсор на те місце, куди слід вставити фрагмент;

- натиснути кнопку  або вибрати в меню **Правка** команду **Вставить**.

Буфер обміну

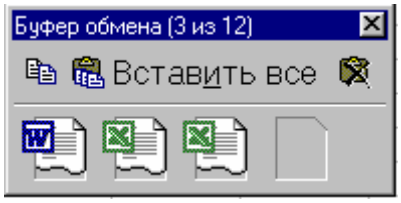


Рисунок 6.14

У Microsoft Word 2000 існує **буфер обміну** на 12 комірок, за допомогою якого можна копіювати фрагменти таблиці не тільки у межах Word але й в інші додатки, наприклад, у Microsoft Excel. Для виведення панелі буферу обміну (рис. 6.14) необхідно в меню **Вид** вибрати – **Панели инструментов**, потім – **Буфер обмена**. Для копіювання фрагмента у буфер його необхідно виділити та клацнути на кнопку  . Для вставлення фрагмента з буферу (у позицію курсора) необхідно клацнути на значок фрагмента. Наприклад, якщо фрагмент скопійовано з Microsoft Word, то він буде мати значок  . Для вставлення усіх фрагментів із буферу одночасно використовується кнопка  . Для очищення буферу слід клацнути на кнопку  . При копіюванні двох фрагментів підряд панель **Буфер обмена** з'являється автоматично.

Вставлення символу

Для вставлення в текст символу, відсутнього на клавіатурі, необхідно:

- установити курсор у позицію, куди слід вставити символ;
- у меню **Вкладка** вибрати команду **Символ**;
- у діалоговому вікні, що відкрилося, (рис. 6.15) вибрати вкладку **Символы;**
- у полі переліку **Шрифт** вибрати необхідний шрифт;
- натиснути мишею потрібний символ;
- натиснути на кнопку **Вставить**;
- для завершення роботи з вікном **Символ –** натиснути на кнопку **Закрыть.**

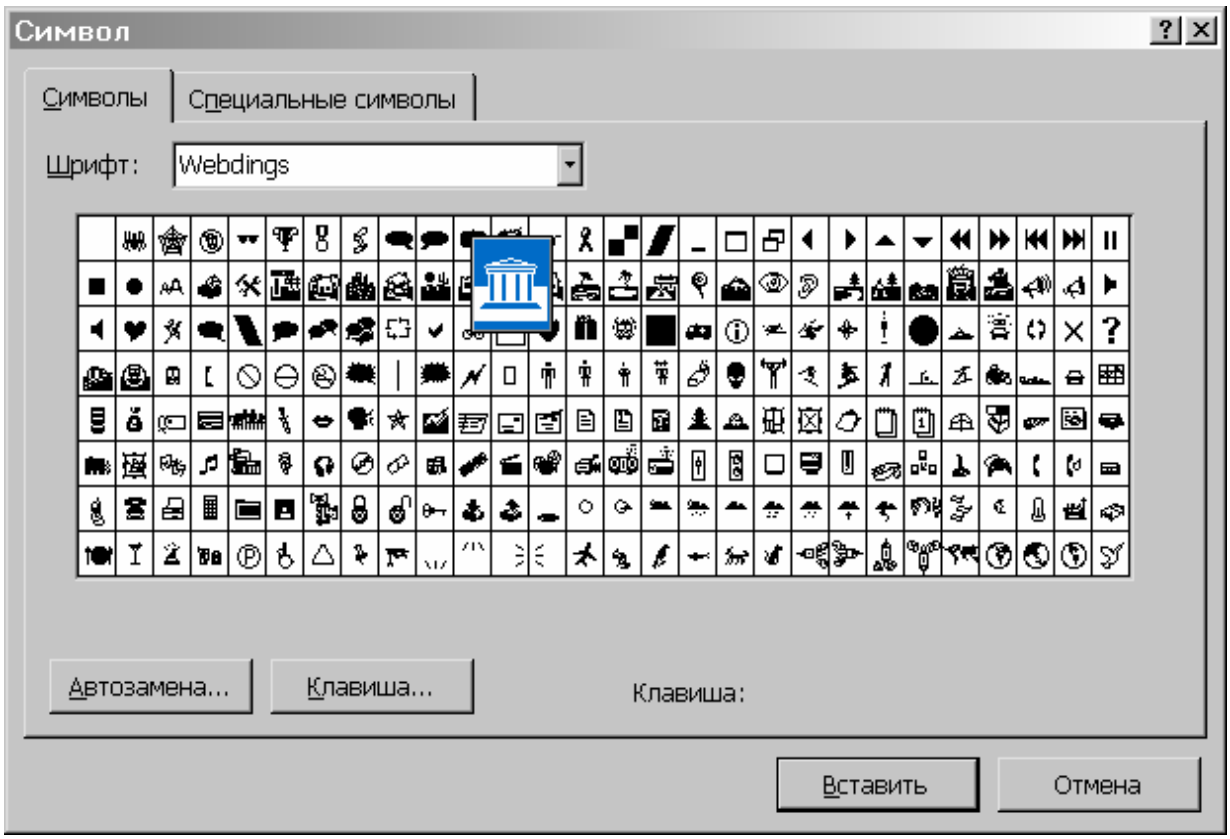


Рисунок 6.15

Пошук і заміна тексту

Для пошуку фрагмента тексту використовується команда **Найти** в меню **Правка.** У діалоговому вікні **Найти и заменить** (рис. 6.16) у полі **Найти** слід ввести фрагмент тексту, який потрібно знайти та натиснути кнопку **Найти далее.** При необхідності можна натиснути на кнопку **Больше** та ввести додаткові умови пошуку.

У полі переліку **Направление** вибирається напрямок пошуку:

- **Везде** – шукати у всьому документі;
- **Вперед –** шукати в тексті над курсором;
- **Назад –** шукати в тексті після курсора.

Встановити потрібні прапорці режимів пошуку:

- **Учитывать регистр –** при пошуку розрізняти великі і малі літери;
- **Только слово целиком** – пошук тільки тих слів, що цілком збігаються з зазначеним;
- **Подстановочные знаки** – використовуються символи шаблону, які вибираються після натискання на кнопку **Специальный**.

Щоб знайти наступне слово за заданими умовами, необхідно знову натиснути на кнопку **Найти далее.**

Для заміни одного фрагмента тексту іншим можна вибрати вставку **Заменить** діалогового вікна **Найти и заменить** або вибрати команду в меню **Правка.** У вставці **Заменить** слід ввести умови пошуку й заміни:

- у полі **Найти** – ввести фрагмент тексту, що необхідно замінити;
- у полі **Заменить на –** ввести фрагмент тексту для заміни;
- натиснути кнопку **Найти далее;**
- для заміни знайденого слова – натиснути кнопку **Заменить;**
- для заміни усіх фрагментів, що задовольняють умови, натиснути кнопку **Заменить все.**

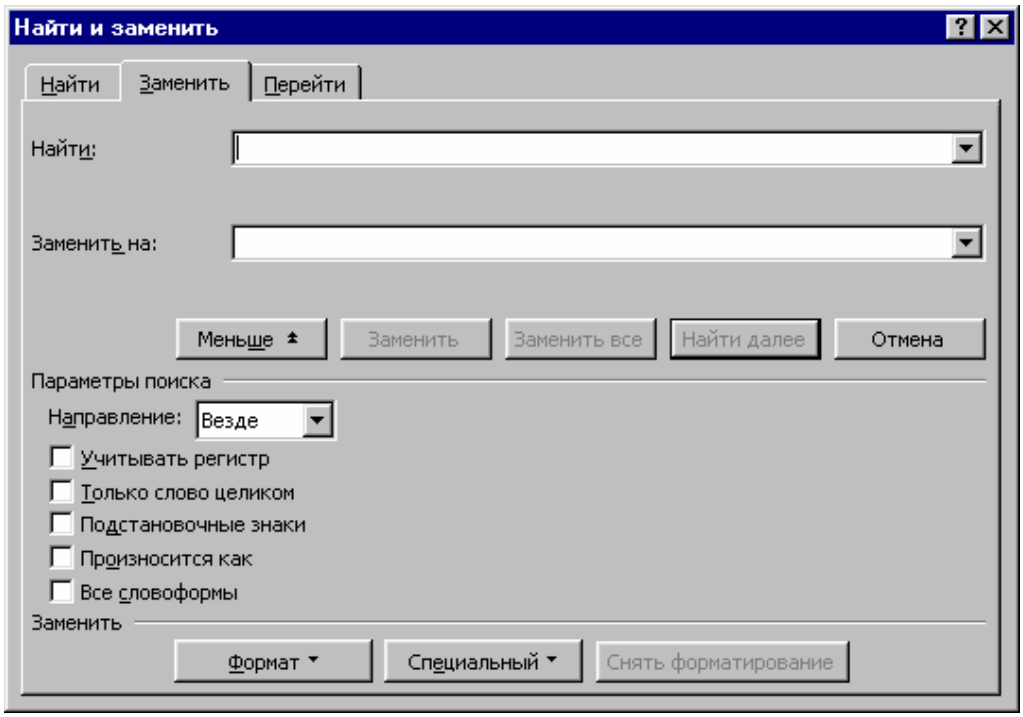


Рисунок 6.16

Контекстне меню

Для виклику контекстного меню (рис. 6.17) слід натиснути правою клавішею миші на оброблюваному об'єкті. Контекстне меню з'являється біля покажчика миші; воно містить команди для обробки виділеного елементу.

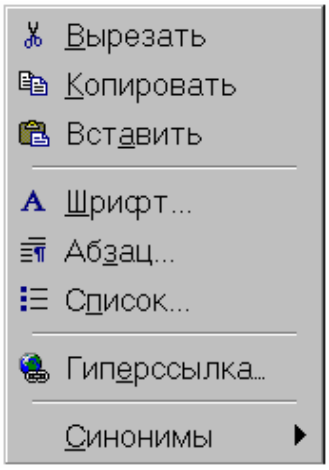


Рисунок 6.17

Форматування тексту

Форматування тексту – процес встановлення параметрів фрагмента тексту, що визначають зовнішній вигляд тексту в цьому фрагменті. Перед зміною параметрів фрагмента тексту його слід виділити. Якщо фрагмент тексту не буде виділений, то змінюватися будуть поточні параметри (параметри тексту, що будуть вводитися з поточної позиції).

Зміна параметрів шрифту

Для зміни параметрів символів використовується команда **Шрифт** в меню **Формат,** що викликає діалогове вікно **Шрифт** (рис. 6.18). Вкладка **Шрифт** використовується для встановлення параметрів шрифту.

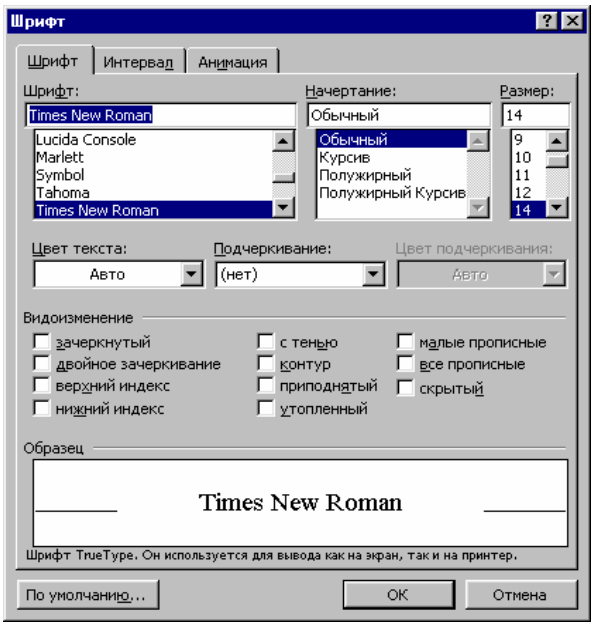


Рисунок 6.18

У полі **Шрифт** вибирається тип шрифту (шрифти типу **TrueType** виглядають однаково на екрані і віддруковані, поруч з їхнім ім'ям установлені спеціальні позначки).

У полі **Начертание** вибирається написання шрифту:

- Обычный – звичайне написання;
- *курсив* – курсивне написання;
- **полужирный** – жирне написання;
- ***полужирный курсив*** – жирне курсивне написання.

У полі **Размер** – розмір шрифту у пунктах (1 пункт = 0,375мм).

У полі **Подчеркивание** – тип лінії підкреслення.

У полі **Цвет** – колір символів.

У рамці **Видоизменение** можна установити прапорці:

- **зачеркнутый** – закреслення тексту одинарною лінією;
- **двойное зачеркивание** – закреслення тексту подвійною лінією;
- **верхний индекс** – розмір символів зменшується, текст розташовується вище;
- **нижний индекс** – розмір символів зменшується, текст розташовується нижче;
- **с тенью** – поруч із символами з’являється тінь;
- **контур** – показується лише контур символів;
- **приподнятый** – символи зображуються піднесеними над поверхнею аркушу;
- **утопленный** – символи зображуються утопленими в поверхню аркушу;
- **скрытый** – робить текст прихованим;

У полі **Образец** показаний фрагмент тексту з обраними параметрами.

Встановити параметри шрифту можна також за допомогою піктографічного меню (рис. 6.19):

1 – стиль форматування;

2 – тип шрифту;

3 – розмір шрифту;

4 – жирне зображення;

5 – курсивне зображення;

6 – підкреслення одинарною лінією.

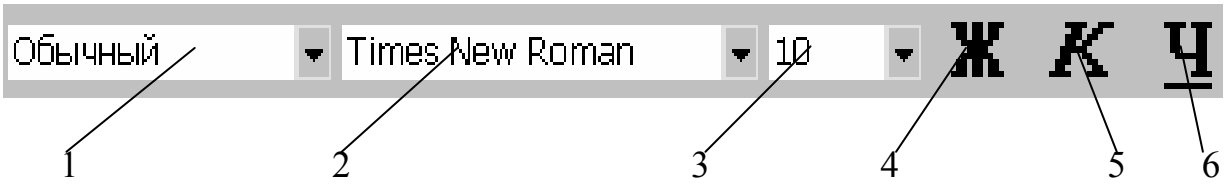


Рисунок 6.19

Зміна інтервалу й положення символів

Для зміни інтервалу й положення символів використовується Вкладка **Интервал** діалогового вікна **Шрифт.**

У полі прихованого переліку **Масштаб** вибирається ступінь розтягування або стиснення символів.

У полі **Интервал** установлюється міжсимвольний інтервал:

- Обычный – звичайний інтервал;

- Разреженный – відстань між символами збільшується до значення, зазначеного в полі **на**;

- Уплотненный – відстань між символами зменшується до значення, зазначеного в полі **на.**

У полі **Смещение** установлюється вертикальне положення символів:

- Нет – звичайне положення;

- Вверх – символи розташовуються вище базової лінії на розмір, зазначений у полі **на;**

- Вниз – символи розташовуються нижче базової лінії на розмір, зазначений у полі **на.**

Зміна регістру символів

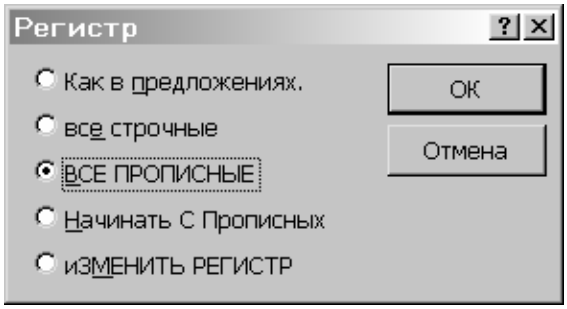


Рисунок 6.20

Для зміни регістру символів у вже набраному тексті необхідно виділити фрагмент тексту і у меню **Формат** вибрати команду **Регистр.** У діалоговому вікні, що з'явилося, (рис. 6.20) слід вибрати з перемикачів:

- Как в предложениях – збільшити першу літеру першого слова речення;

- все строчные – установити усі літери фрагмента в нижній регістр;

- ВСЕ ПРОПИСНЫЕ – установити усі літери фрагмента у верхній регістр;

- Начинать С Прописных – установити перші літери кожного слова у верхній регістр;

- иЗМЕНИТЬ РЕГИСТР – замінити літери верхнього регістру літерами нижнього регістру і навпаки.

Форматування абзаців

Для встановлення параметрів абзацу використовується команда **Абзац** із меню **Формат.** Після вибору цієї команди з'являється діалогове вікно **Абзац** (рис. 6.21). Для встановлення абзацних відступів та інтервалів необхідно вибрати вкладку **Отступы и интервалы.**

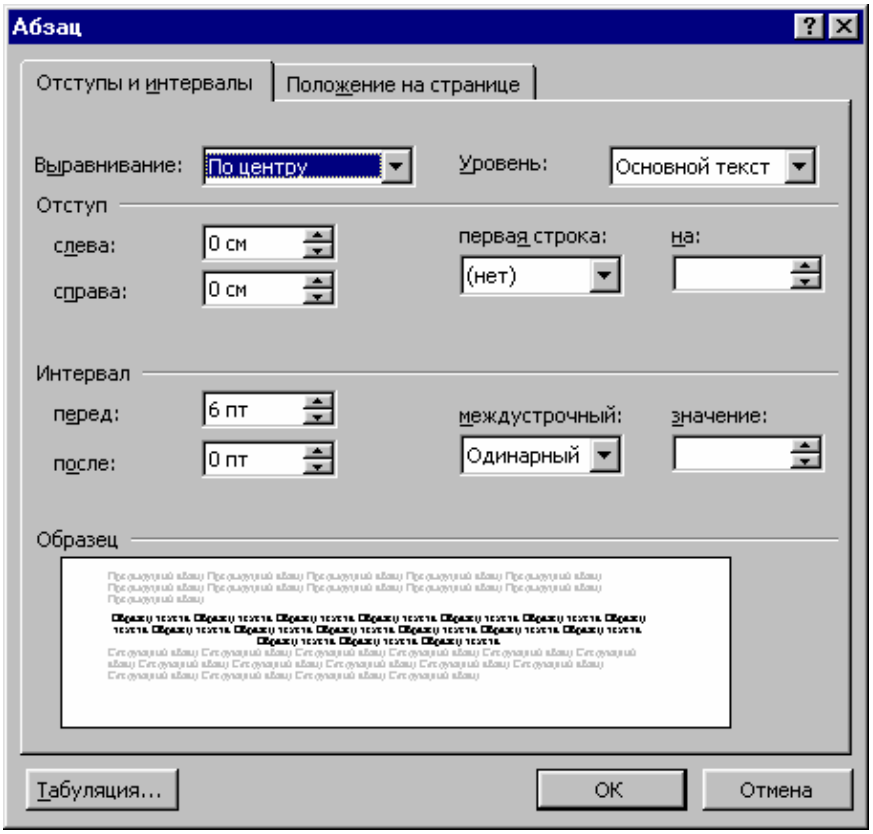


Рисунок 6.21

У полі **Выравнивание** установлюється спосіб вирівнювання абзацу:

- По левому краю – абзац вирівнюється по лівому краю сторінки;

- По центру – абзац вирівнюється по центру сторінки;

- По правому краю – абзац вирівнюється по правому краю сторінки;

- По ширине – абзац вирівнюється по обидва боки.

У полях **слева** і **справа** установлюються відстані від лівого й правого поля до меж абзацу.

У полі **первая строка** – вигляд відступу першого рядку абзацу:

- (нет) – відступ відсутній;

- Отступ – новий рядок, відстань вказується в полі **на;**

- Выступ – негативний відступ, відстань вказується в полі **на.**

У полях **перед** і **после –** відстані відповідно перед першим рядком абзацу й після останнього рядка абзацу.

У полі **междустрочный –** інтервал поміж рядками усередині абзацу:

- Одинарный – інтервал, стандартний для даного типу шрифту;

- Полуторный – інтервал у 1,5 разу більший стандартного;

- Двойной – інтервал у 2 рази більший стандартного;

- Минимум – інтервал не менший, ніж у полі **значение;**

- Точно – інтервал, рівний значенню в полі **значение;**

- Множитель – інтервал, рівний стандартному, помноженому на значення в полі **значение;**

Установлювати спосіб вирівнювання можна також за допомогою кнопок (рис. 6.22).

Рисунок 6.22

Рисунок 6.23

На горизонтальній координатній лінійці (рис. 6.23) знаходяться: маркер першого рядка (1), маркер лівої (2) і правої (3) меж абзацу. Пересуваючи їх за допомогою миші, можна змінювати відповідні параметри абзацу.

Встановлення позицій табуляції

Табуляція використовується для точного вирівнювання колонок тексту або чисел (рис. 6.24). Якщо установити позиції табуляції, то при кожному натисканні клавіші **Tab** курсор буде пересуватися праворуч до найближчої позиції табуляції.

Для встановлення позицій табуляції використовується команда **Табуляция** із меню **Формат,** що викликає діалогове вікно **Табуляция** (рис. 6.25):

- по левому краю – текст вирівнюється по лівому краю;

- по центру – текст вирівнюється по центру щодо позиції табуляції;

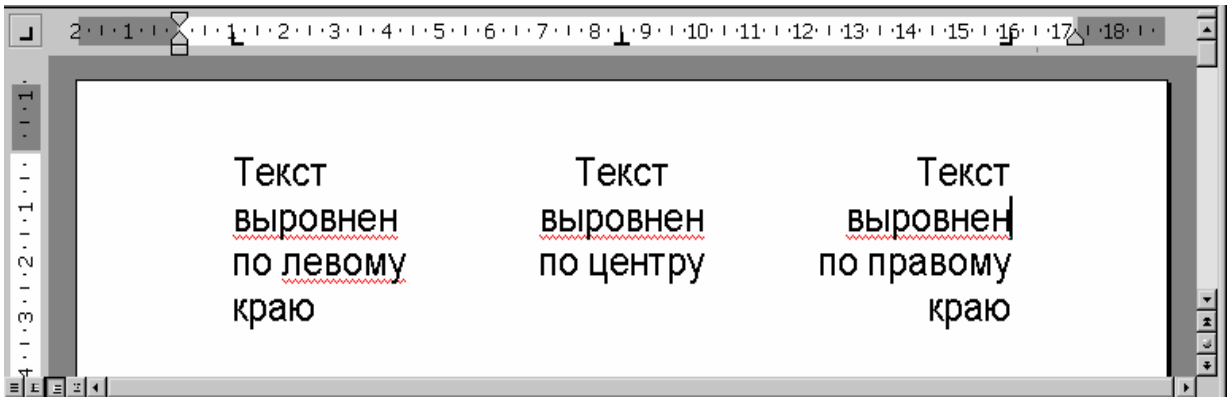


Рисунок 6.24

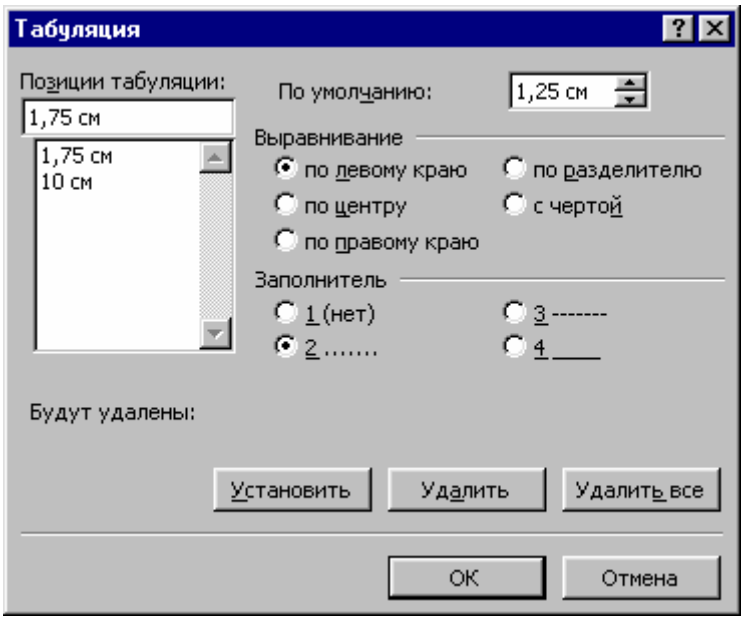


Рисунок 6.25

- по правому краю – текст вирівнюється по правому краю щодо позиції табуляції;

- по разделителю – числа вирівнюються по десятковій крапці, текст вирівнюється по правому краю;

- с чертой – під позиціями табуляції з'являються вертикальні риски.

Для заповнення порожнього місця, що залишається ліворуч від знаку табуляції, можна використовувати ланцюжок символів, які можна вибрати в рамці **Заполнитель.**

Установивши усі необхідні параметри для однієї позиції, слід натиснути кнопку **Установить** і нова позиція буде внесена в перелік **Позиции табуляции,** що містить усі установлені позиції табуляції. Щоб змінити тип вже установленої позиції табуляції, досить вибрати потрібну позицію з переліку **Позиции табуляции** і установити нові значення режимів.

Для видалення позиції табуляції слід вибрати її в переліку **Позиции табуляции** і натиснути на кнопку **Удалить.** Усі наявні позиції табуляції можна видалити натисканням кнопки **Удалить все.**

Установити позицію табуляції можна також натисканням миші на горизонтальній координатній лінійці. Тип позиції табуляції зазначений усередині квадрата в лівому кінці горизонтальної координатної лінійки. Якщо натиснути мишею на цьому квадраті, то тип позиції табуляції зміниться. По черзі можна вибрати такі типи табуляції:

 – вирівнювання по лівому краю;

 – вирівнювання по центру;

 – вирівнювання по правому краю;

 – вирівнювання по десятковому знаку.

Якщо виділити фрагмент тексту, вирівняного по позиції табуляції, й пересунути мишею символ табуляції в нове місце, то текст пересунеться разом із символом табуляції. Щоб видалити позицію табуляції, досить стягнути з координатної лінійки символ табуляції.

Упорядкування переліків

Microsoft Word дозволяє швидко складати переліки з позначками, нумерацією й багаторівневі переліки з нумерацією. Елементом переліку вважається абзац тексту. Для створення переліку необхідно виділити абзаци, які слід зробити елементами переліку або установити курсор у той

абзац, з якого буде починатися перелік. Потім викликати команду **Список** з меню **Формат,** що викликає діалогове вікно **Список** (рис. 6.26).

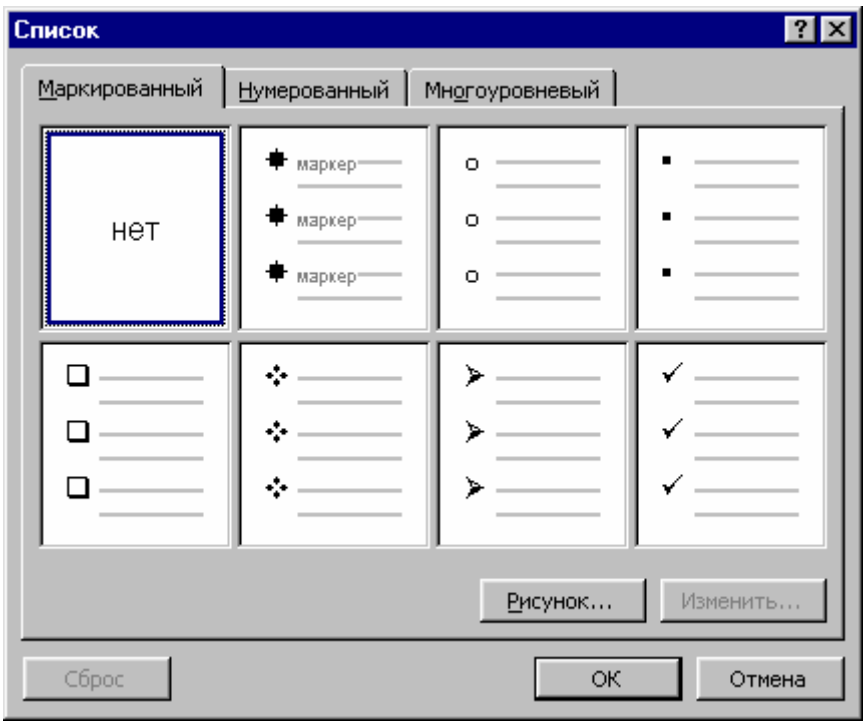


Рисунок 6.26

Для створення переліку з позначками необхідно вибрати вкладку **Маркированный.** Кожний елемент переліку з позначками виділяється за допомогою невеликої позначки, розташованої ліворуч від самого елементу. Серед запропонованих варіантів позначок слід вибрати відповідну (натиснути на ній мишею) й натиснути **OK.**

Для зміни вигляду позначки можна скористатися кнопкою **Изменить.** З'явиться вікно **Изменение маркированного списка**, в якому містяться додаткові позначки**.** При натисканні кнопки **Маркер** з'являється діалогове вікно **Символ,** в якому можна вибрати будь-який із символів як позначку переліку. У рамці **Положение маркера** задається відстань від лівого краю абзацу до позначки, а в рамці **Положение текста** визначається відстань від лівого краю абзацу до лівого краю тексту в переліку.

Для створення переліків із нумерацією використовується вкладка **Нумерованный** діалогового вікна **Список.** Серед запропонованих варіантів нумерації переліку необхідно вибрати потрібний, натиснути **OK** і перелік буде створений. Коли курсор введення знаходиться в переліку,

кожне натискання **Enter** створює новий пронумерований елемент переліку. При доданні нового елементу в перелік або видаленні елементу, номери в переліку **коректуються автоматично.**

Щоб створити свій варіант нумерації, слід натиснути кнопку **Изменить.** З'явиться вікно **Изменение нумерованного списка** (рис. 6.27)**.** У полі **Формат номера** указується текст перед і після номера елементу переліку, наприклад:) або [ ]. У полі **нумерация** вказується стиль нумерації, а в полі **начать с** вказується число (або літера), з якого повинен починатися перелік**.** Шрифт номерів елементів переліку змінюється за допомогою кнопки **Шрифт.**

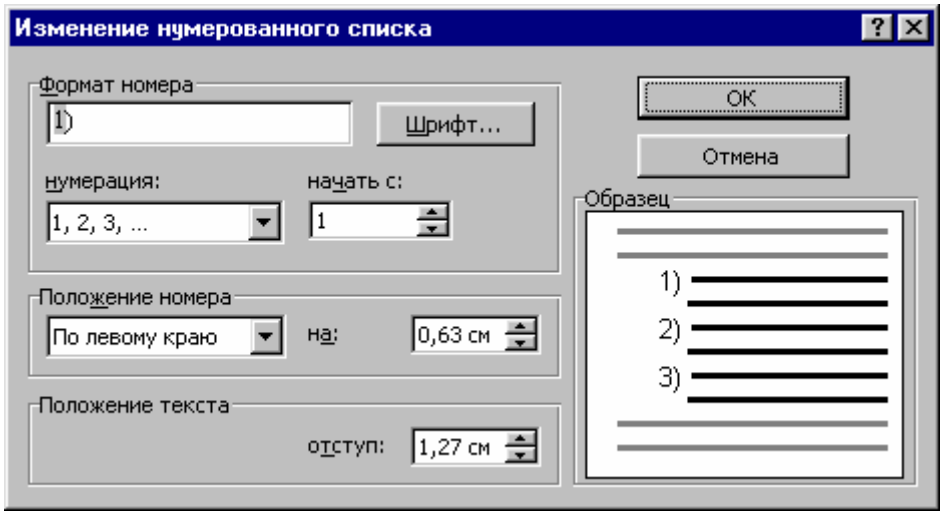


Рисунок 6.27

Швидко створити переліки з маркерами та нумерацією можна за допомогою кнопок і . Для створення переліку з декількома рівнями укладання використовується вкладка **Многоуровневый** діалогового вікна **Список.**

Стилі форматування

Стиль форматування – набір параметрів (шрифту, абзацу тощо), що має унікальне ім'я. Вибрати стиль виділеного фрагмента тексту можна у прихованому переліку **Стиль** на панелі **Форматирование** та у вікні **Стиль** меню **Формат**. У полі **Стили** вікна **Стиль** (рис. 6.28) міститься перелік стилів, що використовуються. Щоб побачити усі стилі слід у прихованому переліку **Список** вибрати – **Всех стилей**. У полях праворуч будуть показані зразки абзаців та символів відформатовані цим

стилем. Для присвоювання фрагмента тексту виділеного стилю слід натиснути кнопку **Применить**.

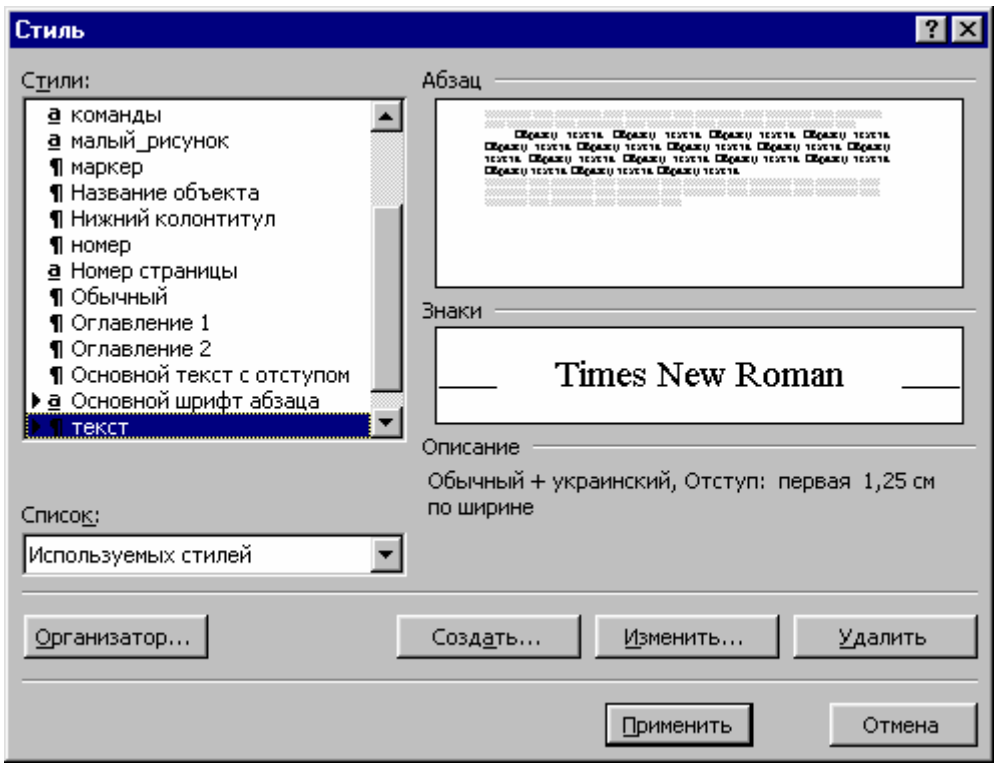


Рисунок 6.28

Для створення нового стилю у вікні **Стиль** використовується кнопка **Создать**. У полі **Имя** вікна **Создание стиля** (рис. 6.29) вводиться ім'я нового стилю. У переліку **Стиль** вибирається різновид стилю: стиль абзацу або стиль символу. У переліку **Основан на стиле** вибирається існуючий стиль, на основі якого буде створено новий. Якщо установити прапорець **Добавить в шаблон**, то новий стиль буде діяти не тільки в активному вікні, а й в усіх документах створених на основі цього ж шаблону. Для встановлення параметрів шрифту, абзацу та ін. слід натиснути кнопку **Формат** і потім вибрати об'єкт форматування. Після натискання кнопки **ОК** новий стиль буде створено. Якщо натиснути кнопку **Применить** у вікні **Стиль**, то новий стиль буде надано виділеному фрагменту тексту. Кнопка **Закрыть** закриває вікно без надання стилю.

Для зміни існуючого стилю слід виділити його у вікні **Стиль** і натиснути кнопку **Изменить**. У вікні **Изменение стиля** можна вибрати нові параметри. Для видалення стилю його слід виділити та натиснути кнопку **Удалить**.

Створити стиль можна також за зразком. Для цього необхідно виділити фрагмент тексту, що буде узятий за зразок, ввести ім'я стилю у поле **Стиль** на панелі **Форматирования** та натиснути **Enter**. Створений стиль буде діяти тільки в активному документі.

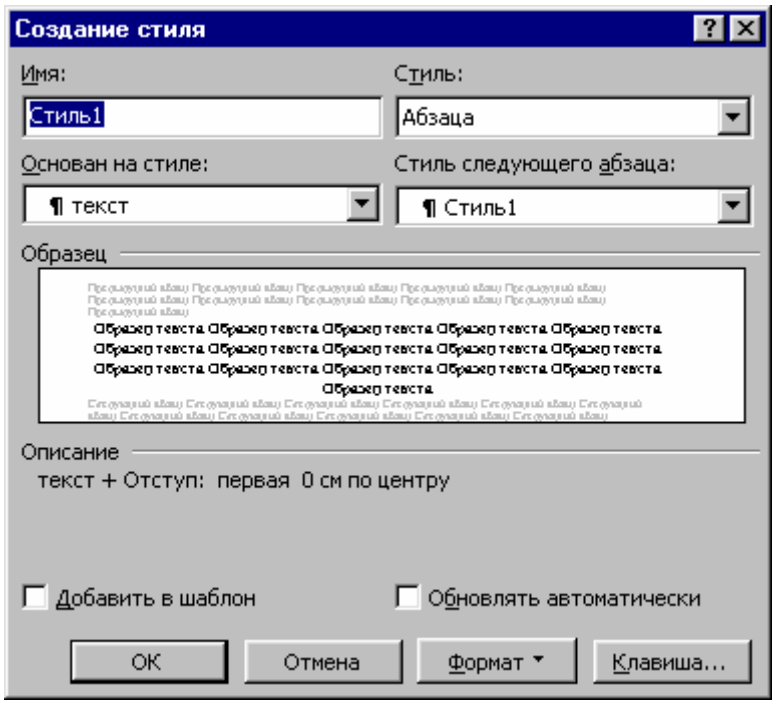


Рисунок 6.29

Оформлення сторінок документа

Встановлення параметрів сторінки

Для встановлення параметрів сторінки використовується команда **Параметры страницы** з меню **Файл**, що викликає діалогове вікно **Параметры страницы.**

Для встановлення поля сторінки використовується вкладка **Поля** (рис. 6.30), у вікнах якого можна установити:

- **Верхнее –** верхнє поле сторінки;
- **Нижнее –** нижнє поле сторінки;
- **Левое –** ліве поле сторінки;
- **Правое** – праве поле сторінки.

У рамці **Образец** показаний зовнішній вигляд сторінки відповідно до обраних параметрів. Якщо сторінки повинні мати дзеркальні поля, необхідно увімкнути прапорець **Зеркальные поля**. В результаті замість полів **Правое** і **Левое** з'являться поля **Внутри** і **Снаружи**.

У полі **Переплет** установлюється ширина поля підшивки**.**

У рамці **От края до колонтитула** установлюється відстань:

- верхнего – від верхнього краю сторінки до верхнього краю верхнього колонтитула;

- нижнего – від нижнього краю сторінки до нижнього краю нижнього колонтитула.

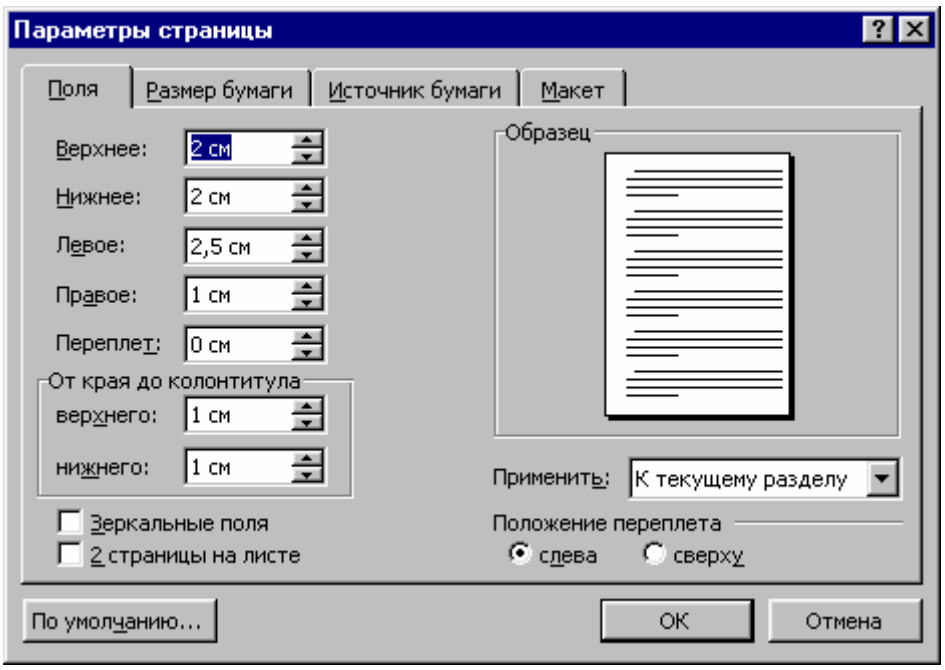


Рисунок 6.30

Слід зазначити, до якої частини документа відносяться обрані параметри, вказавши потрібне значення в полі **Применить:**

- Ко всему документу – параметри використовуються у всьому документі;

- До конца документа – параметри використовуються для тієї частини документа, що розташована після курсора введення.

Установити поля сторінки можна також за допомогою лінійок форматування в режимі **Разметка страниц**. У цьому режимі на екрані присутні і вертикальна, і горизонтальна координатні лінійки. На кожній координатній лінійці поля сторінки позначені сірим кольором. Необхідно установити покажчик миші на межу сірої та білої ділянки (він матиме вигляд двоспрямованої стрілки) й пересунути її в потрібне місце.

Вкладка **Размер бумаги** (рис. 6.31) містить поле переліку **Размер бумаги,** в якому можна вибрати розмір аркуша документа. Якщо необхідні розміри в переліку відсутні, то в полі **Ширина** і **Высота** можна ввести відповідні значення ширини й висоти сторінки**.**

У рамці **Ориентация** вибирається орієнтація сторінки**.** Прапорець **книжная** означає вертикальну орієнтацію сторінки, **альбомная** – горизонтальну.

Вкладка **Макет** вікна **Параметры страницы** дозволяє установити параметри колонтитулів. Для того, щоб на сторінках із парними і непарними номерами були різні колонтитули, слід увімкнути прапорець **четных и нечетных страниц**. Щоб колонтитул першої сторінки відрізнявся від інших, необхідно увімкнути прапорець **первой страницы.** Спосіб вертикального вирівнювання тексту на сторінці вибирається в полі **Вертикальное выравнивание:**

- По верхнему краю – текст вирівнюється по верхньому полю сторінки.

- По центру – текст центрується між верхнім і нижнім полем сторінки.

- По высоте – текст розподіляється між верхнім і нижнім полем.

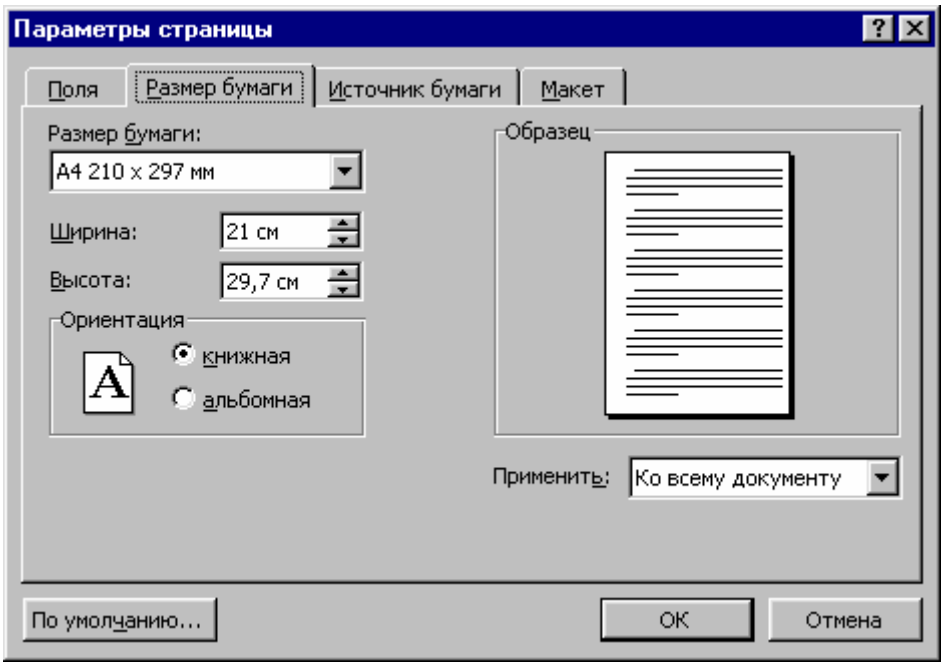


Рисунок 6.31

Встановлення розривів сторінок

Microsoft Word автоматично розбиває текст на сторінки. Для встановлення додаткового розриву сторінки необхідно установити курсор на місце, з якого повинна починатися нова сторінка й викликати команду **Разрыв** із меню **Вкладка. У** діалоговому вікні **Разрыв** (рис. 6.32) потрібно установити перемикач **новую страницу** та натиснути **ОК**.

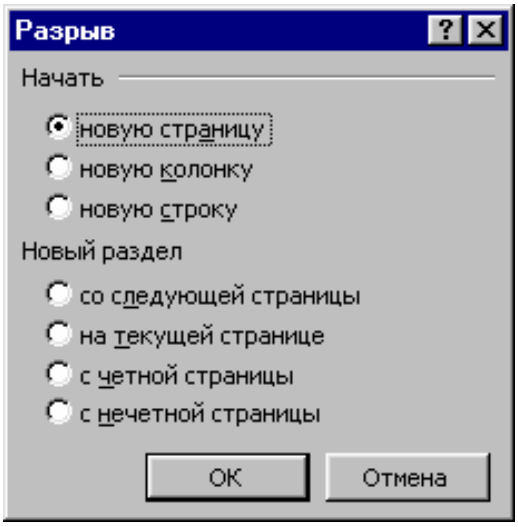


Рисунок 6.32

Якщо документ повинен складатися зі сторінок, що мають різні параметри, то його слід розділити на декілька розділів. Кожний розділ має власні параметри сторінок. Для вставлення в документ нового розділу в діалоговому вікні **Разрыв** необхідно вибрати один з таких перемикачів:

- со следующей страницы – новий розділ починається з наступної сторінки;

- на текущей странице – новий розділ починається безпосередньо після поточного;

- с четной страницы – новий розділ починається з найближчої сторінки, що має парний номер;

- с нечетной страницы – новий розділ починається з найближчої сторінки, що має непарний номер.

Щоб видалити розрив сторінки, вставлений вручну, або розрив розділу слід перейти в режим **Обычный,** або увімкнути режим відображення недрукованих символів**.** У цих режимах розриви сторінок зображуються пунктирними лініями, а розриви розділів подвійними пунктирними лініями. Видаляються знаки розривів як звичайні символи клавішами **Delete** або **Backspace**.

Нумерація сторінок

Для встановлення номерів сторінок необхідно викликати команду **Номера страниц** меню **Вкладка,** що викликає вікно **Номера страниц** (рис. 6.33).

У полі **Положение** слід вибрати розташування номера на сторінці:

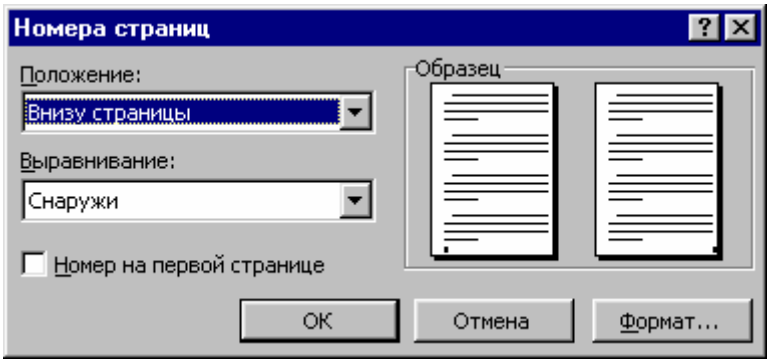


Рисунок 6.33

- Вверху страницы – номер сторінки розташовується угорі (уставляється в верхній колонтитул);

- Внизу страницы – номер сторінки розташовується унизу (уставляється в нижній колонтитул).

У полі **Выравнивание** – розташування номера сторінки відносно полів сторінки:

- Слева – номер сторінки розташовується з лівого краю сторінки;

- От центра – номер сторінки розташовується по центру сторінки;

- Справа – номер сторінки розташовується з правого краю сторінки;

- Внутри – номер сторінки розташовується з внутрішнього краю листа (доступний, тільки якщо документ має дзеркальні поля);

- Снаружи – номер сторінки розташовується з зовнішнього краю листа (доступна, якщо документ має дзеркальні поля).

Якщо зняти прапорець **Номер на первой странице**, то на першій сторінці номер не буде проставлений**.**

Кнопка **Формат** викликає діалогове вікно **Формат номера страницы** (рис. 6.34), в якому задається формат нумерації. У полі переліку **Формат номера** вибирається тип нумерації (арабські і римські цифри або літери латинської абетки).

У рамці **Нумерация страниц** установлюється початок нумерації:

- продолжить – нумерація сторінок поточного розділу починається з числа, наступного за номером останньої сторінки попереднього розділу;

- начать с – нумерація починається з числа, зазначеного у полі праворуч.

Якщо увімкнути прапорець **Включить номер главы,** до номера сторінки буде доданий номер глави або розділу документа. У полі переліку **начинается со стиля** необхідно зазначити, який стиль форматування відповідає рівню глав, номера яких будуть використані. Можна вибрати один із стилів заголовків **Заголовок 1... Заголовок 9.** У полі

разделитель задається роздільник між номером сторінки й номером глави. Установивши усі параметри, слід натиснути **ОК**, після чого знову з'являється вікно **Номера страниц.** Тут також необхідно натиснути **ОК** і усі сторінки документа будуть пронумеровані.

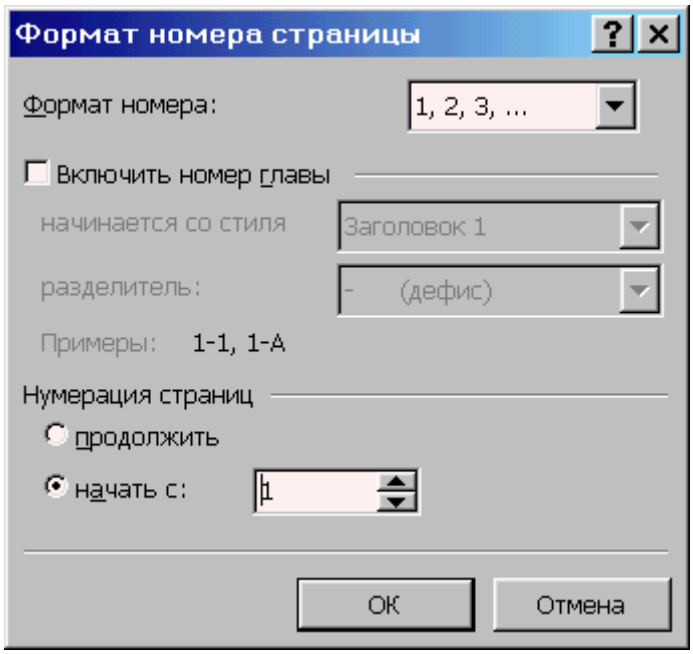


Рисунок 6.34

Встановлення колонтитулів

Колонтитул – текст або рисунок, що друкується унизу або угорі кожної сторінки документа. У колонтитулі, зазвичай, розміщені номера сторінок, назва книги або поточної глави. В залежності від місця розташування (на верхньому або на нижньому полі сторінки) колонтитули бувають верхніми і нижніми. Текст, введений до колонтитула, форматується як звичайний текст.

Для створення колонтитулів слід вибрати команду **Колонтитулы** у меню **Вид.** При цьому відбувається автоматичний перехід у режим екрану **Разметка страниц,** тому що в режимі **Обычный** колонтитули не відображаються. На екрані з'являється піктографічне меню **Колонтитулы** (рис. 6.35).

Для переходу з поля верхнього колонтитула в поле нижнього колонтитула й назад використовується кнопка 1 (рис. 6.35).

Введений текст колонтитула розташовується в пунктирній рамці, що вказує межі колонтитула. Для встановлення номерів сторінок

використовується кнопка 2 (рис. 6.35). Текст колонтитула форматується як звичайний текст. У режимі відображення колонтитула основний текст документа редагувати неможливо.

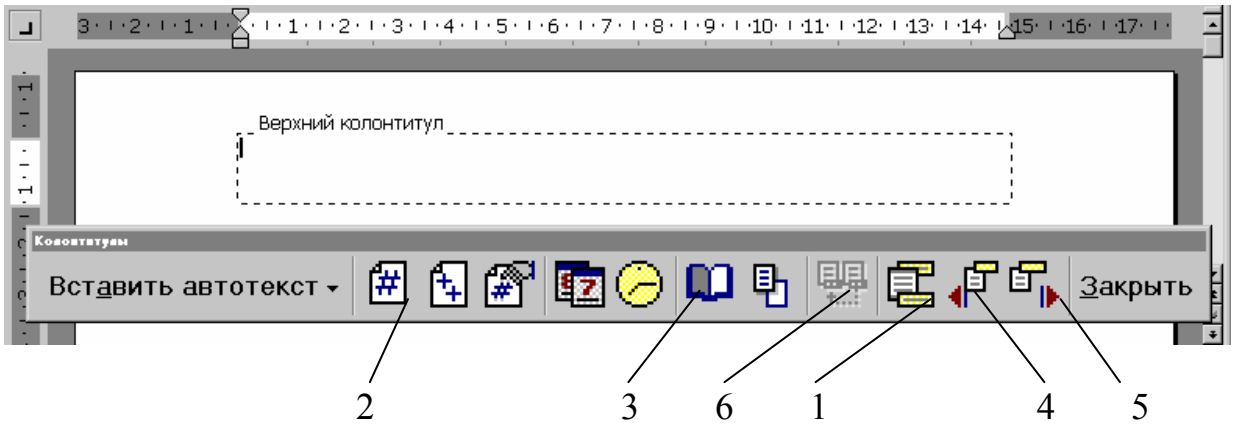


Рисунок 6.35

Для створення на першій сторінці документа колонтитула відмінного від колонтитулів інших сторінок необхідно викликати вікно **Параметры страницы** із меню **Файл** і в вкладці **Макет** установити прапорець **первой страницы.** Якщо в цій вкладці установити прапорець **четных и нечетных страниц,** то можна створити окремо колонтитул для парних і колонтитул для непарних сторінок. Викликати вікно **Параметры страницы** можна за допомогою кнопки 3 (рис. 6.35) із панелі **Колонтитулы.** За допомогою кнопок 4, 5 (рис. 6.35) можна пересуватися між колонтитулом першої сторінки, парної і непарної сторінок. Якщо залишити поле колонтитула порожнім, то колонтитул буде відсутній.

Встановлення прапорця **четных и нечетных страниц** впливає на весь документ, якщо він не поділений на розділи. Якщо документ поділений на декілька розділів, при вставленні колонтитула в один розділ цей же колонтитул автоматично додається в усі розділи документа, якщо натиснута кнопка 6 (рис. 6.35) (приєднати колонтитули поточної секції до колонтитулів попередньої). Щоб створити різні колонтитули для декількох частин документа, слід розірвати зв'язок між розділами. Для цього необхідно вибрати розділ, для якого слід створити інший колонтитул і відтиснути кнопку 6 (рис. 6.35). Після цього необхідно змінити існуючий колонтитул або створити новий.

Для видалення колонтитула слід вибрати команду **Колонтитулы** у меню **Вид,** виділити колонтитул, що необхідно видалити й натиснути клавішу **Delete.** При зміні або видаленні колонтитула в будь-якому

розділі, так само змінюються або видаляються колонтитули в інших розділах, якщо зв'язок із попереднім розділом не буде розірваний примусово за допомогою кнопки 6 (рис. 6.35).

Створення багатошпальтового документа

Microsoft Word дозволяє верстати текст у декілька шпальт. Текст вводиться в них послідовно, переходячи до наступної шпальти після заповнення попередньої. Для багатошпальтової верстки слід перейти в режим **Разметка страниц**, тому що в режимі **Обычный** текст не буде відображений у декілька шпальт.

Існують два варіанти використання багатошпальтової верстки.

1. Весь документ розбитий на однакову кількість шпальт однакової ширини.

2. Різні частини документа розбиті на різне число шпальт або шпальти мають різну ширину. У цьому випадку необхідно розбити документ на розділи, кожен з яких буде мати свій поділ на шпальти.

Для створення шпальт у рамках розділу документа слід установити курсор у текст цього розділу. Якщо весь документ необхідно розбити на однакову кількість шпальт, то курсор може знаходитися в будь-якому місці тексту. Потім слід вибрати команду **Колонки** меню **Формат,** що викликає діалогове вікно (рис. 6.36).

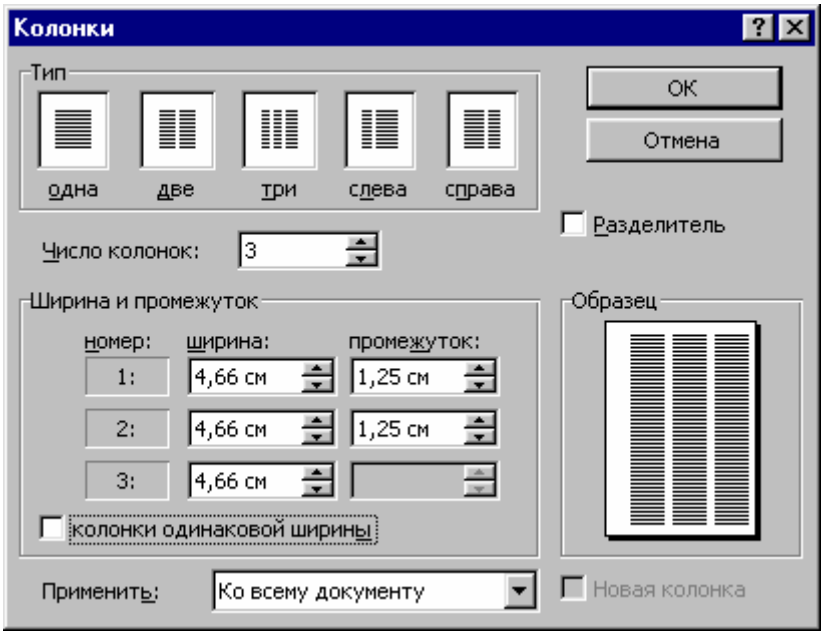


Рисунок 6.36

У полі **Число колонок** необхідно ввести число шпальт або вибрати один із рисунків у рамці **Тип**. Прапорець **Разделитель** накреслює лінію між шпальтами тексту. Якщо увімкнений прапорець **колонки одинаковой ширины,** то усі шпальти будуть мати однакову ширину. Якщо вимкнути цей прапорець, то можна ввести для кожної зі шпальт точні значення її ширини і відстані між шпальтами в поля **Ширина и промежуток.**

У полі **Применить** вказується частина документа, для якої будуть діяти обрані режими:

- К текущему разделу – параметри використовуються тільки у поточному розділі;

- До конца документа – параметри використовуються для тієї частини документа, що розташована після курсора введення;

- Ко всему документу – параметри використовуються у всьому документі.

Створити шпальти однакової ширини можна за допомогою кнопки  . Після натискання на ній з'являється вікно, в якому слід виділити потрібну кількість шпальт і натиснути кнопку миші.

Змінювати ширину шпальт і відстань між ними можна за допомогою горизонтальної координатної лінійки. Коли текст розбитий на шпальти, на лінійці відображаються відповідні символи (рис. 6.37):

1 – символ правої межі;

2 – відстань між шпальтами;

3 – символ лівої межі шпальти.

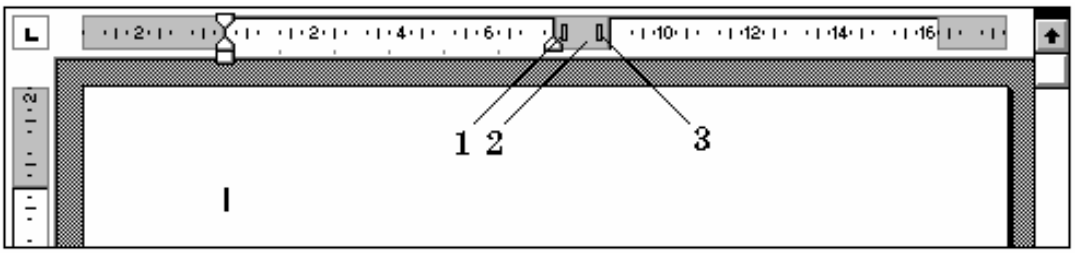


Рисунок 6.37

При пересуванні цих символів будуть змінюватися відповідні параметри шпальт. Для переходу до наступної шпальти можна викликати команду **Разрыв** меню **Вкладка,** у діалоговому вікні увімкнути перемикач **новую колонку** і натиснути **ОК**. Після чого курсор введення й увесь текст нижче курсора пересунеться до початку наступної шпальти.

Видалення шпальт – це операція встановлення однієї шпальти для всього документа.

Друкування документів

Перед друкуванням документа, можна переглянути на екрані як він буде виглядати після друку. Для цього необхідно перейти в режим попереднього перегляду за допомогою команди **Предварительный просмотр** меню **Файл** або кнопки  . У цьому режимі, щоб збільшити зображення слід навести покажчик миші, який матиме вигляд лупи з плюсом, на потрібний фрагмент і натиснути кнопку миші. Покажчик миші матиме вигляд лупи з мінусом і якщо натиснути кнопку миші, то зображення зменшиться. Вийти з режиму попереднього перегляду можна за допомогою кнопки **Закрыть** або клавіші **Esc**.

Для друкування документа використовується команда **Печать** меню **Файл.** У діалоговому вікні **Печать** (рис. 6.38) у полі прихованого переліку **имя** потрібно вибрати принтер, якщо можна друкувати на декількох принтерах.

У рамці **Страницы** задається діапазон сторінок, що будуть надруковані:

- все – надрукується весь документ;

- текущая – надрукується сторінка, в якій знаходиться курсор;

- выделенный фрагмент – друкується тільки виділений фрагмент документа;

- номера – друкується зазначений набір сторінок. Наприклад, щоб надрукувати сторінки 1, 5, 11, 12, 13 необхідно ввести в поле ліворуч: 1, 5, 11-13.

У полі **Копии** вказується кількість копій. Щоб надрукувати цілком першу копію, потім другу й інші слід увімкнути прапорець **разобрать по копиям.** Для друку багатосторінкового документа з двох сторін кожного аркуша можна увімкнути режим виведення на друк тільки парних або непарних сторінок. У переліку **Вывести на печатать** можна вибрати одне із значень:

- Все страницы диапазона – надрукувати весь діапазон сторінок;

- Нечетные страницы – тільки непарні сторінки з зазначеного діапазону;

- Четные страницы – тільки парні сторінки з зазначеного діапазону.

Для друкування однієї копії всього документа достатньо натиснути на кнопоку  .

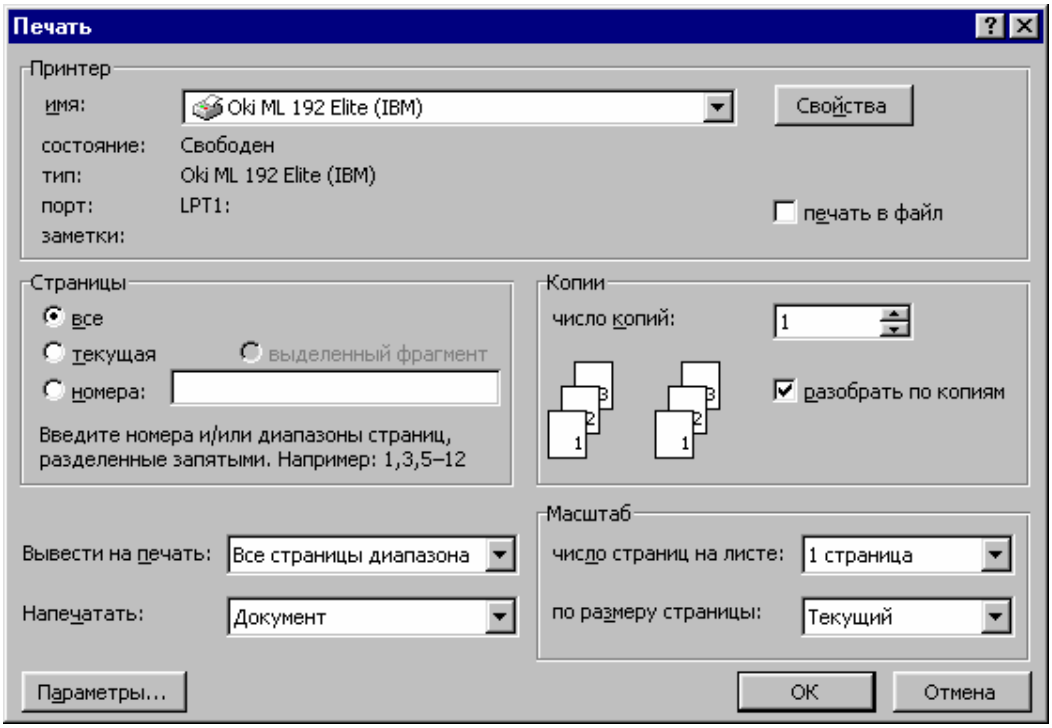


Рисунок 6.38

Вставлення графічних об'єктів

Microsoft Word дозволяє вставляти в документ графічні об'єкти, створені як в інших програмах, так і за допомогою власної панелі рисування. Об'єкти можна копіювати й вставляти на будь-яке місце документа або в інший документ. При доданні рисунка в документ він приєднується до навколишнього тексту. Якщо абзац, що містить рисунок, пересувається угору або униз по сторінці, рисунок пересувається разом із ним.

Викликати панель **Рисование** можна через пункт **Панели инструментов** меню **Вставка** або натиснувши на кнопку  . При цьому слід перейти в режим **Разметка страниц.** За допомогою кнопок панелі рисування можна зображувати лінії, стрілки, еліпси, прямокутники, кола, дуги, сектори й різні криві. Графічний об'єкт можна залити кольором або візерунком, змінити форму, дзеркально відбити або повернути, змінити колір і тип його ліній, додати до них стрілки.

Для вставлення графічного об'єкта, створеного в іншій програмі, необхідно установити курсор у позицію, де повинен стояти об'єкт і у меню

Вставка вибрати пункт **Рисунок,** а потім пункт **Из файла.** У вікні, що з'явилося, (рис. 6.39) у прихованому переліку **Папка** вибрати диск, а в переліку що розташований нижче – папку, в якій знаходиться файл із рисунком. Якщо клацнути мишею на імені файла, що містить рисунок, одночасно в рамці ліворуч буде подано його зображення. Після натискання кнопки **OK** обраний рисунок буде вставлений у документ. Для вставлення рисунків, що поставляються з Microsoft Word, слід після пункту **Рисунок** вибрати пункт **Картинки.**

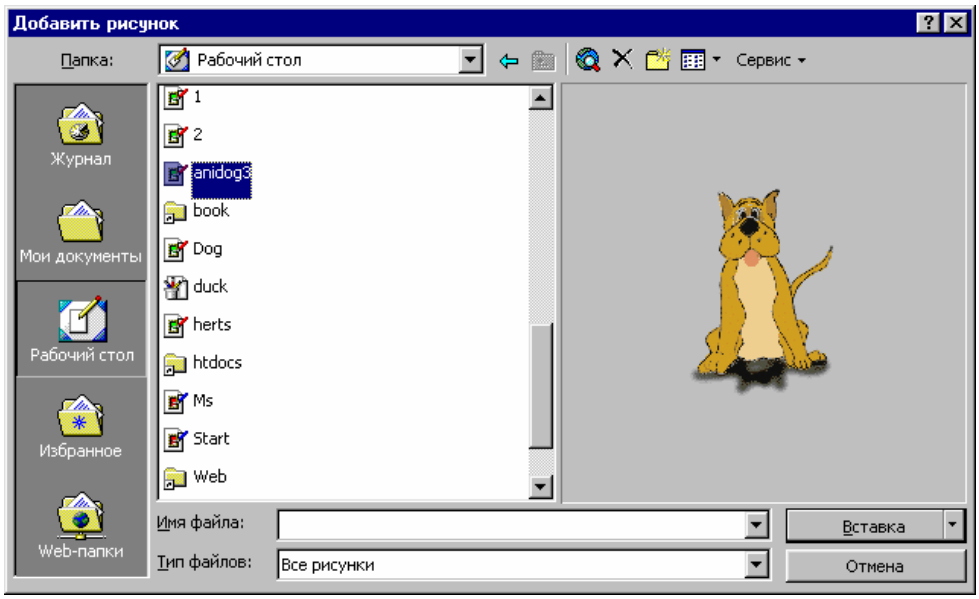


Рисунок 6.49

Щоб змінити розміри й положення рисунка, слід натиснути на ньому мишею, після чого навколо нього з'являться маркери розміру. Пересуваючи кутові маркери мишею, можна змінювати розміри рисунка при зберіганні його пропорцій. При пересуванні інших маркерів буде змінюватися ширина або довжина рисунка. Перемістити рисунок можна за допомогою миші. При переміщенні рисунка за межі видимості екран просунеться в тому ж напрямку.

Якщо натиснути клавішу миші на рисунку, за замовчуванням з'явиться панель **Настройка изображения** з кнопками для зміни параметрів рисунка. Цю панель можна викликати також за допомогою контекстного меню.

Щоб змінити параметри рисунка необхідно викликати вікно **Формат рисунка (Формат объекта)**. Для цього необхідно вибрати відповідний пункт у меню **Вид** або натиснути на кнопку  . Наприклад,

щоб текст міг розташовуватися ліворуч або праворуч від рисунка, слід вибрати вкладку **Положение** і там потрібний вигляд обтікання.

Для видалення рисунка його слід виділити й натиснути клавішу **Delete**.

Вставлення таблиць

Для вставлення в документ таблиці необхідно установити курсор на місце, де повинна починатися таблиця та в меню **Таблица** вибрати пункт **Добавить,** потім **Таблица.** У діалоговому вікні, що з'явилося, слід ввести число стовпців і рядків, і натиснути **OK**. Для вставлення таблиці також можна натиснути на кнопку  . У вікні, що розкрилося, розтягнути виділення на необхідне число комірок і натиснути клавішу миші. Усі команди для роботи з таблицями знаходяться у меню **Таблица**.

Пересування по таблиці здійснюється за допомогою покажчика миші або клавішами: **↑, ↓, ←, →, Tab** (на комірку праворуч), **Shift+Tab** (на комірку ліворуч). Кожна комірка таблиці розглядається як абзац, і дані у комірках форматуються як абзаци тексту. Для роботи з таблицями зручно користуватися панеллю інструментів **Таблицы и границы**, яку можна вивести на екран за допомогою кнопки  .

При наведенні покажчика миші на верхню лінію таблиці, він перетворюється в чорну стрілку. Якщо в цей момент натиснути мишею, то виділиться один стовпець. Пересуваючи мишею чорну стрілку, можна виділити відразу декілька стовпців. Рядки таблиці виділяються як рядки звичайного тексту. Для виділення декількох суміжних комірок необхідно натиснути клавішею миші на одну комірку фрагмента й розтягнути виділення на інші.

Коли курсор введення знаходиться у таблиці, на координатних лінійках з’являються позначки меж стовпців  і рядків  . При пересуванні цих позначок змінюються розміри відповідних стовпців і рядків. За допомогою прихованого переліку  можна вибрати тип вирівнювання тексту у комірках. Кнопка  слугує для зміни напрямку розташування тексту у виділених комірках.

Щоб об'єднати декілька комірок в одне, слід виділити їх і викликати команду **Объединить ячейки** меню **Таблица** або натиснути на кнопку  на панелі **Таблицы и границы**. Для розділення комірки на декілька

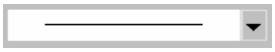
комірок слід вибрати пункт **Разбить ячейки** меню **Таблица** або натиснути кнопку  .

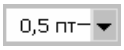
Для додання елементів таблиці (рядків, стовпців, комірок) необхідно виділити елементи, на місці яких необхідно уставити нові та в меню **Таблица** вибрати команду **Добавить**, потім – потрібний пункт (**Столбцы слева, Столбцы справа, Строки выше, Строки ниже, Ячейки**). Для видалення елементів таблиці слід виділити їх та у меню **Таблица** вибрати пункт **Удалить,** потім – потрібний пункт (**Таблица, Столбцы, Строки, Ячейки).**

За замовчуванням лінії сітки таблиці мають товщину 0,5 пт. Змінити товщини та вигляду ліній сітки можна декількома способами.

І спосіб:

- виділити комірки, обрамлення яких потрібно змінити;

- у прихованому переліку **Тип линии**  на панелі **Таблицы и границы** вибрати тип лінії;

- у прихованому переліку **Толщина линии**  – товщину лінії;

- якщо натиснути на кнопку  , з’явиться палітра кольорів, в якій можна вибрати колір обрамлення;

- відкрити прихований перелік  і вибрати вигляд обрамлення.

ІІ спосіб:

- у прихованих переліках панелі **Таблицы и границы** вибрати тип, товщину і колір лінії;

- натиснути на кнопку  ;

- покажчиком миші, який матиме вигляд олівця, вказати початок лінії та розтягнути до її кінця;

- після натискання на кнопку  покажчиком миші можна стирати лінії обрамлення.

Щоб залити комірку кольором необхідно виділити її і у прихованому переліку  вибрати колір.

Контрольні питання

1. Особливості створення і зберігання документа Microsoft Word. Версії програми Word.
2. Параметри сторінок документа Microsoft Word. Виділення тексту. Переміщення по сторінці; пошук і заміна слів; службові клавіші.

3. Захист документа Microsoft Word. Властивості документа.
4. Microsoft Word. Форматування абзацу. Робота з буфером обміну.
5. Microsoft Word. Вставлення символів. Колонки (стовпці). Верхні і нижні індекси (надрядкові і підрядкові знаки).
6. Microsoft Word. Режим колонтитулів. Номера сторінок.
7. Microsoft Word. Маркований і нумерований список. Створення змісту і вказівок.
8. Microsoft Word. Створення, редагування і видалення таблиць. Автоформат. Властивості таблиць.
9. Microsoft Word. Створення і редагування графічних об'єктів. Автофігури.
10. Microsoft Word. Створення формули (редактор формул Microsoft Equation Editor).
11. Microsoft Word. Створення і настроювання діаграми. (об'єкт Microsoft Graph).
12. Microsoft Word. Групування і розгрупування графічних обєктів. Панель настроювання зображень. Експорт рисунків із файлів.
13. Microsoft Word. Формат об'єкта (автофігура, формула, діаграма, рисунок тощо). Настроювання тіні і об'єму.
14. Режими попереднього перегляду документа Word. Настроювання параметрів друку. Перевірка правопису. Допоміжні словники. Тезаурус.

Лабораторна робота №7

Первинне налагодження текстового редактора Microsoft Word. Створення документа, зберігання. Параметри сторінок. Зберігання тексту. Форматування тексту. Спеціальні клавіші

Мета роботи: навчитись створювати документ Word. Навчитись задавати параметри сторінок, правильно набирати, редагувати текст і форматувати його, засвоїти способи виділення тексту і способи переміщування його по сторінці.

Порядок виконання роботи

1. Запустити текстовий редактор.

2. Вивчити інформацію на екрані.

3. Виставити параметри сторінки, полів, абзацу.

4. Вибрати шрифт та його параметри.

5. Набрати довільний текст.

6. Запам’ятати набрану інформацію у файл у своїй папці, використати меню **Файл** та вкладку **Сохранить как**.

7. Скопіювати набрану інформацію та вставити її нижче на тій же сторінці.

8. Відформатувати текст, що скопіювали, по ширині сторінки.

9. Відредагувати текст з використанням виділення, копіювання, та вирізання.

10. Набрати заголовок тексту, виділити його іншим шрифтом та стилем.

11. Запам’ятати файл зі змінами тексту.

12. Скласти звіт.

Питання для самоконтролю

5. Яка послідовність дій необхідна для виконання пунктів 1-11?
6. Види форматування тексту.
7. Чим характеризується розмір та стиль тексту?
8. Як виділити фрагмент?
9. Заміна фрагментів тексту, слів.
10. Які функції виконують кнопки на екрані монітора даного текстового редактора?
11. Команди меню текстового редактора та порядок їх використання.

Лабораторна робота №8

Первинне налагодження параметрів друкованого документу текстового редактора Microsoft Word. Колонтитули, номера сторінок. Додаткове форматування. Перевірка правопису

Мета роботи: засвоїти методи правильного оформлення документації, вивчення правил користування заздалегідь заготовленими шаблонами **Microsoft Word**, створити колонтитул для розміщення номерів сторінок, засвоїти методи перевірки правопису, зберегти документ з використанням для зберігання створеної власної папки.

Порядок виконання роботи

1. Запустити текстовий редактор.

2. Створити новий файл.

3. Скопіювати у цей файл текст із попередньої лабораторної роботи.

4. Вставити у текст таблицю для розпису робочого дня погодинно.

5. Створити у заданому місці колонтитул для розміщення номерів сторінок. Встановити шрифт номерів сторінок.

6. Включити програму перевірки правопису. Використати вкладку **Замена** з меню **Правка** для заміни помилкових слів.

7. Відформатувати створений документ.

8. Запам’ятати створений документ у файл.

9. Скласти звіт.

Питання для самоконтролю

1. Яка послідовність дій необхідна для виконання пунктів 1-8?

2. Як вставити чи видалити рядок або стовпчик таблиці?

3. Опишіть дії, які необхідні для об’єднання або поділу комірок таблиці.

4. Зміна напрямку тексту в таблиці.

5. Виставлення полів сторінки.

6. Опишіть послідовність встановлення параметрів абзацу (відступи, інтервали, вирівнювання).

Лабораторна робота № 9

Введення спеціальних символів. Рисування і вставлення рисунків. Форматування рисунків. Редактор формул Microsoft Equation Editor

Мета роботи: засвоїти можливості створення графічних об’єктів в програмі Word, навчитись рисувати автофігури, задавати формат будь-якої автофігури, засвоїти можливості допоміжного модуля Micrisoft Equation Editor, навчитись редагувати вже написані формули.

Порядок виконання роботи

1. Запустити текстовий редактор.

2. Створити новий файл.

3. Створити графічний об’єкт з використанням автофігур.

4. Згрупувати та розгрупувати об’єкт.

5. Збільшити або зменшити розміри об’єкта.

6. Набрати задані формули та відредагувати їх.

7. Пронумерувати та відцентрувати формули.

8. Запам’ятати створений документ у файл.

9. Скласти звіт.

Питання для самоконтролю

1. Яка послідовність дій необхідна для виконання пунктів 1-8?

2. Як вставити чи видалити графічний об’єкт?

3. Переміщення вставок у тексті.

4. Опишіть порядок набирання та редагування математичних формул.

5. Як вставити фотографію або кліп у текст?

6. Які команди забезпечують обтікання текстом графічного об’єкта справа, зліва, з обох сторін тощо.

7. Як накласти напис на рисунок?

8. Порядок зміни розмірів графічного об’єкта.

9. Опишіть можливі варіанти імпорту рисунків з інших програм.

10. Як виділити рисунок у тексті рамкою або зняти таке виділення?

11. Як встановити кольоровий фон на рисунку або змінити його?

12. Порядок зміни кольору підписів на рисунках та можливість виділення тексту кольором.

ЛІТЕРАТУРА

1. Савуляк В.І., Семичаснова Н.С. Інформатика: Навчальний посібник. – Вінниця: ВНТУ, 1999. – 132 с.
2. Фигурнов В.Э. IBM PC для пользователя. 2-е изд. перераб. и доп. – М.: Финансы и статистика.1991. – 288 с.
3. Зуев Е.А. Система программирования Turbo Pascal. – М.: Радио и связь, 1991. – 288 с.
4. Нортон П. Персональный компьютер фирмы IBM и операционная система MS-DOC: Пер. с англ. – М.: Радио и связь, 1992. – 416 с.
5. Зуев Е.А. Язык программирования Turbo Pascal 6.0. –М.: Унитех, 1992. – 298 с.
6. Сердюченко В.Я. Розробка алгоритмів та програмування мовою Turbo Pascal. – Харків: ВКП ”Парітет” ЛТД, 1995. – 352 с.
7. Руденко В.Д. Збірник практичних робіт з інформатики/ За ред. Мадзігона В.М. – К.: Видавнича група "ВНУ", 1999. – 96с.
8. Аладьев В.З., Хунт Ю.Я., Шищаков М.Л. Основы информатики: Учебное пособие. – М.: "Филинь", 1998. – 496с.
9.Дьяконов В.П. Internet. Настольная книга пользователя. – М.: Солон-Р, 1999. – 576с.
10. Информационные системы в экономике: Учебник / Под ред. В.В. Дика. - М.: Финансы и статистика, 1996. – 272с.
11. Компьютерные системы и сети / Под ред. В.П. Косарева и Л.В. Ярёмина – М.: Финансы и статистика, 1999. – 464.
12. Стинсон К. Эффективная работа в Windows -98. – СПб.: Питер, 1999. – 784 с.
13.Стойкий Ю. Самоучитель Offise 2000. – СПб.: Питер, 1999. – 576 с.

Навчальне видання

Валерій Іванович Савуляк, Тетяна Федорівна Архіпова,

Андрій Васильович Губанов

ІНФОРМАТИКА. Частина 1
Навчальний посібник

Оригінал-макет підготовлено Савуляком В. І.

Редактор О.Д. Скалоцька

Науково-методичний відділ ВНТУ
Свідоцтво Держкомінформу України
серія ДК №746 від 25.12.2001
21021, м. Вінниця, Хмельницьке шосе, 95, ВНТУ

Підписано до друку
Формат 29,7х42 1/4
Друк різографічний
Тираж прим.
Зам. №

Гарнітура Times New Roman
Папір офсетний
Ум. друк. арк.

Віддруковано в комп'ютерному інформаційно-видавничому центрі
Вінницького національного технічного університету
Свідоцтво Держкомінформу України
серія ДК № 746 від 25.12.2001
21021, м. Вінниця, Хмельницьке шосе, 95, ВНТУ