# Тема. Мова програмування Pascal.

## 1. Поняття про мову програмування Turbo Pascal. Головне вікно Turbo Pascal.

Мова програмування Паскаль була створена швейцарським вченим Ніклаусом Віртом в 1968 році. Названа звана на честь видатного французького математика і філософа Блеза Паскаля (1623-1662). Мова Паскаль придатна для створення великих прикладних систем.

Для персональних комп'ютерів фірмою Borland (США) в 1985 році створено середовище програмування Turbo Pascal.

Для запуску середовища Turbo Pascal потрібно перейти в каталог програми і запустити на виконання файл **turbo.exe** або **tpw.exe** (в залежності від версії) або скористатися ярликом на робочому столі.

Після завантаження з'являється головне вікно Turbo Pascal, яке складається із головного меню і вікна редактора програм. Головне меню містить такі пункти:

1. Файл – містить команди для роботи з файлом програми
2. Дії – містить команди для роботи з буфером обміну
3. Пошук – містить команди пошуку фрагментів в тексті програми
4. Запуск програми – містить команди для запуску та налагодження програми
5. Компіляція – містить команди для перетворення програми в автономний exe-файл.
6. Настройка – містить команди для настройки середовища програмування
7. Управління вікнами – містить команди управління вікнами
8. Довідка – отримання довідки по мові програмування та середовищу програмування

Для створення нової програми необхідно виконати команду **Файл – Створити новий**. Для запуску програми необхідно виконати команду **Запуск програми – Запустити програму**.

В загальному випадку програма є послідовністю рядків, що містять команди. В кінці кожного рядка ставиться крапка з комою (крім деяких випадків). Набір допустимих команд обмежений. Кожна мова програмування має свій набір команд і правил їх запису. Для успішного складання програм необхідно вивчити цей набір команд і правил.

## 2. Структура програми на мові Turbo Pascal.

Програма на мові PASCAL складається з трьох основних частин: заголовка, описової частини і виконавчої частини.

Заголовок містить службове слово program та ім'я програми.

Описова частина містить опис розділів об'єктів, із якими буде працювати програма. До таких розділів відносяться: зовнішні модулі, константи, змінні, мітки, масиви, процедури, функції, файли, множини. Для позначення таких розділів використовують такі зарезервовані слова на мові Паскаль:

1. **uses** – опис зовнішніх модулів
2. **label** – опис міток

3. **const** – опис констант
4. **type**– опис типів змінних
5. **var** – опис змінних
6. **procedure** – опис процедур
7. **function** – опис функцій

Не всі перелічені розділи обов'язково мають бути присутні у програмі, це залежить від поставленої задачі.

Виконавча частина програми починається службовим словом begin (початок) і закінчується ключовим словом end (кінець), після якого ставиться крапка. Між begin і end записується основний текст програми, що складається з команд (операторів), розділених крапкою з комою ";". Для зручності запису програми її розбивають на рядки.

Приклад програми:

```
Program Pr; (заголовок програми)
Var x,y,z:integer; (описова частина)
Begin (початок виконавчої частини)
Readln(x);
   Y:=x+5;      (виконавча частина)
   Z:=x*y;
Writeln(z);
End. (закінчення виконавчої частини і програми)
```

## 3. Типи даних та їх описи

При побудові майже кожної програми використовуються змінні, які в процесі виконання програми набувають певних значень. Опис змінних відбувається в описовій частині програми після зарезервованого слова **var.**

При створенні програм найчастіше використовують такі типи даних:

1. Integer – цілі числа в межах від -32768 до 32767
2. Byte - цілі числа в межах від 0 до 255
3. Word - цілі числа в межах від 0 до 65535
4. Real – дійсні числа в межах від $-2{,}9*10^{39}$ до $1{,}7*10^{38}$
5. Double - дійсні числа в межах від $-5*10^{324}$ до $1{,}7*10^{308}$
6. Char – змінні, що набувають символьних значень (A := 'Медицина');
7. Boolean – логічний тип даних, що можуть приймати два значення «Так» або «Ні».
8. Array – масив даних – це набір однотипних компонентів.

Приклад:

```
Var
i: integer;
a,b: real;
c: char;
x: array [1..10] of real;
```

## 4. Арифметичні функції

Мова Pascal має досить широкий набір математичних функцій, які можна застосовувати для розв'язання різноманітних завдань. Найчастіше використовують наступні:

| № | Функція | Призначення | Тип |
|---|---|---|---|
| 1 | ABS(X) | \|X\| | Integer, real |
| 2 | ARCTAN(X) | Arctg x | Real |
| 3 | COS(X) | Cos x | Real |
| 4 | EXP(X) | $e^x$ | Real |
| 5 | LN(X) | Ln x | Real |
| 6 | RANDOM(X) | Випадкове число в діапазоні [0;X] | Integer, real |
| 6 | SIN(X) | Sin x | Real |
| 7 | SQR(X) | $X^2$ | Integer, real |
| 7 | SQRT(X) | $\sqrt{X}$ | Real |
| 8 | EXP(Y*LN(X)) | $X^y$, X>0 | Real |

## 5. Оператори мови Pascal

Виконавча частина програми на мові Pascal складається із операторів. Кожний оператор використовується для виконання певної дії. Розглянемо оператори, які найчастіше використовуються при побудові простих програм.

### 5.1. Оператор присвоєння

Значення змінних змінюють за допомогою оператора присвоєння. Загальний вигляд оператора присвоєння такий:

**<змінна> := <вираз>**

Приклад використання оператора присвоєння:

```
var
a, b, c, d: integer;
Тоді можна записати
a:=1; b:=2; c:=20; d:=a+b*c;
```

Після цього значення *d* дорівнюватиме 41.

### 5.2. Оператори введення-виведення

Оператори введення-виведення використовуються для введення даних з клавіатури або виведення даних на екран монітора.

Для виведення інформації використовують оператори **Write( )** або **Writeln( )**. Відмінність полягає в тому, що оператор **Write** виводить інформацію і курсор залишається в тому ж самому рядку, а оператор **Writeln** виводить інформацію і курсор переводиться на наступну стрічку. В дужках вказують 'текст' (в лапках) або список змінних через кому, які потрібно вивести.

Наприклад:

`write(A,B)` - виведення значення А і В в одну стрічку;

`writeln(A)` - виведення значення А і перехід на наступний рядок;

`writeln(B)` - виведення значення В і перехід на наступний рядок.

`writeln('Уведітьдані')` - виведення тексту і перехід на наступний рядок.

Для введення даних використовують оператори **Read( )** або **Readln( )**. Відмінність аналогічна попереднім операторам.

Наприклад:

read (A) – ввести значення змінної А;

read (A,B,C) – ввести через кому в одному рядку значення змінних (наприклад 10,15,20)

readln (A,B,C) – ввести три значення змінних, натискаючи після кожного клавішу Enter, тобто кожне значення вводиться з нового рядка.

Приклад програми:

```
Program prog1;
uses wincrt;
var
  a,b,c: real;
begin
  writeln (‘Введіть значення змінних А і В’);
  readln (a,b);
  c:=a*b;
  writeln (‘Результат:’,c);
end.
```

### 5.3. Умовний оператор

Загальний вигляд умовного оператора такий:

**if<логічний вираз>then<оператор>else<оператор>.**

Якщо значення логічного виразу (умови) є «істинним», то виконується оператор, що стоїть після **then**, якщо значення логічного виразу є «хибним», то виконується оператор, що стоїть після **else.**

Таким чином, залежно від результату логічного виразу виконується один з альтернативних операторів.

Прикладом, коли логічний вираз в операторі if має складнішу структуру, може бути задача про те, чи можна побудувати трикутник з відрізків a, b, *c:*

```
Program prog2;
uses wincrt;
var
  a,b,c: integer;
begin
  read (a,b,c);
  if (a+b>c) and (a+c>b) and (b+c>a)  then
     writeln ('Трикутник побудувати можна')
```

```
  else
     writeln ('Трикутник побудувати не можна');
end.
```

«and» – означає логічну операцію «і», ще часто також використовують операцію «or» - означає логічну операцію «або».

Умовний оператор може і не мати конструкції **else**, така форма називається скороченою:

**if<логічний вираз>then<оператор>**

У випадку такої конструкції умовного оператора, якщо вираз є «істинним», то виконується оператор, що є після **then,** а якщо вираз є «хибним», то жодні дії не виконуються.

Кожен з операторів, що виконується після **then** може бути складеним, наприклад,

```
if a>b then
 begin
       x:=x+1;
       y:=y+1;
 end;
else
 begin
       x:=abs(x);
       y:=abs(y);
 end;
```

Складений оператор використовують тоді, коли після **then** або **else** потрібно записати групу операторів, однак згідно з синтаксисом тут повинен бути лише один оператор.

Оператори, що стоять після **then** або **else,** самі можуть бути умовними, тоді маємо вкладену конструкцію умовного оператора.