# Тема. Програмування основних алгоритмічних структур.

## 1.     Оператори присвоювання, вводу-виводу; запис лінійних алгоритмів.

Практично кожна програма повинна виконувати такі дії: вводити вихідні дані, проводити опрацювання цих даних (як правило, обчислення) та виводити результати роботи. Для виконання цих дій використовуються оператори вводу, присвоювання та виводу.

Одним з основних операторів є оператор присвоювання. Він використовується для обчислення значення виразу і присвоювання його змінній величині.

Вид оператора:

і**м’я змінної := вираз**

При цьому значення виразу, розташованого справа від знака присвоювання **: =,** присвоюється змінній, ідентифікатор якої розташований зліва від знака присвоювання. При цьому результат виконання оператора на екрані не відображається.

Оператор присвоювання можна використовувати для присвоювання значень змінній будь-якого типу, крім файлового. Єдиною умовою є, щоб змінна в лівій частині і вираз в правій частині оператору мали однаковий або сумісний тип. Виняток складають випадки, коли змінна в лівій частині оператора має тип **real**, тоді вираз може бути цілочисельного типу.

ПРИКЛАДИ:

a :=41;

b := false;

c := a+8;

c := c div 2;

d := ‘a’;

Оператор присвоювання не виводить результат обчислень на екран. Він лише запам’ятовує ці результати в пам’яті у вигляді значень змінних.

Типовими помилками при використанні цього оператора, є:

             несумісні типи даних змінної та виразу;

             використання в лівій частині оператора замість змінної виразу.

Розповсюдженими є також синтаксичні помилки при записі виразів в лівій частині оператора.

ПРИКЛАДИ:

var

 a: integer;

 b:real;

…

a:=a/2;  операція ділення дає результат дійсного типу, а змінна a – цілочисельна;

b+1:=a–2;  в лівій частині оператора записаний вираз;

b:=sqrt(sqr(a)+1; в лівій частині не вистачає закритої дужки.

Оператор виводу інформації призначений для обчислення значень виразів і виводу їх на екран. Є два оператори виводу: **write**  та **writeln**. Вид оператора**write** :

**write(список виразів)**

Список задає вирази, значення яких обчислюються та виводяться на екран. При цьому ці значення не запам’ятовуються. В більшості випадків в ролі виразів використовуються константи або змінні. Наступний оператор виводу **write** продовжує вивід інформації у тому ж рядкові екрану, у якому закінчилась дія попереднього оператора.

Оператор **writeln** має аналогічний формат і призначення. Відмінність від попереднього оператора полягає в тому, що вивід інформації закінчується переходом на новий рядок, тобто наступний оператор виводу буде виводити інформацію з нового рядка екрану. При цьому допускається використання “порожнього” оператора **writeln** (тобто оператора, у якому відсутній список виразів). Такий оператор використовують для переходу при виводі інформації на новий рядок або пропуск рядка.

ПРИКЛАД:

1. **write**(‘Моя програма’);

write(a,b,c);

write(b\*b-4\*a\*c);

2. writeln(x,y);

writeln(x/y);

writeln;

Для кращої наочності виводу інформації існує форматний вивід даних. Такий спосіб виводу дозволяє виділяти певну кількість позицій під кожен елемент списку виводу. Форматний вивід розглянемо на прикладі оператора **write.** Для даних цілого, логічного та літерного типів форматний вивід має вигляд:

**write(вираз:n),**

де **n** задає кількість позицій екрану, які відводяться для виводу значення виразу. Якщо ця кількість менша за потрібну, мова ПАСКАЛЬ автоматично розширяє поле виводу. Якщо кількість позицій більша за потрібну, значення вирівнюється по правому краю поля, а зайві позиції заповнюються пробілами.

Для даних дійсного типу форматний вивід має вигляд:

**write(вираз:n:m),**

де **n** задає загальну кількість позицій екрану, які відводяться для виводу значення виразу, а **m -** кількість позицій для дробової частини числа. Слід пам’ятати, що загальна кількість позицій виводу повинна включати одну позицію на знак числа, і одну позицію на десяткову точку. Якщо значення **m** менше за дійсну кількість цифр у дробовій частині, то зайві цифри відкидаються з округленням. Якщо значення **n** менше за потрібне, мова ПАСКАЛЬ автоматично розширяє поле виводу. Якщо кількість позицій більша за потрібну, значення вирівнюється по правому краю поля, а зайві позиції заповнюються пробілами.

ПРИКЛАДИ:

write(a:4,b:6,c:1);

write(b\*b-4\*a\*c:5);

writeln(x:8:3,y:6:2);

Оператор вводу даних призначений для вводу даних з клавіатури та присвоювання їх змінним. Є два оператори вводу даних: **read** та **readln . О**ператор **read**   має такий вид :

**read(список змінних)**

де список задає ті змінні, значення яких необхідно ввести з клавіатури. Можна вводити з клавіатури значення всіх стандартних типів даних, крім **boolean**.

При виконанні оператора очікується введення даних вказаного типу. Значення повинні вводитись у строгій відповідності з синтаксисом мови ПАСКАЛЬ. При цьому ніякого додаткового повідомлення про очікування вводу даних на екрані не виводиться. Дані можна вводити списком, розділяючи їх символом “пробіл”, або по одному. Введення закінчується натискуванням клавіші **Enter**.

Оператор вводу **readln** має аналогічний формат. Для скалярних типів при вводі даних з клавіатури обидва оператори працюють аналогічно.

Типовими помилками використання оператора є :

        розбіжність типу змінної і типу значення, що вводиться;

        введення виразів, а не значень.

Для наочності рекомендується перед оператором вводу використовувати оператор виводу з підказкою про тип та кількість значень, що вводяться.

ПРИКЛАДИ:

write(‘введіть цілі значення змінних a,b,c’);

readln(a,b,c);

write(‘введіть два довільних дійсних числа’);

readln(x,y);

Для покращення наочності тексту програми рекомендується використовувати коментарі. Їх наявність не обов’язковою, але дозволить пояснити особі, що працює з текстом програми, зміст окремих її блоків. Коментар являє собою послідовність будь-яких символів, укладених у фігурні дужки. Коментар не є оператором, і його не потрібно відокремлювати від інших об’єктів програми символом “;”.Навпаки, він є роздільником і може використовуватись для відокремлення складових рядків програми аналогічно символу “пропуск”. Але таке використання коментарю, очевидно, було б дуже незручним.

ПРИКЛАД:

p:=(a+b+c)/2; {обчислення півпериметра трикутника}

s:=sqrt(p\*(p-a)\*(p-b)\*(p-c)); {обчислення площі за формулою Герона}

Лінійним називається алгоритм, у якому оператори виконуються в порядку їх слідування в програмі. Для запису таких алгоритмів достатньо наведених операторів. Розглянемо приклади запису лінійних алгоритмів.

Задача 1 Два робітники виконували спільну роботу протягом місяця на умовах погодинної оплати праці. Перший відпрацював t1годин, а другий – t2 годин. Вартість однієї години першого робітника становить p1 грн., другого – p2 грн. Обчислити середню заробітну плату робітників.

Розв’язання. Середня заробітна плата обчислюється за формулою:

pc=(p1 \* t1 + p2 \* t2)/(t1 + t2)

Програма для обчислень має вигляд:

program zarplata;

var p1,p2:real;

 t1,t2:integer;

 pc:real;

 x:real;

 y:integer;

begin

 write(‘введіть кількість відпрацьованих годин робітниками’);

 readln(t1,t2);

 write(‘введіть вартості годин:’);

 readln(p1,p2);

 x:=p1\*t1+p2\*t2;

 y:=t1+t2;

 pc:=x/y;

 writeln(‘Середня зарплата становить pc=’,pc:7:2,’грн’);

end.

## 2.     Умовні оператори; запис розгалужень.

Лінійні алгоритми зустрічаються на практиці рідко. В більшості випадків обчислення носять розгалужений або циклічний характер. При цьому оператори об’єднуються в групи, які виконуються як один оператор. Для цього призначений складений оператор. Складений оператор використовується також в тому випадку, якщо необхідно виконати декілька операторів, а синтаксис мови ПАСКАЛЬ припускає наявність лише одного оператора. Складений оператор може містити довільну кількість операторів, відділених один від одного крапкою з комою. Отже, він являє собою сукупність операторів мови ПАСКАЛЬ, об’єднаних операторними дужками **begin** та **end.**

ПРИКЛАД

begin

 readln(x,y);

 z:=x+y;

 writeln(‘z=’,z);

end;

Операторні дужки не є операторами, тому символ “;” перед службовим словом **end** не обов’язковий. Складений оператор використовується в умовних операторах та операторах циклу, які будуть розглянуті далі.

Умовний оператор дозволяє виконувати один з декількох операторів, що входять у його склад,  у залежності від виконання певної умови. До умовних відносять оператори **if** та **case**.

Оператор умовної передачі управління **if** призначений для вибору однієї з двох альтернатив в залежності від значення логічного виразу.

Вид оператора:

**if** логічний вираз **then**  оператор1 **else** оператор2

Якщо вираз приймає значення **true**, то виконується **оператор1** (гілка **then**), у противному випадку, якщо вираз приймає значення **false**, то виконується **оператор2** (гілка **else**). Після цього виконується наступний за оператором **if** оператор.

В будь-якому випадку виконується лише один оператор із вказаних у гілках . Неможлива ситуація, коли виконується або обидва оператори, або не виконується жоден із них.

Якщо по якійсь із гілок має бути декілька операторів, вони повинні бути об’єднані операторними дужками у складений оператор.

Якщо одна із гілок не містить ніяких дій, то можна скористуватись скороченою формою оператора:

**if** логічний вираз **then**  оператор

Якщо вираз приймає значення **true**, то виконується **оператор** (гілка **then**), у противному випадку ця гілка пропускається (обходиться ). Далі виконується наступний за цим оператор.

ПРИКЛАДИ:

1. Пошук більшого із двох значень: z=max(x,y)

**if**  x>y **then**  z:=x **else** z:=y;

2. Впорядкування значень двох змінних за зростанням

if x>y  then

**begin**

z:=x; x:=y; y:=z

**end;**

“Розширенням” оператора **if** є оператор варіанта **case.** Вінпризначений для вибору однієї з декількох альтернатив в залежності від значення селектора варіанта.

Вид оператора

**Case** селектор **of**

мітка1: оператор1;

мітка2: оператор2;

. . .

міткаN: операторN;

**end;**

Оператор **case** складається з виразу (селектора) і списку операторів, кожному з який передує мітка. Ці мітки називаються мітками варіанта і вони не повинні оголошуватись у розділі опису міток програми. При цьому мітки варіанта повинні мати той же тип, що і селектор. З визначення оператора **case** випливає, що в кожному випадку виконується лише один оператор - той, мітка якого дорівнює поточному значенню селектора. Якщо жодна з міток не дорівнює поточному значенню селектора, то ніякі оператори не виконуються.

Оператор варіанта має ще одну форму:

**Case** селектор **of**

мітка1: оператор1;

мітка2: оператор2;

. . .

міткаN: операторN;

**else** оператор(N+1);

end;

Цей оператор варіанта відрізняється від попереднього тим, що, якщо жодна з міток не дорівнює поточному значенню селектора, то виконується оператор, що слідує за зарезервованим словом **else**. Наявність альтернативи **else** в операторі **case** є розширенням стандартної мови ПАСКАЛЬ. Мітка оператора **case** може складатися з довільної кількості констант або діапазонів, відділених один від одного комами. Слідом за міткою ставиться двокрапка. Діапазон записується у виді двох констант, відділених один від одного роздільником “**..**”. Тип констант повинний збігатися з типом селектора. Оператор, записаний слідом за міткою оператора **case**, виконується в тому випадку, якщо значення селектора дорівнює однієї з констант або потрапляє в межі одного з зазначених діапазонів.

У ролі типу селектора можуть використовуватись всі типи, за винятком дійсного.

ПРИКЛАДИ:

1. case i of

1: write(‘понеділок’);

2: write(‘вівторок’);

3: write(‘середа’);

4: write(‘четвер’);

5: write(‘п’ятниця’);

6: write(‘субота’);

7: write(‘неділя’);

 end;

2.case i of

1..5: write(‘робочий день’);

6,7: write(‘вихідний’);

 end;

Розглянемо приклад запису розгалуженого алгоритму.

Задача 3. Розмір відрахувань на заробітну плату становить 1% з суми, що не перевищує 150 грн, і 2% у противному випадку. Програма має вигляд:

program vidrah;

var s:integer;

 x:real;

begin

 write(‘введіть значення фонду зарплати s:’);

 readln(s);

 if s<=150 then x:=s\*0.01 else x:=s\*0.02;

 writeln(‘відрахування становлять x=’,x:7:2, ‘грн’)

end.

## 3.     Оператори циклу; запис циклів.

Для організації циклічних процесів призначені оператори повтору. Вони вказують на те, що певну групу операторів необхідно виконати декілька разів. Ця група операторів називається тілом циклу. Якщо кількість повторень заданої групи операторів відома заздалегідь, то в такій ситуації цілком підходить оператор циклу з параметром **for**. Якщо ж кількість повторів невідома, то необхідно використовувати оператори **while** або **repeat**.

Оператор циклу з параметром **for** призначений для програмування циклів із заздалегідь відомою кількість повторень тіла циклу.

Вид оператора:

**for** параметр **=** поч.зн **to** кін.зн **do** оператор,

**де for, =, to, do -** службові слова**;**

**параметр** - змінна, яка рахує кількість повторень тіла циклу;

**поч.зн**.  та **кін.зн**. – початкове та кінцеве значення параметру циклу (діапазон його значень);

**оператор** - тіло циклу.

Параметром може бути будь-яка неіндексована змінна довільного скалярного типу (окрім типу **real**).

Параметр може приймати лише послідовні значення заданого діапазону за їх зростанням. Поки значення параметру входить в діапазон, тіло циклу виконується. Параметр циклу може не приймати участь в обчисленнях, які проводяться в тілі циклу. По завершенні виконання оператора циклу виконується наступний за ним оператор.

Існує ще одна форма оператора **for**

**for** параметр **=** поч.зн **to** кін.зн **downto** оператор

Відмінність від попереднього полягає у тому, що параметр приймає значення вказаного діапазону за їх спаданням.

Для програмування циклів із заздалегідь невідомою кількістю повторень або циклів, в яких параметр циклу приймав би значення із діапазону з певним кроком (наприклад 2, 4, 6, ...) використовуються оператор циклу з передумовою та оператор циклу з постумовою.

Оператор циклу з передумовою **while** має вигляд

**while** лог. вираз **do** оператор ,

де **while, do** -службові слова;

**лог. вираз** - умова (логічний вираз), яка управляє виконанням тіла циклу;

**оператор** - задає тіло циклу.

Поки логічний вираз приймає значення **true**, виконується тіло циклу. Зрозуміло, що зміну істинності умови необхідно передбачити у тілі циклу (тобто тіло циклу повинно містити оператор, який впливає на зміну істинності умови).

Оператор циклу з постумовою **repeat** має вигляд

**repeat**  оператор **until** лог. вираз ,

де **repeat** ,**until** -службові слова;

**логічний вираз** - умова (логічний вираз), яка управляє виконанням тіла циклу;

**оператор** - задає тіло циклу.

Поки логічний вираз не прийме значення **true** (тобто, приймає значення **false**), виконується тіло циклу. Зміну істинності умови для цього оператора також необхідно передбачити у тілі циклу .

В усіх операторах циклу (окрім **repeat**) тіло циклу може складатись лише з одного оператора . Якщо тіло циклу містить групу операторів, їх необхідно об’єднати операторними дужками.

ПРИКЛАД :

обчислити добуток перших n натуральних чисел (тобто n!) кожним із операторів циклу

а) оператор циклу з параметром **for-to**:

p:=1;

for i:=1 to n do p:=p\*i;

б) оператор циклу з параметром **for-downto**:

p:=1;

for i:=n downto 1 do p:=p\*i;

в) оператор циклу з передумовою

p:=1;

i:=1;

while i<=n do

begin

p:=p\*i;

i:=i+1

end;

г) оператор циклу з постумовою

p:=1;

i:=1;

repeat

p:=p\*i;

i:=i+1

until i>n;

# Узагальнення по темі.

Практично кожна програма повинна виконувати такі дії: вводити вихідні дані, проводити опрацювання цих даних (як правило, обчислення) та виводити результати роботи. Для виконання цих дій використовуються оператори вводу, присвоювання та виводу.

Одним з основних операторів є оператор присвоювання. Він використовується для обчислення значення виразу і присвоювання його змінній величині.

Оператор виводу інформації призначений для обчислення значень виразів і виводу їх на екран. Є два оператори виводу: **write**  та **writeln**.

Оператор вводу даних призначений для вводу даних з клавіатури та присвоювання їх змінним. Є також два оператори вводу даних: **read** та **readln .**

Можна вводити з клавіатури значення всіх стандартних типів даних, крім **boolean**. При виконанні оператора очікується введення даних вказаного типу. Значення повинні вводитись у строгій відповідності з синтаксисом мови ПАСКАЛЬ. Дані можна вводити списком, розділяючи їх символом “пробіл”, або по одному. Введення закінчується натискуванням клавіші **Enter**.

Для покращення наочності тексту програми рекомендується використовувати коментарі. Їх наявність не обов’язковою, але дозволить пояснити особі, що працює з текстом програми, зміст окремих її блоків. Коментар являє собою послідовність будь-яких символів, укладених у фігурні дужки. Коментар є роздільником і може використовуватись для відокремлення складових рядків програми аналогічно символу “пропуск”.

Лінійним називається алгоритм, у якому оператори виконуються в порядку їх слідування в програмі. Для запису таких алгоритмів достатньо використання операторів присвоювання, вводу та виводу.

Лінійні алгоритми зустрічаються на практиці рідко. В більшості випадків обчислення носять розгалужений або циклічний характер. При цьому оператори об’єднуються в групи, які виконуються як один оператор. Для цього призначений складений оператор. Такий оператор використовується також в тому випадку, якщо необхідно виконати декілька операторів, а синтаксис мови ПАСКАЛЬ припускає наявність лише одного оператора. Складений оператор являє собою деяку сукупність операторів мови ПАСКАЛЬ, об’єднаних операторними дужками **begin** та **end.**

Умовний оператор дозволяє виконувати один з декількох операторів, що входять у його склад,  у залежності від виконання певної умови. До умовних відносять оператори **if** та **case**.

Оператор умовної передачі управління **if** призначений для вибору однієї з двох альтернатив в залежності від значення логічного виразу.

“Розширенням” оператора **if** є оператор варіанта **case.** Вінпризначений для вибору однієї з декількох альтернатив в залежності від значення селектора варіанта.

Для організації циклічних процесів призначені оператори повтору. Вони вказують на те, що певну групу операторів необхідно виконати декілька разів. Ця група операторів називається тілом циклу. Якщо кількість повторень заданої групи операторів відома заздалегідь, то в такій ситуації цілком підходить оператор циклу з параметром **for**. Якщо ж кількість повторів невідома, то необхідно використовувати оператори **while** або **repeat**.

Оператор циклу з параметром **for** призначений для програмування циклів із заздалегідь відомою кількість повторень тіла циклу.

Для програмування циклів із заздалегідь невідомою кількістю повторень або циклів, в яких параметр циклу приймав би значення із діапазону з певним кроком (наприклад 2, 4, 6, ...) використовуються оператор циклу з передумовою та оператор циклу з постумовою.