Тема №5.

ПРОГРАМУВАННЯ ЦИКЛІЧНИХ ОБЧИСЛЮВАЛЬНИХ ПРОЦЕСІВ

Циклічним називається такий обчислювальний процес, який повторює одну і ту ж визначену послідовність операторів декілька разів. В Borland Pascal розрізняють три види циклів: оператор циклу з лічильником – *FOR*, оператор циклу з передумовою – *WHILE*, оператор циклу з післяумовою (постумовою) – *REPEAT*.

У випадках, коли заздалегідь відома кількість повторень, використовують оператор циклу з лічильником:

```
FOR <змінна>:=<початкове_значення_лічильника> TO
            <кінцеве_значення_лічильника> DO
            <оператор>
```

де змінна, якій спочатку присвоюється початкове значення, виконує роль лічильника, і при кожному повторенні збільшується на одиницю або зменшується на одиницю, якщо замість ключового слова *TO* записати інше ключове слово – *DOWNTO* (при цьому початкове значення повинно бути більше за кінцеве). Саме ця змінна і показує на кількість повторень.

Два інші оператори циклу використовують коли кількість повторень наперед невідома, ознакою закінчення циклу виступає виконання або невиконання деякої заданої умови.

Структура запису оператору циклу з передумовою:

```
WHILE <умова> DO
      <оператор>
```

При виконанні цього оператора спочатку перевіряється умова (обраховується деякий логічний вираз) і, у випадку, коли це значення правдиве, виконується оператор. Якщо є

потреба виконання в циклі декількох операторів, їх включають до складового оператора. Виконання оператора буде проводитися по циклу до тих пір, поки умова зберігає значення *TRUE*. Як тільки умова стане *FALSE*, виконання циклу припиниться і управління передасться наступному за циклом оператору.

Структура запису оператору циклу з післяумовою:

```
REPEAT
      <оператор>
UNTIL <умова>
```

Тут спочатку виконується оператор, тобто тіло циклу, потім перевіряється умова (вичислюється деякий логічний вираз), який йде за службовим словом *UNTIL*. У випадку, коли результат *FALSE*, виконується тіло циклу ще раз. В протилежному випадку управління передається наступному оператору, який розташований відразу після циклу.

Оператор *REPEAT* відрізняється від оператора *WHILE* тим, що умова перевіряється після виконання циклу, а також тим, що цикл виконується, доки умова рівна *FALSE* (а не *TRUE*). Якщо ж умова рівна *TRUE* до початку циклу, тіло циклу виконується один раз. У випадку, коли умова не змінюється в циклі і рівна *FALSE*, цикл буде нескінченним. Отже, в операторі *WHILE* цикл виконується при істинності умови, а цикл *REPEAT* – поки умова хибна. Ще одна відмінна риса – для циклу *REPEAT* при записі в тілі циклу декількох операторів немає необхідності записувати складовий оператор *begin…end*, оскільки цей оператор сам виступає в ролі складового: *repeat…until*.

Використовуючи оператор *WHILE* і оператору *REPEAT*, необхідно зважати на те, що змінні чи вирази, які записані в умові циклу – між словами *WHILE* і *DO* та після *UNTIL*, в тілі циклу повинні змінювати своє значення, в іншому випадку отримаємо “вічний” цикл.

Під час складання програм з циклами потрібно проаналізувати технічну постановку задачі та вибрати такий опе-

ратор циклу, який оптимально підходить для розв'язку. Звичайно, можна скористатися будь-яким циклом для будь-якої задачі циклічного обчислювального процесу, але саме один з операторів підходить найкраще для вирішення поставленого завдання.

Якщо до виконання циклу відома кількість повторень – найкраще використовувати цикл *FOR*. Якщо кількість повторень невідома, необхідно проаналізувати інший фактор: чи обов'язково потрібно хоч раз виконати тіло циклу, чи, можливо, цикл може ні разу не виконуватися. В першому випадку треба застосовувати *REPEAT*, в другому – *WHILE*.

Завдання 1

Скласти блок-схему алгоритму і написати програму обчислення середньомісячного прибутку фірми за рік відповідно до значень щомісячних прибутків, які вводяться з клавіатури. Введення та виведення даних оформити у вигляді таблиці.

Методичні рекомендації

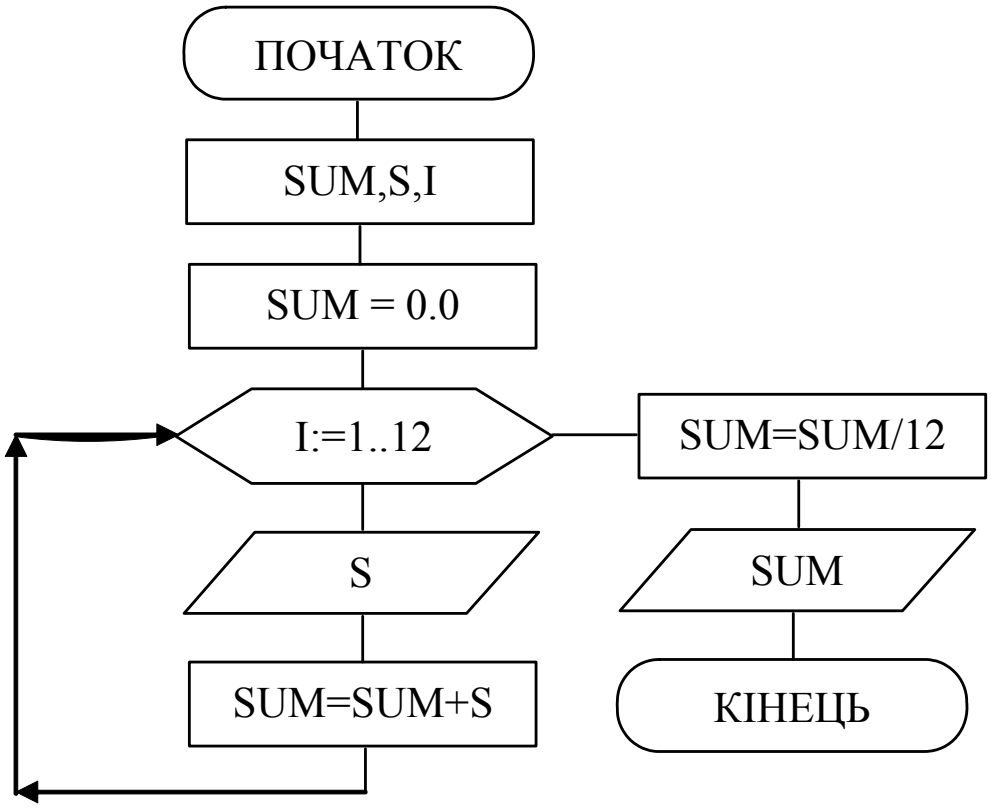
1. Проаналізувати поставлене завдання.

   Для реалізації поставленого завдання потрібно по черзі зчитати прибуток за кожен місяць та просумувати його. Потім отриману загальну суму розділити на кількість місяців – 12. Щоб здійснити таке обчислення доцільно використати цикл *FOR*, оскільки наперед відома кількість повторень, тобто скільки раз потрібно зчитати суму прибутку в місяць та додати до загальної суми (за рік). Так як рік має 12 місяців, то і кількість повторень також 12. Для оформлення діалогу програми у вигляді таблиці необхідно здійснити такий зовнішній вигляд (лінії створювати за допомогою знаків мінус, а їх перетин – за допомогою знаків плюс):

Місяць	**Сума прибутку**
1	
2	
3	
……	……
11	
12	

2. Скласти блок-схему розв'язку задачі.

   Для заданої функції блок-схема задачі з використанням оператору циклу FOR, буде мати такий вигляд:



3. Завантажити інтегроване середовище Borland Pascal.
4. Написати програму реалізації розв'язку задачі з використанням циклу FOR.

   Відповідно до блок-схеми реалізації даної задачі та структури циклу FOR програма буде мати такий вигляд:

```
{************************************************
 * SumFirm – програма знаходження середньомісячного*
 *           прибутку фірми за зчитаними сумами    *
 *           щомісячних прибутків                  *
 * Copyright (c) Шищук В.В. гр.ІС-04-1, 17/10/2004 *
 ************************************************}
PROGRAM SumFirm;
CONST
    line = '+----------+----------------+;'
VAR
    sum,s: real;        {sum - загальна сума,
                         s - сума за місяць}
    i: integer;         {змінна циклу}
BEGIN
    writeln('СЕРЕДНЬОМІСЯЧНИЙ ПРИБУТОК');
    sum:=0.0;
    writeln('Введіть суми прибутків за всі місяці:');
    writeln(line);      {лінія-роздільник}
    writeln('|  Місяць  |  Сума прибутку  |');
    for i:=1 to 12 do
         begin                  {цикл}
            writeln(line);
            write('|',i:6,'|   ':8); readln(s);
            sum:=sum+s;
         end;
    writeln(str);
    sum:=sum/12;
    writeln;
    writeln('Середньомісячний прибуток ', sum:11:3);
    write('Для завершення натисніть <Enter>:');
    readln
END.
```

5. Виконати компіляцію, виправити помилки та запустити програму на виконання.
6. Протестувати програму, вводячи різні значення аргументу.
7. Зберегти написану програму в персональну папку.

Завдання 2

Скласти блок-схему алгоритму і написати програму обчислення середньомісячного прибутку фірми за деякий період, межі якого (номери місяців) вводяться користувачем. Передбачити перевірку коректності вводу номера місяця. Значення прибутку за кожен місяць вводиться з клавіатури. Введення та виведення даних оформити у вигляді таблиці.

Методичні рекомендації

1. Проаналізувати індивідуальне завдання.

   На відміну від попередньої, в даній задачі наперед невідомо, скільки разів потрібно зчитувати суму за місяць та сумувати загальну. В таких випадках використовують цикли з передумовою або післяумовою. Спочатку в програмі потрібно зчитати номер початкового та кінцевого місяця, потім – перевірити, чи зчитані значення належать проміжку [1;12], якщо ні – вивести повідомлення, якщо так – розпочати цикл. Цикл буде повторюватися доки, починаючи з початкового, не дійдемо до кінцевого місяця, до того ж, у випадку, коли кінцевий менший за початковий, цикл жодного разу не повинен виконуватися. Отже, для вирішення цієї задачі доцільно застосувати *WHILE*.

2. Написати програму реалізації розв'язку задачі.

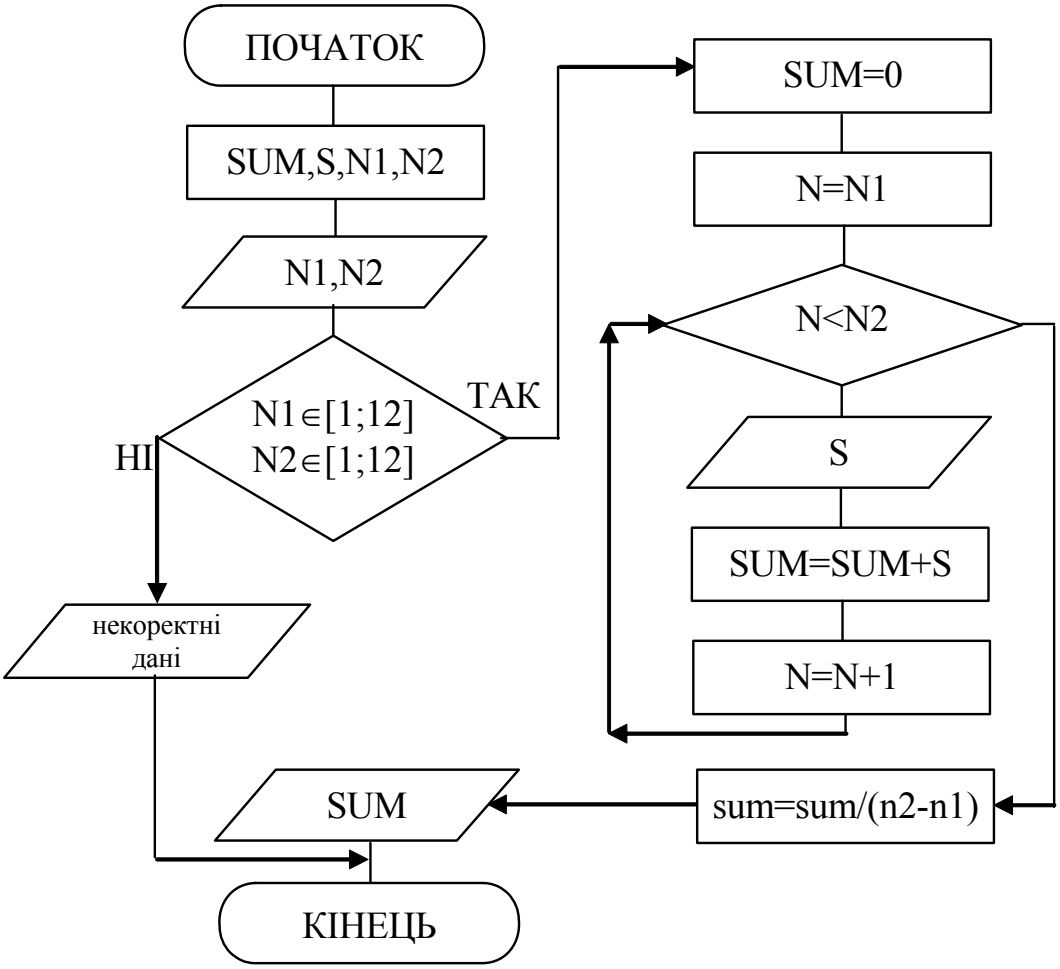
   Перед циклом потрібно записати умовний оператор на перевірку входження до потрібного інтервалу зчитаних номерів місяців:

```
IF (n1>=1) AND (n1<=12) AND (n2>=1)
    AND (n2<=12) THEN
        … … …
```

   Потім записати цикл WHILE з умовою поки початковий місяць менший за кінцевий.

```
WHILE (n1<n2) DO
        … … …
```

При чому значення n2 в тілі циклу повинно збільшуватися на 1. Для такого завдання складемо блок-схему:



За складеною блок-схемою алгоритму розв'язку поставленої задачі записуємо код програми, який буде матиме такий вигляд:

```
{*************************************************
 * SSum - програма знаходження середньомісячного   *
 *        прибутку за вказаний період              *
 * Copyright (c) Шищук В.В. гр.ІС-04-1, 17/10/04   *
 *************************************************}
PROGRAM SSum;
CONST
    line = '+----------+----------------+;'
VAR
    sum,s: real;        {sum - загальна сума,
                         s - сума за місяць}
```

```
    n1,n2                {n1 – початковий місяць
                          n2 – кінцевий місяць}
    n                    {n – для нарощування місяця}
BEGIN
    writeln('СЕРЕДНЬОМІСЯЧНИЙ ПРИБУТОК ЗА ПЕРІОД');
    write('Введіть початковий місяць:');
    readln(n1);
    write('Введіть початковий місяць:');
    readln(n2);
    if (n1>=1) and (n1<=12) and (n2>=1) and (n2<=12)
    then
       begin
          writeln('Суми прибутків за місяці:');
          writeln(line);        {лінія-роздільник}
          writeln('|  Місяць  |  Сума прибутку  |');
          writeln(line);
          n:=n1;           {поточний місяць – початок}
          sum:=0.0;
          while (n<=n2) do {доки не досягне кінцев.}
             begin
                write('|',x:6,'|   ':8); readln(s);
                writeln(line);
                sum:=sum+s;
                n:=n+1;
             end;
          sum:=sum/(n2-n1);
          writeln;
          writeln('Середньомісячний прибуток ',
                   sum:11:3);
       end
    else
       writeln('Введені некоректні номери місяців');
    write('Для завершення натисніть <Enter>:');
    readln
END.
```

3. Протестувати програму.
4. Зберегти створену програму в персональну папку.

Завдання 3

Скласти блок-схему алгоритму і написати програму обчислення кількості місяців, починаючи з першого, загальна сума прибутку за які у фірми досягне 100 тис. грн. Якщо такої суми не буде досягнуто і до 12-го місяця – вивести відповідне повідомлення. Значення прибутку за кожен місяць вводиться з клавіатури. Введення та виведення даних оформити у вигляді таблиці.

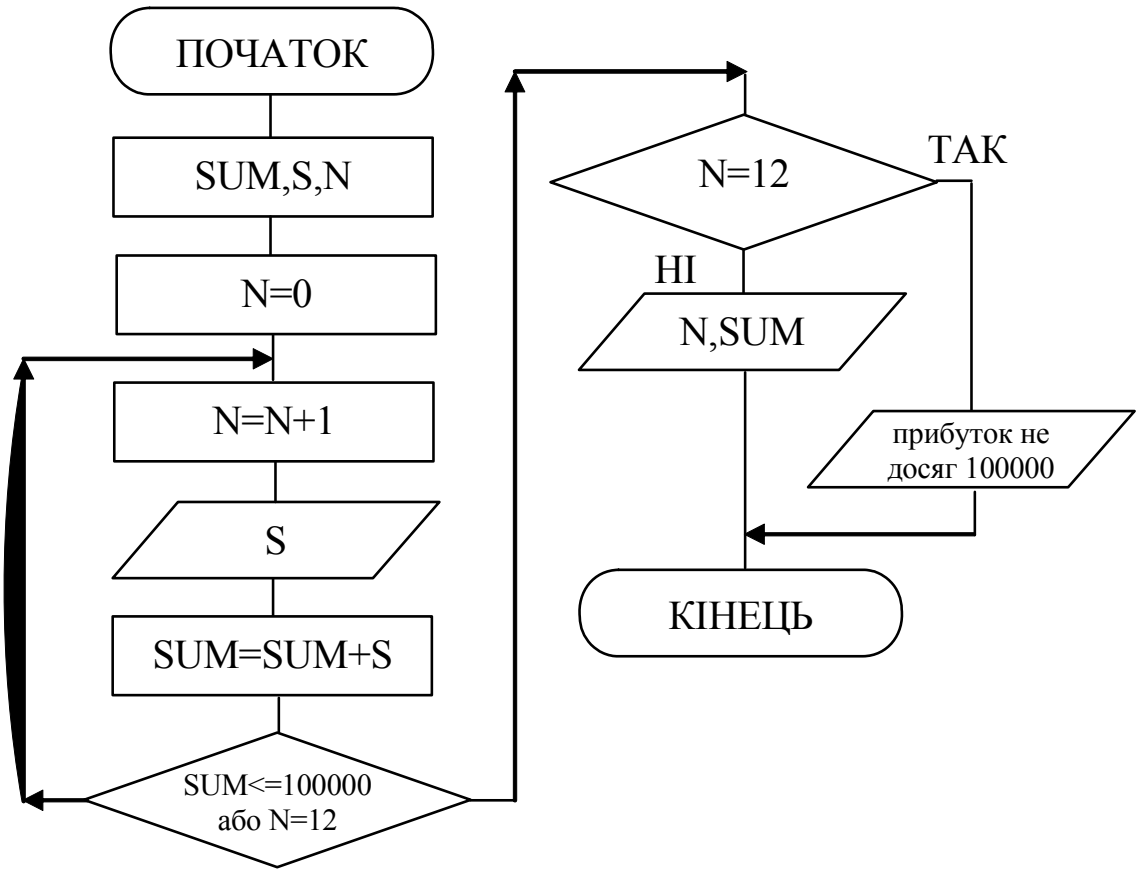
Методичні рекомендації

1. Проаналізувати індивідуальне завдання.

   Для вирішення цієї задачі потрібно скористатися оператором циклу з післяумовою, оскільки, по-перше, кількість повторень наперед невідома, по-друге, потрібно спочатку просумувати значення прибутку, а потім порівняти отриману суму з 100 тис. і так повторювати доки сума не досягне зазначеної величини або доки кількість місяців не досягне 12. В другому випадку потрібно вивести на екран повідомлення про те, що ця фірма за рік не отримала такого прибутку.

2. Написати програму реалізації розв'язку задачі з використанням циклу `REPEAT`.

   Умову циклу потрібно записати так, що її невиконання приведе до повторення тіла циклу. В цьому випадку запишемо таку складову `(sum>=100000) or (n=12)` – сума перевищила задану величину або перебрали всі 12 місяців; справдження будь-якої частини приведе до завершення циклу. Блок-схема буде мати такий вигляд:



Код програми реалізації створеної блок-схеми буде мати такий вигляд:

```
{*************************************************
 * CountMounth – програма знаходження кількості    *
 *               місяців, загальна сума прибутку   *
 *               яких досягає 100 000 грн.         *
 * Copyright (c) Шищук В.В. гр.ІС-04-1, 17/10/2004 *
 *************************************************}
PROGRAM CountMounth;
CONST
    line = '+----------+-----------------+;'
VAR
    sum,s: real;        {sum - загальна сума,
                         s - сума за місяць}
    n: integer;         {кількість місяців}
BEGIN
    writeln('КІЛЬКІСТЬ МІСЯЦІВ ДО 100 000 грн.');
    n:=0;
    sum:=0.0;
    writeln('Введіть суми прибутків за всі місяці:');
```

```
    writeln(line);      {лінія-роздільник}
    writeln('|  Місяць  |  Сума прибутку  |');
    repeat
        writeln(line);
        n:=n+1;
        write('|',n:6,'|   ':8); readln(s);
        sum:=sum+s;
    until (sum>=100000) or (n=12);
    writeln(str);
    writeln;
    if (n=12) then
       writeln('За рік прибуток не досяг 100000 грн.')
    else
       writeln('За ',n,' місяців прибуток становив ',
               sum,' грн.');
    write('Для завершення натисніть <Enter>:');
    readln
END.
```

3. Протестувати програму.
4. Зберегти створену програму в персональну папку.
5. Оформити звіт про виконану роботу (три програми).

Завдання 4

Скласти блок-схему алгоритму та написати програму знаходження найменшого номера n члена послідовності $(-1)^n \frac{3^n n!}{(2n-1)!}$, $n = 1,2,3...$, для якого різниця між знайденим та попереднім членами послідовності за модулем не перевищує заданої точності, тобто виконується умова $\left|a_n - a_{n-1}\right| < \varepsilon$, де $\varepsilon = 0.000001$. Вивести на екран всі елементи послідовності до n-го включно.

Методичні рекомендації

1. Проаналізувати технічну постановку задачі та вибрати варіант її розв'язку.

   Для знаходження елемента нескінченного ряду, для якого виконується умова $\left|a_n - a_{n-1}\right| < \varepsilon$ необхідно скористатися циклом, в якому будемо порівнювати поточний елемент ряду з попереднім. Використання циклу `FOR` недоцільне (хоча можливе разом з оператором `IF - GOTO`), бо кількість проходжень циклу невідома. При цьому хоча б один раз, але порівняння (першого з другим) повинно відбутися. За такої умови краще використовувати оператор циклу з післяумовою – `REPEAT`. Перед циклом за основу потрібно взяти перший елемент (той, який буде відніматися). Потім в циклі виконати приріст номера n та обчислити наступний елемент. І так повторювати доки різниця між поточним та базовим (попереднім, який на кожному кроці буде змінюватися) не стане меншою за вказану точність. Тобто, при досягненні наступним членом значення різниці між елементами меншої, ніж 0.000001, потрібно вийти з циклу.

2. Скласти загальний план розв'язку задачі, звернути увагу на окремі моменти, такі, як знаходження факторіалу та піднесення до степеня числа.

   Спочатку необхідно записати вираз 3^n за синтаксисом мови Pascal, але в цій мові не існує функції піднесення до степеня, тоді, за правилами математики, якщо число додатне, піднесення його до степеня можна записати в такому вигляді: $e^{n\ln 3}$. Такий вираз вже можна записати на мові Pascal, оскільки є стандартна функція експоненти:

```
EXP(n*ln(3))
```

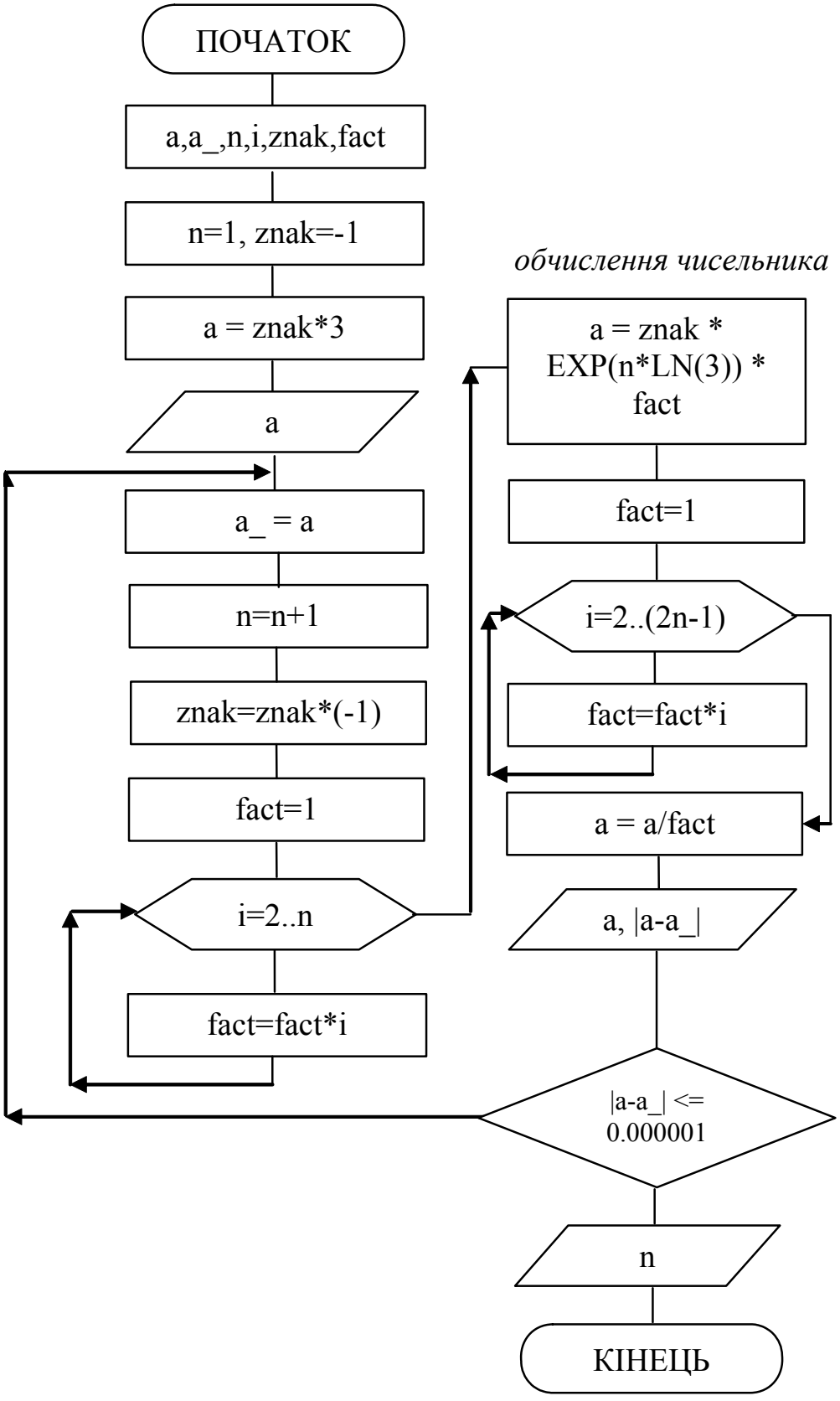
Крім піднесення до степеня формула загального члена ряду містить факторіал числа, але стандартної функції в мові Pascal, яка реалізує знаходження факторіала, також немає. Тому необхідно самостійно написати блок програми, який виконує цю задачу. Для цього доцільно скористатися змінною, яку за циклом щоразу множити на число, більше на одиницю від попереднього. Тут можна застосувати оператор циклу *FOR*, адже кількість повторень наперед відома. Код програми для знаходження факторіала буде мати такий вигляд (значення факторіала обчислюється в змінній *FACT*):

```
FACT:=1;
FOR i:=2 TO N DO
    FACT:=FACT*i;
```

де *N* і є те число, факторіал якого потрібно обчислити.

3. Скласти блок-схему алгоритму розв'язку задачі.

Для знаходження номера члена ряду, для якого виконується зазначена умова, використаємо цикл з післяумовою, в середині якого потрібно по черзі записати два рази цикл *FOR* перший – для знаходження факторіала числа n, другий – для знаходження факторіала числа $2n-1$ відповідно до формули загального члена ряду. Отже, для даної задачі блок–схема алгоритму буде мати такий вигляд:



4. Завантажити інтегроване середовище Borland Pascal.

5. Написати програму реалізації задачі.

   Відповідно до блок-схеми реалізації даної задачі програма буде мати такий вигляд:

```
{************************************************
 * RowElem – програма знаходження елементу нескінч. *
 *           ряду по заданій формулі заг. члена та  *
 *           точності різниці між сусідніми елемент.*
 * Copyright (c) Шищук В.В. гр.ІС-04-1, 22/10/04    *
 ************************************************}
PROGRAM RowElem;
CONST
    line = '--------------------------';
VAR
    a,a_: real;         {члени ряду}
    i,n,znak,fact: integer;
BEGIN
    writeln('ЕЛЕМЕНТ НЕСКІНЧЕННОГО РЯДУ');
    n:=1;
    znak:=-1;
    a:=znak*3;          {перший елемент ряду}
    wtieln(line)
    writeln('N = ',n, ' елемент = ',a:12:10);
    repeat
       a_:=a;           {попередній елемент}
       n:=n+1;
       znak:=znak*(-1);
       fact:=1;
       for i:=2 to n do
         fact:=fact*i;      {знаходження факторіала}
       a:=znak*exp(n*ln(3))*fact;       {чисельник}
       fact:=1;
       for i:=2 to (2*n-1) do
         fact:=fact*i;      {знаходження факторіала}
       a:=a/fact;   {значення члену ряду}
       wtieln(line)
       write('N = ',n, ' елемент = ',a:12:10);
       writeln('Різниця = ',abs(a-a_):12:10);
    until abs(a-a_)<=0.000001;
    writeln; writeln('Шуканий елемент - ',n, '-й');
    write('Для завершення натисніть <Enter>:');
    readln
END.
```

6. Виконати компіляцію та запустити програму на виконання.
7. Зберегти написану програму в персональну папку.
8. Оформити звіт про виконану роботу.

Індивідуальні завдання:

рівень А

Зчитати з клавіатури необхідні дані (відповідно до варіанту) та виконати необхідні дії.

Варіант 1.

Знайти суму перших N натуральних чисел.

Варіант 2.

Знайти необхідну кількість натуральних чисел, починаючи з одиниці, щоб їх сума дорівнювала N^2.

Варіант 3.

Підрахувати суму натуральних чисел від одиниці до числа, квадрат якого більше за N.

Варіант 4.

Знайти найбільший спільний дільник двох натуральних чисел.

Варіант 5.

Знайти факторіал числа N.

Варіант 6.

Знайти суму чисел, які послідовно вводяться з клавіатури. Підрахунок зупинити під час вводу наступного числа, рівного нулю.

Варіант 7.

Перевірити, чи можна задане число N записати у вигляді квадрата простого числа.

Варіант 8.

Визначити, чи задане число N є простим.

Варіант 9.

Знайти всі прості числа, що знаходяться в заданому діапазоні.

Варіант 10.

Визначити, чи можна записати задане число N у вигляді добутку двох простих чисел.

Варіант 11.

Знайти суму дільників натурального числа N.

Варіант 12.

Знайти наступне просте число відносно іншого заданого простого числа N.

Варіант 13.

Записати програму вгадування попередньо заданого числа. Вихід з програми здійснити лише при вгадуванні, при цьому необхідно вказати кількість спроб.

Варіант 14.

Записати програму розкладу натурального числа N на прості множники.

Варіант 15.

Знайти середнє арифметичне чисел, послідовно введених з клавіатури. Підрахунок зупинити після вводу наступного числа, рівного нулю.

рівень Б

Знайти суму перших *N* членів ряду для значення *X*, які зчитуються з клавіатури, а також суму членів ряду з точністю *E* для елемента ряду, заданої користувачем (до елемента, значення якого менше вказаної точності). Формула загального члена ряду має такий вигляд:

Варіант 1.

$$\frac{(-1)^i x^{2i+5}}{(2i+5)!}$$

Варіант 2.

$$\frac{(-1)^i (2x)^{i+7}}{(2i+1)!}$$

Варіант 3.

$$\frac{(-1)^i (i^2+1)}{i!}\left(\frac{x}{2}\right)^2$$

Варіант 4.

$$\frac{(-1)^i (2x)^{2i+3}}{(2i+3)!}$$

Варіант 5.

$$\frac{(-1)^i x^{2i+5}}{(3i+1)!}$$

Варіант 6.

$$\frac{(-1)^i (2x)^{2i}}{(2i+1)!}$$

Варіант 7.

$$\frac{(-1)^i x^{4i+2}}{(2i+1)!}$$

Варіант 8.

$$\frac{(-1)^i x^i}{(i+4)!}$$

Варіант 9.

$$\frac{(-1)^i (3x)^i}{(i+2)!}$$

Варіант 10.

$$\frac{(-1)^i (2x)^{2i}}{(3i+1)!}$$

Варіант 11.

$$\frac{(-1)^i \ln^i 3}{i!}x^i$$

Варіант 12.

$$\frac{(-1)^i x^{3i+2}}{(2i+5)!}$$

Варіант 13.

$$\frac{(-1)^i (2x)^{2i+1}}{(i+3)!}$$

Варіант 14.

$$\frac{(-1)^i x^{i+3}}{(2i+3)!}$$

Варіант 15.

$$\frac{(-1)^i x^{4i+2}}{(2i+1)!}$$

рівень В

Вивести на екран у вигляді таблиці значення заданої функції для аргументу на інтервалі від $X_{поч}$ $X_{кін}$ та з постійним кроком H, значення яких зчитати з клавіатури. Підрахувати середнє арифметичне виведених значень функції, а також найбільше та найменше її значення на зазначеному проміжку.

Варіант 1.

$$y=\begin{cases}2x^2+4x-5, \text{ якщо } x<0\\ \sqrt{x}+27x, \text{ якщо } 0\le x<10\\ \dfrac{2x}{5}, \text{ в інших випадках}\end{cases}$$

Варіант 2.

$$y=\begin{cases}\dfrac{-5-x}{5x}, & \text{якщо} \quad x\le 1\\ x^3+5x^2-x+4, & \text{якщо} \quad 1<x<5\\ \dfrac{x^2+6}{-\sqrt{x}}, & \text{в інших випадках}\end{cases}$$

Варіант 3.

$$y=\begin{cases}x^4+2x^3, \text{ якщо } x<0\\ \dfrac{x-1}{2x}, \text{ якщо } 0\le x<10\\ \dfrac{4x}{5-3x}, \text{ в інших випадках}\end{cases}$$

Варіант 4.

$$y=\begin{cases}\dfrac{1}{x^3-2x}, & \text{якщо } x\le 2\\ \sqrt[4]{x-2}, & \text{якщо } 2<x<15\\ \dfrac{x-1}{x+2}, & \text{в інших випадках}\end{cases}$$

Варіант 5.

$$y=\begin{cases}\dfrac{-2}{\sqrt{x+1}}, & \text{якщо } x>-1\\ \dfrac{-x}{\sqrt{2}}, & \text{якщо } -10<x\le -1\\ ax^2, & \text{в інших випадках}\end{cases}$$

Варіант 6.

$$y=\begin{cases}-7x+4, & \text{якщо } x\le 0\\ \dfrac{3-x}{2x}, & \text{якщо } 0<x<5\\ \dfrac{x}{2}+\dfrac{\sqrt{x}}{x+5}, & \text{в інших випадках}\end{cases}$$

Варіант 7.

$$y=\begin{cases}(2x+4)^4, & \text{якщо } x>-2\\ 3x^2+\dfrac{2}{x}, & \text{якщо } x>0\\ x^3-\dfrac{x^2}{\sqrt{5}}, & \text{в інших випадках}\end{cases}$$

Варіант 8.

$$y=\begin{cases}\dfrac{x}{-\sqrt{2x}}, \text{ якщо } x>0\\ \dfrac{-x}{2}+\dfrac{2}{x}, \text{ якщо } x<0\\ \dfrac{x^2}{-3}, \text{ в інших випадках}\end{cases}$$

Варіант 9.

$$y=\begin{cases}\dfrac{x-5}{x+1}, \text{ якщо } x<0\\ \dfrac{-\sqrt[4]{x}}{25x^2}, \text{ якщо } x>0\\ 1+\dfrac{\sqrt{x}}{(x+1)^2}, \text{ в інших випадках}\end{cases}$$

Варіант 10.

$$y=\begin{cases}\dfrac{x-4}{5-x^4}+4\sqrt{|x|}, \text{ якщо } x<0\\ tgx, \text{ якщо } 0\le x<1\\ x+10, \text{ в інших випадках}\end{cases}$$

Варіант 11.

$$y=\begin{cases}-2x^2-8, \text{ якщо } -5<x\le 0\\ \dfrac{2}{-2x^2-x}, \text{ якщо } x>0\\ x^2\sqrt{2}, \text{ в інших випадках}\end{cases}$$

Варіант 12.

$$y=\begin{cases}\dfrac{3x}{5-x}, \text{ якщо } x\geq 3\\ \left|2-x^2\right|, \text{ якщо } x<10\\ 3x+\dfrac{2}{x^3}, \text{ в інших випадках}\end{cases}$$

Варіант 13.

$$y=\begin{cases}\dfrac{1}{2x}-4, \text{ якщо } x\geq 1\\ x^3-4x, \text{ якщо } -5<x\leq 0\\ \dfrac{10x}{2-x^2}, \text{ в інших випадках}\end{cases}$$

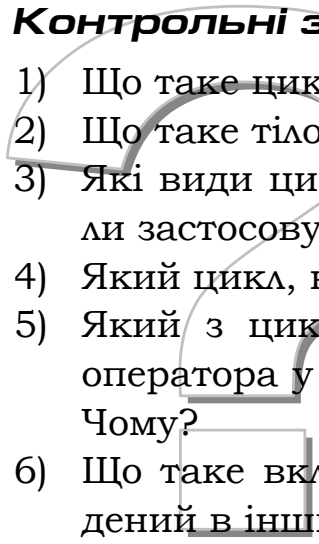
Варіант 14.

$$y=\begin{cases}3x^3-\dfrac{2x}{\sqrt{x}}, \text{ якщо } -10<x\leq 10\\ -2x-5, \text{ якщо } x>15\\ \dfrac{4}{(2-x^3)^2}, \text{ в інших випадках}\end{cases}$$

Варіант 15.

$$y=\begin{cases}2-\dfrac{x^2}{10-2x}, \text{ якщо } -1<x\leq 1\\ 2x^2+4x-5, \text{ якщо } x<-1\\ \dfrac{|x|-5}{3x}, \text{ в інших випадках}\end{cases}$$

Контрольні запитання:



1) Що таке цикл і як він працює?
2) Що таке тіло циклу? Коли відбувається вихід з нього?
3) Які види циклів існують і чим вони відрізняються? Коли застосовують кожен з циклів?
4) Який цикл, на вашу думку, найбільш зручний і чому?
5) Який з циклів не потребує використання складового оператора у випадку декількох операторів в тілі циклу? Чому?
6) Що таке вкладений цикл? Який цикл може бути вкладений в інший? Наведіть приклад.
7) Чи завжди можна замінити один цикл іншим? Чому?
8) Що таке лічильник? Що таке прапорець? В яких випадках їх використовують?
9) Що таке цикл з передумовою і післяумовою? Чим вони відрізняються?
10) Яка необхідна дія повинна бути записана в тілі циклу з передумовою і тілі циклу з післяумовою?
11) Вкажіть які з нижченаведених записів будуть циклами:

```
WHILE x>0 DO x:=x+1;
WHILE x<0 DO x:=x-100;
WHILE x<0 DO x:=x+10;
WHILE x>y DO BEGIN x:=x-1; y:=y+1 END;
```

12) Що таке булевий вираз? Яку роль він виконує в циклах? Наведіть приклади.
13) Які різновиди циклу `FOR` існують?
14) Що таке нескінченний цикл? Коли трапляються такі випадки?
15) Чи існують обмеження для застосування вкладених циклів? Якщо так, то які?