## Лабораторна робота №2.

**Тема: "Використання основних операторів мови С"**

**Мета:** Отримання навичок у виборі й використанні операторів С; знайомство з ітераційними процесами.

## 1. Короткі теоретичні відомості

Оператори керування роботою програми називають керуючими конструкціями програми. До них належать:

- складені оператори;
- оператори вибору;
- оператори циклів;
- оператори переходу.

### 1.1. Складені оператори

До складених операторів відносять власне складені оператори й блоки. В обох випадках це послідовність операторів, взятих у фігурні дужки. Блок відрізняється від складеного оператора наявністю визначень у тілі блоку. Наприклад:

```
{
n++;                    це складений оператор
summa+=n;
}

{
int n=0;
n++;                    це блок
summa+=n;
}
```

## 1.2. Оператори вибору

Оператори вибору - це умовний оператор і перемикач. Умовний оператор має повну й скорочену форму.

if ( <вираз-умова> ) <оператор>; //скорочена форма

В ролі <виразу-умови> можуть використовуватися арифметичний вираз, відношення й логічний вираз. Якщо значення <виразу-умови> відрізняється від нуля (тобто істина), то виконується оператор. Наприклад:

if (x<y&&x<z)min=x;

if ( <вираз-умова> ) <оператор1>; //повна форма

else <оператор2>;

Якщо значення <виразу-умови> відрізняється від нуля, то виконується оператор1, при нульовому значенні <виразу-умови> виконується оператор2. Наприклад:

```
if (d>=0)
{
x1=(-b-sqrt(d))/(2*a);
x2=(-b+sqrt(d))/(2*a);
cout<< “\nx1=”<<x1<<“x2=”<<x2;
}
else cout<<“\nрозв’язку немає”;
```

Перемикач визначає множинний вибір.

```
switch (<вираз>)
{
case <константа1> : <оператор1 >;
case <константа2> : <оператор2 >;
. . . . . .
default: <оператори>;
```

При виконанні оператора switch, обчислюється вираз, записаний після switch і його значення послідовно порівнюється з константами, які записані слідом за case. При першому ж співпаданні виконуються оператори позначені даною міткою. Якщо виконані оператори не містять оператора переходу, то

далі виконуються оператори всіх наступних варіантів, поки не з'явиться оператор переходу або не закінчиться перемикач. Якщо значення виразу, записаного після switch не співпало з жодною константою, то виконуються оператори, які стоять за міткою default. Мітка default може бути відсутньою.

Приклад:

```
switch ( number )
{
case 1 :  cout<< “число=1”;break;
case 2 :  cout<< “2 * 2”<<number * number;
case 3 :  cout<< “3 * 3”<<number * number; break;
case 4 :  cout<< number<<“- це чудове число”; break;
default: cout<< “Кінець роботи програми”;
}
```

## 1.3. Оператори циклів

1. Цикл із предумовою:

while (<вираз-умова>)

<тіло_циклу> ;

В ролі <виразу-умови> найчастіше використовується відношення або логічний вираз. Якщо він істинний, тобто не дорівнює 0, то тіло циклу виконується доти поки <вираз-умова> не стане помилковим.

2. Цикл із післяумовою:

do

<тіло_циклу>;

while (<вираз-умова>);

Тіло циклу виконується доти, поки <вираз-умова> істинний.

3. Цикл із параметром:

for ( <вираз_1>;<вираз-умова>;<вираз_3>)

тіло_циклу;

<Вираз_1> й <вираз_3> можуть складатися з декількох виразів, розділених комами. <Вираз_1> - задає початкові умови для циклу (ініціалізація). <Вираз-

умова>  визначає умову виконання циклу, якщо він не дорівнює 0, цикл виконується, а потім обчислюється значення  <вираз_3>. <Вираз_3> - задає зміну параметра циклу або інших змінних (корекція). Цикл триває доти, поки <вираз-умова> не стане дорівнює 0. Будь-який вираз може бути відсутнім, але розділювачі  « ; » повинні бути обов'язково.

Приклади використання циклу з параметром.

1) Зменшення параметра:

```
for ( n=10; n>0; n--)
{ <тіло циклу>};
```

2) Зміна кроку коректування:

```
for ( n=2; n>60; n+=13)
{ <тіло циклу>};
```

3) Можливість перевіряти умову, яка відрізняється від умови, що накладається на число ітерацій:

```
for ( num=1;num*num*num<216; num++)
{ <тіло циклу>};
```

4) Корекція може здійснюватися не тільки за допомогою додавання або віднімання:

```
for ( d=100.0; d<150.0;d*=1.1)
{ <тіло циклу>};
for (x=1;y<=75;y=5*(x++)+10)
{ <тіло циклу>};
```

5) Можна використовувати декілька ініціалізуючих або коригувальних виразів:

```
for (x=1, y=0; x<10;x++;y+=x);
```

## 1.4. Оператори переходу

Оператори переходу виконують безумовну передачу керування.

1) break - оператор переривання циклу.

```
{
< оператори>
if (<вираз_умова>) break;
```

<оператори>

}

Тобто оператор break доцільно використовувати, коли умову продовження ітерацій потрібно перевіряти в середині циклу.

Приклад:

```
// шукає суму чисел, що вводяться з клавіатури доти, поки не буде введено 100 чисел або 0
for(s=0, i=1; i<100;i++)
{
cin>>x;
if( x==0) break; // якщо ввели 0, то підсумовування закінчується
s+=x;
}
```

2) continue - перехід до наступної ітерації циклу. Він використовується, коли тіло циклу містить розгалуження.

Приклад:

```
//шукає кількість і суму додатніх чисел
for( k=0,s=0,x=1;x!=0;)
{
cin>>x;
if (x<=0) continue;
k++;s+=x;
}
```

## 2. Постановка завдання

Використовуючи оператор циклу, знайти суму елементів, зазначених у конкретному варіанті. Результат надрукувати, надавши відповідний заголовком.

## 3. Варіанти

1) Знайти суму цілих додатніх чисел, кратних 3 і менших 200.

2) Знайти суму цілих додатніх парних чисел, менших 100.

3) Знайти суму цілих додатніх непарних чисел, менших 200.

4) Знайти суму цілих додатніх чисел, більших 20, менших 100 і кратних 3

5) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \frac{(-1)^{n-1}}{n^n}$$

6) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \frac{1}{2^n} + \frac{1}{3^n}$$

7) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \frac{1}{((3n-2)(3n+1))}$$

8) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \frac{(2n-1)}{2^n}$$

9) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \frac{10^n}{n!}$$

10) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \frac{n!}{(2n)!}$$

11) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \frac{n!}{n^n}$$

12) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \frac{2^n n!}{n^n}$$

13) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \frac{3^n n!}{(3n)!}$$

14) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \frac{n!}{3n^n}$$

15) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \frac{(n!)^2}{2^{n^2}}$$

16) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \lg(n!)e^{-n\sqrt{n}}$$

17) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = 10^{-n}(n-1)!$$

18) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \frac{n^3}{(3n-3)!}$$

19) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = \frac{n}{(n-1)^2}$$

20) Знайти суму ряду з точністю ε=0.0001, загальний член якого

$$a_n = e^n \cdot 100^{-n^2}$$

21) Знайти суму 13 членів ряду, у якому

$$a_n = \frac{\ln(n!)}{n^2}$$

22) Знайти суму 15 членів ряду, у якому

$$a_n = \frac{n^{\ln n}}{(\ln n)^n}$$

23) Знайти суму 10 членів ряду, у якому

$$a_n = \frac{n!}{n^{\sqrt{n}}}$$

24) Знайти суму 9 членів ряду, у якому

$$a_n = e^{-\sqrt{n}}$$

25) Знайти суму 7 членів ряду, у якому

$$a_n = n^2 e^{-\sqrt{n}}$$

## 3. Зміст звіту

1. Постановка завдання.

2. Текст програми.

3. Результат розв’язку конкретного варіанту.

## 4. Методичні вказівки

1. При визначенні суми членів ряду варто використовувати рекурентну формулу для отримання наступного члена ряду.

Наприклад, потрібно знайти суму ряду з точністю ε=0.0001, загальний член якого $a_n = \frac{2(n!)^2}{(3(2n)!)!}$.

Для одержання рекурентної формули обчислимо відношення:

$$\frac{a_{n+1}}{a_n} = \frac{2((n+1)!)^2 \cdot 3(2n)!}{3(2n+2)! \cdot 2(n!)^2} = \frac{n+1}{2(2n+1)},$$

звідки:

$$a_{n+1} = a_n \cdot \frac{(n+1)}{2(2n+1)}.$$

2. При складанні програми вважати, що точність досягнута, якщо $a_n < \varepsilon$