Міністерство освіти і науки України Національний університет “Львівська Політехніка”

Лабораторна робота №12

з дисципліни «Програмування частина 2»

Виконав:

Студент групи АП-11

Іщенко Василь

Прийняв:

Чайковський І.Б.

Львів – 2024

Тема: Оператори циклу

Мета роботи: ознайомитися з особливостями функціонування операторів циклу та навчитись їх використовувати у процесі програмування.

Теоретичні відомості

У мові С, як і в інших мовах програмування, оператори циклу служать для багаторазового виконання послідовності операторів до тих пір, поки виконується деяка умова. Умова може бути встановлена заздалегідь (як в операторі for) або змінюватися при виконанні тіла циклу (як в while або do-while). Цикл for. У всіх процедурних мовах програмування цикли for дуже схожі. Однак в С цей цикл особливо гнучкий і потужний. Загальна форма оператора for наступна:

for (необов'язковий вираз 1;необов'язковий вираз 2;необов'язковий вираз 3) оператор;

Цикл while. Синтаксис данного циклу:

while (вираз) оператор;

Виконання оператора повторюється, доки значення виразу залишається ненульовим. Перевірка виконується перед кожним виконанням оператора while. Вираз може бути арифметичного або логічного типів. Цикл може бути виконаний один раз, декілька разів або не виконуватися жодного разу.

Цикл do while. В циклі do while перевірка умови здійснюється після виконання тіла циклу. Синтаксис циклу:

do оператор while (вираз);

Виконання оператора повторюється доти, поки значення виразу залишається ненульовим. Перевірка виконується після кожного виконання оператора do. Вираз може бути арифметичним або логічним. Оператор у циклі do while на відміну від циклу while буде виконуватися хоча б один раз завжди

Приклад 1

```
#include <stdio.h>
int main (void){
   int x;
   for (x = 1; x <= 100; x ++)
   printf ( "% d", x);
   return 0;
}
```

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52
53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100
```

Приклад 2

```
#include<stdio.h>
main(){
   int i=2;
   while (i<=1024){
   i = i*2;
   printf("%d\n",i);}
}
```

```
4
8
16
32
64
128
256
512
1024
2048
```

Приклад 8

```
/* Celsius and Fahrengeit */
/* C=(5/9)*(F-32) */
#include<stdio.h>
#include<conio.h>
main(){
   int fahr,celsius;
   int lower,upper,step;
   lower=0;
   upper=300;
   step=20;
   fahr=lower;
   printf("\n\nCelsius Fahrengeit\n");
   while( fahr <= upper ){
    celsius = 5*(fahr-32)/9;
   printf("%10d\t%8d\n",fahr,celsius);
   fahr=fahr+step;}
   getch();}
```

```
Celsius Fahrengeit
      0        -17
     20         -6
     40          4
     60         15
     80         26
    100         37
    120         48
    140         60
    160         71
    180         82
    200         93
    220        104
    240        115
    260        126
    280        137
    300        148
```

Приклад 9

```
#include <stdio.h>
int main() {
   int n, i, j;
   printf("Введіть розмір трикутника (кількість рядків): ");
   scanf("%d", &n);
   for (i = 0; i < n; i++) {
      for (j = 0; j <= i; j++) {
         printf("* ");
      }
      printf("\n");
   }
   return 0;
}
```

```
Введіть розмір трикутника (кількість рядків): 5
*
* *
* * *
* * * *
* * * * *
```

Приклад 10

```
#include <stdio.h>
int main() {
   unsigned  long total_grains = 0; // Загальна кількість зерен
   unsigned  long grains = 1; // Кількість зерен для поточної клітини
   int squares = 64; // Кількість клітин на шахівниці
   // Обчислення кількості зерен для кожної клітини та загальної кількості
   for (int i = 1; i <= squares; i++) {
      total_grains += grains;
      grains *= 2; // Подвоєння кількості зерен для наступної клітини
   }
```

```
    // Виведення результату
    printf("Загальна кількість зерен, що потрібно видати: %llu\n", total_grains);
    return 0;
}
```

```
Загальна кількість зерен, що потрібно видати: 4294967295
```

Приклад 11

```
#include <stdio.h>
#include <math.h>
int main() {
    int i;
    printf("Число\tКвадрат\tКуб\tКорінь 4-го степеня\n");
    for (i = 1; i <= 20; i++) {
        printf("%d\t%d\t%d\t%.2f\n", i, i * i, i * i * i, pow(i, 0.25));
    }}
```

```
Число   Квадрат Куб     Корінь 4-го степеня
1         1       1       1.00
2         4       8       1.19
3         9       27      1.32
4         16      64      1.41
5         25      125     1.50
6         36      216     1.57
7         49      343     1.63
8         64      512     1.68
9         81      729     1.73
10        100     1000    1.78
11        121     1331    1.82
12        144     1728    1.86
13        169     2197    1.90
14        196     2744    1.93
15        225     3375    1.97
16        256     4096    2.00
17        289     4913    2.03
18        324     5832    2.06
19        361     6859    2.09
20        400     8000    2.11
```

Приклад 12

```
#include <stdio.h>
#include <math.h>
#define PI 3.14159265
#define EXP 2.71828182
int main(void) {
    float y;
    int N = 32;
    float a = INFINITY;     // Початкове значення для мінімуму
    float b = -INFINITY;    // Початкове значення для максимуму
    float res[N];           // Масив для зберігання значень функції
    for (int i = 0; i < N; i++) {
```

```
        y = pow(i, 2) * pow(EXP, (-pow(i, 2) / 100.0)) * sin((2 * PI / N) * i);
        res[i] = y;
        b = fmax(b, y);     // Використовуємо fmax для визначення максимуму
        a = fmin(a, y);     // Використовуємо fmin для визначення мінімуму
        printf("i = %d, y = %f\n", i, y);
    }
    printf("max = %f\n", b);
    printf("min = %f\n", a);
    return 0;
}
```

```
i = 0, y = 0.000000
i = 1, y = 0.193149
i = 2, y = 1.470713
i = 3, y = 4.569777
i = 4, y = 9.640906
i = 5, y = 16.188730
i = 6, y = 23.204479
i = 7, y = 29.441893
i = 8, y = 33.746716
i = 9, y = 35.341129
i = 10, y = 33.987629
i = 11, y = 30.000978
i = 12, y = 24.124784
i = 13, y = 17.324791
i = 14, y = 10.565220
i = 15, y = 4.626533
i = 16, y = 0.000000
i = 17, y = -3.133448
i = 18, y = -4.855909
i = 19, y = -5.425541
i = 20, y = -5.180445
i = 21, y = -4.457038
i = 22, y = -3.535700
i = 23, y = -2.615844
i = 24, y = -1.815040
i = 25, y = -1.183351
i = 26, y = -0.723988
i = 27, y = -0.413587
i = 28, y = -0.218239
i = 29, y = -0.104020
i = 30, y = -0.042504
i = 31, y = -0.012572
max = 35.341129
min = -5.425541
```

Приклад 13 //FLOAT

```
#include <stdio.h>
#include <math.h>
int main(void) {
int i = 0;
float precision = 1.0, a = 1.0 + precision;
for (precision = 1.0; a > 1.0; ++i) {
precision = precision / 2;
a = 1.0 + precision;}
printf("\nЧисло ділень на 2: %6d\n", i);
```

```
printf("Машинний нуль: %e\n", precision);
}
```

```
Число ділень на 2:     24
Машинний нуль: 5.960464e-008
```

```
#include <stdio.h>
#include <math.h>
int main(void) {
int i = 0;
float precision = 1.0;
float a = 1.0 + precision;
while (a > 1.0) {
precision = precision / 2;
a = 1.0 + precision;
++i;}
printf("\nЧисло ділень на 2: %6d\n", i);
printf("Машинний нуль: %e\n", precision);
}
```

```
Число ділень на 2:     24
Машинний нуль: 5.960464e-008
```

```
#include<stdio.h>
#include<math.h>
void main(void){
int i=0;
float precision,a;
precision = 1.0;
do{
precision = precision/2;
a = 1.0 + precision;
++i;}
while(a>1);
printf("\nчисло ділень на 2: %6d\n",i);
printf("машинний нуль: %e\n ",precision);
}
```

```
число ділень на 2:     24
машинний нуль: 5.960464e-008
```

Приклад 14

```
#include <stdio.h>
// Функція для обчислення факторіалу
unsigned long long factorial(int n) {
   unsigned long long fact = 1;
   for (int i = 1; i <= n; i++) {
```

```
        fact *= i;}
    return fact;}
int main() {
    int N = 5;
    int M = 5;
    // Обчислення факторіалів чисел M та N
    unsigned  long fact_M = factorial(M);
    unsigned  long fact_N = factorial(N);
    // Обчислення факторіалу суми M та N
    unsigned  long fact_MN = factorial(M + N);
    // Обчислення значення виразу
    double result = (double)(fact_M + fact_N) / fact_MN;
    printf("Результат: %lf\n", result);
    return 0;
}
```

Результат: 0.000066

Приклад 15

```
#include <stdio.h>
#include <math.h>
#include <windows.h>
float factorial(float n);
float sin_x(float x,float y);
float cos_x(float x,float y);
float exp_x(float x,float y);
int z=0;
void main(){
 SetConsoleCP(65001);
 SetConsoleOutputCP(65001);
 float a = 0.00001;
 float x;
 printf("Введіть значення x в межах 0 ≤ X ≤ π/2:");
 scanf("%f",&x);
 printf("Значення sin(x) за допомогою ітераційного процесу: %f\n", sin_x(x,
a));
 printf("%d\n", z);
 printf("Значення sin(x) за допомогою бібліотечної функції: %f\n\n", sin(x));
 printf("Значення cos(x) за допомогою ітераційного процесу: %f\n", cos_x(x,
a));
 printf("%d\n", z);
 printf("Значення cos(x) за допомогою бібліотечної функції: %f\n\n", cos(x));
 printf("Значення cos(x) за допомогою ітераційного процесу: %f\n", exp_x(x,
a));
 printf("%d\n", z);
 printf("Значення cos(x) за допомогою бібліотечної функції: %f\n", exp(x));}
float factorial(float n){
```

```
  if(n==0){
        return 1;}
  else{
        return n*factorial(n-1);}}
float sin_x(float x,float y){
  float n = 0;
  float X = x;
  float sum = x;
  z=0;
  while (fabs(X)>y){
        n = n+1;
        X = pow(-1,n)*(pow(x,2*n+1)/factorial(2*n+1));
        sum = sum + X;
        z = z+1;}
  return sum;}
float cos_x(float x,float y){
  float n = 0;
  float X = x;
  float sum = 1;
  z=0;
  while (fabs(X)>y){
        n = n+1;
        X = pow(-1,n)*(pow(x,2*n)/factorial(2*n));
        sum = sum + X;
        z = z+1;}
  return sum;}
float exp_x(float x,float y){
  float n = 0;
  float X = x;
  float sum = 1;
  z=0;
  while (fabs(X)>y){
        n = n+1;
        X = (pow(x,n)/factorial(n));
        sum = sum + X;
        z = z+1;}
  return sum;}
```

```
Введіть значення x в межах 0 ≤ X ≤ π/2:1
Значення sin(x) за допомогою ітераційного процесу: 0.841471
4
Значення sin(x) за допомогою бібліотечної функції: 0.841471
Значення cos(x) за допомогою ітераційного процесу: 0.540302
5
Значення cos(x) за допомогою бібліотечної функції: 0.540302
Значення cos(x) за допомогою ітераційного процесу: 2.718282
9
Значення cos(x) за допомогою бібліотечної функції: 2.718282
```

Відповіді на контрольні запитання

1)Призначення операторів циклу:

Оператори циклу використовуються для повторення виконання певних дій або блоку коду доти, доки виконується певна умова.

2) Конструкція оператора циклу while:

Оператор циклу while використовується для повторення виконання блоку коду, доки певна умова залишається істинною.

```
while (умова) {
   // Блок коду, який виконується, поки умова істинна
}
```

3) Конструкція оператора циклу do-while:

Оператор циклу do-while виконує блок коду один раз, а потім перевіряє умову. Якщо умова виконується, цикл повторюється.

```
do {
   // Блок коду, який виконується принаймні один раз
} while (умова);
do {
   // Блок коду, який виконується принаймні один раз
} while (умова);
```

4) Конструкція оператора циклу for:

Оператор циклу for дає більш компактний спосіб оголошення, ініціалізації та ітерації змінних, які використовуються у циклі.

```
for (ініціалізація; умова; ітерація) {
   // Блок коду, який повторюється, доки умова істинна
}
```

5) Поясніть призначення виразів у конструкції циклу for:

Ініціалізація: Це вираз, який ініціалізує змінні, які використовуються у циклі. Він виконується один раз перед входом у цикл.

Умова: Це вираз, який перевіряється перед кожною ітерацією циклу. Якщо він істинний, то цикл продовжується; якщо ні, то він завершується.

Ітерація: Це вираз, який виконується після кожної ітерації циклу. Він зазвичай використовується для зміни значень змінних, що контролюють хід циклу.

Висновок: ознайомитися з особливостями функціонування операторів циклу та навчитись їх використовувати у процесі програмування.