Лекція 4

Логічні операції

В мові С передбачені логічні операції, які дозволяють побудувати складні умови шляхом об'єднання простих. Логічними операціями є такі операції:

&& - логічне множення;

|| - логічне додавання;

! – логічне заперечення.

Логічне множення

- Якщо в деякому місці програми необхідно забезпечити правдивість двох умов одночасно для вибору деякої гілки її виконання, застосовується логічне множення.
- if ((умова1) && (умова2)) дія;
- Ця умова є правдивою тоді і лише тоді, коли обидві прості умови правдиві.
- Якщо хоча б одна з цих простих умов не правдива, або є не правдивими обидві прості умови, тоді програма ігнорує оператор виведення і переходить до оператора, який є наступним після **if**.

Логічне множення

Вираз A	Вираз B	A&&B
T	T	T
T	F	F
F	T	F
F	F	F

Логічне додавання

- Якщо в деякому місці програми необхідно забезпечити правдивість хоча б однієї з двох умов одночасно для вибору деякої гілки її виконання, застосовується логічне додавання.
- if ((умова1) || (умова2)) дія;

Логічне додавання

Вираз A	Вираз B	A||B
T	T	T
T	F	T
F	T	T
F	F	F

Логічна операція заперечення

- Операція логічного заперечення дозволяє програмістові “обернути” умову.
- Ця операція, на відміну від операцій && і ||, є унарною, тобто у якості операнда використовується тільки одна умова.
- Логічна операція заперечення розміщується перед умовою тоді, коли необхідно вибрати гілку виконання програми з неправдивою умовою

Логічна операція заперечення

Вираз A	!A
T	F
F	T

Приклад 1

Визначити правдивість виразу(створити таблицю істинності виразу)

A||B&&C

Послідовність операцій:

1) D=B&&C
2) A||D

Приклад 1

B	C	D=B&&C
T	T	T
T	F	F
F	T	F
F	F	F

Приклад 1(невірно!!!)

A	D=B&&C	A||D
T	T	T
T	F	T
F	F	F
F	F	F

Приклад 1

A	B	C	B&&C	A||B&&C
T	T	T	T	T
T	T	F	F	T
T	F	T	F	T
T	F	F	F	T
F	T	T	T	T
F	T	F	F	F
F	F	T	F	F
F	F	F	F	F

Приклад 1

Кожна змінна може приймати 2 значення T або F

Кількість рядків в таблиці

$$2^N$$

де N – кількість змінних

Для прикладу 1 :

$$2^3=8$$

Приклад 2

Побудувати таблицю істинності для виразу

- !A&&(B||C)||A

Приклад 2

A	B	C	!A	B||C	!A&&(B||C)	!A&&(B||C)||A
T	T	T	F	T	F	T
T	T	F	F	T	F	T
T	F	T	F	T	F	T
T	F	F	F	F	F	T
F	T	T	T	T	T	T
F	T	F	T	T	T	T
F	F	T	T	T	T	T
F	F	F	T	F	F	F

Приклад 3

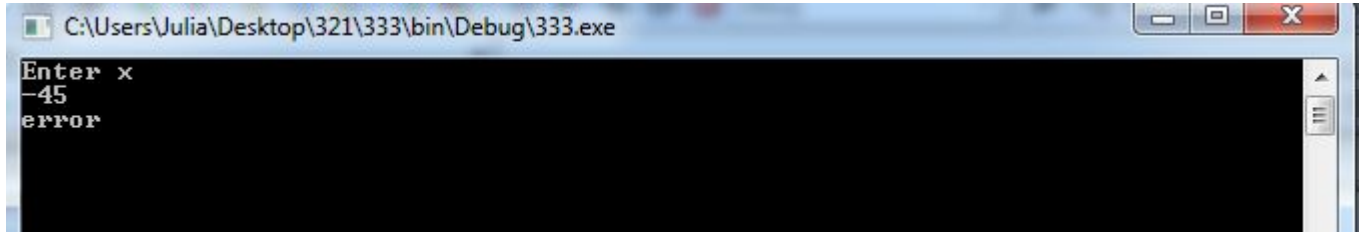
Написати програму мовою С, що обчислює функцію

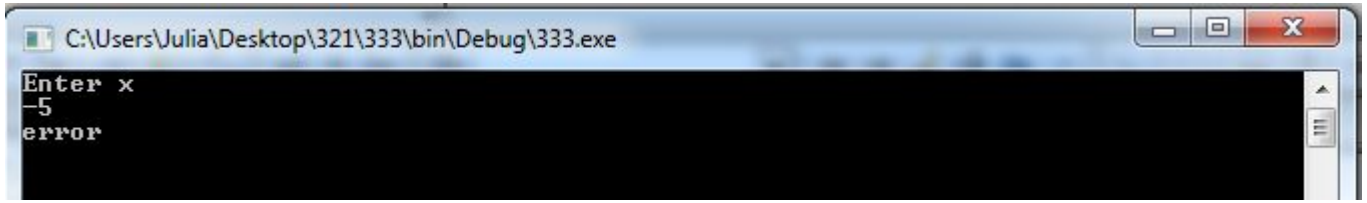
$$y = \begin{cases} x^2 - 5, & \text{якщо x} \in (-5; 0] \cup [5; 10) \\ \frac{2}{3}x - x^3, & \text{якщо x} \in (0; 5] \cup [10; 15) \end{cases}$$

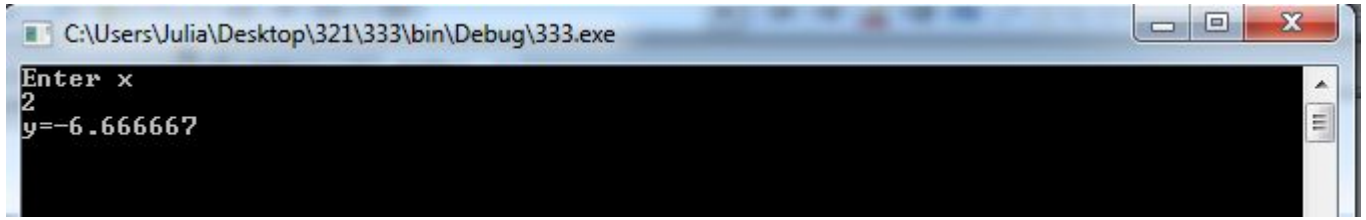
1 спосіб

```
#include <stdio.h>
#include <conio.h>
int main()
{
   float x;
   printf("Enter x\n");
   scanf("%f",&x);
   if (x<=-5)     printf("error\n");
   else
   if (x>15)     printf("error\n");
```

```
 if (x>-5)
        if (x<=0) printf("y=%.5f\n",x*x-5);
    if (x>=5)
        if (x<10) printf("y=%.5f\n",x*x-5);
     if (x>0)
        if (x<=5) printf("y=%f\n",2.0/3.0*x-x*x*x);
        else if (x>=10)
        if (x<15) printf("y=%f\n",2.0/3.0*x-x*x*x);
getch();
    return 0;
}
```


C:\Users\Julia\Desktop\321\333\bin\Debug\333.exe
Enter x
-45
error


C:\Users\Julia\Desktop\321\333\bin\Debug\333.exe
Enter x
-5
error

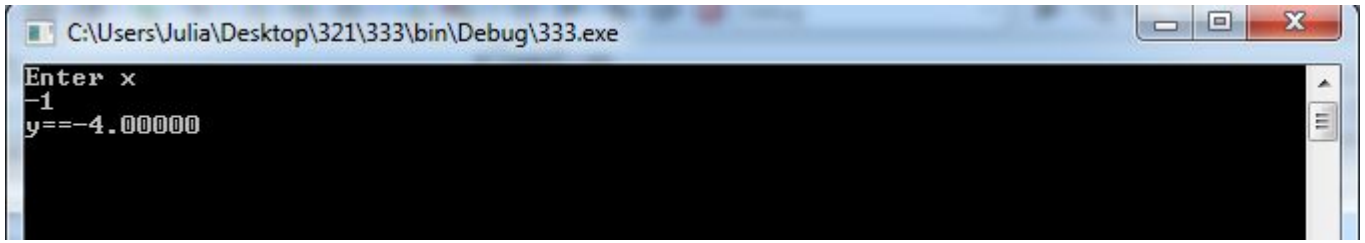

C:\Users\Julia\Desktop\321\333\bin\Debug\333.exe
Enter x
2
y=-6.666667

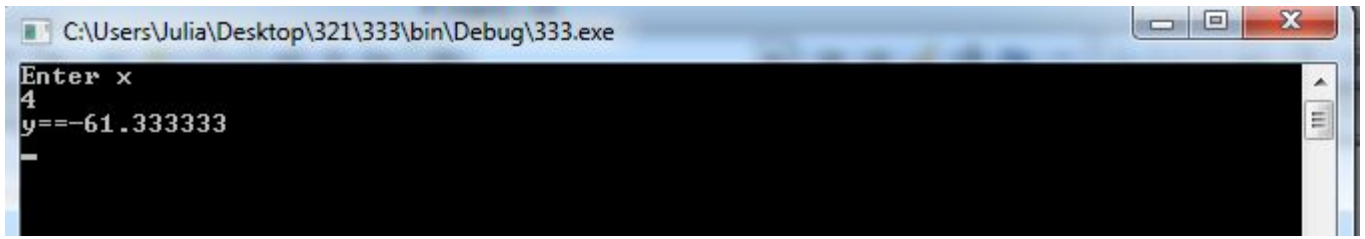
2 спосіб

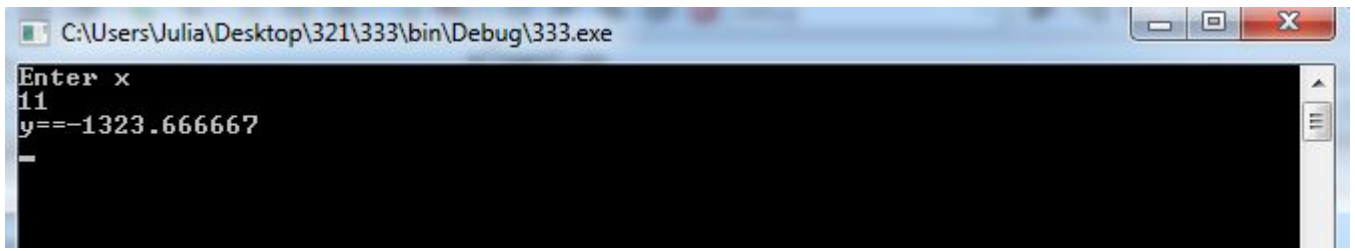
```
#include <stdio.h>
#include <conio.h>

int main()
{
    float x;
    printf("Enter x\n");
    scanf("%f",&x);
```

```
 if ((x<=-5)||(x>=15))     printf("error!!!\n");
    if (((x>-5)&&(x<=0))||((x>=5)&&(x<10)))
     printf("y==%.5f\n",x*x-5);
    if (((x>0)&&(x<=5))||((x>=10)&&(x<15)))
     printf("y==%f\n",2.0/3.0*x-x*x*x);
       getch();
    return 0;
}
```


C:\Users\Julia\Desktop\321\333\bin\Debug\333.exe
Enter x
55
error!!!


C:\Users\Julia\Desktop\321\333\bin\Debug\333.exe
Enter x
-1
y==-4.00000


C:\Users\Julia\Desktop\321\333\bin\Debug\333.exe
Enter x
4
y==-61.333333


C:\Users\Julia\Desktop\321\333\bin\Debug\333.exe
Enter x
11
y==-1323.666667

Структури повторення

Більшість програм включає повторення, тобто цикли.

Цикл – це група команд, які неодноразово виконуються комп'ютером, поки деяка ***умова продовження*** залишається правдивою.

Оператори, які включені до структури повторень, складають тіло цієї структури. Тіло структури повторень може бути простим (один оператор) або складеним оператором (блок).

Структура повторення *for*

Такі повторення іноді називають визначеними повтореннями, оскільки заздалегідь відомо, скільки разів буде виконаний цикл. Для підрахунку кількості повторень використовується керуюча змінна.

Керуюча змінна змінюється кожний раз (як правило, збільшується на 1), коли виконується тіло циклу.

Коли значення керуючої змінної показує, що виконана необхідна кількість повторень, цикл завершується, комп'ютер продовжує виконання програми з оператора, який є наступним за структурою повторення.

Загальний формат структури ***for***

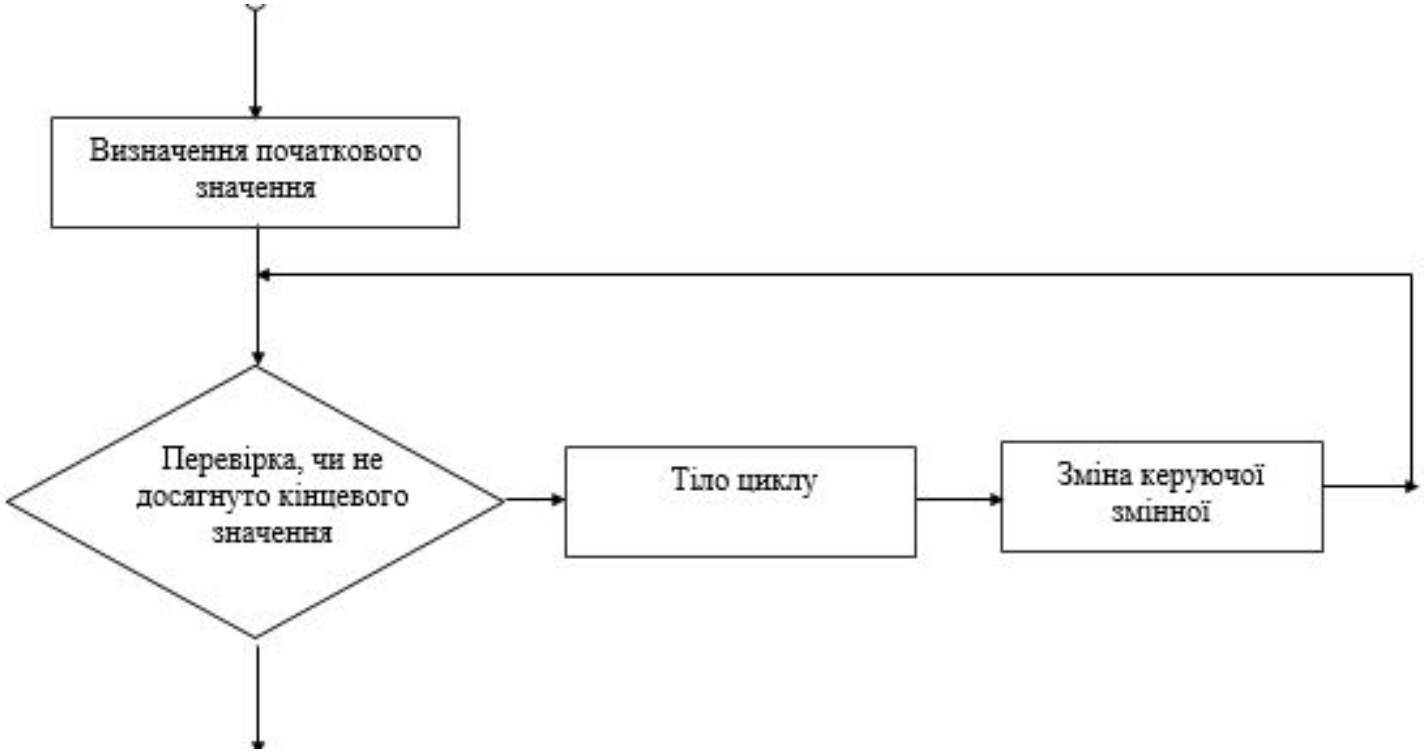
for (вираз1; вираз2; вираз3) тіло циклу ,

де ***вираз1*** ініціює змінну керування циклом,

вираз2 є умовою продовження циклу,

вираз3 вказує, як змінюється змінна керування циклом.

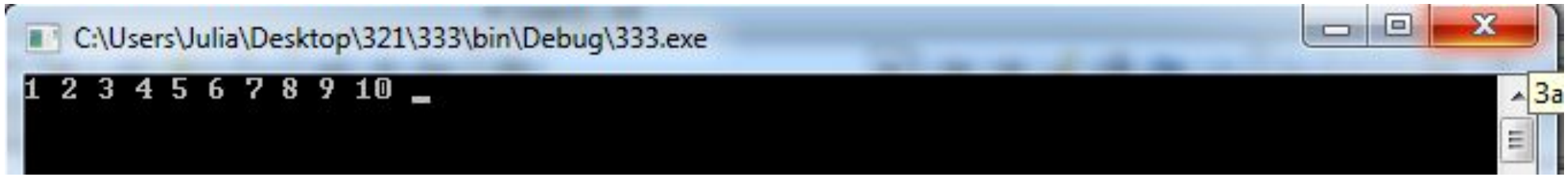
- Для реалізації структури повторення, що керується лічильником, у мові С передбачена структура ***for***. Ця структура автоматично контролює всі деталі такого повторення



Приклад 1

Розглянемо просту програму, яка виводить числа від 1 до 10.

```
#include <conio.h>
#include <stdio.h>
int main()
{   int counter = 1;
for (counter=1; counter<=10; counter++)
        printf("%d ",counter);
        getch();
return 0;
}
```


C:\Users\Julia\Desktop\321\333\bin\Debug\333.exe
1 2 3 4 5 6 7 8 9 10 _