

**Міністерство освіти і науки України**  
**Національний університет “Львівська політехніка”**



**Лабораторна робота №7**  
**з дисципліни «Програмування**  
**частина 2»**

**Виконав:**  
Студент групи АП-11  
Братейко Вадим

**Прийняв:**  
Чайковський І.Б.

Львів 2024

## «Арифметичні операції та вирази мови С»

**Мета роботи:** ознайомитися з синтаксисом арифметичних операцій, їх пріоритетом застосувань, навчитися їх використовувати для обчислень математичних виразів.

### Теоретичні відомості

Мова С була розроблена в процесі створення операційної системи UNIX, тому можна зрозуміти, які принципові можливості в ній реалізовані: це максимальна гнучкість при діалоговому режимі роботи комп'ютера, представлення повідомлень системи і користувача в максимально простій і зрозумілій формі і, водночас, спроможність вибору адекватної реакції в найскладніших ситуаціях. Мова С поєднує в собі можливості прямої адресації і побітових операцій, як в Ассемблері, з використанням великої кількості (декілька сотень) функцій найвищого рівня. При використанні бібліотеки графічних функцій мова С отримала практично необмежені можливості для розробки діалогових програмних засобів. Проте, мова С має суттєвий недолік з точки зору потреб розробки радіотехнічних задач: тут недостатньо розвинені операції арифметики, зокрема, повністю відсутня комплексна арифметика, і її імітація призводить до генерування недостатньо ефективних кодів, що значно збільшує потреби часу при проведенні значних за обсягом математичних обчислень. Фірма Microsoft розробила власну версію мови С з інтерфейсом подібним до мови ФОРТРАН, найбільш пристосованою для математичних розрахунків і генеруючою найефективніші машинні коди. Паралельно на фірмі Borland велась розробка іншої версії мови С, перші варіанти якої мали назву "Turbo C", а пізніші - "Borland C", "C++", причому в версіях "C++" комплексну арифметику реалізують за допомогою класу об'єкта. Сервісна оболонка мови С призначена для розробки та відладки програм і включає в себе засоби роботи з файлами, їх редагування, запуску виконуваних файлів, а також різноманітні режими компіляції і збірки виконуваного модуля, розвинуті засоби відлагоджування програми - детальна діагностика помилок, 2 можливість виконання по кроках з переглядом проміжних результатів, можливість прослідковувати вміст певних змінних тощо.

#### Приклад 1

```
#include <stdio.h>
main()
{
    int a=67;
    int b=33;
    int c=a+b+7;
    printf("a+b+7=%d\n", c);
}
a+b+7=107
```

#### Приклад 2

```
#include <stdio.h>
main()
{
```

```
int a=8;
int b=7;
int c=a+5*b;
printf("c=%d\n",c);
}
C=43
```

### **Приклад 3**

```
#include <stdio.h>
main()
{
    int a=8;
    int b=7;
    int c=(a+5)*b;
    printf("c=%d\n",c);
}
C=91
```

### **Приклад 4**

```
#include <stdio.h>
main()
{
    int a=8;
    int b= ++a;
    printf("a=%d\n",a);
    printf("b=%d\n",b);
}
A=9
B=9
```

### **Приклад 5**

```
#include <stdio.h>
main()
{
    int a=8;
    int b= a++;
    printf("a=%d\n",a);
    printf("b=%d\n",b);
}
A=9
B=8
```

### **Приклад 6**

```
#include <stdio.h>
main()
{
    int a=8;
```

```

    int b= a--;
    printf("a=%d\n",a);
    printf("b=%d\n",b);
}
A=7
B=8

```

### Приклад 7

```

#include <stdio.h>
main()
{
    int a=8;
    int b= --a;
    printf("a=%d\n",a);
    printf("b=%d\n",b);
}
A=7
B=7

```

### Приклад 8

Додано пропущені дужки {} для функції main.

Додано int перед main() для вказання типу повернення функції main.

Додано крапку з комою ; після оголошення масиву char name[50];.

Виправлено sizeof PRAISE на sizeof(PRAISE) у функції printf.

```

#include <stdio.h>
#include <string.h>
#define PRAISE "О, яке чудове ім'я!"
int main()
{
    char name[50];
    printf("Як Вас звати?\n");
    scanf("%s", name);
    printf("Привіт, %s. %s\n", name, PRAISE);
    printf("Ваше ім'я складається з %lu літер і займає %lu комірок пам'яті.\n",
    strlen(name), sizeof(name));
    printf("Вітальна фраза складається з %lu літер і займає %lu комірок
пам'яті.\n", strlen(PRAISE), sizeof(PRAISE));
    return 0;
}

```

```

Vadym
Привіт, Vadym. О, яке чудове ім'я!
Ваше ім'я складається з 5 літер і займає 50 комірок пам'яті.
Вітальна фраза складається з 33 літер і займає 34 комірок пам'яті.

```

### Приклад 9

```

#include <stdio.h>
#include <string.h>
void main()

```

```

{
    float x=1.4,y=2.0;
    int z;
    z=x/2*7 + y/4-1;
    printf("z=%d\n",z);
    getch();
}
Z=4

```

### Приклад 10

```

#include <stdio.h>
#include <string.h>
void main()
{
    int x=2,z;
    float y;
    z=0.5*(y=2.3*x)+x++/3*y;
    printf("z=%d\n",z);
    getch();
}
Z=2

```

### Приклад 11

```

#include <stdio.h>
#include <string.h>
void main()
{
    int x, y=3;
    float z;
    z=1.1*(x=++y/2.)+0.3*x;
    printf("z=%d\n",z);
    getch();
}
Z=118281368

```

## Контрольні запитання

1) Призначення та структура програми, написаної мовою С:

Мова програмування С використовується для створення різноманітних програм, від найпростіших до складних системних програм. Структура програми на мові С складається з заголовних файлів, визначення функцій, головної функції `main()` та інших користувацьких функцій.

2) Різновиди типів величин:

У мові С є такі основні типи величин:

Цілі числа (`int`, `short`, `long`, `char`)

Дійсні числа (`float`, `double`)

Символи (`char`)

Вказівники (`pointer`)

Похідні типи (структури, об'єднання, переліки)

3)Що таке константи і змінні:

Константи - це значення, яке не може бути змінено під час виконання програми.

Змінні - це області пам'яті, які призначені для зберігання даних, і можуть бути змінені під час виконання програми.

4)Порядок виконання операцій:

В мові C порядок виконання операцій може змінюватися за допомогою дужок (), а також за допомогою пріоритету операцій. Зазвичай операції виконуються в порядку: дужки, знаки поділу, множення, додавання, віднімання, порівняння тощо.

5)Особливості операцій інкремента і декремента:

Оператор інкремента ++ збільшує значення змінної на 1.

Оператор декремента -- зменшує значення змінної на 1.

Обидва оператори можуть бути застосовані до змінних типу цілих чисел.

6)Операції присвоєння:

Операція присвоєння в мові C використовується для присвоєння значення змінній. Синтаксис такий: змінна = вираз;

7)Пояснення змісту і обґрунтування результатів виконаних прикладів:

Для пояснення змісту і обґрунтування результатів прикладів, потрібно конкретизувати самі приклади, що були виконані. Якщо ви надаєте деякі приклади коду або ситуації, я можу надати вам пояснення і обґрунтування результатів на їх основі.

**Висновок:** на цій лабораторній роботі я ознайомився з синтаксисом арифметичних операцій, їх пріоритетом застосувань, навчився їх використовувати для обчислень математичних виразів.