Practical Work No. 2
Expressions and Operators

Goals:

Learn how to manipulate expressions using the main C operators such as:

- the simple assignment operator = and compound +=, -=, /=, *=, %=…
- the arithmetic operators +, -, *, / and %
- the comparison operators <, <=, >, >=, == and !=
- the increment ++ and decrement operators --

I. Expressions

An expression consists of variables and constants connected by operators. In C, there are 3 types of operators: unary (one operand), binary (2 operands), and ternary (3 operands). The main classes of operators are: arithmetic, assignment, comparison, and logical operators.

II. Operators

II.1. Arithmetic operators

The C language knows two unary arithmetic operators: + and - and five binary operators: addition (+), subtraction (-), division (/), multiplication (*), and modulo (%) (remainder of integer division)

II.2. Assignment operators

Simple assignment is performed by the (=) operator.

For all binary arithmetic operators Δ (-,+,*,/…), the C language defines a compound assignment operator Δ= (without spaces). These compound assignment operators allow you to simplify expressions.

Examples:

operator	use	equivalent
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y

II.3 Increment and decrement operators

The (++) and (--) operators are unary operators that allow you to add 1 to the contents of their operand and subtract 1 from them, respectively. This operation is performed after or before the evaluation of the expression, depending on whether the operator follows or precedes its operand.

Examples:

```
int i=17 , j=2, k ;
i--; /*equivalent to i= i-1*/
k =5+i++ - j; /*equivalent to k=5+i-j then i++*/
k =5+++i - j; /*equivalent to ++i then k=5+i-j */
```

II.3 Comparison operators

The comparison operators are as follows:

Opérateur	Dénomination	Effet	Exemple	Résultat (avec x valant 7)
== A ne pas confondre avec le signe d'affectation (=)!!	opérateur d'égalité	Compare deux valeurs et vérifie leur égalité	x==3	Retourne 1 si X est égal à 3, sinon 0
<	opérateur d'infériorité stricte	Vérifie qu'une variable est strictement inférieure à une valeur	x<3	Retourne 1 si X est inférieur à 3, sinon 0
<=	opérateur d'infériorité	Vérifie qu'une variable est inférieure ou égale à une valeur	x<=3	Retourne 1 si X est inférieur ou égal à 3, sinon 0
>	opérateur de supériorité stricte	Vérifie qu'une variable est strictement supérieure à une valeur	x>3	Retourne 1 si X est supérieur à 3, sinon 0
>=	opérateur de supériorité	Vérifie qu'une variable est supérieure ou égale à une valeur	x>=3	Retourne 1 si X est supérieur ou égal à 3, sinon 0
!=	opérateur de différence	Vérifie qu'une variable est différente d'une valeur	x!=3	Retourne 1 si X est différent de 3, sinon 0

II.4 Logical (Boolean) operators

This type of operator allows you to check whether several conditions are true:

Opérateur	Dénomination	Effet	Syntaxe
||	OU logique	Vérifie qu'une des conditions est réalisée	((condition1)||(condition2))
&&	ET logique	Vérifie que toutes les conditions sont réalisées	((condition1)&&(condition2))
!	NON logique	Inverse l'état d'une variable booléenne (retourne la valeur 1 si la variable vaut 0, 0 si elle vaut 1)	(!condition)

III. Work requested

Exercise 1:
In each of the following statements, assume that A= 27, B= 7, C=5 before execution.
Write a program to display the values of A, B, and C after each instruction is executed.

1. **C+=A-8-B;**
2. **C=A%B ;**
3. **C=A%B++ ;**
4. **C=A%++B ;**
5. **C+=++AB ;**
6. **C+=B ;**
7. **C*=B ;**
8. **C+=--B ;**
9. **C-=C-- ;**
10. **C+=A--+--B ;**

Exercise 2:
What results does the following program provide?

```
#include <stdio.h>
void main() {
        int i, j, n;
        i = 0; n=i++;
        printf("A : i=%d n=%d \n", i, n);
        i = 10; n=++i;
        printf("B : i=%d n=%d \n", i, n);
        i = 20; j=5; n=i++ *++j;
        printf("C : i=%d j=%d n=%d \n", i, j, n);
        i = 15; n=i+=3;
        printf("D : i=%d n=%d \n", i, n);
        i = 3; j = 5; n=i+=--j;
        printf("E : i=%d j =%d n=%d \n", i, j, n);
}
```

Exercise 3:

Write a C program that calculates the sum of the digits of a number entered by the user.

- **Case 1:** Assume that the number has at most 3 digits.
- **Case 2:** Assume that the number has at most 4 digits.