Unit 2: Arithmetic Expression, and Conditional Statements, Loops, Expression.

Operators.

- Arithmetic operators (+, -, *, /, %)
- Relational operators (<, <=, >, >=, ==, !=)
- Logical operators (&&, ||, !)
- Assignment operators (+=, -=, *=, /=)
- Increment and decrement operators (++, --)
- Conditional operators (?:)
- Bitwise operators (&, |, ^, <<, >>)
- Special operators ()

~~Adith~~ Arithmetic Operators.

These are used in/for mathematical calculation

operator → meaning → example → Description

+ → Addition → a+b → Addition of a and b

- → Subtraction → a-b → Subtraction of b from a.

* → multiplication → a*b → Multiplication of a and b

/ → Division → a/b → Division of a by b

% → Modulo division - remainder → a%b - Modulo of a by b

Relational operators.

- These are used to compare two numbers and taking decisions based on their relation.
- Relational expressions are used in decision statements such as if, for, while, etc.

operator → meaning → example → Description

< → is less than → a<b → a is less than b

<= → is less than or equal to → a<=b → a is less than or equal to b.

> → is greater than → a > b → a is greater than b
>= → is greater than or equal to → a >= b → a is greater than or equal
== → is equal to → a == b → a is equal to b
!= → is not equal to → a != b → a is not equal

Logical operators:
- These are used to test more than one condition and make decisions.

Operator → meaning
&& → logical AND (Both non zero then true, either zero then fa
|| → logical OR (Both zero then false, either is no then tru
! → logical NOT (is greater than.

a	b	a && b	a || b	!a	!b
0	0	0	0	1	1
0	1	0	1	1	0
1	0	0	1	0	1
1	1	1	1	0	0

Examples:
1. (a == b) && (b == c) will be true if a = b = c.
2. (a == b) && (b == c) will be false if the val of even a single variable is different.

1. Logical AND operator (&&)

- If both operands are non-zero then the condition becomes true. Otherwise, the result has a value of 0
- The return type of the result is int
- Syntax: (operand-1 && operand-2)

Example

```
#include <stdio.h>

void
int main(){
  int a = 10, b = 20;
  if(a > 0 && b > 0) {
      printf("Both values are greater than 0\n");
    }else {
     printf("Both values are less than 0\n");
       }
  return 0;
  }
```

Output:
Both values are greater than 0.

2. Logical OR Operator (||)

- The condition becomes true if any one of them is non-zero. Otherwise, it returns false i.e; 0 as the value.
- Syntax: (operand1 || operand-2)

Example

```
#include <stdio.h>
  int main(){
   int a = -1 , b = 20;
 if ( a > 0 || b > 0) {
      printf("Any one value is greater than 0\n");
   }else{
 printf(" Both values are less than 0");
 } return 0; }
```

Output - Any one of the given value is greater than 0.

3. Logical NOT Operator (!)

- If the ~~cod~~ condition is true then the logical operator will make it false and vice-versa
- Syntax : !(operand1 && operand_2) or !(operand
- Below is the truth table for the logical NOT operator

x	!x
0	1
1	0

Example

```
#include <Stdio.h>

int main() {
int a = 10, b=20;
   if (!(a>0 && b>0)) {
        printf ("Both values are greater than 0");
   }else{
    printf ("Both values are less than 0\n");
    }
   return 0;
   }
```

Assignment Operators.

- They are used to assign the result of a expression to a variable.
- Assignment operator stores a value in memo
- C also supports shorthand assignment opera which simplify operation with assignment.

Operator	Meaning
=	Assign value of right side to left si
+=	a+=1 is same as a=a=a+1
-=	a-=1 is same as a=a-1
*=	a*=1 is same as a=a*1