LECTURE 11:
C OPERATORS

Types of C Operators

- Arithmetic Operators
- Relational Operators
- Assignment Operators

Arithmetic Operators

- Assume variable **A** holds 10 and variable **B** holds 20 then –

Operator	Description	Example
+	Adds two operands.	A + B = 30
–	Subtracts second operand from the first.	A – B = -10
*	Multiplies both operands.	A * B = 200
/	Divides numerator by de-numerator.	B / A = 2
%	Modulus Operator and remainder of after an integer division.	B % A = 0
++	Increment operator increases the integer value by one.	A++ = 11
--	Decrement operator decreases the integer value by one.	A-- = 9

Example of Arithmetic Operators

```
#include <stdio.h>

int main()
{
  int a=40,b=20, add, sub, mul, div, mod;
  add = a+b;
  sub = a-b;
  mul = a*b;
  div = a/b;
  mod = a%b;
  printf("Addition of a, b is : %d\n", add);
  printf("Subtraction of a, b is : %d\n", sub);
  printf("Multiplication of a, b is : %d\n", mul);
  printf("Division of a, b is : %d\n", div);
  printf("Modulus of a, b is : %d\n", mod);
}
```

Output:

```
Addition of a, b is : 60
Subtraction of a, b is : 20
Multiplication of a, b is : 800
Division of a, b is : 2
Modulus of a, b is : 0
```

Increment & Decrement Operators in C

Operator type	**Operator**	**Description**
Increment	i ++	Value of i is incremented
Decrement	i – –	Value of i is decremented

Example of increment & decrement operators

```
#include <stdio.h>
 int main()
{
    int x = 5;
    int y = 8;
    printf("x=%d\n", x--);
    printf("y=%d\n", y++);

    return 0;
}
```

Output

```
x=4
y=9
```

Relational Operators

- Assume variable **A** holds 10 and variable **B** holds 20 then -

Operator	Description	Example
==	Checks if the values of two operands are equal or not. If yes, then the condition becomes true.	(A == B) is not true.
!=	Checks if the values of two operands are equal or not. If the values are not equal, then the condition becomes true.	(A != B) is true.
>	Checks if the value of left operand is greater than the value of right operand. If yes, then the condition becomes true.	(A > B) is not true.
<	Checks if the value of left operand is less than the value of right operand. If yes, then the condition becomes true.	(A < B) is true.

Example Of Relational Operators

```
#include <stdio.h>
int  main()
 {
   int a = 21, b = 10;
     if( a == b )
          printf(" Line 1 - a is equal to b\n" );
    else
          printf(" Line 2 - a is not equal to b\n" );

   if ( a < b )
          printf(" Line 3 -  a is less than b\n" );
   else
          printf(" Line 4 - a is not less than b\n" );

   if ( a > b )
          printf(" Line 5 - a is greater than b\n" );
   else
          printf(“Line 6 -  a is not greater than b\n" );

   return 0;
}
```

Condition statement:
if-else

Output

```
Line 2 - a is not equal to b
Line 4 - a is not less than b
Line 5 - a is greater than b
```

Assignment Operators

- The following table lists the assignment operators supported by the C language −

Operator	Description	Example
=	Simple assignment operator. Assigns values from right side operands to left side operand	C = A + B
+ =	Add AND assignment operator. It adds the right operand to the left operand and assign the result to the left operand.	C + = A means C = C + A
- =	Subtract AND assignment operator. It subtracts the right operand from the left operand and assigns the result to the left operand.	C - = A means C = C – A

EXAMPLE OF ASSIGNMENT OPERATORS

```
#include <stdio.h>
int main() {
   int a = 21, c ;
   c =  a;
   printf("Line 1, Value of c = %d\n", c );

   c +=  a;
   printf("Line 2, Value of c = %d\n", c );

   c -=  a;
   printf("Line 3, Value of c = %d\n", c );
}
```

Output

```
Line 1, Value of c = 21
Line 2, Value of c = 42
Line 3, Value of c = 21
```

EXAMPLE OF ASSIGNMENT OPERATORS

```
// Print numbers from 1 to 10

#include <stdio.h>
int main()
 {
     int i, m;
     for (i = 1; i < 11; ++i)
        {
           m = i;
           printf("%d ", m);
        }
     return 0;
 }
```

Iteration statement:
for

Output

```
1 2 3 4 5 6 7 8 9 10
```

Thank You