OPERATORS AND CONTROL STATEMENTS IN JAVA

1. Operators :

Operators are used to perform operations on variables and values.
These include arithmetic, relational, logical, bitwise, assignment, and miscellaneous operators.

a) Arithmetic Operators :

These are used for basic math operations.

Operator	Example	Description
+	a+b	Addition
-	a-b	Substraction
*	a*b	Multiplication
/	a/b	Division
%	a%b	Modulus (remainder)

classmate
Date
Page

Example :

```
int a = 10 , b=5;
int sum = a+b ;          // sum = 15
int remainder = a % b    // remainder = 0
```

b) Relational Operators :

These operators compare values to see if they are equal , greater , or smaller than each other.

Operator	Example	Description
==	a=b	Equal to
!=	a!=b	Not equal to
>	a>b	Greater than
<	a<b	Less than
>=	a>=b	Greater than or equal to
<=	a<=b	Less than or equal to

Example :

```
if (a>b) {
    printf ("a is greater than b");
}
```

@ Curious_programmer

c) Logical Operators :

These are used when you want to check multiple conditions at once (for example, if two things are true).

Operator	Example	Description
&&	a && b	AND (both must be true)
||	a || b	OR (one of them is true)
!	!a	NOT (inverts the condition)

Example :

```
if (a>0) && b<10) {
    // Executes if both conditions are true.
}
```

Logical operators help combine conditions like checking if two conditions are true at the same time.

d) Bitwise Operators :

These work with the binary form of numbers (bits) and perform operations on individual bits.

Operator	Example	Description
&	a & b	AND
|	a | b	OR
^	a ^ b	XOR
~	~a	NOT
<<	a << 2	Left shift
>>	a >> 2	Right shift

Example :

```
int result = a&b ;   // bitwise and of a and b
```

These are used for low-level programming and dealing with individual bits of numbers.

@ Curious_.programmer

e) Assignment Operators :

These are used to assign a value to a variable.

Operator	Example	Description
=	a=b	Assigns b to a
+=	a+=b	Adds b to a and stores the results in a
-=	a-=b	Subtracts b from a and stores the result in a
*=	a*=b	Multiplies a and b and stores result in a
/=	a/=b	Divides a by b and stores the result in a
%=	a%=b	Stores the remainder of a dividend by b in a

Example :

```
int  a = 5 ;
 a +=3 ;    // a = 8
```

Assignment operators make it easy to store results in variables after doing operations.

@ Curious_. programmer

f) Miscellaneous Operators :

These are special operators used for specific tasks.

Operator	Example	Description
?:	a>b ? a : b	Ternary (shortend if-else)
sizeof	sizeof (a)	Gives the size of a variable in memory

Example :

```
int max = (a>b) ? a : b ;
                // max = a if a>b , else max=b
```

These help perform tasks like quick decision - making or finding the size of a variable.

@Curious_.programmer

CLASSMATE
Date :
Page :

2. Control Flow Statements :

Control flow statements help your program decide what to do based on conditions or repeat certain actions.

a) if statement :

The if statement runs a block of code only if a condition is true.

Example:

```
if (a > b) {
    printf ("a is greater than b");
}
```

It is used when you want to perform an action only if a condition is met.

b) if-else Statement :

The if-else statement lets you choose between two blocks of code : one if the condition is true, and one if it's false.

@ Curious –. programmer

Example :

```
if (a>b) {
    printf ("a is greater than b");
} else {
    printf ("b is greater than a");
}
```

It is used when there are two parallel actions to take based on conditions.

c) Switch Statement :

The switch statement is a cleaner way to handle multiple possible conditions by checking one variable against many options.

```
switch (day) {
      Case 1 : printf ("Monday"); break;
      case 2 : printf ("Tuesday"); break;
      default : printf ("Invalid day");
}
```

It is used when you have many different options to check for, like menu choices.

3. Loops:

Loops are used to repeat a set of actions multiple times, like running a task over and over until a condition changes.

a) for Loop:

A for loop is used when you know how many times you want to repeat an action.

Example:

```
for (int i=0; i<5; i++){
    printf("%d", i);  //Prints  0 1 2 3 4
}
```

It's used when you need to repeat something a fixed number of times, like looping through a list of items.

b) while Loop:

A while loop repeats an action ~~an~~ as long as a condition is true. You might not know how many times it will repeat.

Examples :

```
int i = 0;
while (i<5) {
  printf ("%d", i);   // Prints 0 1 2 3 4
   i++;
}
```

It's useful when you don't know how many repetitions you need, but just want to keep going until something changes.

c) do-while Loop:

A do-while loop is similar to a while loop, but it always runs at least once before checking the condition.

```
int i = 0;
 do {
     printf ("%d ", i);
     i++;
} while (i <5);
```

@ Curious_. programmer

It's used when you want to ensure the code runs at least once like showing a menu before checking the condition.

4. break and continue Statement :

a) break

The break statement stops the loop completely and moves on to the next part of the program.

Example :

```
for (int i = 0 ; i < 10 ; i++) {
      if (i==5) break ;  // Stops the loop when i is 5
}
```

Use break when you want to stop a loop before it finishes.

b) Continue :

The continue statement skips the current iteration of a loop and moves on the next one.

Example :

```
for (int  i = 0 ; i < 10 ; i++) {
      if (i==5) continue ;   // skips printing 5
      printf ("%d", i) ;
                  // Prints  0 1 2 3 4 6 7 8 9
  }
```

Use continue when you want to skip certain steps in a loop but keep the loop going.

@ Curious -. programmer