Comments: Comments can start at the beginning of a line or on the same line following a program statement and are used for documentation purpose. Comments are ignored by compiler. In Java comments can be given in following ways:

a) **Single line comments** start with **//** (double slash) symbol and terminate at the end of the line.

b) **Multiline comments** start with **/*** symbol and terminate with */ symbol.

For example:

```
/* Program to find the sum of two
numbers */
int a;  //declares a variable of integer type
```

c) **Multiline comments** start with **/**** symbol and terminate with */ symbol. Comments that start with **/**** symbol and terminate with */ symbol are automatically extracted for documentation from java software, when we use a program named javadoc. These comments are also ignored by compiler

For example:

```
/** Program to find the remainder of two
numbers */
```

Syntax for switch…case construct: In a switch…case construct the value of the switch variable is compared against a set of constant values. Wherever, a match is found, the statements associated with that particular case are executed. If the value of the switch variable does not match any of the case constants, then the default case is executed.

Syntax	Example
switch(<switch variable>) **{** **case <constant1>: statement1;** **.** **.** **statement n;** **case <constant2>: statement1;** **.** **.** **statement n;** **.** **.** **.** **case <constantn>: statement1;** **.** **.** **statement n;** **.** **.**	**switch(color)** **{** **case 3: System.out.println(”Red”);** **case 2: System.out.println(”Orange”);** **case 7: System.out.println(”Black”);** **}**

.
 	
default : statement1;
.
.
statement n;
}
 	
The underlined is optional.
	switch(choice)
	{
	case 1: System.out.println("Sum=");
	System.out.println(a+b);
	break;
	case 2: System.out.println("Diff=");
	System.out.println(a-b);
	break;
	case 3: System.out.println("Product=");
	System.out.println(a*b);
	break;
	case 4: System.out.println("Quot=");
	System.out.println(a/b);
	break;
	case 5: System.out.println("Rem=");
	System.out.println(a%b);
	break;
	default:System.out.println("wrong input");
	}

Break statement in switch…case construct: The break keyword causes the entire switch statement to exit, and the control is passed to statement following the switch…case construct. Without break, the execution just continues to the next case.*The break statement is optional in the switch…case construct.*

Use of default case in switch…case construct: This keyword gives the switch…case construct a way to take an action if the value of the switch variable does not match with any of the case constants. No break statement is necessary after default case, since the control is already at the end of switch…case construct. *The default case is optional in the switch…case construct.*

Difference between if…else construct and switch…case construct

If…else construct	Switch…case construct
Any kind of relation can be checked in any condition of this construct. The condition can include any relational operator (<, >, <=,>=,==,!=).	This construct checks only for equality.
Float values/variables can be used in the condition.	Float variable cannot be used as switch variable.

Example:
`if(a>=b)`
`  System.out.println(a);`
`else`
`  System.out.println(b);`
	Example:
	`switch(a)`
	`{ case 1:System.out.println("red");`
	`  break;`
	`  default:System.out.println("Orange");`
	`}`

Arithmetic Assignment Operators: Arithmetic assignment operator, is one, which combines arithmetic operator and an assignment operator. The arithmetic assignment operators present in Java are: += , -=, *= , /= , %= .
Example: a+=b; **// is same as a=a+b;**

Increment/Decrement Unary Operators:

Increment (++) Operator: It is used to increment the value of variable by 1. It can be used in the following two ways:

Pre Increment	Post Increment
In this, the operator precedes the variable. Example: `++a`	In this, the operator follows the variable. Example: `a++`

`b=++a;`
This statement is equivalent to the following two statements:-
`a=a+1;`
`b=a;`
So, in pre increment the value of the variable is incremented, before it is used in any other operation (Assignment in this case)
	`b=a++;`
	This statement is equivalent to the following two statements:-
	`b=a;`
	`a=a+1;`
	So, in post increment the value of the variable is incremented, after it is used in any other operation (Assignment in this case)

Decrement (--) Operator: It is used to decrement the value of variable by 1. It can be used in the following two ways:

Pre Decrement	Post Decrement
In this, the operator precedes the variable. Example: `--a`	In this, the operator follows the variable. Example: `a--`

`b= --a;`
This statement is equivalent to the following two statements:-
`a=a-1;`
`b=a;`
So, in pre decrement the value of the variable is decremented, before it is used in any other operation (Assignment in this case)
	`b=a--;`
	This statement is equivalent to the following two statements:-
	`b=a;`
	`a=a-1;`
	So, in post decrement the value of the variable is decremented, after it is used in any other operation (Assignment in this case)

Conditional Operator(?:): The question mark(?) and the colon(:) make up the conditional operator. The operator operates on three operands, that is why it is also called the **ternary operator.** The whole expression is called the conditional expression. The expression before the question mark is called the test expression. If the test expression is true, then the entire conditional expression takes on the value of the operand after the question mark. If the test expression evaluates to false, then the conditional expression takes on the value of the operand following the colon.

Syntax:

<test expression>? <true expression> : <false expression>

Example:

res = (x<y) ? x : y ;

In the above example, if the test expression evaluate to true then the variable res will be assigned the value of the variable after the question mark, i.e. x else it will be assigned the value of the variable after the colon i.e. y.

Example: System.out.println(((a>b)?(a-b):(b-a))); //displays a-b if the condition is //true, else displays b-a