* Java follows unicode principles.

Conditional Statements

* if (condition) {
    } // Executes this block if condition is true.

* if-else

```
if(condition) {
    // Executes if true.
}
else {
    // Executes if condition is false.
}
```

* else-if

```
if (condition) {
    // Executes if condition is true.
}
else if (condition) {
    // it will executes if condition is true
}
else if (condition) {
    // Executes if condition is true.
}
else {
    // Executes if none of the above are true.
}
```

(iv) **if-else**

Syntax

```
if (boolean expression T or F) {
    //body - execute this.
}
else {
//execute this
}
```

Example:

```
public class Conditionals {
    pvsm(String [] args) {
        int salary = 25000;
        if (salary > 10000) {
            salary = salary + 2000;   // output, this block
        }                                will execute as
        else                             salary is 25000 ie; >10000
        {                                So, it updates salary =
            salary = salary + 1000;      28000;
        }
        S.o.p(salary); // 28000.
    }
}
```

→ In if-else either one of blocks will only executes.

else-if

↳ We can add multiple else-if statements as per requirements.

Example:

```
public class Conditionals {
  psvm(String [] args) {
   int salary = 25000;
   if (salary > 10000)
   {salary + = 2000;
   }
   else if (salary > 20000)
   {
     salary + = 3000;    // this will execute.
   }
   else {
     salary + = 1000;
   }
   S..p(salary) : // 28000;
}
```

Loops :-

It is a control-flow statement that allows code to be executed repeatedly based on a given condition.

Loops

for | while | do-while.

① for( ) [It generally used when we know how many times loop will iterate.]

Syntax:

```
for (initialization; condition; updation)
{
  // body
}
```

Ex: Printing numbers from 1 to 10;

Code snippet:

```
for (int i=1; i<=10; i++)
{
  S.out.println(i);   // Output
}
```

Output
1
2
3
4
5
6
7
8
9
10

② while( ) [Used when we dont know how many times loop will iterate]

Syntax:

```
initialization
while (condition) {
  // body
  } // updation.
```

Ex: Printing numbers from 1 to 10.

```
int i=1
while( i<=10)
{
  System.out.println(i); // output 1
                                   2
                                   3
                                   4
                                   5
                                   6
                                   7
                                   8
                                   9
                                   10
  i++;
}
```

③ do-while() [Used when we want to execute our statement atleast one time].

↳ Also called Exit-Control loop - because it checks the condition after the execution of statement.

Syntax:

```
initialization
do {
    //body
    // updation
} while(condition);
```

Ex: Printing 1 to 5

```
int n=1
do{
    S.o.p(n);      // output 1
                              2
                              3
                              4
                              5
    n++;
} while (n<=5);
```