Switch Case Statements

A control structure used to simplify the decision making when multiple options are there

SYNTAX:

```
switch(expression){
                        case value1:
                        // code block
                        Break;

                        case value2:
                        // Code block
                        Break;

                        // more cases

                        default:
                        //Default block of code
}
```

```java
public class SwitchCaseExample {
    public static void main(String[] args) {
        int day = 3;

        switch(day){
            case 1:
                System.out.println("Monday!");
                break;

            case 2:
                System.out.println("Tuesday!");
                break;

            case 3:
                System.out.println("Wednesday!");
                break;

            case 4:
                System.out.println("Thursday!");
                break;

            default:
                System.out.println("Invalid Input!!!!");

        }
    }
}
```

o/p:

Wednesday!

Explanation:

Day variable value is 3 and it matches with case 3, so Wednesday is printed.

The break prevents the execution of further cases

The default is optional but it executes when no other cases matches.

Looping Statements

It is used to execute a block of code repeatedly.

Types of loops:

1. for loop
2. while loop
3. do-while loop

- for loop:
  - It is used when the number of iterations are known

SYNTAX:

```
for(initialization,condition,update){
        //Code Block
}
```

```java
public class ForLoopExample {

    public static void main(String[] args) {
        for (int i=1; i <= 50; i++){
            System.out.println("Count= "+i);

            //1st Iteration
            /* i=1
            1 <= 50 -> True
             Print => Count = 1
             */
        }
    }
}
```

Explanation:

Initialization: int i = 1;

Condition: i <= 50
Update: i++

* While loop

  - It is used when the condition is checked before each iteration.

SYNTAX:
while(condition){

        //Code block

}

```
public class WhileLoopExample {
    public static void main(String[] args) {
        int i = 1;
        while(i <= 50){
            System.out.println("Count: "+ i);
            i++;
        }
    }
}
```

- do-while loop
  - It gurantees the execution of the loop body at least once.

SYNTAX:

```
do{
  // code block
}while(condition);
```

```
public class DoWhileLoopExaple {
    public static void main(String[] args) {
        int i = 100;

        do {
            System.out.println("Count: "+ i);
            i++;
        }while(i<5);
        // 100<5 -> false -> wont execute the loop again
    }
}
```

O/P: 100

```
public class DoWhileLoopExaple {
    public static void main(String[] args) {
        int i = 1;

        do {
            System.out.println("Count: "+ i);
            i++;
        }while(i<5);
    }
}
```

O/P:

```
Count: 1
Count: 2
Count: 3
Count: 4
```

Note:

1. Use for loops when number of iterations are known
2. Use while oops when the condition needs to be checked before the execution
3. Use do-while loops when we want the loop execute for at least once.
4. Use switch for multiple conditional branches.