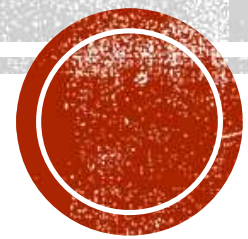
CONTROL FLOW STATEMENT



CONTROL FLOW STATEMENT

1. Program execution flow is always in the sequence.
2. To execute the program based on some scenarios or based on some conditions, you can use the control flow statement
3. There are different typed of control flow statement
   a. **Conditional Statement**
      i. Using this statement you execute the statements based on condition.
      ii. Example: if condition and its variable or Switch case
   b. **Looping Statement**
      i. Using this statement you can execute the statements multiple time.
      ii. Example: while, do-while, for loop, nested loop or enhance for loop

IF CONDITION AND VARIATIONS

1. If you wanted to execute the statement or block of statement based on some scenario/condition.
2. There are multiple variations of for If statement

if Condition

Syntax:

```
if(condition/Boolean Expression)
{
        Statement(s)
}
```

Note: **The statements from the if block will only executes if the condition is true.**

if-else condition

Syntax:

```
if(condition/Boolean Expression)
{
        Statement(s)
}
else
{
        Statement(s)
}
```

Note: **In this case the if statements execute when the condition is true and else statement executes when the condition is false. Else block has to use with if block only.**

IF CONDITION AND VARIATIONS

Else-if condition/ladder

Syntax:

```
if(condition)
{
    Statement(s)
}
else if(condition)
{
    Statement(s)
}
else if(condition)
{
    Statement(s)
}
else
{
    Statement(s)
}
```

Note: **Can write as many else-if you want. If any of the condition is true then no further conditions will be checked.**

IF CONDITION AND VARIATIONS

Nested if

Syntax:

```
if(condition)
{
    if(condition)
    {
    }
}
```

- **Task**

WAP to find the vaccination slot
age is between 60-120
    Your are in 1st slot of vaccination
age is between 40-60
    Your are in 2st slot of vaccination
age is between 18-40
    Your are in 3st slot of vaccination
If age is between 1-18
    Your are in 4st slot of vaccination
If age not between 1-120
    Invalid age

SWITCH CASE

Switch Case

- Switch can be use to improve the performance of the else-if structure which created to check the exact equality of the values.
- In the switch there is not value comparison happens or condition check happened inside the switch.
- Syntax:

```
switch(value)
{
        case label:
                Statement(s)
                break;
        case label:
                Statement(s)
                break;
        case label:
                Statement(s)
                break;
        default:
                statement(s)
}
```

- Rules to use Switch case
  a. In Switch case value, only byte, short, int, char, enum, String(jdk1.7) data type is allowed.
  b. Every case label must be unique.
  c. Case label data type must be match with the value data type.
  d. If Multiple cases having same execution then you can combine a cases.
     - **Example:**

```
case 1:
case 8:
case 15:
case 23:
    System.out.println("Monday");
    break;
```

LOOPING STATEMENT

While Loop

1. While loop is use to execute a statement or block of statement multiple time.
2. It is a **pre condition check**. That is it will execute the statements from the loop only if the condition is true.
3. Syntax:

```
Variable Declaration and initialization (start point)
while(condition/Boolean expression)
{
    Statement(s)
    Increment/decrement the variable
}
```

Do-while Loop

1. Do-While loop is use to execute a statement or block of statement multiple time.
2. It is a **post condition check**. This will execute the loop at least once even after condition is false.
3. Syntax:

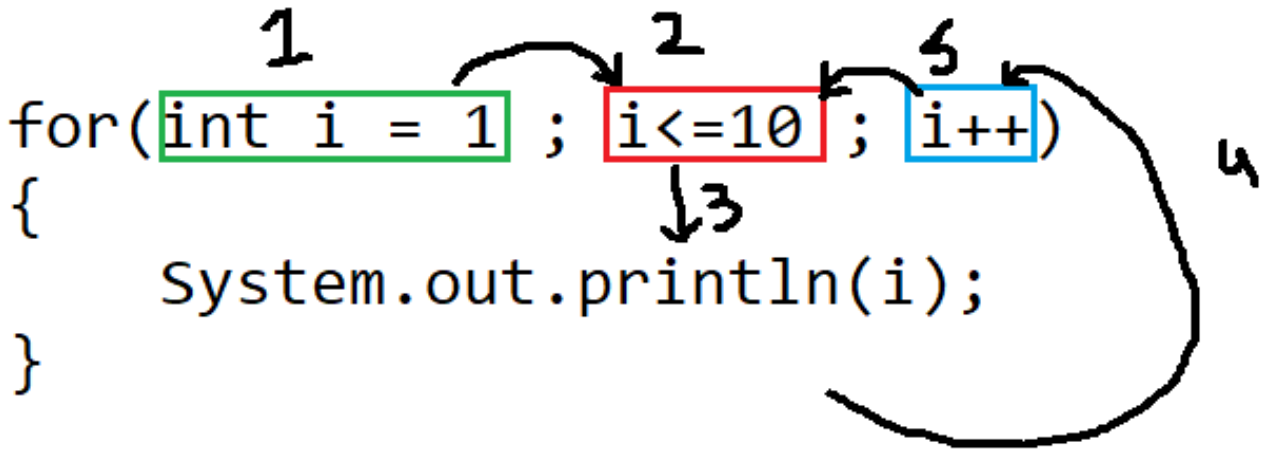
```
Variable Declaration and initialization (start point)
do
{
    Statement(s)
    Increment/decrement of the variable
}
while(condition/Boolean expression);
```

LOOPING STATEMENT

For Loop

1. For loop is also used for executing a statement(s) multiple time.
2. Syntax

```
for(Declare/initialize variable ; Condition ; Increment/Decrement/statements )
{
        Statement(s)
}
```



LOOPING STATEMENT

Nested Loop

1. One loop can be created and execute inside another loop.
2. Nested looping is used to work with table structure. (On the row and column)
3. Nested loop can be achieve using any of the type of loop.
4. Syntax:

```
for(Declare/initialize variable ; Condition ; Increment/Decrement)   // outer loop // rows
{
        for(Declare/initialize variable ; Condition ; Increment/Decrement) //inner loop //column
        {
        }
}
```

Enhanced For Loop / For Each Loop

1. This loop is use to iterate/get collection values one by one.
2. Using Enhance for loop you can get all the values from the collection. It internally maintains the index.
3. Syntax:

```
for( Variable x : collection_object)
{

}
```

TASK

Task-1

WAP to print following Output (using switch)
Create char variable
If char is ‘M’ or ‘m’ then print Monday
If char is ‘T’ or ‘t’ then print Tuesday or Thursday
If char is ‘W’ or ‘w’ then print Wednesday
If char is ‘S’ or ‘s’ then print Saturday or Sunday
If char is ‘F’ or ‘f’ then print Friday

Task-2

Print the table of number.
If number is 5
Output: 5

10
15
20
.
.
.
50

Task-3 using for loop

WAP to Print the even numbers from 1-50
Output: 2

4
6
.
.
.
.
50

Task-4

Print the following pattern

1
1 2
1 2 3
1 2 3 4
1 2 3 4 5