Module no : 3 Operators & Control Statements 14/10/2022

Agenda :-

1. Arithmatic operators.
2. Relational Operators.
3. Assignment vs Equality
4. Compound Assignment
5. Postfix & Prefix operators
6. Branching - if else
7. Branching - Switchcase
8. Logical Operators.

Arithmetic Operators :

Definition : Arithmetic operators are used to perform arithmetic operations on variables and data.

Program :

```
class Main {
    public static void main (String args []) {
        int a = 2;
        int b = 4;
        int add = a + b;
        int sub = a - b;
        int mul = a * b;
        double div = a / b;
        system. out. println ("add = " + add);
        System. out. println ("sub = " + sub);
        System. out. println ("Mul = " + mul);
        System. out. println ("div = " + div);
    }
}
```

Arithmetic Operators:

Program:

```
public class Main{
    public static void main(String[] args){
        int a = 2;
        int b = 5;
        int add = a+b;
        int sub = a-b;
        int mul = a*b;
        float div = a / (float)b;
        int mod = a % b;
        System.out.println("Add = " + add);
        System.out.println("Sub = " + sub);
        System.out.println("Mul = " + mul);
        System.out.println("Div = " + div);
        System.out.println("Mod = " + mod);
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe"
Add = 7
Sub = -3
Mul = 10
Div = 0.4
Mod = 2

Process finished with exit code 0
```

Relational Operators :-

Definition : Relational operators are used to check some relationship between two operands.

Program :

```
class Main {
    public static void main (String args[]) {
        int a = 5;
        int b = 3;
        System.out.println (a<b);
    }
}
```

Operator	Description	Example
==	is equal to ?	100 == 50 returns false
>	Greater than ?	10 > 5 returns true
<	Less than ?	10 < 5 returns false
>=	Greater than or equal to ?	10 >= 5 returns true
<=	Less than or equal to ?	10 <= 5 returns false
!=	Not equal to ?	10 != 5 returns true

Relational Operators:

Program:

```
public class Main{
    public static void main(String[] args){
        int a = 2;
        int b = 5;
        System.out.println("a == b ?     "+(a == b));
        System.out.println("a <= b ?     "+(a <= b));
        System.out.println("a >= b ?     "+(a >= b));
        System.out.println("a != b ?     "+(a != b));
        System.out.println("a < b ?      "+(a < b));
        System.out.println("a > b ?      "+(a > b));
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe
a == b ?     false
a <= b ?     true
a >= b ?     false
a != b ?     true
a < b ?      true
a > b ?      false

Process finished with exit code 0
```

Assignment vs Equality :-

① The `=` is an assignment operator is used to assign the value on the right to the variable on the left.

② The `==` is the equality operator which is used to check whether two items are equal in value.

Program & Example :-

```
class Main {
    public static void main(String args[]) {
        int x = 5;  // x is assigned to a value 5
        float y = 2.3; // Y is assigned to a value 2.3
        System.out.println(x == 5); // comparing or checking
        System.out.println(y == 2); // comparing or checking
    }
}
```

Branching - if else :-

→ The if statement is used to test a condition. It checks boolean condition : true or false.

→ It can optionally have an else if and an else statement attached with it as well.

Program :-

```
class Main {
    public static void main (String args[]) {
        String weather = "rainy";
        if (weather == "rainy") {          → condition which outputs boolean
            System.out.println("Take an umbrella");
        }
        else if (weather == "Sunny") {
            System.out.println("Wear Sunglasses");
        }
        else { System.out.println("Check weather forecast");
        }
    }
}
```

Branching – if else:

Program:

```
import java.util.Scanner;
public class Main{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter a Number to Check Whether it is Positive or Negative");
        int numberFromUser = sc.nextInt();
        if (numberFromUser < 0){
            System.out.println("The Number you had entered is Negative");
        } else if (numberFromUser > 0) {
            System.out.println("The Number you had entered is Positive");
        } else {
            System.out.println("The Number you had entered is neither Positive nor Negative (0)");
        }
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Progra
Enter a Number to Check Whether it is Positive or Negative
0
The Number you had entered is neither Positive nor Negative (0)

Process finished with exit code 0
```

Example - [Odd or Even checker] :

Given an integer N, check whether it is odd or even.

Sample input : N = 5 , Sample output : Odd.

Solution program :-

```
import java.util.Scanner;
class Main {
    public static void main (String args[]) {
        Scanner sc = new Scanner (System.in);
        int a = sc.nextInt ();
        if (a % 2 == 0) {
            System.out.println ("Even");
        } else {
            System.out.println ("Odd");
        }
    }
}
```

Example [Odd or Even Checker]:

Program:

```
package com.company;

import java.util.Scanner;
public class Main{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter a Number to Check Whether it is Odd or Even:");
        int numberFromUser = sc.nextInt();
        if (numberFromUser % 2 == 0){
            System.out.println("The Number you had entered is Even");
        } else {
            System.out.println("The Number you had entered is Odd");
        }
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Pr
Enter a Number to Check Whether it is Positive or Negative
3
The Number you had entered is Odd

Process finished with exit code 0
```

Switch Case :

Definition : In Java, a switch statement is used to transfer control to a particular block of code, based on the value of the variable being ~~created~~ tested.

Program :-

```
class Main {
    public static void main(String args[]) {
        String weather = "Rainy";
        switch (weather) {
            case "rainy" : // in case weather is rainy
                System.out.println("Take an umbrella");
                break;
            case "sunny" : // in case weather is sunny
                System.out.println("Wear Sunglasses");
                break;
            default : // if it is neither rainy nor sunny
                System.out.println("Check weather forecast");
        }}
}
```

Switch Case:

Program:

```
package com.company;

public class Main{
    public static void main(String[] args){
        String weather = "sunny";

        switch (weather){
            case "rainy":
                System.out.println("Take an umbrella");
                break;
            case "sunny":
                System.out.println("Wear coolers");
                break;
            default:
                System.out.println("Check the weather forecasting");
        }
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe"
Wear coolers

Process finished with exit code 0
```

Example - [Days in a month]:

Question:

Given number of the month, print its name and no. of days as output. (Assume non-leap year).

Sample Input:

N = 6

Sample Output:

June

30 days

Program:

```
package com.company;

import java.util.Scanner;
public class Main{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the no. of. the. month which you need to check:");
        int monthNumber = sc.nextInt();

        switch (monthNumber){
            case 1:
                System.out.println("January");
                System.out.println("31 days");
                break;
            case 2:
                System.out.println("Febraury");
                System.out.println("28 days");
                break;
            case 3:
                System.out.println("March");
                System.out.println("31 days");
                break;
            case 4:
                System.out.println("April");
                System.out.println("30 days");
                break;
            case 5:
                System.out.println("May");
                System.out.println("31 days");
                break;
            case 6:
                System.out.println("June");
                System.out.println("30 days");
                break;
            case 7:
                System.out.println("July");
                System.out.println("31 days");
```

```
                break;
            case 8:
                System.out.println("August");
                System.out.println("31 days");
                break;
            case 9:
                System.out.println("September");
                System.out.println("30 days");
                break;
            case 10:
                System.out.println("October");
                System.out.println("31 days");
                break;
            case 11:
                System.out.println("November");
                System.out.println("30 days");
                break;
            case 12:
                System.out.println("December");
                System.out.println("31 days");
                break;
            default:
                System.out.println("Enter the number from 1 to 12");
        }
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C
Enter the no. of. the. month which you need to check:
6
June
30 days

Process finished with exit code 0
```

Ternary Operator :-

Definition :- Ternary operator is a condensed form of if-else statement which evaluates a condition and executes the code based on the evaluated condition.

Syntax :- `result = (condition) ? expression1 : expression2 ;`

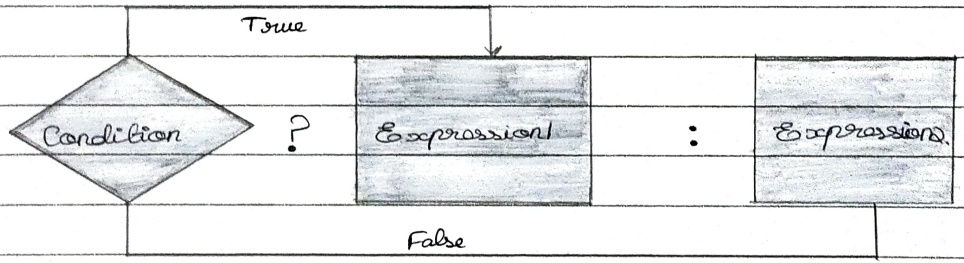
where ; result : The final variable which gets the value.

condition : This is the test condition statement that gets evaluated to true or false.

expression1 : If condition gets evaluated to true then expression1 is assigned to result.

expression2 : If condition gets evaluated to false then expression2 is assigned to result

Flowchart :-



Ternary Operator:

Program:

```
package com.company;

import java.util.Scanner;
public class Main{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the number to check whether it is Even or not:");
        int number = sc.nextInt();
        boolean isEven = (number%2 == 0) ? true : false ;                    // ternary operator
        String answer = (isEven) ? "Even" : "Odd";                           // ternary operator
        System.out.println("The Entered number is : " + answer);
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:
Enter the number to check whether it is Even or not:
845468464
The Entered number is : Even

Process finished with exit code 0
```

Logic Operators :-

Definition :- Logical operators are used making decisions in programming based on other conditions.

Types :-
1. AND
2. OR
3. NOT.

Logical AND - [&] :-

Definition :- The AND operator takes two expressions and returns True if both the Expressions are True, else it returns false It is Denoted by "&".

Expression 1	Expression 2	Result
True	True	True
True	False	False
False	True	False
False	False	False

Logical Short Circuit AND - && :-

Definition :- Short circuit AND behaves just like Logical AND however, it only evaluates the second expression if the first expression yields true. It is denoted by "&&".

Logical (or) - "|" :-

Definition :- The OR operator takes two expressions and returns true if either one of the two expressions is true, else false. it is denoted by "|".

Expression 1	Expression 2	Result
True	True	True
True	False	True
False	True	True
False	False	False

Logical Short Circuit OR - || :-

Definition : Short circuit OR behaves just like Logical OR however, it only evaluates the second expression if the first expression yields false. It is denoted by "||".

Logical NOT - ! :-

Definition :- The Logical NOT operator return the negation of a boolean value. That is, it returns true for false and vice-versa. It is denoted by "!".

Logical AND:

Program:

```
package com.company;

import java.util.Scanner;
public class Main{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter your age to identify whether you are eligible to vote or not :");
        int age = sc.nextInt();
        String nationality = "Indian";
        String result = ((age >= 18) & (nationality == "Indian")) ? "You are eligible to vote" : "You are not
eligible to vote" ;
        System.out.println(result);
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\
Enter your age to identify whether you are eligible to vote or not :
19
You are eligible to vote

Process finished with exit code 0
```

Logical Short Circuit AND - &&:

Program:

```
package com.company;

import java.util.Scanner;
public class Main{
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter your age to identify whether you are eligible to vote or not :");
        int age = sc.nextInt();
        String nationality = "Indian";
        String result = ((age >= 18) && (nationality == "Indian")) ? "You are eligible to vote" : "You are not
eligible to vote" ;
        System.out.println(result);
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\
Enter your age to identify whether you are eligible to vote or not :
19
You are eligible to vote

Process finished with exit code 0
```

Logical OR - |:

Program:

```
package com.company;

public class  Main{
    public static void main(String[] args){
        String org = "Scaler";
        String result = ((org == "Scaler") | (org == "great learning")) ? "Your certificate is eligible" : "Your
certificate is not valid" ;
        System.out.println(result);
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe"
Your certificate is eligible

Process finished with exit code 0
```

Logical Short Circuit OR - ||:

Program:

```
package com.company;

public class  Main{
    public static void main(String[] args){
        String org = "Scaler";
        String result = ((org == "Scaler") || (org == "great learning")) ? "Your certificate is eligible" : "Your
certificate is not valid" ;
        System.out.println(result);
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe"
Your certificate is eligible

Process finished with exit code 0
```

Logical NOT - !:

Program:

```
package com.company;

public class  Main{
    public static void main(String[] args){
        String org = "Scaler";
        String result = ((org == "Scaler") || (org == "great learning")) ? "Your certificate is eligible" : "Your
certificate is not valid" ;
        System.out.println(result);
    }
}
```

Output:

```
"C:\Program Files\Java\jdk-21\bin\java.exe"
Hello World

Process finished with exit code 0
```