SNS COLLEGE OF ENGINEERING

Kurumbapalayam (Po), Coimbatore – 641 107

An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade

Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

COURSE NAME : 23CSB101- OBJECT ORIENTED PROGRAMMING

I YEAR /II SEMESTER

Unit I – INTRODUCTION TO OOP AND JAVA

Topic : CONTROL STATEMENTS

Control Statements

Decision Making statements

- if if-else nested-if if-else-if switch-case

Jump Statements

– break, continue, return

Loop statements

- for loop while loop do-while loop

Decision Making statements

Types :

- if
- if-else
- nested-if
- if-else-if
- switch-case
- jump – break, continue, return

Decision Making statements

```
if(condition)  {
 // Statements to execute if condition is true
}


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

Decision Making statements

if

```
if(condition)  {
 // Statements to execute if condition is true
}
```

if-else

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

Decision Making statements

nested-if

```
if (condition1)  {
// Executes when condition1 is true
if (condition2)
{
// Executes when condition2 is true
}
}
```

Decision Making statements

if-else-if ladder

```
if (condition1) {
    // code to be executed if condition1 is true
} else if (condition2) {
    // code to be executed if condition2 is true
} else {
    // code to be executed if all conditions are false
}
```

Decision Making statements

Switch Case

```
switch (expression) {
    case value1:
        // code to be executed if expression == value1
        break;
    case value2:
        // code to be executed if expression == value2
        break;
    // more cases…
 default:
        // code to be executed if no cases match
}
```

Jump Statements

break, continue and **return**.

break: Terminate a sequence in a switch statement. To exit a loop.

continue: it is useful to force an early iteration of a loop.

return : The return statement is used to explicitly return from a method. That is, it causes program control to transfer back to the caller of the method.

Looping Statements

for loop

- initialize the loop variable, check the condition, and increment/decrement

```
for(initialization, condition, increment/decrement) {
//block of statements
}
```

Looping Statements

Nested for:

```
for (initialization; termination; increment) {
    // Outer loop block
    for (initialization; termination; increment) {
        // Inner loop block
    }
}
```

Looping Statements

for-each Loop

The for-each loop is used to traverse array or collection in Java. It is easier to use than simple for loop because we don't need to increment value and use subscript notation.

It works on the basis of elements and not the index. It returns element one by one in the defined variable.

General form:

```
for(data_type variable : array_name)
{
//code to be executed
}
```

```
int arr[]={12,23,44,56,78};
    //Printing array using for-each loop
    for(int i:arr){
        System.out.println(i);
    }
```

Looping Statements

Labelled for Loop

A labelled for loop in Java is a for loop that has been assigned a label to provide a way to identify a block of code, making it possible to break out of or continue an outer loop from within a nested loop. This feature is particularly useful when working with nested loops and we need to control the flow of the outer loop from within an inner loop. Labels enhance the control flow in complex looping structures, allowing for more precise and flexible loop management.

Looping Statements

Infinitive for Loop

An infinitive for loop in Java is a loop that has no termination condition, or the condition is always true, causing the loop to run indefinitely until the program is manually terminated or interrupted by a break statement or an exception. This type of loop is used when you want to create a continuous loop that keeps running until an external condition or user interaction dictates otherwise.

If we use a pair of semicolons (;;) in the for loop, it will be infinitive for loop.

Looping Statements

Labelled for Loop

labelname:

for(initialization; condition; increment/decrement)

{

//code to be executed

}

```
aa:
        for(int i=1;i<=3;i++){
            bb:
                for(int j=1;j<=3;j++){
                    if(i==2&&j==2){
                        break aa;
                    }
```

Infinitive for Loop

```
for(;;){
//code to be executed
}
```

```
 for(;;)
{
        System.out.println("infinitive loop");
    }
```

Decision Making statements

```
// Java program to illustrate if statement without curly block
import java.util.*;
class Exif {
    public static void main(String args[])
    {
        int i = 10;

        if (i < 15)
// part of if block(immediate one statement after if condition)
            System.out.println("Inside If block");
// always executes as it is outside of if block
        System.out.println("10 is less than 15");
// This statement will be executed as if considers one statement
//by default again below statement is outside of if block
        System.out.println("I am Not in if");
    }
}
```

Output:

Inside If block
10 is less than 15
I am Not in if

Decision Making statements

```
// Java program to demonstrate
// the working of if-else statement
import java.util.*;

class Exifelse {
    public static void main(String args[])
    {
        int i = 10;

        if (i < 15)
            System.out.println("i is smaller than 15");
        else
            System.out.println("i is greater than 15");
    }
}
```

Output:

i is smaller than 15

Decision Making statements

```
// Java program to demonstrate the
//  working of if-else-if ladder
import java.util.*;

class Exifelseif {
    public static void main(String args[])
    {
        int i = 20;
        if (i == 10)
            System.out.println("i is 10");
        else if (i == 15)
            System.out.println("i is 15");
        else if (i == 20)
            System.out.println("i is 20");
        else
            System.out.println("i is not present");
    }
}
```

Output:

i is smaller than 15
i is smaller than 12 too

Decision Making statements

```
// Java program to demonstrate the working of nested-if statement
import java.util.*;
class Exnestedif {
    public static void main(String args[])
    {
        int i = 10;
        if (i == 10 || i < 15) {
            // First if statement
            if (i < 15)
                System.out.println("i is smaller than 15");
// Nested - if statement Will only be executed if statement above  it is true
            if (i < 12)
                System.out.println( "i is smaller than 12 too");
        }
        else { System.out.println("i is greater than 15");
        }
    }
}
```

Output:

i is 20

Decision Making statements

```
// Java program to demonstrates the
//  working of switch statements
import java.io.*;

class Exswitch {
    public static void main(String[] args)
    {
        int num = 20;
        switch (num) {
        case 5:
            System.out.println("It is 5");
            break;
        case 10:
            System.out.println("It is 10");
            break;
case 15:
            System.out.println("It is 15");
            break;
        case 20:
            System.out.println("It is 20");
            break;
        default:
            System.out.println("Not present");
        }
    }
}
```

Output:

It is 20

Decision Making statements

```
// Java program to demonstrates the use of continue in an if statement
import java.util.*;
class Excontinue{
    public static void main(String args[])
    {
        for (int i = 0; i < 10; i++) {

            // If the number is even
            // skip and continue
            if (i % 2 == 0)
                continue;

            // If number is odd, print it
            System.out.print(i + " ");
        }
    }
}
```

Output:

1 3 5 7 9

Decision Making statements

```
// Java program to demonstrate the use of return
import java.util.*;
public class Exreturn{
    public static void main(String args[])
    {
        boolean t = true;
        System.out.println("Before the return.");
        if (t)
            return;
        // Compiler will bypass every statement
        // after return
        System.out.println("This won't execute.");
    }
}
```

Output:

Before the return.

Decision Making statements

```
public class ContinueExample {

public static void main(String[] args) {

for(int i = 0; i<= 2; i++) {

for (int j = i; j<=5; j++) {

if(j == 4) {
continue;
}
System.out.println(j);
}
}
}

}
```

Output:

0
1
2
3
5
1
2
3
5
2
3
5

Looping Statements

```
//A Java program to demonstrate the use of labeled for loop
public class LabeledForExample {
public static void main(String[] args) {
    //Using Label for outer and for loop
    aa:
        for(int i=1;i<=3;i++){
            bb:
                for(int j=1;j<=3;j++){
                    if(i==2&&j==2){
                        break aa;
                    }
                    System.out.println(i+" "+j);
                }
        }
}
}
```

Looping Statements

Example:

```
//Java For-each loop example which prints the
//elements of the array
public class ForEachExample {
public static void main(String[] args) {
    //Declaring an array
    int arr[]={12,23,44,56,78};
    //Printing array using for-each loop
    for(int i:arr){
        System.out.println(i);
    }
}
}
```

Output:

```
12
23
44
56
78
```

Looping Statements

Example:

```
public class Sumnatural {
public static void main(String[] args) {
int sum = 0;
for(int j = 1; j<=10; j++) {
sum = sum + j;
}
System.out.println("The sum of first 10 natural numbers is " + sum);
}
}
```

Looping Statements

Example:

```
public class NestedForExample {
public static void main(String[] args) {
//loop of i
for(int i=1;i<=3;i++){
//loop of j
for(int j=1;j<=3;j++){
        System.out.println(i+" "+j);
}//end of i
}//end of j
}
}
```

Looping Statements

Example:

```
public class PyramidExample {
public static void main(String[] args) {
for(int i=1;i<=5;i++){
for(int j=1;j<=i;j++){
        System.out.print("* ");
}
System.out.println();//new line
}
}
}
```

Looping Statements

Example:

```
//Java program to demonstrate the use of infinite for loop
//which prints an statement
public class ForExample {
public static void main(String[] args) {
    //Using no condition in for loop
    for(;;){
        System.out.println("infinitive loop");
    }
}
}
```


Thank
you