**LOOPS IN JAVA**

* while Loop
* do-while Loop
* for Loop

**while Loop (Entry check loop)**

* Java while loop is used to iterate a part of the program several times.
* If the number of iterations is not fixed, it is recommended to use while loop.

**Syntax**

while(condition)

{

//statement

//increment/decrement statement

}

**Example: Direct declaration value**

public class WhileExample1{

public static void main(String[] args){

int i = 1;

while(i<11){

System.out.println(i);

i++;

}

System.out.println(“The final value is:”+ i);

}

}

**Example: User defined value**

package com.demo.Example;  
import java.util.Scanner;

public class WhileExample2{

public static void main(String[] args){

Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("Enter the value: ");  
 int num = sc.nextInt();

while(num<=15){

System.out.println(num);

++num;

}

System.out.println(“The final value is:”+ num);

}

}

**do-while (Exit check loop)**

The java do-while loop is used to iterate a part of the program several times.

Use it if the number of iterations is not fixed and you must have to execute the loop at least once.

**Syntax**

do{

//statement

//increment/decrement statement

}while(condition);

**Example: Direct declaration value**

public class DoWhileExample1{

public static void main(String[] args){

int i = 1;

do{

System.out.println(i);

i++;

}while(i<11);

System.out.println(“The final value is:”+ i);

}

}

**Example: User defined value**

package com.demo.Example;  
import java.util.Scanner;

public class DoWhileExample2{

public static void main(String[] args){

Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("Enter the value: ");  
 int num = sc.nextInt();

do{

System.out.println(num);

++num;

}while(num<15);

System.out.println(“The final value is:”+ num);

}

}

**for loop (Entry controlled loop)**

* The java for loop is used to iterate a part of the program several times.
* If the number of iterations is fixed, it is recommended to use for loop.

**3 Types of for loop**

* Simple for loop
* For-each or enhanced for loop
* Labeled for loop.

**Simple for loop**

1. **Initialization:** It is the initial condition which is executed once when the loop starts. Here, we can initialize the variable, or we can use an already initialized variable. It is an optional condition.
2. **Condition:** It is the second condition which is executed each time to test the condition of the loop. It continues execution until the condition is false. It must return boolean value either true or false. It is an optional condition.
3. **Increment/Decrement**: It increments or decrements the variable value. It is an optional condition.
4. **Statement:** The statement of the loop is executed each time until the second condition is false.

**Syntax**

for(initialization; condition; increment/decrement){

//statement

}

**Example: Direct declaration value**

public class ForExample{

public static void main(String[] args){

for(int i = 1; i<=15; i++){

System.out.println(i);

}

}

}

**Example: User defined value**

package com.demo.Example;  
import java.util.Scanner;

public class ForExample1{

public static void main(String[] args){

int num;

int sum = 0;

Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("Enter the value: ");  
 int num = sc.nextInt();

for(int i = 1; i<=num; i++){

sum += i;

}

System.out.println(“The sum of num natural no.: ”+ num + “ natural number is: ”+ sum);

}

}

**Nested for loop**

* If we have a for loop inside the another for loop, it is known as nested for loop.
* The inner loop executes completely whenever outer loop executes.

**Syntax**

for(initialization; condition; increment/decrement){

//statement

for(initialization; condition; increment/decrement){

//statement

}

}

**Example: Direct declaration value**

public class NestedForLoop{

public static void main(String[] args){

for(int i = 1; i<=4; i++){

for(int j = 1; j<=4; j++){

System.out.println(i+“ ”+j);

}

}

}

}

**Example: pattern1**

package com.demo.JavaCode;  
  
public class Pattern1 {  
 public static void main(String[] args){  
 for (int i = 1; i<=5; i++){  
 for (int j = 1; j<=5; j++){  
 //System.out.print(j+" ");  
 System.out.print(i+" ");  
 }  
 System.out.println();  
 }  
 }  
}

**Example: pattern2**

package com.demo.JavaCode;  
  
public class Pattern2 {  
 public static void main(String[] args){  
 for (int i = 1; i<=5; i++){  
 //for (int j = 1; j<=i; j++)  
 for (int j = 1; j<i+1; j++){  
 System.out.print(j+" ");  
 }  
 System.out.println();  
 }  
 }  
}

**Example: pattern3**

package com.demo.JavaCode;  
  
public class Pattern3 {  
 public static void main(String[]args){  
 int num = 6; //rows  
 for (int i =0; i<=num; i++){  
 //for (int j = num; j>=i; j--)  
 for (int j = 1; j<=num-i; j++){  
 System.out.print("\* ");  
 }  
 System.out.println();  
 }  
 }  
}

**Example: Diamond Pattern**

import java.util.Scanner;

public class DiamondPattern {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Ask the user for the number of rows in the top half of the diamond

System.out.print("Enter the number of rows for the diamond pattern: ");

int rows = scanner.nextInt();

// Top half of the diamond

for (int i = 1; i <= rows; i++) {

// Print leading spaces

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

System.out.print(" ");

}

// Print stars

for (int k = 1; k <= (2 \* i - 1); k++) {

System.out.print("\*");

}

System.out.println();

}

// Bottom half of the diamond

for (int i = rows - 1; i >= 1; i--) {

// Print leading spaces

for (int j = rows; j > i; j--) {

System.out.print(" ");

}

// Print stars

for (int k = 1; k <= (2 \* i - 1); k++) {

System.out.print("\*");

}

System.out.println();

}

}

}