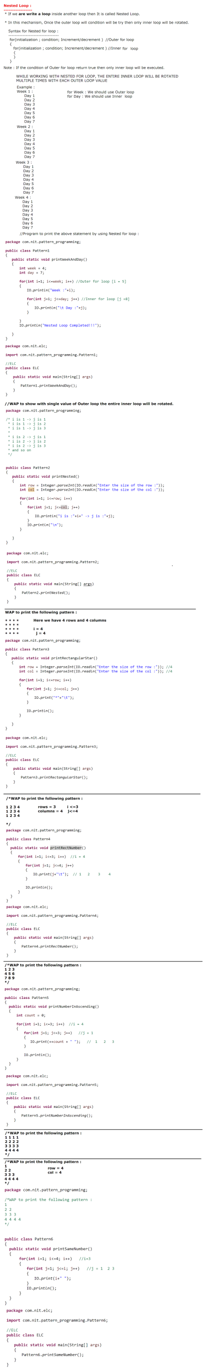

Nested Loop :

* If we **are write a loop** inside another loop then It is called Nested Loop.

* In this mechanism, Once the outer loop will condition will be try then only inner loop will be rotated.

Syntax for Nested for loop :

```
for(initialization ; condition; Increment/decrement )  //Outer for loop
{
   for(initialization ; condition; Increment/decrement ) //Inner for  loop
   {
   }
}
```

Note : If the condition of Outer for loop return true then only inner loop will be executed.

WHILE WORKING WITH NESTED FOR LOOP, THE ENTIRE INNER LOOP WILL BE ROTATED
MULTIPLE TIMES WITH EACH OUTER LOOP VALUE

```
Example :
Week 1 :                    for Week : We should use Outer loop
      Day 1                 for Day : We should use Inner  loop
      Day 2
      Day 3
      Day 4
      Day 5
      Day 6
      Day 7
Week 2 :
      Day 1
      Day 2
      Day 3
      Day 4
      Day 5
      Day 6
      Day 7
Week 3 :
      Day 1
      Day 2
      Day 3
      Day 4
      Day 5
      Day 6
      Day 7
Week 4 :
      Day 1
      Day 2
      Day 3
      Day 4
      Day 5
      Day 6
      Day 7
```

//Program to print the above statement by using Nested for loop :

```
package com.nit.pattern_programming;

public class Pattern1
{
   public static void printWeekAndDay()
   {
      int week = 4;
      int day = 7;

      for(int i=1; i<=week; i++) //Outer for loop [i = 5]
      {
         IO.println("Week :"+i);

         for(int j=1; j<=day; j++) //Inner for loop [j =8]
         {
            IO.println("\t Day :"+j);
         }

      }
      IO.println("Nested Loop Completed!!!");

   }
}
```

```
package com.nit.elc;

import com.nit.pattern_programming.Pattern1;

//ELC
public class ELC
{
    public static void main(String[] args)
    {
       Pattern1.printWeekAndDay();
    }
}
```

//WAP to show with single value of Outer loop the entire inner loop will be rotated.

```
package com.nit.pattern_programming;

/* i is 1 -> j is 1
 * i is 1 -> j is 2
 * i is 1 -> j is 3
 *
 * i is 2 -> j is 1
 * i is 2 -> j is 2
 * i is 2 -> j is 3
 * and so on
 */


public class Pattern2
{
   public static void printNested()
   {
      int row = Integer.parseInt(IO.readln("Enter the size of the row :"));
      int col = Integer.parseInt(IO.readln("Enter the size of the col :"));

      for(int i=1; i<=row; i++)
      {
         for(int j=1; j<=col; j++)
         {
            IO.println("i is :"+i+" -> j is :"+j);
         }
         IO.println("\n");
      }

   }
}
```

```
package com.nit.elc;

import com.nit.pattern_programming.Pattern2;

//ELC
public class ELC
{
    public static void main(String[] args)
    {
       Pattern2.printNested();
    }
}
```

WAP to print the following pattern :

```
* * * *        Here we have 4 rows and 4 columns
* * * *
* * * *        i = 4
* * * *         j = 4
```

```
package com.nit.pattern_programming;

public class Pattern3
{
   public static void printRectangularStar()
   {
      int row = Integer.parseInt(IO.readln("Enter the size of the row :")); //4
      int col = Integer.parseInt(IO.readln("Enter the size of the col :")); //4

      for(int i=1; i<=row; i++)
      {
         for(int j=1; j<=col; j++)
         {
            IO.print("*"+"\t");
         }

         IO.println();
      }

   }
}
```

```
package com.nit.elc;

import com.nit.pattern_programming.Pattern3;

//ELC
public class ELC
{
    public static void main(String[] args)
    {
       Pattern3.printRectangularStar();
    }
}
```

/*WAP to print the following pattern :

```
1 2 3 4        rows = 3      i <=3
1 2 3 4        columns = 4   j<=4
1 2 3 4

*/
```

```
package com.nit.pattern_programming;

public class Pattern4
{
  public static void printRectNumber()
  {
     for(int i=1; i<=3; i++)  //i = 4
     {
        for(int j=1; j<=4; j++)
        {
           IO.print(j+"\t");  // 1   2    3    4
        }

        IO.println();
     }
  }
}
```

```
package com.nit.elc;

import com.nit.pattern_programming.Pattern4;

//ELC
public class ELC
{
    public static void main(String[] args)
    {
       Pattern4.printRectNumber();
    }
}
```

/*WAP to print the following pattern :

```
1 2 3
4 5 6
7 8 9
*/
```

```
package com.nit.pattern_programming;

public class Pattern5
{
  public static void printNumberInAscending()
  {
     int count = 0;

     for(int i=1; i<=3; i++)  //i = 4
     {
        for(int j=1; j<=3; j++)   //j = 1
        {
           IO.print(++count + " ");   //  1   2   3
        }

        IO.println();
     }
  }
}
```

```
package com.nit.elc;

import com.nit.pattern_programming.Pattern5;

//ELC
public class ELC
{
    public static void main(String[] args)
    {
       Pattern5.printNumberInAscending();
    }
}
```

/*WAP to print the following pattern :

```
1 1 1 1
2 2 2 2
3 3 3 3
4 4 4 4
*/
```

/*WAP to print the following pattern :

```
1
2 2                  row = 4
3 3 3                col = 4
4 4 4 4
*/
```

```
package com.nit.pattern_programming;

/*WAP to print the following pattern :
1
2 2
3 3 3
4 4 4 4
*/


public class Pattern6
{
  public static void printSameNumber()
  {
     for(int i=1; i<=4; i++)   //i=3
     {
        for(int j=1; j<=i; j++)   //j = 1  2 3
        {
           IO.print(i+" ");
        }
        IO.println();
     }
  }
}
```

```
package com.nit.elc;

import com.nit.pattern_programming.Pattern6;

//ELC
public class ELC
{
    public static void main(String[] args)
    {
       Pattern6.printSameNumber();
    }
}
```