GAURAV KISHOR PATIL
ROLL NO: 54
DIV:2 BATCH:C

Experiment No. 14
Multithreading.
Date of Performance:6/10/23
Date of Submission:6/10/23

Aim:- To implement the concept of Multithreading.

Objective:- To implement multithreading in a program to display multiplication tables of 2 and 10. Create two independent threads to display multiplication tables of 2 and 10 independently.

Theory :-

Multithreading is a Java feature that allows concurrent execution of two or more parts of a program for maximum utilization of CPU. Each part of such program is called a thread. So, threads are light-weight processes within a process.

Threads can be created by using two mechanisms:

1. Extending the Thread class
2. Implementing the Runnable Interface

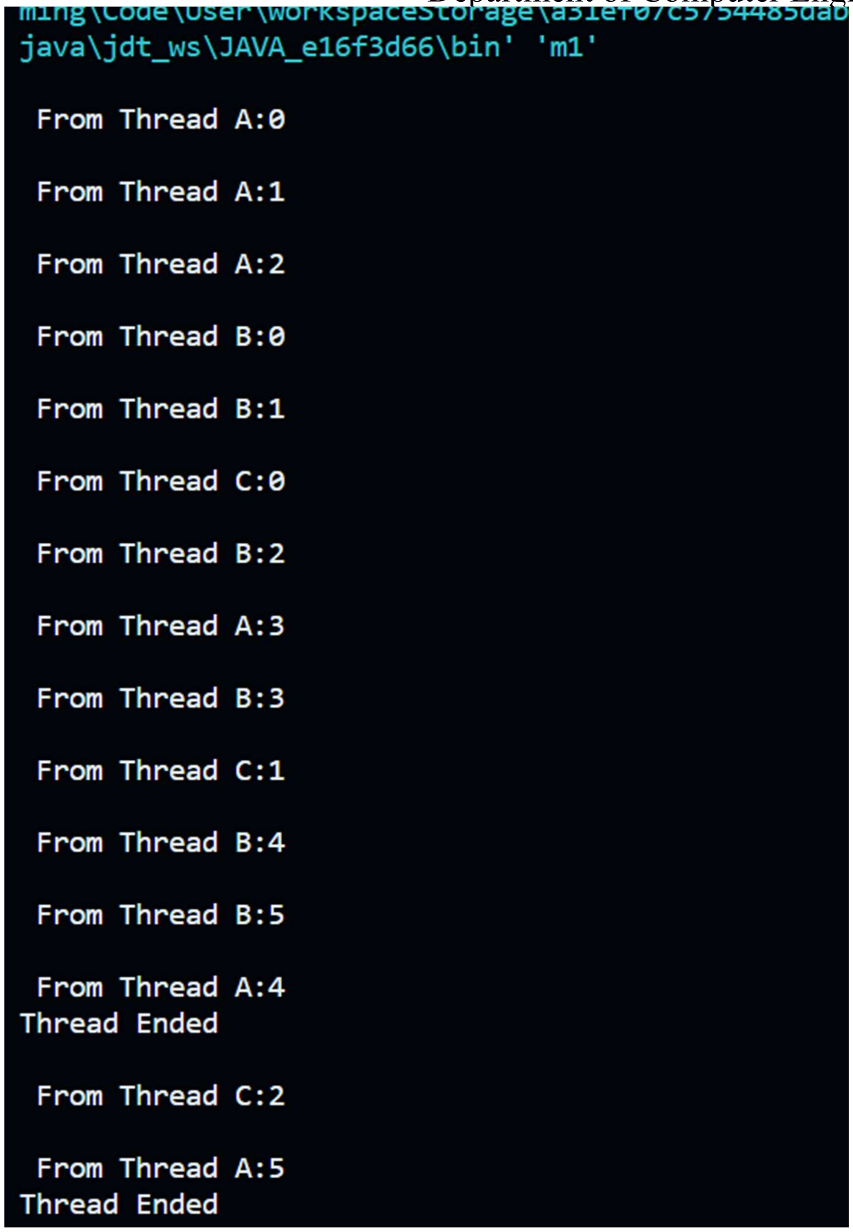
Thread creation by extending the Thread class:

We create a class that extends the java.lang.Thread class. This class overrides the run() method available in the Thread class. A thread begins its life inside run() method. We create an object of our new class and call start() method to start the execution of a thread. Start() invokes the run() method on the Thread object.

Code :-

1) From Thread Class

```
class A extends Thread{
public void run(){
for (int i=0;i<=5;i++) {
   System.out.println("\n From Thread A:"+i);
}
System.out.println("Thread Ended");
}
}
class B extends Thread{
   public void run(){
for (int i=0;i<=5;i++) {
   System.out.println("\n From Thread B:"+i);
}
System.out.println("Thread Ended");
}}
class C extends Thread{
   public void run(){
for (int i=0;i<=5;i++) {
   System.out.println("\n From Thread C:"+i);
}
System.out.println("Thread Ended");

}


}

public class m1 {

  public static void main(String[] args) {

    new A().start();

    new B().start();

    new C().start();

  }


}
```

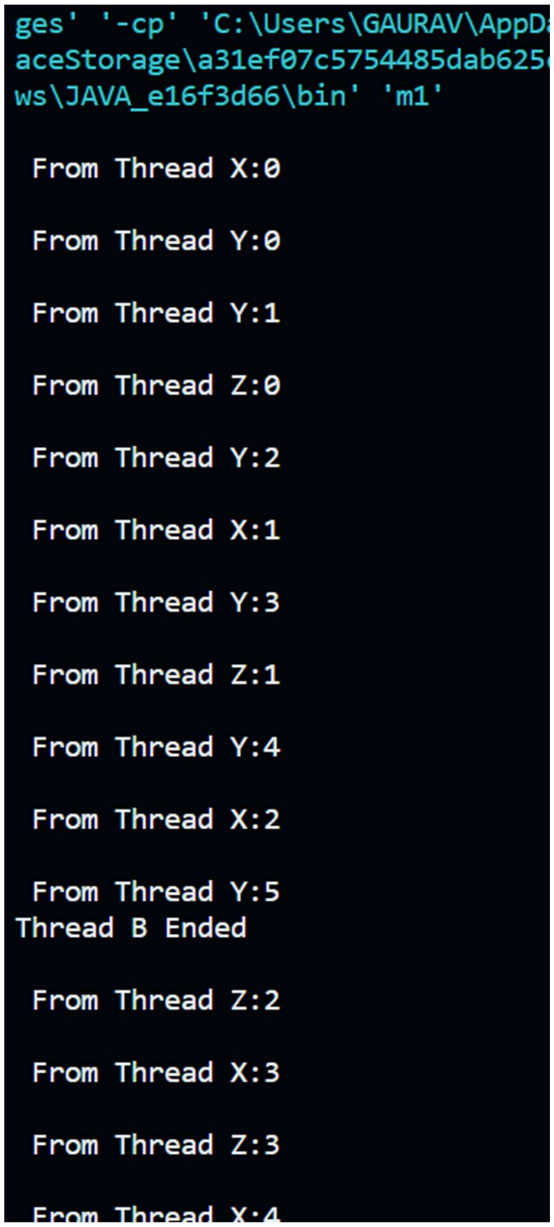


```
ming\Code\User\workspaceStorage\a31ef07c57544850ab
java\jdt_ws\JAVA_e16f3d66\bin' 'm1'

 From Thread A:0

 From Thread A:1

 From Thread A:2

 From Thread B:0

 From Thread B:1

 From Thread C:0

 From Thread B:2

 From Thread A:3

 From Thread B:3

 From Thread C:1

 From Thread B:4

 From Thread B:5

 From Thread A:4
Thread Ended

 From Thread C:2

 From Thread A:5
Thread Ended
```

2)From Interface Runnable

```
class X implements Runnable {
   public void run() {
      for (int i = 0; i <= 5; i++) {
         System.out.println("\n From Thread X:" + i);
      }
      System.out.println("Thread A Ended");
   }
}
```

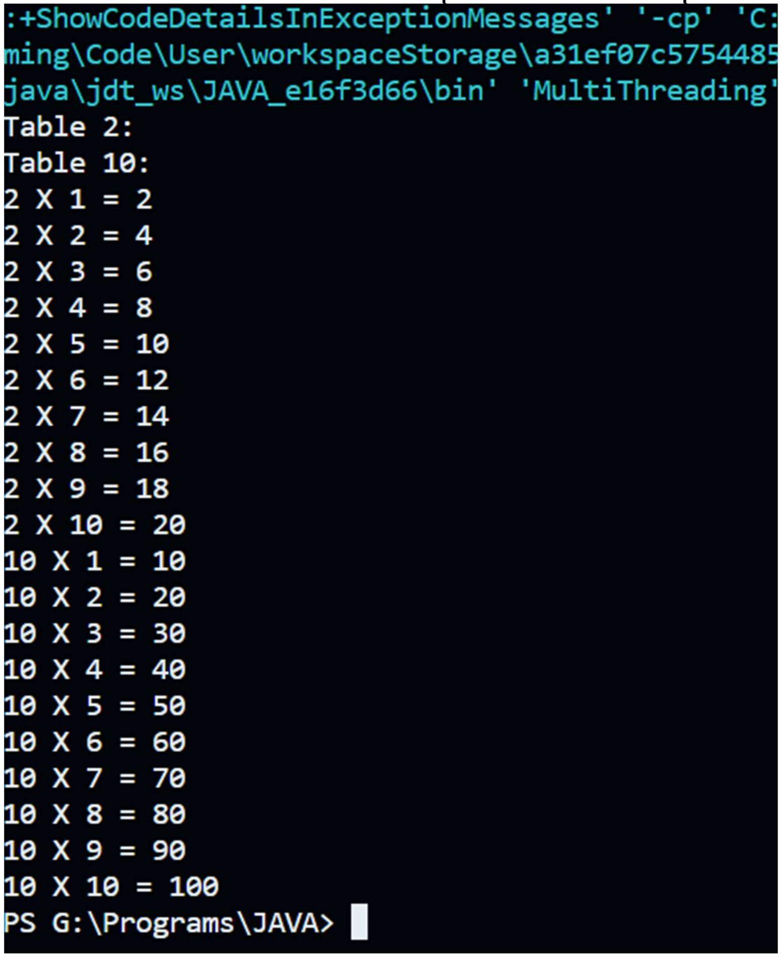
```
class Y implements Runnable {
    public void run() {
        for (int i = 0; i <= 5; i++) {
            System.out.println("\n From Thread Y:" + i);
        }
        System.out.println("Thread B Ended");
    }
}

class Z implements Runnable {
    public void run() {
        for (int i = 0; i <= 5; i++) {
            System.out.println("\n From Thread Z:" + i);
        }
        System.out.println("Thread C Ended");
    }
}

class m1 {
    public static void main(String[] args) {
        Runnable taskA = new X();
        Runnable taskB = new Y();
        Runnable taskC = new Z();

        Thread threadA = new Thread(taskA);
        Thread threadB = new Thread(taskB);
        Thread threadC = new Thread(taskC);

        threadA.start();
        threadB.start();

        threadC.start();

    }

}
```



```
ges' '-cp' 'C:\Users\GAURAV\AppD
aceStorage\a31ef07c57544485dab625
ws\JAVA_e16f3d66\bin' 'm1'

 From Thread X:0

 From Thread Y:0

 From Thread Y:1

 From Thread Z:0

 From Thread Y:2

 From Thread X:1

 From Thread Y:3

 From Thread Z:1

 From Thread Y:4

 From Thread X:2

 From Thread Y:5
Thread B Ended

 From Thread Z:2

 From Thread X:3

 From Thread Z:3

 From Thread X:4
```

3)Problem Statement

```
class table2 extends Thread{


public void run(){

for(int i=1;i<=10;i++){
   int c=2*i;
System.out.println("2 X "+i+" = "+c);
}
}


}
class table10 extends Thread{
   public void run(){
try{
   sleep(5000);
}catch(Exception e){

}
for(int i=1;i<=10;i++){
   int c=10*i;
System.out.println("10 X "+i+" = "+c);
}
}

}
class MultiThreading{
   public static void main(String[] args) {
      System.out.println("Table 2:");
     new table2().start();
     System.out.println("Table 10:");
      new table10().start();
   }
}
```



```
:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:
ming\Code\User\workspaceStorage\a31ef07c5754485
java\jdt_ws\JAVA_e16f3d66\bin' 'MultiThreading'
Table 2:
Table 10:
2 X 1 = 2
2 X 2 = 4
2 X 3 = 6
2 X 4 = 8
2 X 5 = 10
2 X 6 = 12
2 X 7 = 14
2 X 8 = 16
2 X 9 = 18
2 X 10 = 20
10 X 1 = 10
10 X 2 = 20
10 X 3 = 30
10 X 4 = 40
10 X 5 = 50
10 X 6 = 60
10 X 7 = 70
10 X 8 = 80
10 X 9 = 90
10 X 10 = 100
PS G:\Programs\JAVA>
```

Conclusion:-

Implementing multithreaded Java code is reasonably straightforward. Even converting existing single-threaded Java GUI code to a multithreaded format is not difficult. The results are actually quite impressive:Multithreading is a feature in Java that concurrently executes two or more parts of the program for utilizing the CPU at its maximum. The part of each program is called Thread which is a lightweight process.According to the definition, it can be deduced that it expands the concept of multitasking in the program by allowing certain operations to be divided into smaller units using a single application.Each Thread operates concurrently and permits the execution of multiple tasks inside the same application.