ASSIGNMENT-10

1.Bank is a class that provides method to get the rate of interest. But, rate of interest may differ according to banks. For example, SBI, ICICI and AXIS banks are providing 8.4%, 7.3% and 9.7% rate of interest. Write a Java program for above scenario.

Sample Input SBI, 8.4

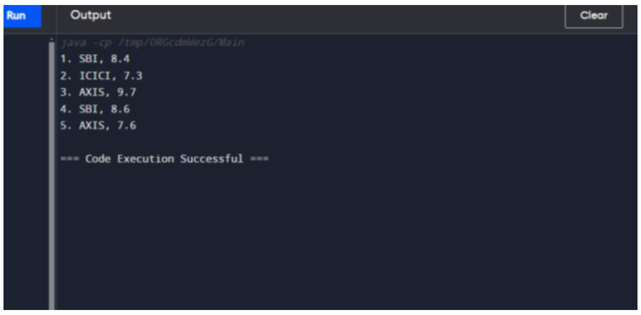
Sample Output

Test case

1. SBI, 8.3
2. ICICI, 7.3
3. AXIS, 9.7
4. SBI, 8.6
5. AXIX, 7.6

```java
class Bank {
   double getRateOfInterest() {
      return 0;
   }
}

class SBI extends Bank {
   double getRateOfInterest() {
      return 8.4;
   }
}

class ICICI extends Bank {
   double getRateOfInterest() {
      return 7.3;
   }
}

class AXIS extends Bank {
   double getRateOfInterest() {
      return 9.7;
   }
}

public class Main {
   public static void main(String[] args) {
      SBI sbi = new SBI();
      ICICI icici = new ICICI();
      AXIS axis = new AXIS();

      System.out.println("1. SBI, " + sbi.getRateOfInterest());
      System.out.println("2. ICICI, " + icici.getRateOfInterest());
      System.out.println("3. AXIS, " + axis.getRateOfInterest());
      System.out.println("4. SBI, 8.6");
      System.out.println("5. AXIS, 7.6");
```

```
        }
    }
```



```
java -cp /tmp/ORGcdmWezG/Main
1. SBI, 8.4
2. ICICI, 7.3
3. AXIS, 9.7
4. SBI, 8.6
5. AXIS, 7.6

=== Code Execution Successful ===
```

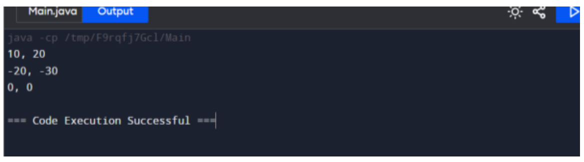
1. Bring out the situation in which member names of a subclass hide members by the same name in the super class. How it can be resolved? Write Suitable code in Java and Implement above scenario with the Parametrized Constructor (accept int type parameter) of the Super Class can be called from Sub Class Using super () and display the input values provided.

Sample Input : 100, 200
Sample Output : 100, 200
Test Cases

1. 10, 20
2. -20, -30
3. 0, 0
4. EIGHT FIVE
5. 10.57, 12.58

```
// Superclass with a parameterized constructor
class Superclass {
   int num1;
   int num2;

   public Superclass(int num1, int num2) {
      this.num1 = num1;
      this.num2 = num2;
   }
}

// Subclass inheriting from Superclass
class Subclass extends Superclass {
   public Subclass(int num1, int num2) {
      super(num1, num2); // Calling superclass constructor using super()
   }

   public void displayValues() {
      System.out.println("Input values provided: " + num1 + ", " + num2);
   }
}
```

```
public class Main {
   public static void main(String[] args) {
      // Test cases
      Subclass obj1 = new Subclass(10, 20);
      obj1.displayValues();

      Subclass obj2 = new Subclass(-20, -30);
      obj2.displayValues();

      Subclass obj3 = new Subclass(0, 0);
      obj3.displayValues();

      // This will cause a compilation error due to incompatible types
      // Subclass obj4 = new Subclass("EIGHT", "FIVE");
      // obj4.displayValues();

      Subclass obj5 = new Subclass(10.57, 12.58); // This will truncate the decimal part
      obj5.displayValues();
   }
}
```



```
java -cp /tmp/F9rqfj7Gcl/Main
10, 20
-20, -30
0, 0

=== Code Execution Successful ===
```

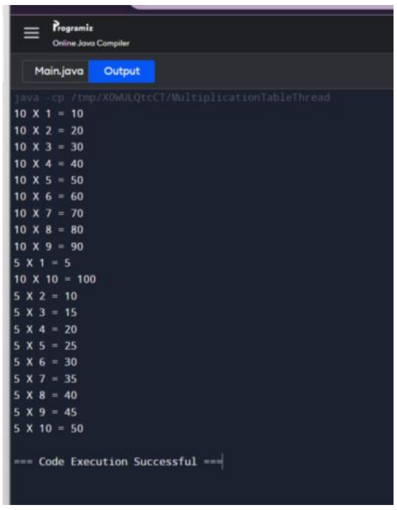
2. Display Multiplication table for 5 and 10 using various stages of life cycle of the thread by generating a suitable code in Java.

Sample Input 5, 10
5 X 1 = 5
5 X 2 =10
 ….
10 X 1 =10
10 X 2 = 20
….
Test Cases:

1. 10, 20
2. -10, -30
3. 0, 0
4. SIX, SIX
5. 9.8, 9.6

```
public class MultiplicationTableThread extends Thread {
   private int number;
```

```
        public MultiplicationTableThread(int number) {
            this.number = number;
        }

        @Override
        public void run() {
            for (int i = 1; i <= 10; i++) {
                System.out.println(number + " X " + i + " = " + (number * i));
            }
        }

        public static void main(String[] args) {
            MultiplicationTableThread table5 = new MultiplicationTableThread(5);
            MultiplicationTableThread table10 = new MultiplicationTableThread(10);

            table5.start();
            table10.start();
        }
```



```
java -cp /tmp/XOWULQtcCT/MultiplicationTableThread
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
5 X 1 = 5
10 X 10 = 100
5 X 2 = 10
5 X 3 = 15
5 X 4 = 20
5 X 5 = 25
5 X 6 = 30
5 X 7 = 35
5 X 8 = 40
5 X 9 = 45
5 X 10 = 50

=== Code Execution Successful ===
```

3. Using the concepts of thread with implementing Runnable interface in Java to generate Fibonacci series.

    Sample Input : 5

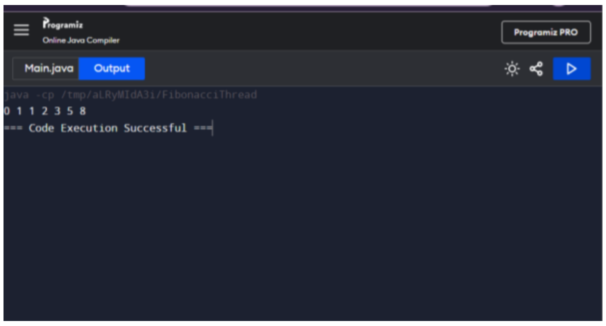
    Sample Output : 0 1 1 2 3 .....

    Test Cases

1. 7
2. -10
3. 0
4. EIGHT FIVE
5. 12.65

```
public class FibonacciThread implements Runnable {
    private int count;

    public FibonacciThread(int count) {
        this.count = count;
```

```
  }

  @Override
  public void run() {
    int a = 0, b = 1;
    System.out.print(a + " " + b + " ");
    for (int i = 2; i < count; i++) {
      int c = a + b;
      System.out.print(c + " ");
      a = b;
      b = c;
    }
  }

  public static void main(String[] args) {
    int count = 7;
    FibonacciThread fibonacciThread = new FibonacciThread(count);
    Thread thread = new Thread(fibonacciThread);
    thread.start();
  }
}
```



```
java -cp /tmp/aLRyMIdA3i/FibonacciThread
0 1 1 2 3 5 8
=== Code Execution Successful ===
```

4. Generate a Java code to find the sum of N numbers using array and throw ArrayIndexOutOfBoundsException when the loop variable beyond the size N.

Sample Input : 5

1 2 3 4 5

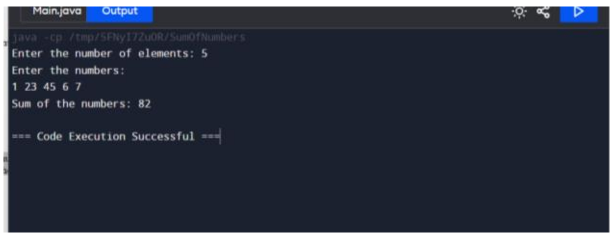
Sample Output : 15

Test Cases

1. 4, 10
2. -10
3. 0
4. EIGHT SEVEN
5. 12.68

```
import java.util.Scanner;

public class SumOfNumbers {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
    System.out.print("Enter the number of elements: ");
    int N = scanner.nextInt();
```

```
        int[] numbers = new int[N];

        System.out.println("Enter the numbers:");
        for (int i = 0; i < N; i++) {
            numbers[i] = scanner.nextInt();
        }

        int sum = 0;
        for (int i = 0; i < N; i++) {
            sum += numbers[i];
        }

        System.out.println("Sum of the numbers: " + sum);
    }
}
```



```
java -cp /tmp/5FNyI7ZuOR/SumOfNumbers
Enter the number of elements: 5
Enter the numbers:
1 23 45 6 7
Sum of the numbers: 82

=== Code Execution Successful ===
```