Objected Oriented Programming Lab

Ashish Wilson

Roll no : 28

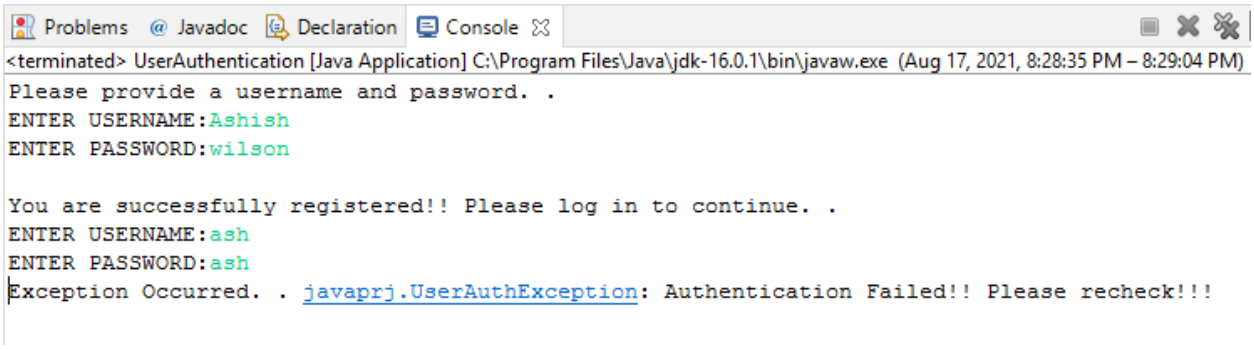
S2MCA_A

3. Write a user defined exception class to authenticate the user name and password.

```
package javaprj;

import java.util.Scanner;

class UserAuthException extends Exception {

		private static final long serialVersionUID = 1L;

		UserAuthException(String s){
			super(s);
		}

	}

public class UserAuthentication {

		public static void main(String[] args) {
			 String username, u_name, password,p_wrd;

			Scanner sc = new Scanner(System.in);
			System.out.print("Please provide a username and password. . ");
			System.out.print("\nENTER USERNAME:");
			username = sc.nextLine();
			System.out.print("ENTER PASSWORD:");
			password = sc.nextLine();
			System.out.print("\nYou are successfully registered!! Please log in to
continue. .");
			System.out.print("\nENTER USERNAME:");
			u_name = sc.nextLine();
			System.out.print("ENTER PASSWORD:");
			p_wrd = sc.nextLine();
			sc.close();
			try
			{
			  authentication(username,password,u_name,p_wrd);
				}catch(Exception e)
				 {
					System.out.println("Exception Occurred. . "+e);
				 }
			  // TODO Auto-generated method stub
```

```
        }

        public static void authentication(String uname, String pwd, String u, String p)
throws UserAuthException{

                if((u=="") || (p="")) {
                        throw new UserAuthException("Fields cannot be empty!!!");
                }
                else if(u.equals(uname) && p.equals(pwd)){
                        System.out.println("Authentication Successful!!!");
                }
                else if((u!=uname) || (p!=pwd)) {
                 throw new UserAuthException("Authentication Failed!! Please
                 recheck!!!");
                }

                else {
                        System.exit(0);
                }

   }

}
```

OUTPUT



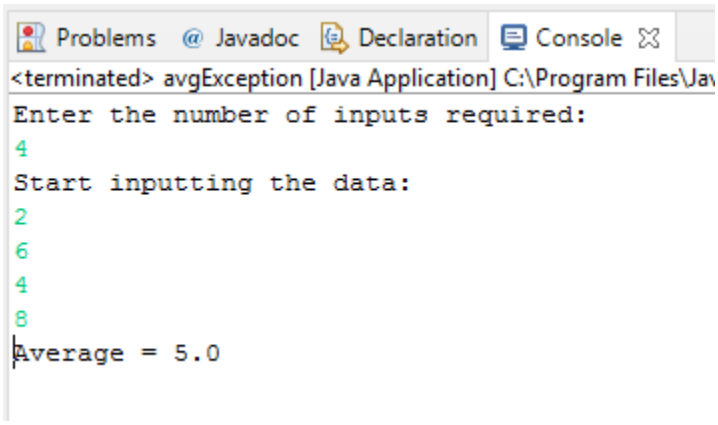
4. Find the average of N positive integers, raising a user defined exception for each negative input.

```
package javaprj;

import java.util.Scanner;
class NegInputException extends Exception {

	private static final long serialVersionUID = 1L;

	NegInputException(String s){
		super(s);
	}

}
public class avgException {

	public static void main(String[] args) {

	    int N;
	    float Sum = 0,avg;
	    Scanner sc=new Scanner(System.in);
	    System.out.println("Enter the number of inputs required:");
	    N = sc.nextInt();
	    float[] numbers = new float[N];
	    System.out.println("Start inputting the data:");

	    for( int i=0; i < N ; i++)
	    {
	        numbers[i]=sc.nextInt();
	        try{
	          if(numbers[i]<0)
	           {
	        throw new NegInputException("Negative inputs not allowed!");
	           }
	          else
	            {
	              Sum += numbers[i];

	            }
	          }catch(NegInputException e)
	            {
	              System.out.println("Exception Occurred. . "+e);
	              System.exit(0);
	            }
	     }
	    sc.close();
```

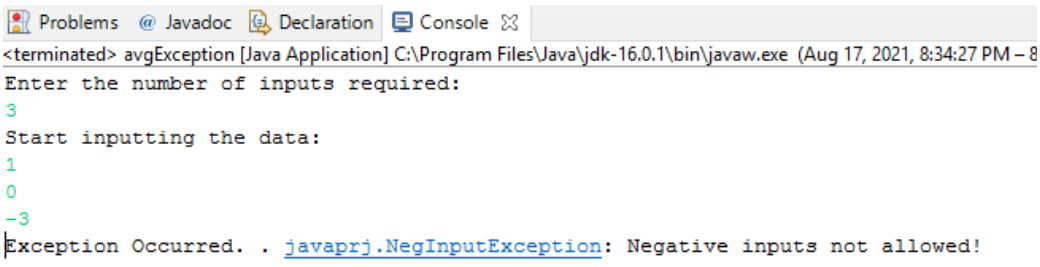
```
            avg = Sum / N;
            System.out.println("Average = "+ avg);

        }


    }
```

OUTPUT

1,



2,

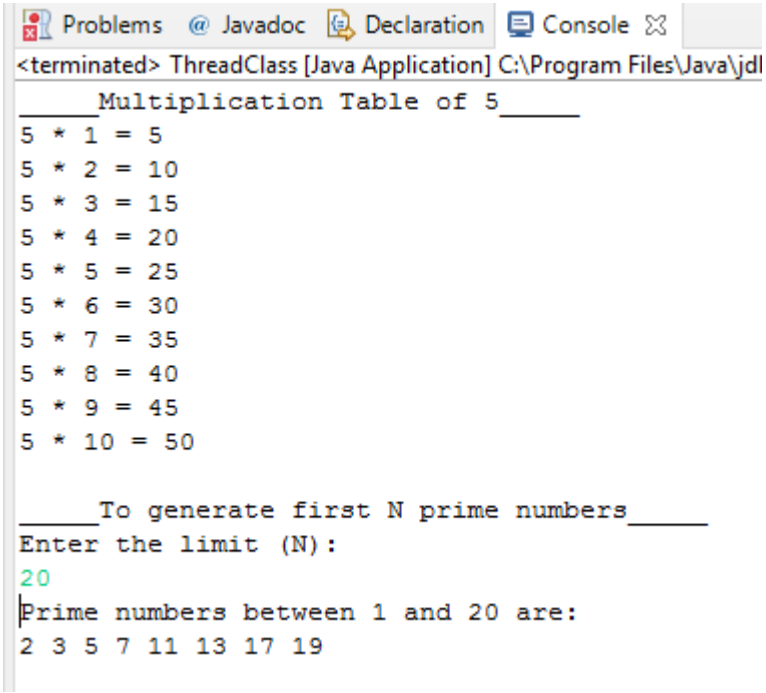


5. Define 2 classes; one for generating multiplication table of 5 and other for displaying first N prime numbers. Implement using threads. (Thread class)

```
package javaprj;

import java.util.Scanner;

class MulTable extends Thread{
  public void run() {
        int num = 5;
        System.out.printf("_____Multiplication Table of 5_____\n");
   for(int i = 1; i <= 10; ++i)
   {
        System.out.printf("%d * %d = %d \n", num, i, num * i);
   }
  }

}

class PrimeNo extends Thread{
        public void run() {
         int i, j,flag;
         Scanner s = new Scanner(System.in);
         System.out.println("\n_____To generate first N prime numbers_____");
         System.out.println("Enter the limit (N):");
         int N = s.nextInt();


         System.out.println("Prime numbers between 1 and " + N + " are:");

           for (i = 1; i <= N; i++)
           {

              if (i == 1 || i == 0)
                 continue;

              flag = 1;

              for (j = 2; j <= i / 2; ++j)
              {
                 if (i % j == 0)
                 {
                    flag = 0;
                    break;
                 }
```

```
            }

            if (flag == 1)
                System.out.print(i + " ");
        }
    }
}


public class ThreadClass {

    public static void main(String[] args) throws InterruptedException {
        MulTable m = new MulTable();
        m.start();
        m.sleep(200);

        PrimeNo p = new PrimeNo();
        p.start();
        p.sleep(200);
        // TODO Auto-generated method stub

    }

}
```

OUTPUT



6. Define 2 classes; one for generating Fibonacci numbers and other for displaying even numbers in a given range. Implement using threads. (Runnable Interface)

```
package javaprj;
import java.util.Scanner;

class Fibonacci implements Runnable{
        public void run(){
                int first = 0, second = 1, next;
                Scanner sc= new Scanner(System.in);
                System.out.println("___TO GENERATE FIBONACCI SERIES____");
                System.out.println("Enter the no.of terms required:");
                int n=sc.nextInt();

                System.out.println("Series Generated!!!");
                for (int i = 1; i <= n; ++i){
                        System.out.print(first + "  ");
                   next = first + second;
                        first = second;
                        second = next;
        }
                }
  }

class EvenNo implements Runnable{
        public  void run(){
                Scanner sc= new Scanner(System.in);
                int lower, upper;
                System.out.println("\n\n_____TO GENERATE EVEN NUMBERS OF GIVEN
RANGE_____");
                System.out.println("Enter the lower limit:");
                lower=sc.nextInt();

                System.out.println("Enter the upper limit:");
                upper=sc.nextInt();

                System.out.println("Even numbers from " + lower + " and " + upper + " are:");
                for (int i = lower; i <= upper; i++){
             if (i%2!=0)
                continue;

            else
                {
                  System.out.print(i+" ") ;
                }
      }
     }
```

```
  }


public class ThreadRunner {
        public static void main(String arg[]) throws InterruptedException
        {
        Fibonacci obj1 = new Fibonacci();
        Thread a=new Thread(obj1);
        a.start();
        a.sleep(2000);
        EvenNo obj2 = new EvenNo();
        Thread b= new Thread(obj2);
        b.start();
        b.sleep(1000);

        }
}
```

OUTPUT