Department of Computer Science & Engineering, SDMCET, Dharwad-2

AOOP Assignment Submission Report

[Submitted as part of CTA Assignment No-1]

Course:	Advanced Object-Oriented Programming	Course Code:	18UCSE508
Semester:	V	Division:	A

Submitted by:

USN:	2SD20CS070	Name:	POOJA BASAVARAJ DAMBAL

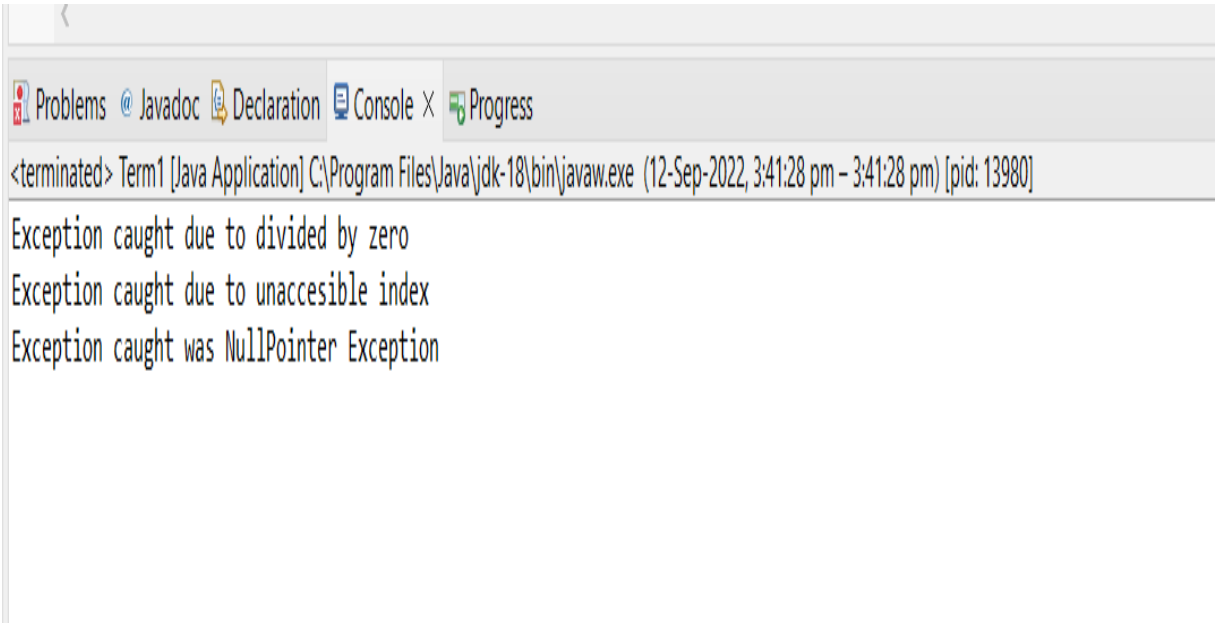
1. Problem Definition:

Q1. Write a Java program to generate and handle any three built-in exceptions and display appropriate error messages.

2. Java Program:

```
package aoop;
import java.util.*;
public class Term1 {


public static void main(String[] args) {
		// TODO Auto-generated method stub
		int d,b=5;
		int a=0;
		int c[]= {0,1,12,34,1};
		String s=null;


		try {
			d=b/a;//value of d is infinity
			System.out.println("res is " +d);
		}
		catch(ArithmeticException ae){
			System.out.println("Exception caught due to divided by zero");
		}
		try {
			System.out.println(c[6]);//array with index 6 is not accessible
		because size is 5
		}
		catch(ArrayIndexOutOfBoundsException aae){
			System.out.println("Exception caught due to unaccesible index");
		}

		try {
			System.out.println(s.toLowerCase());
		}
		catch(NullPointerException ne){
			System.out.printf("Exception caught was NullPointer Exception");
		}

	}

}
```

3. Screen Shots of Execution:



1. Problem Definition:

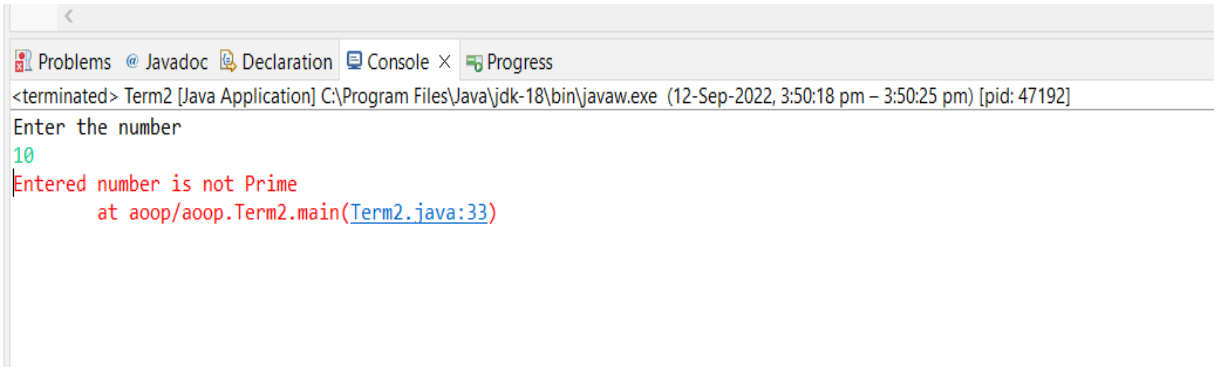
Q2. Write a Java program to read an integer and check whether the number is prime or not. If negative number is entered, throw an exception NegativeNumberNotAllowedException and if entered number is not prime, then throw NumberNotPrimeException.

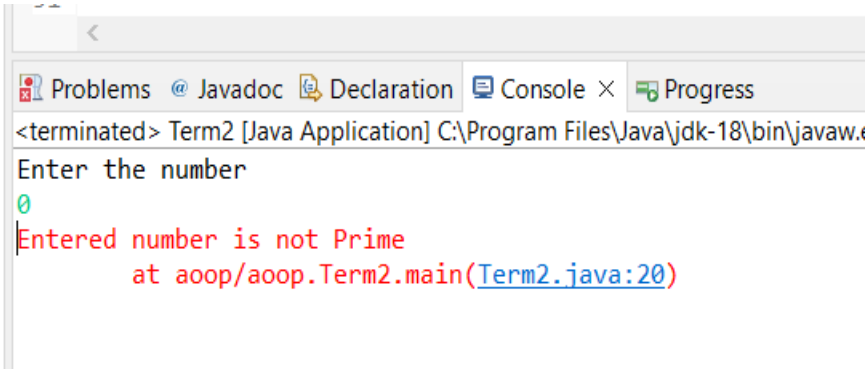
2. Java Program:

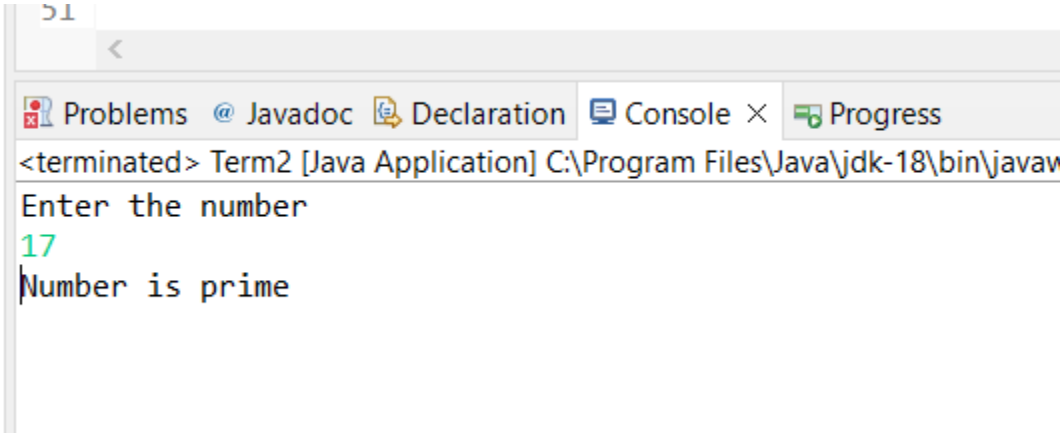
```
package aoop;
import java.util.*;
import java.util.Scanner;

public class Term2 {

	public static void main(String[] args) {
		Scanner sc =new Scanner(System.in);
		// TODO Auto-generated method stub
		int i,count=0;
		System.out.println("Enter the number");
		int num=sc.nextInt();

		try {
		if(num<0) {
		   throw new NegativeNumException("Entered num is Negative");
		}

		else if(num==0||num==1) {
			throw new NotPrimeException("Not Prime");
		}
		else {

		for(i=1;i<=num;i++)
		{
			if(num%i==0)
				count++;
		}

		if(count==2)
			System.out.println("Number is prime ");
		else
			throw new NotPrimeException("Not Prime");

		    }
		}

		catch(NotPrimeException e) {
```

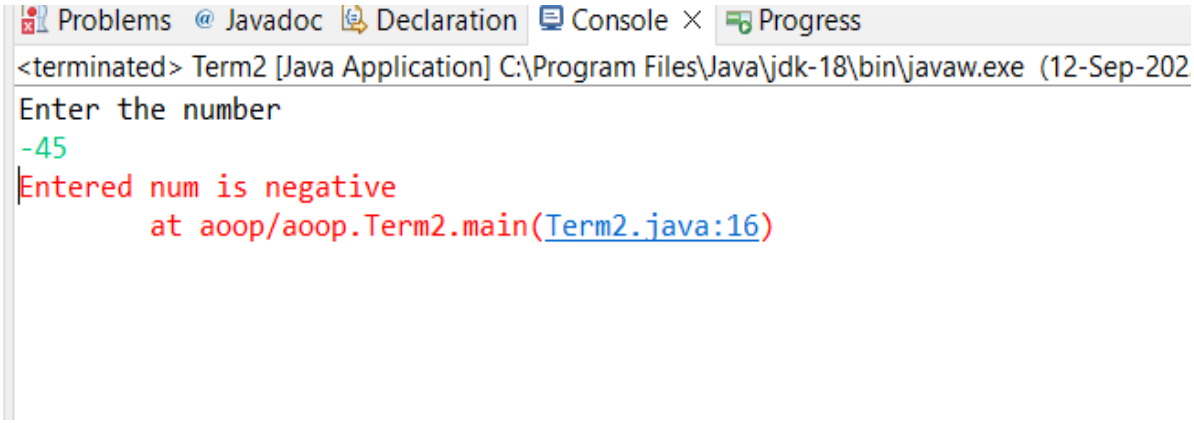
```
                        e.printStackTrace();
                }



                catch(NegativeNumException ne) {

                        ne.printStackTrace();

                }

        }

}


package aoop;
public class NotPrimeException extends Exception {
            String s;
            NotPrimeException(String s){
               this.s=s;
            }

        public String toString() {
          return "Entered number is not Prime";
        }

}

package aoop;
public class NegativeNumException extends Exception {
        String e;
         NegativeNumException(String e){
                this.e=e;
         }

        public String toString() {
                return "Entered num is negative";
        }

}
```

3. Screen Shots of Execution:






Problems
Javadoc
Declaration
Console
Progress
<terminated> Term2 [Java Application] C:\Program Files\Java\jdk-18\bin\javaw
Enter the number
17
Number is prime


Problems
Javadoc
Declaration
Console
Progress
<terminated> Term2 [Java Application] C:\Program Files\Java\jdk-18\bin\javaw.exe (12-Sep-202
Enter the number
-45
Entered num is negative
at aoop/aoop.Term2.main(Term2.java:16)

1. Problem Definition:

Q3. Write a Java program to perform the following operations:

a) Read a line of text

b) Search for a sub-string SDMCET (case insensitive search)

c) If found, then print success message

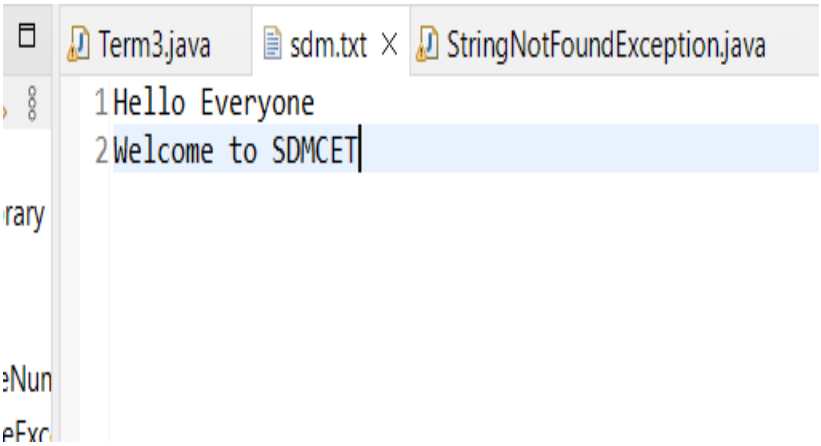
d) Otherwise throw an exception SubStringNotFoundException with appropriate message

2. Java Program:

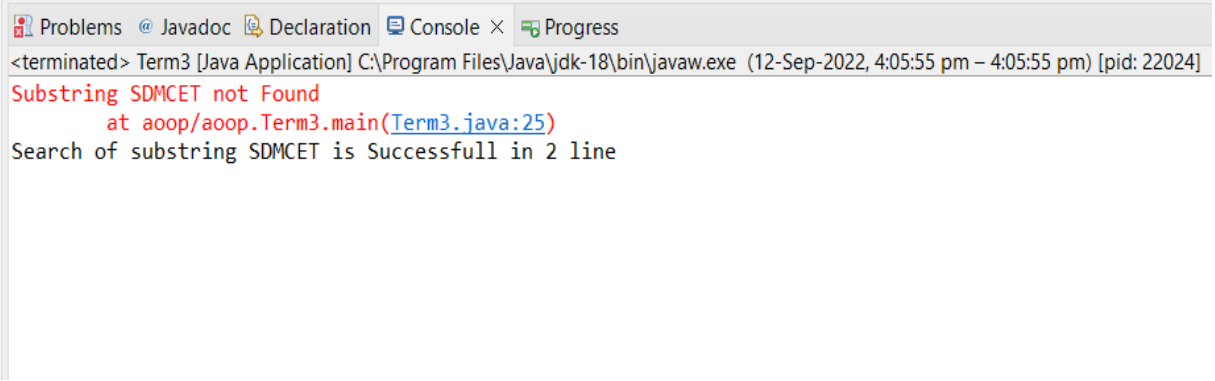
```
package aoop;
import java.io.FileReader;
import java.io.BufferedReader;

public class Term3 {

	public static void main(String[] args) throws Exception {
		// TODO Auto-generated method stub
		FileReader fr=new FileReader("sdm.txt");
		BufferedReader br=new BufferedReader(fr);

		String str;
		String str1="SDMCET";

		int i=0;
		while((str=br.readLine())!=null) {
			i++;
			try {
			if(str.contains(str1)) {
				System.out.println("Search of substring SDMCET is
						Successfull in "+ i +" line");
			}
				else
					throw new StringNotFoundException("String not
							found");
			}
```

```
                    catch(StringNotFoundException se) {
                          se.printStackTrace();

                           }

                }
            }
}
```

```
package aoop;
public class StringNotFoundException extends Exception{
         String s;
         StringNotFoundException(String s){
                this.s=s;
         }
       public  String toString() {
                return "Substring SDMCET not Found";
         }

}
```



3. Screen Shots of Execution:



1. Problem Definition:

Q4. Write a Java program to perform the following operations:

a) Create a file named Alphabets.txt and insert appropriate data into it

b) Read the file and copy all the consonants into another file named Consonants.txt

c) If vowel is encountered, throw an exception VowelNotAllowedException and continue until end of file

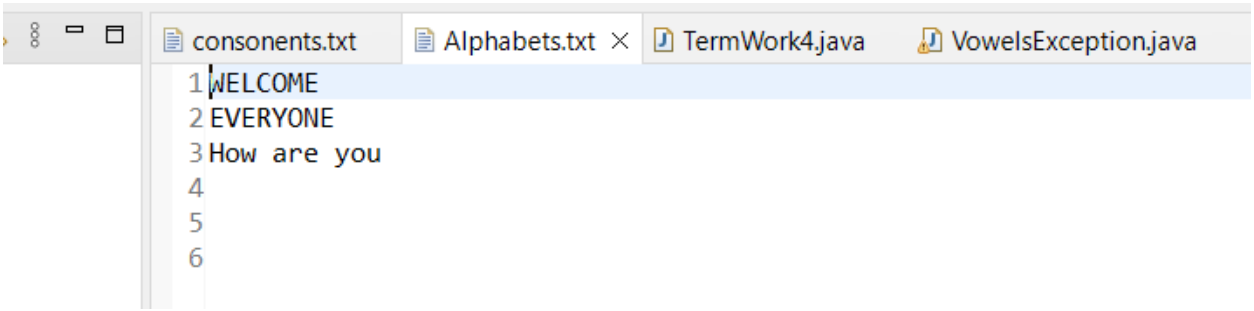
2. Java Program:

```
package aoop;

import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;

public class TermWork4 {

	public static void main(String[] args) throws IOException{
		// TODO Auto-generated method stub
		FileInputStream	fr =new FileInputStream ("Alphabets.txt");
		 FileOutputStream fw =new FileOutputStream("consonents.txt");

				int ch;

		  while(( ch=fr.read())!=-1){
		   try {
	 if(ch=='a'||ch=='e'||ch=='i'||ch=='o'||ch=='u'||ch=='A'||ch=='E'
		||ch=='I'||ch=='O'||ch=='U')	{
		 throw new VowelsException("Vowels Found");
			}
		 else {

				fw.write(ch);
		}
		  }

					catch(VowelsException ve) {
				ve.printStackTrace();
```

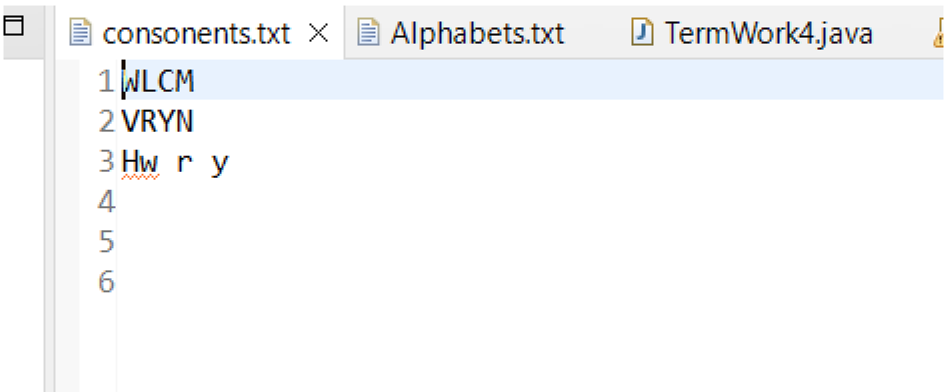
```
		}

	}
```

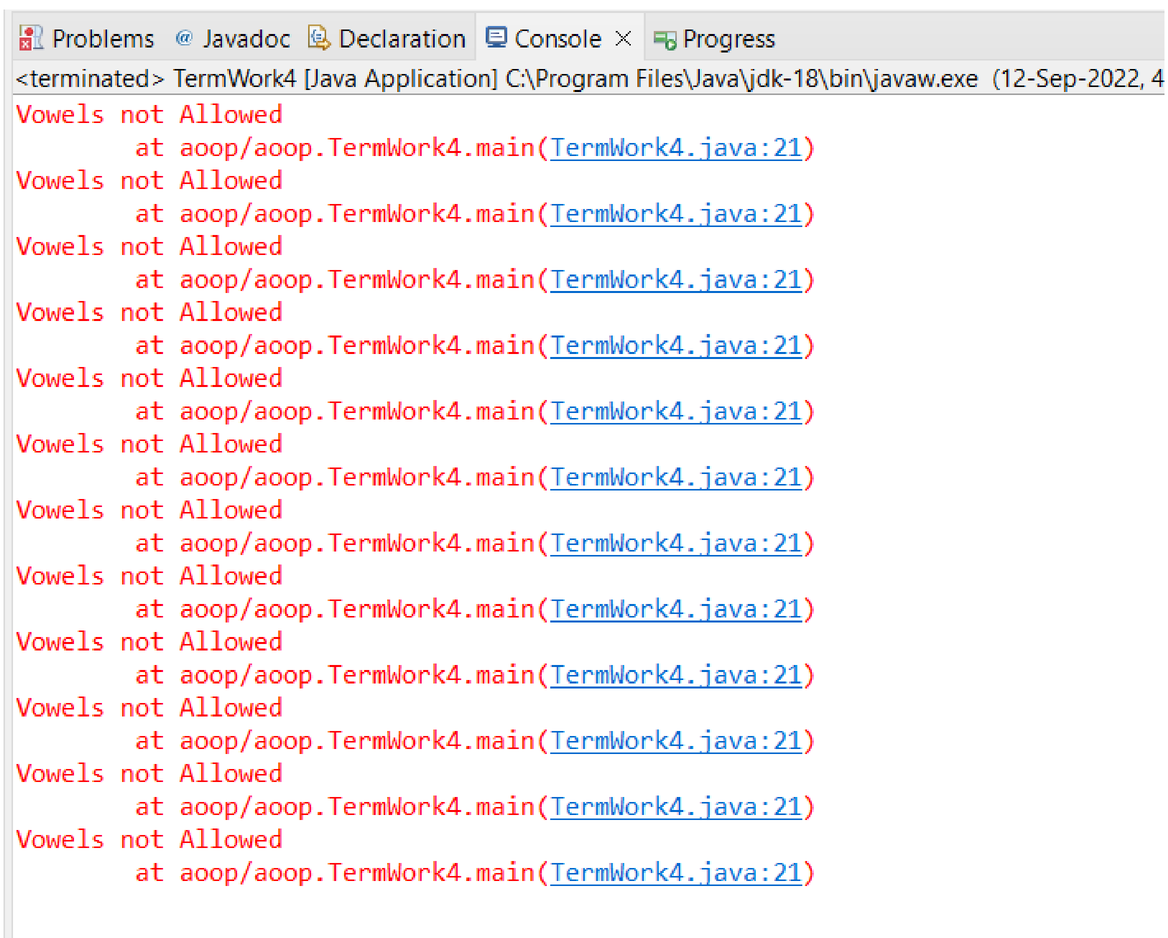
```
                                fr.close();
                                 fw.close();


                        }

                }
```

```
package aoop;
public class VowelsException extends Exception {
	String s;
	VowelsException(String s){
		this.s=s;

	}
  public String toString() {
	  return "Vowels not Allowed";
  }
}
```



3. Screen Shots of Execution:





1. Problem Definition:

Q5. Write a Java program to implement the following scenario:

a) Create a file named Integers.txt and insert n-random integers into it

b) Create three threads T1, T2 and T3 that read n/3 integers in sequence of occurrence of numbers from the file and sort the read n/3 integers

c) Thread T4 waits for all the threads T1, T2 and T3 to complete sorting, then sorts and outputs

the entire list of sorted numbers to another file named SortedIntegers.txt

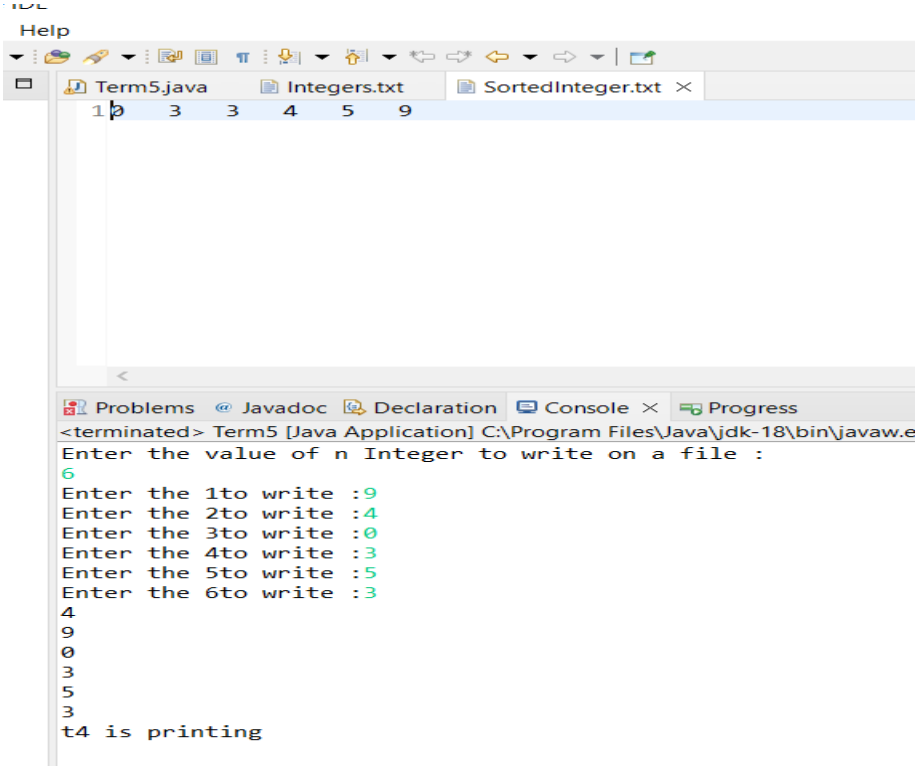
2. Java Program:

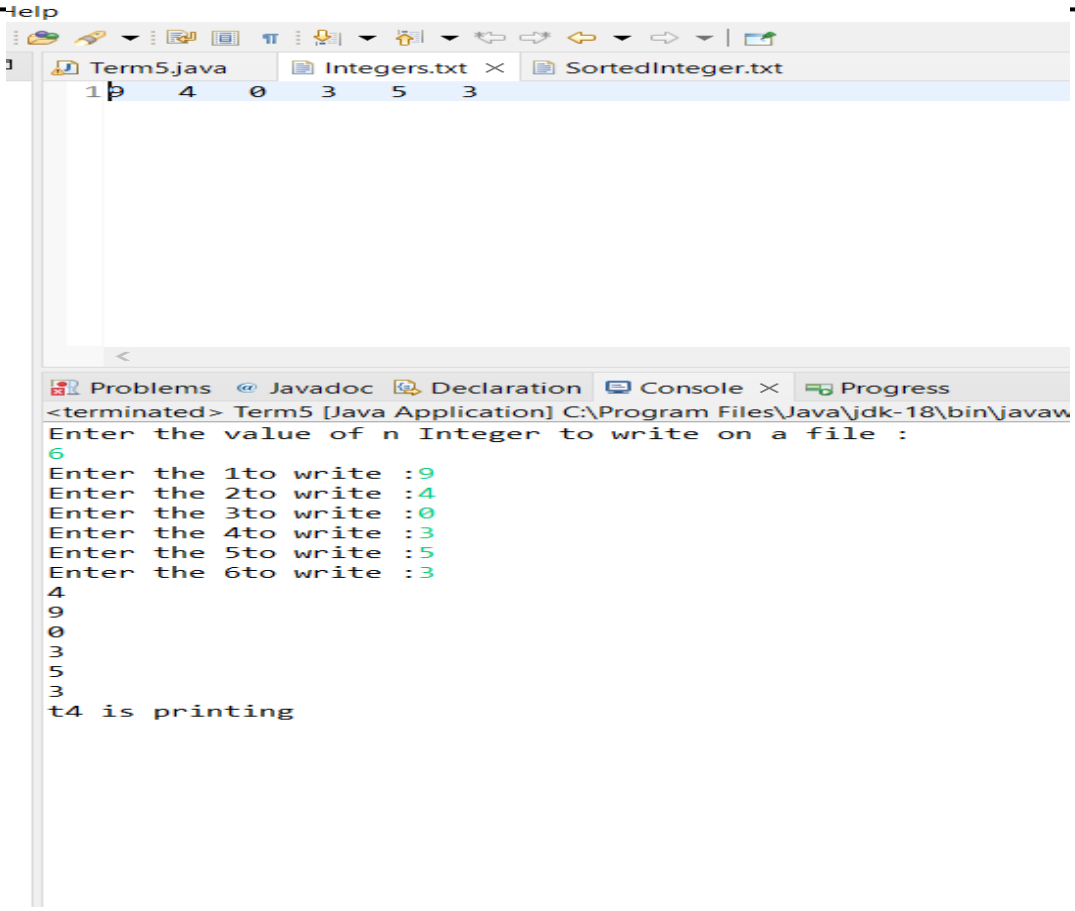
```
package aoop;
import java.util.*;
import java.util.Scanner;
import java.io.*;

public class Term5 {
	public static void main(String[] args) {
		try{
		FileWriter w = new FileWriter("Integers.txt");
		Scanner sc= new Scanner(System.in);


		System.out.println("Enter the value of n Integer to write on a file
			:");
		int n = sc.nextInt();
		for (int i = 0; i < n; i++) {
		    System.out.print("Enter the " + (i + 1) +  "to write :" );
		    int input = sc.nextInt();
		    w.write(input + "\t");
	     }w.close();
	     int i=0;
	     int arr[] = new int[n];
	    File file = new File("Integers.txt");
	    Scanner read = new Scanner(file);
	    while(read.hasNext()){
	     arr[i++] = Integer.valueOf(read.next());
	    }
```

```
Thread t1= new Thread(){

   public void run(){
     Arrays.sort(arr, 0, (arr.length/3));
     for (int j = 0; j < (arr.length/3); j++) {
       System.out.println(arr[j]);
     }

   }

};
Thread t2= new Thread(){

  public void run(){
    Arrays.sort(arr, (arr.length/3), (2*(arr.length/3)));
    for (int j =  (arr.length/3); j < (2*(arr.length/3)); j++) {
      System.out.println(arr[j]);
     }
   }

 };
 Thread t3= new Thread(){

   public void run(){
     Arrays.sort(arr,  (2*(arr.length/3)),(n-1));
     for (int j = (2*(arr.length/3)); j < n; j++) {
       System.out.println(arr[j]);
     }

   }

 };
 Thread t4= new Thread(){

   public void run(){
     Arrays.sort(arr);
     // Arrays.sort(arr,  0,n);

     StringBuilder s = new StringBuilder();
     try{
     FileWriter write  =new FileWriter("SortedInteger.txt");
     System.out.println("t4 is printing");
     for (int j = 0; j < n; j++) {

       s.append(String.valueOf(arr[j]) + "\t");

     }
     write.write(s.toString());
     write.close();
   }catch (Exception e){
     System.out.println(e);
   }
   }

 };
```

```
                t1.start();
                 t1.join();
                 t2.start();
                 t2.join();
                 t3.start();
                 t3.join();
                 t4.start();

                }catch(Exception e){
                       System.out.println(e);
                   }
                 }
}
```

3. Screen Shots of Execution:




Help
Term5.java
Integers.txt
SortedInteger.txt
1 9 4 0 3 5 3
Problems
Javadoc
Declaration
Console
Progress
<terminated> Term5 [Java Application] C:\Program Files\Java\jdk-18\bin\javaw
Enter the value of n Integer to write on a file :
6
Enter the 1to write :9
Enter the 2to write :4
Enter the 3to write :0
Enter the 4to write :3
Enter the 5to write :5
Enter the 6to write :3
4
9
0
3
5
3
t4 is printing