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:	2SD20CS022	Name:	ANUSHA SHASHIDHAR RAIBAGI

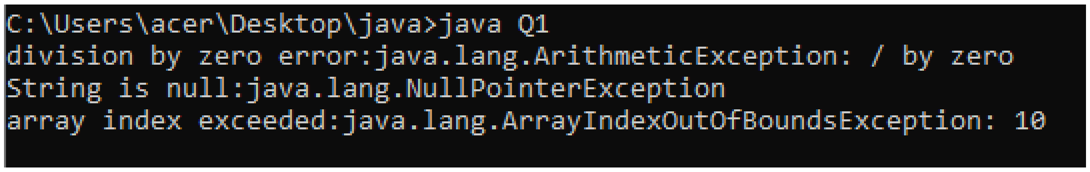
1. Problem Definition:

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

2.Java program:

```
public class Q1{
  public static void main(String[] args){
     int a=10;
     int b=5;
     int c=5;
     String s=null;
     int d[]=new int[5];
     try{
      System.out.println(a/(b-c));
     }
     catch(ArithmeticException ae){
      System.out.println("division by zero error:"+ae);
     }
     try{
      System.out.println(s.length());
     }
     catch(NullPointerException ne){
      System.out.println("String is null:"+ne);
     }
     try{
      d[10]=50;
     }catch(ArrayIndexOutOfBoundsException aoe){
       System.out.println("array index exceeded:"+aoe);
     }
  }
}
```

3.Screenshots of execution:



```
C:\Users\acer\Desktop\java>java Q1
division by zero error:java.lang.ArithmeticException: / by zero
String is null:java.lang.NullPointerException
array index exceeded:java.lang.ArrayIndexOutOfBoundsException: 10
```

1.Problem Definition:

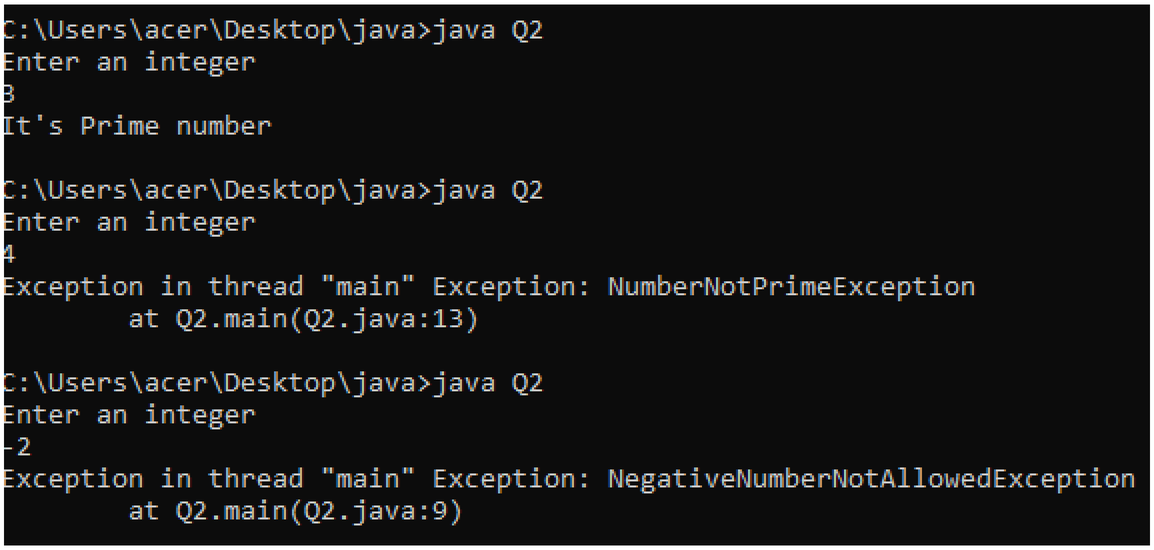
Q2. Write a Java program to read an integer and check whether the number is prime or not. If a negative number is entered, throw an exception NegativeNumberNotAllowedException and if entered number is not prime, then throw a NumberNotPrimeException.

2.Java program:

```
import java.util.Scanner;


 public class Q2{
   public static void main(String[] args)throws numException{
            Scanner sc=new Scanner(System.in);
        System.out.println("Enter an integer");
            int n=sc.nextInt();
        if(n<0){
          throw new numException("NegativeNumberNotAllowedException");
        }
        for(int i=2;i<n;i++){
          if(n%i==0)
            throw new numException("NumberNotPrimeException");
        }
        System.out.println("It's Prime number");
   }
 }
class numException extends Exception{
            String msg;
            public numException(String msg){
               this.msg=msg;
            }
        public String toString(){
          return "Exception: "+msg;
        }
 }
```

3.Screenshots of execution:



```
C:\Users\acer\Desktop\java>java Q2
Enter an integer
3
It's Prime number

C:\Users\acer\Desktop\java>java Q2
Enter an integer
4
Exception in thread "main" Exception: NumberNotPrimeException
        at Q2.main(Q2.java:13)

C:\Users\acer\Desktop\java>java Q2
Enter an integer
-2
Exception in thread "main" Exception: NegativeNumberNotAllowedException
        at Q2.main(Q2.java:9)
```

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

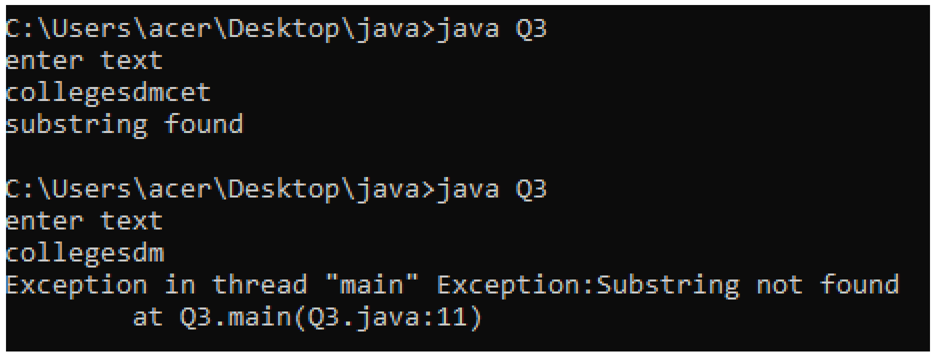
2.Java program:

```
import java.util.Scanner;


 public class Q3{
   public static void main(String[] args)throws SubStringNotFoundException{
            Scanner sc=new  Scanner(System.in);
            System.out.println("enter text");
        String str=sc.next();
            String substr="SDMCET";
            boolean b=str.toLowerCase().contains(substr.toLowerCase());
            if(!b)
            throw new SubStringNotFoundException();
            else
            System.out.println("substring found");
   }
 }

class SubStringNotFoundException extends Exception{
    public String toString(){
            return"Exception:Substring not found";
    }
}
```

3.Screenshots of execution:



```
C:\Users\acer\Desktop\java>java Q3
enter text
collegesdmcet
substring found

C:\Users\acer\Desktop\java>java Q3
enter text
collegesdm
Exception in thread "main" Exception:Substring not found
        at Q3.main(Q3.java:11)
```

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

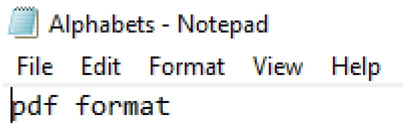
c) If vowel is encountered, throw an exception VowelNotAllowedException and continue until

end of file

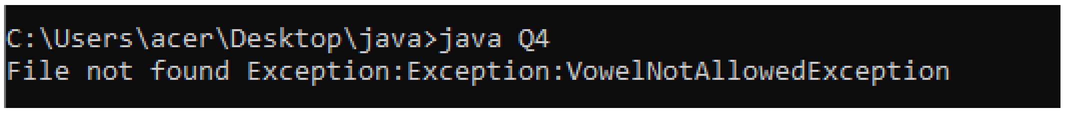
2.Java program:

```
import java.io.FileInputStream;
import java.io.FileOutputStream;
  public class Q4{
   public static void main(String[] args){
            FileInputStream fin=null;
            FileOutputStream fout=null;
            try{
              fin=new FileInputStream("C:/Users/acer/Desktop/java/Alphabets.txt");
              fout=new FileOutputStream("C:/Users/acer/Desktop/java/Consonants.txt");
              int ch;
              while((ch=fin.read())!=-1){
                if(ch=='a'||ch=='e'||ch=='i'||ch=='o'||ch=='u')
                     throw new VowelNotAllowedException();
            else
                     fout.write(ch);
           }
        }catch(Exception e){
                System.out.println("File not found Exception:"+e);
             }
       }
 }
 class VowelNotAllowedException extends Exception{
            public String toString(){
               return "Exception:VowelNotAllowedException";
        }
 }
```

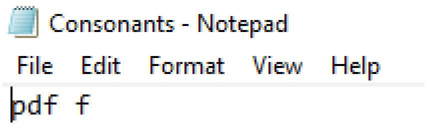
3.Screenshots of execution:



```
pdf format
```



```
C:\Users\acer\Desktop\java>java Q4
File not found Exception:Exception:VowelNotAllowedException
```



```
pdf f
```

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:

```
import java.util.*;

import java.util.Scanner;

import java.io.*;



public class Q5 {


  public static void main(String[] args)throws Exception{


            Scanner sc=new Scanner(System.in);

            int n;

        int i=0;

            System.out.println("enter the number of integers\n");

            n=sc.nextInt();

            int arr[] = new int[n];

            File file = new File("Integer.txt");

            Scanner read = new Scanner(file);

            while(read.hasNext()){

             arr[i++]=Integer.valueOf(read.next());

            }



 Thread t1= new Thread(){

  public void run(){

   synchronized(this){

    try{
     Arrays.sort(arr, 0, (arr.length/3));
      for (int j = 0; j < (arr.length/3); j++) {
        System.out.println(arr[j]);
      }
    }catch(Exception e){
    }
   }
  }
 };

 Thread t2= new Thread(){
  public void run(){
   synchronized(this){
    try{
     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]);
     }
    }catch(Exception e){
     }
  }
 };

  Thread t3= new Thread(){
   public void run(){
     synchronized(this){
      try{
       Arrays.sort(arr,  (2*(arr.length/3)),(n-1));
       for (int j = (2*(arr.length/3)); j < n; j++) {
        System.out.println(arr[j]);
       }
      }catch(Exceprion e){
      }
     }
   }
  };

 Thread t4= new Thread(){
    public void run(){
          synchronized(this){
        try{
         Arrays.sort(arr);
         StringBuilder s = new StringBuilder();
            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();


 }

 }
```