



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:
Niveditha Pise

USN:	2SD20CS064	Name:	NIVEDITHA P
------	------------	-------	-------------

Problem Definition:

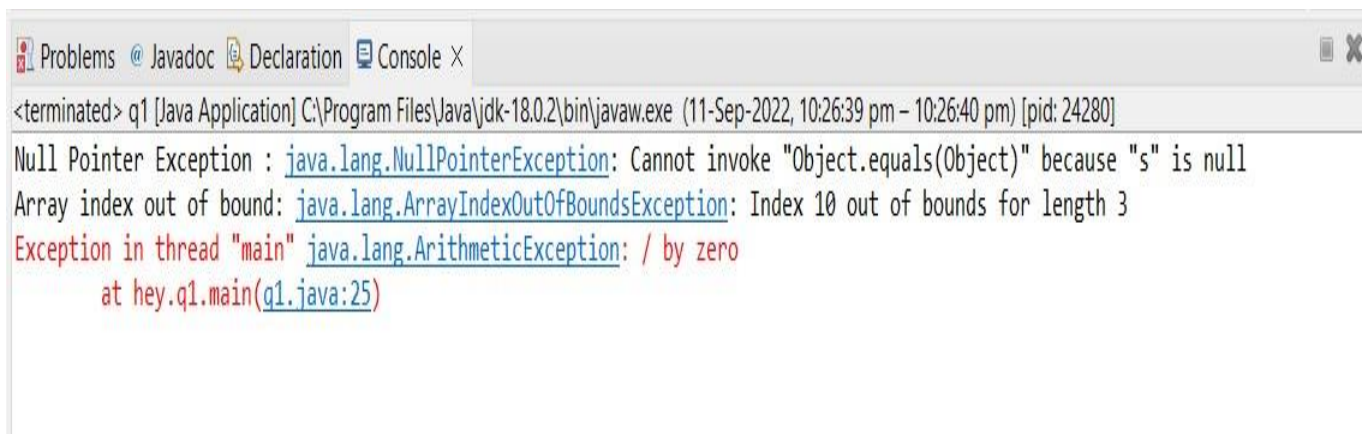
Q1. Write a Java program to generate and handle any three built-in exceptions and display appropriate error messages.

Java Program:

```
1. public class Q1 {
2.     public static void main(String [] args) {
3.         int n=0;
4.         int a[]={2,4,5};
5.         try {
6.             String s[]=null;
7.             if(s.equals("loop")) {
8.                 System.out.println("equal");
9.             }
10.        }catch(NullPointerException ne) {
11.            System.out.println("Null Pointer Exception : "+ne);
12.        }
13.        try {
14.            a[10]=20;
15.        }catch(ArrayIndexOutOfBoundsException ae) {
16.            System.out.println("Array index out of bound: " +ae);
17.        }
18.
19.        try {
```

```
        int z=a[2]/n;  
20        } catch(ExceptionInInitializerError e) {  
21        System.out.println("Divide by zero error: "+e);  
22        }  
23        }
```

Screen Shots of Execution:



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`.

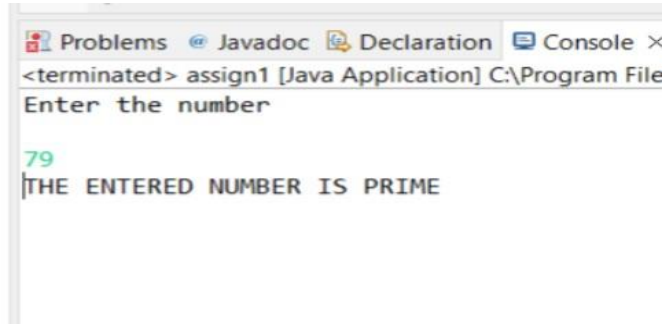
Java Program:

```
1          import java.util.Scanner;
2          import java.util.*;
3          public class Q2{
4              public static void main(String[] args) {
5                  Q2 a=new Q2();
6                  Scanner sc=new Scanner(System.in);
7                  System.out.println("Enter the number\n");
8                  int inum=sc.nextInt();
9                  try {
10                     for(int i=2;i<inum/2;i++) {
11                         if(inum%i==1) {
12                             System.out.println("THE ENTERED NUMBER IS PRIME\n")
13                             break;
14                         } else
15                             throw new NumberNotPrimeException();
16                     }
17                 }
18                 catch(NumberNotPrimeException e) {
19                     e.printStackTrace();
20                 }
```

```
21         }
22
23
24         try {
25             if(inum<0)
26                 throw new NegativeNumberNotAllowedException();
27         } catch(NegativeNumberNotAllowedException e) {
28             e.printStackTrace();
29         }
30     }
31 }
32
33 class NegativeNumberNotAllowedException extends Exception {
34     private String e1;
35     void NegativeNumberNotAllowedException(String e1){
36         this.e1=e1;
37     }
38     public String toString() {
39         return "NEGATIVE NUMBER NOT ALLOWED";
40     }
41 }
42
43 class NumberNotPrimeException extends Exception {
44     private String e2;
45     void NumberNotPrimeException(String e2){
46         this.e2=e2;
47     }
48     public String toString() {
49         return "NOT A PRIME NUMBER";
```

```
48         }  
49     }
```

Screen Shots of Execution:



```
<terminated> assign1 [Java Application] C:\Program Files\Java\jdk-18.0.  
Enter the number  
-6  
NEGATIVE NUMBER NOT ALLOWED  
    at assign1.main(assign1.java:30)
```

Problem Definition:

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

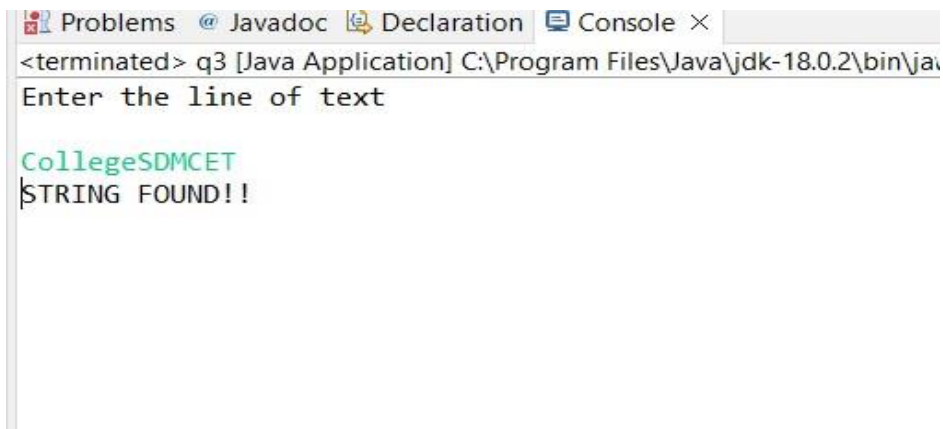
- a) Read a line of text
- b) Search for a sub-string SDMCET (case insensitive search)
- c) If found, then print success message
- d) Otherwise throw an exception SubStringNotFoundException with appropriate message

Java Program:

```
1      import java.util.Scanner;  
2      public class q3 {  
3          public static void main(String args[]) {  
4              Scanner sc=new Scanner(System.in);  
5              System.out.println("Enter the line of text\n");  
6              String s=sc.next();  
7              String sub="SDMCET";  
8              boolean substr=s.toLowerCase().contains(sub.toLowerCase());  
9              try {  
10
```

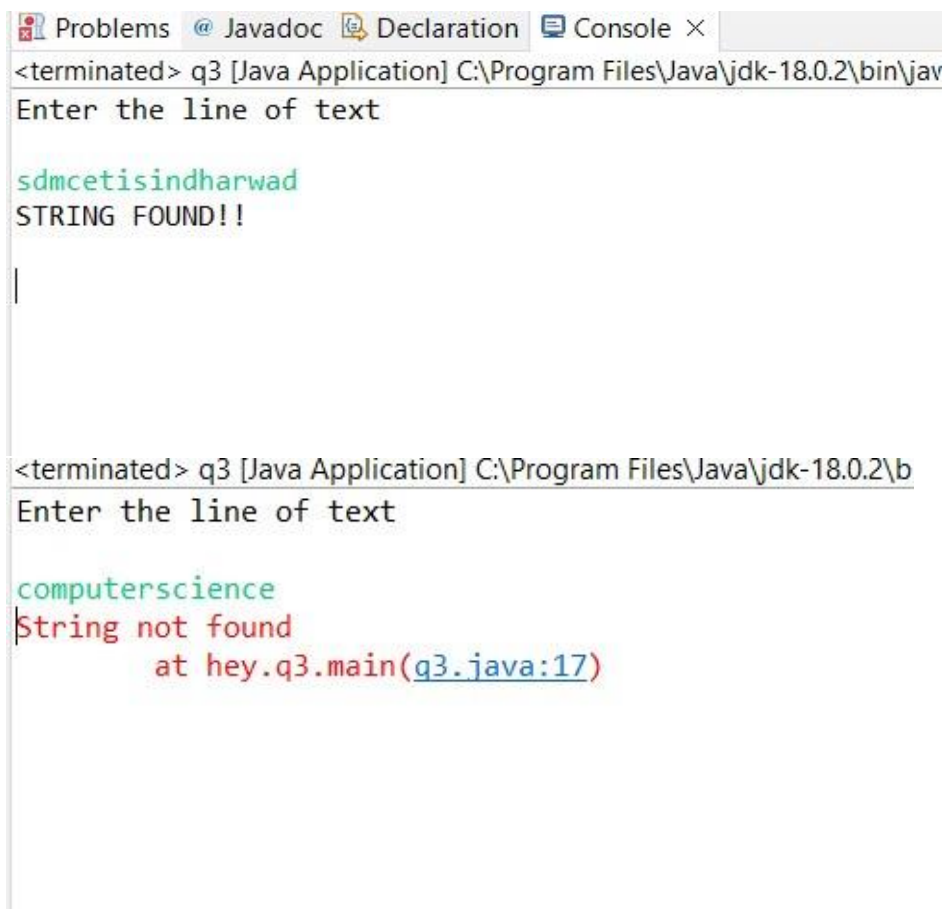
```
11         if(substr) {
12             System.out.println("STRING FOUND!!!");
13         }
14         else
15             throw new SubStringNotFoundException();
16     } catch(SubStringNotFoundException e) {
17         e.printStackTrace();
18     }
19 }
20 }
21 class SubStringNotFoundException extends Exception {
22     private String e1;
23     void SubStringNotFoundException(String e1) {
24         this.e1=e1;
25     }
26     public String toString() {
27         return"String not found";
28     }
29 }
```

Screen Shots of Execution:



The screenshot shows an IDE console window with tabs for Problems, Javadoc, Declaration, and Console. The console output indicates a terminated Java application named 'q3' running from 'C:\Program Files\Java\jdk-18.0.2\bin\java'. It prompts the user to 'Enter the line of text', and the input 'CollegeSDMCET' is shown. The output is 'STRING FOUND!!'.

```
<terminated> q3 [Java Application] C:\Program Files\Java\jdk-18.0.2\bin\java
Enter the line of text
CollegeSDMCET
STRING FOUND!!
```



The first screenshot shows the same IDE console window as above, but with the input 'sdmcetisindharwad'. The output is 'STRING FOUND!!'. The second screenshot shows the input 'computerscience', which results in an error: 'String not found' at 'hey.q3.main(q3.java:17)'.

```
<terminated> q3 [Java Application] C:\Program Files\Java\jdk-18.0.2\bin\java
Enter the line of text
sdmcetisindharwad
STRING FOUND!!

<terminated> q3 [Java Application] C:\Program Files\Java\jdk-18.0.2\bin\java
Enter the line of text
computerscience
String not found
    at hey.q3.main(q3.java:17)
```

Problem Definition:

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

- 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

Java Program:

```
1      import java.io.File;
2      import java.io.FileInputStream;
3      import java.io.FileOutputStream;
4      import java.io.IOException;
5      public class q4 {
6      public static void main(String args[]) throws IOException{
7      FileInputStream fin=null;
8      FileOutputStream fout=null;
9      try {
10     fin=new FileInputStream("C:\\Users\\Dell\\eclipse-
        workspace\\hey\\src\\Alphabet.txt");11
        fout=newFileOutputStream("C:\\Users\\Dell\\eclipse-
        workspace\\hey\\src\\Consonants.txt");
12
        int c;
13
        while((c=fin.read())!=-1) {
14
            if(c=='a' || c=='e' || c=='i' || c=='o' || c=='u' || c=='A' || c=='E' || c=='I' ||
            c=='O' || c=='U')
15
                throw new VowelNotAllowedException();
16
            else
17
                fout.write(c);
18
            }
19
        }catch(VowelNotAllowedException e) {
```

```
20         e.printStackTrace();
21     }
22 }
23 } class VowelNotAllowedException extends Exception{
24     private String e1;
25     void VowelNotAllowedException(String e1) {
26         this.e1=e1;
27     }
28     public String toString() {
29         return"Vowel not Allowed";
30     }
31 }
```

Screen Shots of Execution:

