



# **AOOP Assignment Submission Report**

[Submitted as part of CTA Assignment No-1]

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

Submitted by:

USN:	2SD20CS044	Name:	JASHANDEEP SINGH
------	------------	-------	------------------

## 1. Problem Definition:

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

## Java Program:

```
import java.util.Scanner;

//class creation

class animal{

    String type; //instance variable

    String food;

    void ChangeFood(String food) { //method declaration

        this.food=food;

    }

}

public class TestDemo {

    public static void main(String[] args){

        Scanner sc = new Scanner(System.in);

        //try catch block

        try {

            System.out.println("Enter value of a and b:");

            int a = sc.nextInt();

            int b = sc.nextInt();

            int divi = a/b;

            int[] intarray = new int[4];

            intarray [0]=1;

            intarray [1]=2;

            intarray [2]=3;

            intarray [3]=4;

            intarray [4]=5;
```

---

```
    animal an = new animal();  
    an.setfood("meat");
```

```
    } catch(ArithmeticException ae) { //ArithmeticException handling  
        System.out.println("Error 1:"+ae);  
    }
```

```
    catch(ArrayIndexOutOfBoundsException ao) { //ArrayIndexOutOfBoundsException Exception handling  
        System.out.println("Error 2:"+ao);  
    }
```

```
    catch(NoSuchMethodException np) { //NoSuchMethodException handling  
        System.out.println("Error 3:"+np);  
    }
```

```
    }
```

```
}
```

---

## Screen Shots of Execution:

The screenshot shows the Eclipse IDE with a Java project named 'AOOP Assignment1'. The main class is 'TestDemo.java'. The code defines a 'TestDemo' class with a 'main' method. The 'main' method uses a 'Scanner' to read input, but in this case, it's not shown. It then performs a division 'divi = a/b;' where 'a' is 9 and 'b' is 0. This results in an 'ArithmeticException: / by zero'. The console output shows the error message: 'Error 1: java.lang.ArithmeticException: / by zero'. The Outline view on the right shows the class structure: 'animal' (with 'type' and 'food' attributes and a 'ChangeFood' method) and 'TestDemo' (with a 'main' method).

```

10 }
11 }
12 public class TestDemo{
13
14     public static void main(String[] args){
15         Scanner sc = new Scanner(System.in);
16         //try catch block
17         try {
18             int a = 9;
19             int b = 0;
20             int divi = a/b;
21
22             /*int[] intarray = new int[4];
23             intarray [0]=1;
24             intarray [1]=2;
25             intarray [2]=3;
26             intarray [3]=4;
27             intarray [4]=5;
28             */
29
30             //animal an = new animal();
31             //an.setfood("meat");
32
33     }
34 }

```

Declaration Console ×  
 <terminated> TestDemo [Java Application] C:\Users\LeNovo\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86\_64.17.0.4.v20220903-1038\jre\bin\javaw.exe (12-Sep-2022, 5:13:06 pm - 5:13:06 pm) [pid: 14536]  
 Error 1: java.lang.ArithmeticException: / by zero

The screenshot shows the Eclipse IDE with the same Java project. The code in 'TestDemo.java' is updated to include exception handling. It defines an 'int[] intarray' with 4 elements. The 'main' method sets the first three elements to 1, 2, and 3, and the fourth to 5. It then attempts to access 'intarray[4]', which is out of bounds, resulting in an 'ArrayIndexOutOfBoundsException: Index 4 out of bounds for length 4'. The console output shows the error message: 'Error 2: java.lang.ArrayIndexOutOfBoundsException: Index 4 out of bounds for length 4'. The Outline view on the right shows the class structure: 'animal' (with 'type' and 'food' attributes and a 'ChangeFood' method) and 'TestDemo' (with a 'main' method).

```

22     int[] intarray = new int[4];
23     intarray [0]=1;
24     intarray [1]=2;
25     intarray [2]=3;
26     intarray [3]=4;
27     intarray [4]=5;
28
29
30     //animal an = new animal();
31     //an.setfood("meat");
32
33     }
34 }
35
36 //catch (ArithmeticException ae) { //ArithmeticException handling
37 //System.out.println("Error 1:"+ae);
38 //}
39
40 catch (ArrayIndexOutOfBoundsException ao) { //ArrayIndexOutOfBoundsException handling
41     System.out.println("Error 2:"+ao);
42 }
43
44 //catch (NoSuchMethodException np) { //NoSuchMethodException handling
45 //System.out.println("Error 3:"+np);
46 //}
47
48 }
49 }

```

Declaration Console ×  
 <terminated> TestDemo [Java Application] C:\Users\LeNovo\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86\_64.17.0.4.v20220903-1038\jre\bin\javaw.exe (12-Sep-2022, 5:17:02 pm - 5:17:03 pm) [pid: 11900]  
 Error 2: java.lang.ArrayIndexOutOfBoundsException: Index 4 out of bounds for length 4

---

## 2.Problem Definition:

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:

```
import java.util.*;

class NegativeNumberNotAllowedException extends Exception{
    public String toString() {
        return "NegativeNumberNotAllowedException[Negative Number]";
    }
}

class NumberNotPrimeException extends Exception{
    public String toString() {
        return "NumberNotPrimeException[Number not prime]";
    }
}

public class Main {

    public static void main(String[] args) throws Exception{
        int num,i;
        boolean temp=false;

        Scanner sc=new Scanner(System.in);

        //input an integer number
        System.out.print("Enter any integer number: ");
        num= sc.nextInt();

        //check prime
        for(i=2; i<=(num/2); i++)
        {
            if(num%i==0)
            {
                temp=true;
                break;
            }
        }
        if(num<0) {
            throw new NegativeNumberNotAllowedException();
        }
    }
}
```

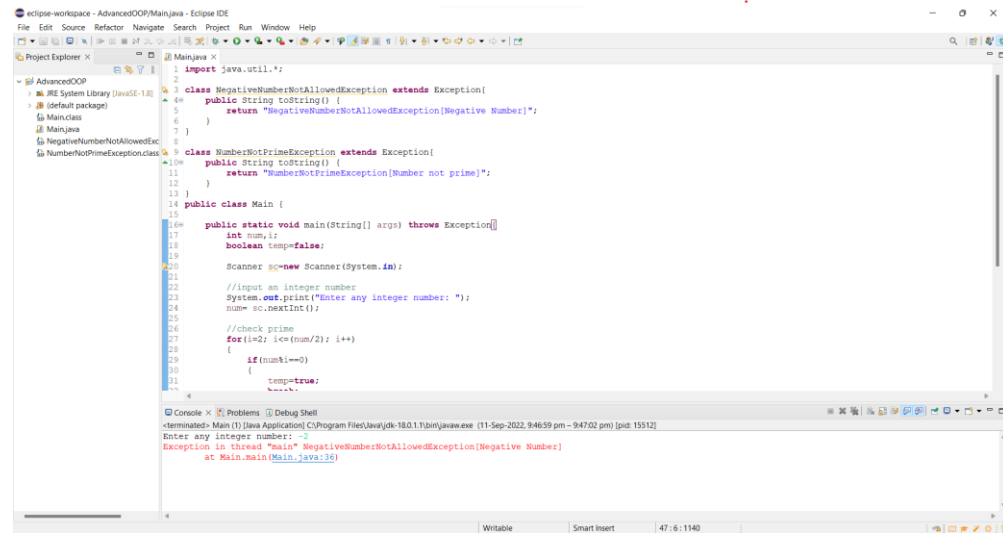
---

```
        else if(temp==true) {  
            throw new NumberNotPrimeException();  
        }  
  
        else {  
            System.out.println(num + " is a prime number.");  
        }  
  
    }  
  
} //End of Main
```

---



# Screen Shots of Execution:



This screenshot shows the Eclipse IDE with a Java project named 'AdvancedOOP'. The 'Main.java' file is open, displaying the following code:

```
import java.util.*;

class NegativeNumberNotAllowedException extends Exception {
    public String toString() {
        return "NegativeNumberNotAllowedException[Negative Number]";
    }
}

class NumberNotPrimeException extends Exception {
    public String toString() {
        return "NumberNotPrimeException[Number not prime]";
    }
}

public class Main {
    public static void main(String[] args) throws Exception {
        int num, i;
        boolean temp=false;

        Scanner sc=new Scanner(System.in);

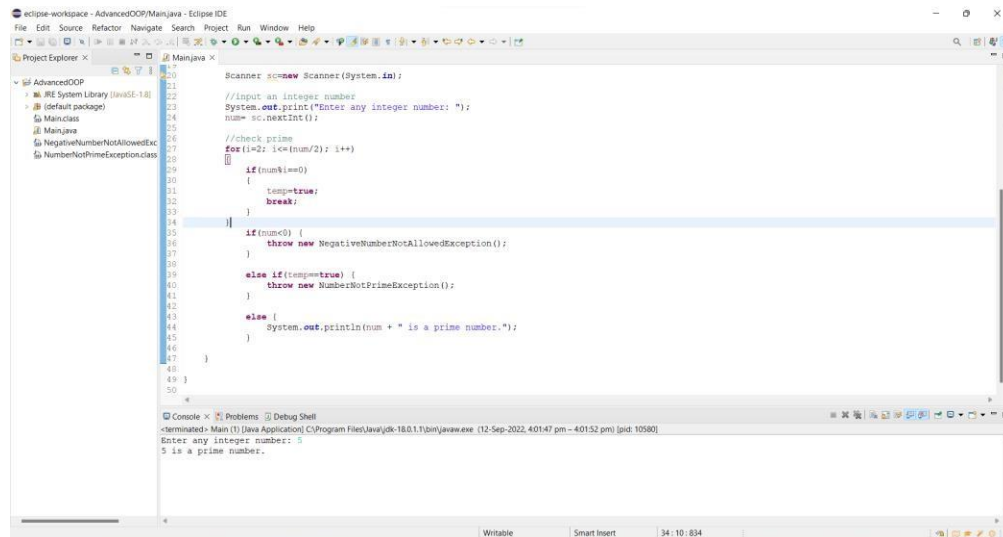
        //input an integer number
        System.out.print("Enter any integer number: ");
        num= sc.nextInt();

        //check prime
        for(i=2; i<=(num/2); i++)
        {
            if(num%i==0)
            {
                temp=true;
                break;
            }
        }

        if(num<0) {
            throw new NegativeNumberNotAllowedException();
        }
        else if(temp==true) {
            throw new NumberNotPrimeException();
        }
        else {
            System.out.println(num + " is a prime number.");
        }
    }
}
```

The console window at the bottom shows the following output:

```
<terminated> Main (1) [Java Application] C:\Program Files\Java\jdk-18.0.1\bin\java.exe (11-Sep-2022, 9:46:59 pm - 94702 pm) [pid: 15512]
Enter any integer number: -5
Exception in thread "main" NegativeNumberNotAllowedException[Negative Number]
at Main.main(Main.java:36)
```



This screenshot shows the Eclipse IDE with the 'Main.java' file open, displaying the following code:

```
Scanner sc=new Scanner(System.in);

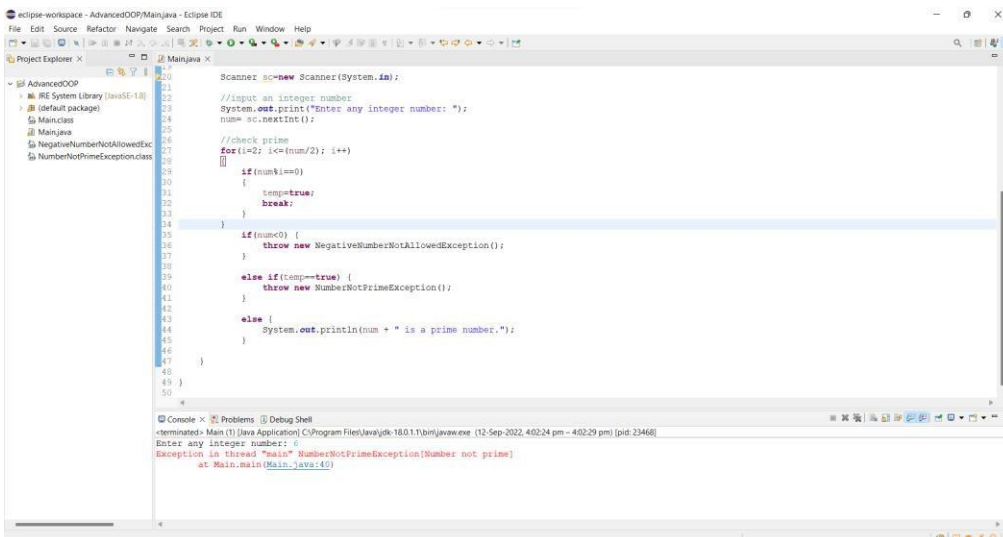
//input an integer number
System.out.print("Enter any integer number: ");
num= sc.nextInt();

//check prime
for(i=2; i<=(num/2); i++)
{
    if(num%i==0)
    {
        temp=true;
        break;
    }
}

if(num<0) {
    throw new NegativeNumberNotAllowedException();
}
else if(temp==true) {
    throw new NumberNotPrimeException();
}
else {
    System.out.println(num + " is a prime number.");
}
}
```

The console window at the bottom shows the following output:

```
<terminated> Main (1) [Java Application] C:\Program Files\Java\jdk-18.0.1\bin\java.exe (12-Sep-2022, 4:01:47 pm - 40152 pm) [pid: 10580]
Enter any integer number: 5
5 is a prime number.
```



This screenshot shows the Eclipse IDE with the 'Main.java' file open, displaying the following code:

```
Scanner sc=new Scanner(System.in);

//input an integer number
System.out.print("Enter any integer number: ");
num= sc.nextInt();

//check prime
for(i=2; i<=(num/2); i++)
{
    if(num%i==0)
    {
        temp=true;
        break;
    }
}

if(num<0) {
    throw new NegativeNumberNotAllowedException();
}
else if(temp==true) {
    throw new NumberNotPrimeException();
}
else {
    System.out.println(num + " is a prime number.");
}
}
```

The console window at the bottom shows the following output:

```
<terminated> Main (1) [Java Application] C:\Program Files\Java\jdk-18.0.1\bin\java.exe (12-Sep-2022, 4:02:24 pm - 40229 pm) [pid: 23468]
Enter any integer number: 6
Exception in thread "main" NumberNotPrimeException[Number not prime]
at Main.main(Main.java:40)
```

### **3.Problem Definition:**

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:

```
import java.util.Scanner;

class Main {
    public static void main(String[] args) throws Exception{
        Scanner sc=new Scanner (System.in);
        System.out.println("Enter the line of text");
        String txt=sc.next();
        // create a string
        //String txt = "This is Programiz";
        String str1 = "SDMCET";
        //String str2 = "Programming";

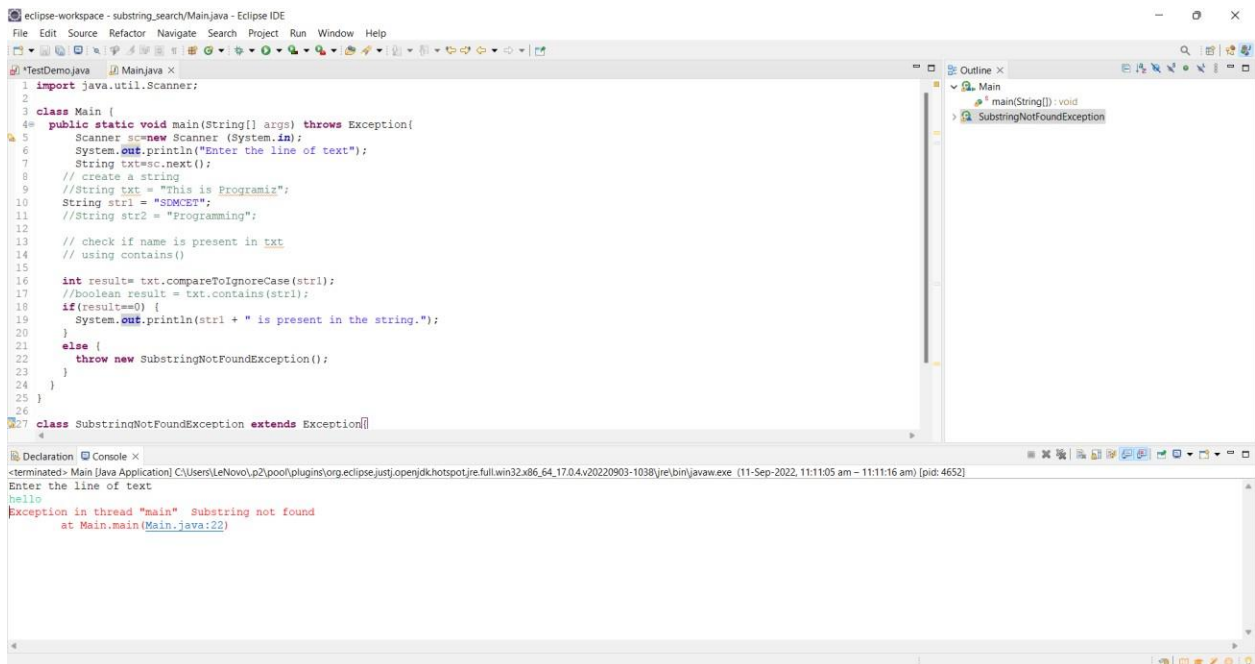
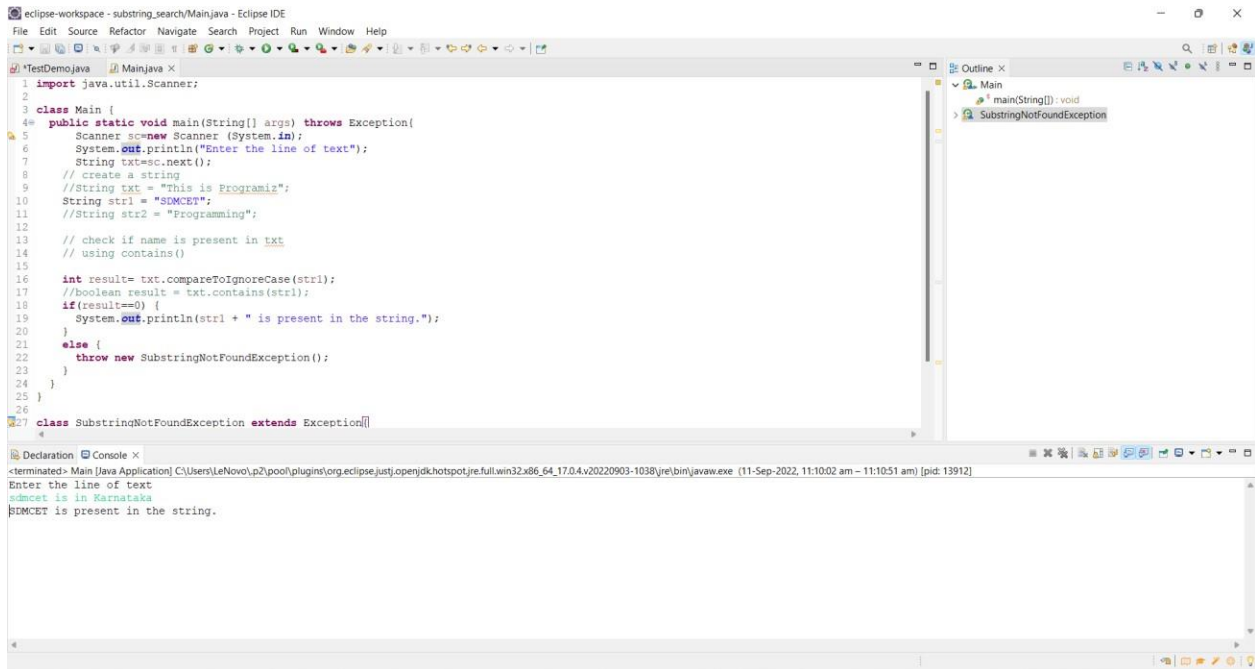
        // check if name is present in txt
        // using contains()

        int result= txt.compareToIgnoreCase(str1);
        //boolean result = txt.contains(str1);
        if(result==0) {
            System.out.println(str1 + " is present in the string.");
        }
        else {
            throw new SubstringNotFoundException();
        }
    }
}

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

---

# Screen Shots of Execution:



## 4.Problem Definition:

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
-

## Java Program:

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

class alq4 {

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

        FileWriter w = new FileWriter("Alphabets.txt");
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the data to write in the file :");
        String str = sc.nextLine();
        w.write(str);
        w.close();
        File file = new File("Alphabets.txt");

        Scanner reader = new Scanner(file);
        StringBuilder s = new StringBuilder();
        FileWriter write = new FileWriter("Consonant.txt");
        while(reader.hasNext()){
            String data = reader.next();
            for (int i = 0; i < data.length(); i++) {
                if(isVowel(data.charAt(i))){
                    throw new VowelNotAllowedException();

                }else{

                    s.append(data.charAt(i));
                }
            }
            write.write(s.toString());
        }

        write.close();

    }
}
```

---

```
static boolean isVowel(char c) {  
    if(c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u' || c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U'){  
  
        return true;  
    }else{  
        return false;  
    }  
}  
  
}
```

---

# Screen Shots of Execution:

This screenshot shows the Eclipse IDE with a Java file named `atq4.java`. The code defines a `isVowel` method and a `VowelNotAllowedException` class. The `main` method reads input from the user and checks if it's a vowel. An exception is thrown when the input is 'a'. The console output shows the error message: `Exception in thread "main" Vowel Not allowed at atq4.main(atq4.java:26)`.

```
28         }else{
29             s.append(data.charAt(i));
30         }
31     }
32     write.write(s.toString());
33 }
34
35 write.close();
36
37 }
38
39 }
40
41 static boolean isVowel(char c) {
42     if(c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u' || c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U'){
43
44     }
45     return true;
46 }else{
47     return false;
48 }
49 }
50
51 class VowelNotAllowedException extends Exception{
52
53     public String toString(){
54         return "Vowel Not allowed";
55     }
56 }
57
58 }
```

Console Output:

```
<terminated> atq4 [Java Application] C:\Program Files\Java\jdk-18.0.1\bin\java.exe (12-Sep-2022, 6:10:09 pm) [pid: 13140]
Enter the data to write in the file :aahff
Exception in thread "main" Vowel Not allowed
at atq4.main(atq4.java:26)
```

This screenshot shows the Eclipse IDE with the same `atq4.java` file. The `main` method now reads input from the user and checks if it's a vowel. The console output shows the successful execution: `Enter the data to write in the file :o;g`.

```
23 String data = reader.next();
24 for (int i = 0; i < data.length(); i++) {
25     if (isVowel(data.charAt(i))) {
26         throw new VowelNotAllowedException();
27     }
28     }else{
29         s.append(data.charAt(i));
30     }
31 }
32 write.write(s.toString());
33
34 write.close();
35
36 }
37
38 }
39
40
41 static boolean isVowel(char c) {
42     if(c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u' || c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U'){
43
44     }
45     return true;
46 }else{
47     return false;
48 }
49 }
50
51 class VowelNotAllowedException extends Exception{
52
53     public String toString(){
54         return "Vowel Not allowed";
55     }
56 }
57
58 }
```

Console Output:

```
<terminated> atq4 [Java Application] C:\Program Files\Java\jdk-18.0.1\bin\java.exe (12-Sep-2022, 6:07:52 pm - 6:07:59 pm) [pid: 2336]
Enter the data to write in the file :o;g
```

This screenshot shows a Notepad window titled `Alphabets.txt - Notepad`. The text content is `ٲٲٲ`.

Ln 1, Col 1 | 100% | Windows (CRLF) | UTF-8



## 5.Problem Definition:

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
-

# Java Program:

```
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Arrays;
import java.util.Scanner;

class a1q5 {
    private static int arr[];

    public static void main(String[] args) throws FileNotFoundException, InterruptedException,
    IOException {
        File inputFile = new File("Integers.txt");

        File outputFile = new File("SortedIntegers.txt");

        FileWriter opWriter = new FileWriter(outputFile);

        Scanner sc = new Scanner(inputFile);

        int size = sc.nextInt();

        arr = new int[size];
        int i = 0;
        while (sc.hasNext()) {

            arr[i++] = sc.nextInt();
        }
        sc.close();

        Thread T1 = new Thread() {
            public void run() {
                ThreadSorting(arr, 0, (size / 3) - 1);
            }
        };

        Thread T2 = new Thread() {
            public void run() {
                ThreadSorting(arr, (size / 3), ((size / 3) * 2) - 1);
            }
        };
    }
}
```

---

```
Thread T3 = new Thread() {
    public void run() {
        ThreadSorting(arr, ((size / 3) * 2), (size - 1));
    }
};

Thread T4 = new Thread() {
    public void run() {
        ThreadSorting(arr, 0, size - 1);
    }
};

T1.start();
T1.join();
T2.start();
T2.join();
T3.start();
T3.join();
T4.start();
T4.join();

for (int num : arr) {

    opWriter.append(String.valueOf(num) + " ");
}
opWriter.close();

}

public static void ThreadSorting(int arr[], int start, int end) {
    int tempArr[] = new int[end - start + 1];
    int tempIndex = 0;
    for (int i = start; i <= end; i++) {
        tempArr[tempIndex++] = arr[i];
    }
    Arrays.sort(tempArr);
    int index = start;

    for (int n : tempArr) {
        arr[index++] = n;
    }
}
}
```

---

## Screen Shots of Execution:

