United International University

QUEST FOR EXCELLENCE

Assignment :

Course Name: Object Oriented Programming

Course Code: CSE 1115

Section: “F”

Submitted by:

Name: Yeasir Arafat

ID: 011 201 035

Submitted to:

Subangkar Karmaker Shanto

United International University

6. Write only the numeric values of a file to corresponding line of another file for each line in the input

Sample Input file	Output file
There are 12 pillars in the building. Each pillar is 5.6ft in width and 1ft in height No pillar in the roof. 4 flats on each floor. 2 windows kept in every room.	12 5.6 1 4 2

CODE :

```
package exam.oop212.assignment;

import java.io.*;


public class FIleException6 {
  public static void main(String[] args) {
    try {
      FileReader file1 = new FileReader("src/exam/oop212/assignment/file1.txt");
      BufferedReader br = new BufferedReader(file1);
      FileWriter file2 = new FileWriter("src/exam/oop212/assignment/file2.txt");
      BufferedWriter bw = new BufferedWriter(file2);
      String str;
      while ((str=br.readLine()) != null) {    //read a line from the file in each iteration
        String[] temp = str.split(" "); //splitting the line based on space
        for (String s : temp) {            // iterate the splitted string
          // replace all the character with empty string except 0 to 9 and dot(.)
          s = s.replaceAll("[^0-9.]","");
          // now the string can be three thing, Number/Empty String/Single dot (full stop of sentence)
          // so when the string contains digits/numbers, then the condition will be true
          if(!s.equals("") && !s.equals(".")) {
            bw.write(s+" ");    // writing to file
          }
        }
        bw.write("\n");  // writing to file
      }
      bw.close();  // close file / flash the content to file
      br.close();  // close reader
```

```
    } catch (FileNotFoundException e) {  // Catch exceptions
      System.out.println(e.getMessage());
      e.printStackTrace();
    }
    catch (IOException e) {
      e.printStackTrace();
    }
  }
}
```

8. Find the word given as input from console in a file and report the word indices for each line if it's present as following:

Sample Input file	For the word "first" (case insensitive matching)
Here goes the first line. First line was of 5 words and the first word of the first line was here. Second line was in past tense unlike the first line. This is the first document in the directory	Found [3] Found [0, 8, 12] Not Found Found [2, 7] Not Found

CODE:

```
package exam.oop212.assignment;
import java.io.*;
import java.util.ArrayList;
import java.util.Scanner;

public class FIleException8 {
  public static void main(String[] args) {
    String matchWith;
    String str;
```

```
        try {
            FileReader file3 = new FileReader("src/exam/oop212/assignment/file3.txt");
            FileWriter file4 = new FileWriter("src/exam/oop212/assignment/file4.txt");
            BufferedReader br = new BufferedReader(file3);          // using BufferedReader to read file
            BufferedWriter bw = new BufferedWriter(file4);          // using BufferedWriter to write file
            Scanner sc = new Scanner(System.in);
            System.out.println("Enter a word");
            matchWith = sc.next().toLowerCase();                // Convert the input word to lowercase
            while((str = br.readLine()) != null){               // reading a line from the file in each iteration
                boolean found = false;                          // for determining if the word is found or not
                str = str.toLowerCase();                        // Converting the whole line to lowercase
                String[] str_split = str.split(" ");            //splitting the line by space and adding to string array
                ArrayList<Integer> list = new ArrayList<>();   //using arraylist to store the indexes of the word
                for(int i=0; i< str_split.length; i++){
                    if(str_split[i].equals(matchWith)){     // checking if the splitted word is matched with the file
                        list.add(i);                        // adding the index to arraylist
                        found = true;                       // set found to true so that I can write the correct output later
                    }
                }
                if(found) {                             // Checking if the word is found in splitted strings
                    bw.write("Found " + list + "\n");       // writing to file
                }
                else{                           //If the line doesn't contain the user input word, then write Not Found
                    bw.write("Not Found\n");            // writing to file
                }
            }
            bw.close();                         // closing file // flush the written content to file
            br.close();                         // closing reader
        } catch (FileNotFoundException e) {   // Catch exceptions
            System.out.println(e.getMessage());
            e.printStackTrace();
        } catch (IOException e){
            e.printStackTrace();
        }
    }
}
```

3. Create a class named **WifiLANNode** that contains an instance variable named double **usedMB**, a method named **downloadFile** & a **constructor** which takes the initial value of usedMB. Method downloadFile takes a single double argument: **amount** and increases the usedMB by that amount. However, the class maintains a global **upper-limit for all of its objects**. The global upper-limit for all of its objects **can be increased globally for all the objects** of the class by calling a method named **increaseGlobalLimit** of the class. Whenever the downloadFile method or the constructor tries to set usedMB beyond that limit a user-defined exception named **BandwidthLimitExceeded** is **thrown** from the method downloadFile with the message:
You requested to use 1510MB which crosses the current maximum limit of 1500MB.
Here, 1510 would have been the value of the usedMB if the amount could be added to the current usedMB & 1500 is the currently set global upper-limit for all of its objects. For both of these methods or the constructor if the argument amount passed is negative, they will **throw** another user defined exception called **InvalidAmountArgument.**

CODE :

```
package exam.oop212.assignment;

class WifiLANNode {
   private static double upper_limit = 1500; // Declaring a static variable for global upper limit.
   double usedMB;
   public WifiLANNode(double usedMB) throws BandwidthLimitExceeded, InvalidAmountArgument {
      if(usedMB<0){
         throw new InvalidAmountArgument(); // Throwing exception when the value is negative.
      }
      if(usedMB>upper_limit){
         throw new BandwidthLimitExceeded(usedMB,upper_limit);   // Throwing exception when the value is
//greater than upper limit
      }
      this.usedMB = usedMB;  // If none of the exception is generated, then set the value of usedMb
   }
   public void downloadFile(double amount) throws BandwidthLimitExceeded, InvalidAmountArgument {
      if(amount<0){
         throw new InvalidAmountArgument(); // Throwing exception when the value of the amount is negative.
      }
      if((usedMB+amount)>upper_limit){
         throw new BandwidthLimitExceeded(usedMB+amount,upper_limit);  // Throwing exception when the
//value of usedMB is tried to set beyond upper limit.
      }
```

```
      usedMB += amount;  // If none of the exception is generated, then increase the value of usedMb
   }
   public static void increaseGlobalLimit(double increase_upper_limit){
      upper_limit += increase_upper_limit;   // increase the value of static variable which is used as upper limit
//for all the object
   }
}

class BandwidthLimitExceeded extends Exception{  // creating custom exception for BandwidthLimitExceeded
   public BandwidthLimitExceeded(double requested_size,double upper_limit){
     super("You requested to use "+requested_size+"MB"+" which crosses the current maximum limit of "+upper_limit+"MB.");
   }
}
class InvalidAmountArgument extends Exception{ // creating custom exception for InvalidAmountArgument
   InvalidAmountArgument(){
      super("Please input positive value.");

   }
}

public class Exception3{
   public static void main(String[] args) {
      try {
         WifiLANNode obj1 = new WifiLANNode(1510);  // BandwidthLimitExceeded exception will be thrown
//as the value passed is greater than the upper limit.
      }
      catch (BandwidthLimitExceeded | InvalidAmountArgument e) {  // Catch the exception that was thrown
         System.out.println(e.getMessage());  // Printing the message that was passed in Exception using super
//keyword
      }

      try {
         WifiLANNode.increaseGlobalLimit(1000);   // increase upper limit for all the objects | Now upper limit
//for all the objects are 1500+1000 = 2500
         WifiLANNode obj2 = new WifiLANNode(1800);
         obj2.downloadFile(1200);  // BandwidthLimitExceeded exception will be thrown as the value usedMB
//will be tried to set beyond upper limit.
      }
      catch (BandwidthLimitExceeded | InvalidAmountArgument e) {   // Catch the exception that was thrown
```

```
        System.out.println(e.getMessage());  // Printing the message that was passed in Exception using super
//keyword
    }

    try {
        WifiLANNode obj3 = new WifiLANNode(-1000);  // InvalidAmountArgument exception will be thrown
//as the value of the parameter is negative.
    }
    catch (BandwidthLimitExceeded | InvalidAmountArgument e) {  // Catch the exception that was thrown
        System.out.println(e.getMessage()); //Printing the message that was passed in Exception using super keyword
    }
  }
}
```

Java Files: [GitHub]