1. **Reading and Writing to a file**

**package** file;

**import** java.io.BufferedReader;

**import** java.io.BufferedWriter;

**import** java.io.File;

**import** java.io.FileReader;

**import** java.io.FileWriter;

**import** java.io.IOException;

**public** **class** FileRW {

**private** **static** **final** String ***inputFileName*** = "C://Users/ragini.kulkarni/Desktop/InputFilePractice2.txt";

**private** **static** **final** String ***outputFileName*** = "C://Users/ragini.kulkarni/Desktop/outfile.txt";

**private** **static** **final** String ***eeroredFile*** = "C://Users/ragini.kulkarni/Desktop/errorfile.txt";

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

File inputFile = **new** File(***inputFileName***);

FileReader fileReader = **new** FileReader(inputFile);

BufferedReader br = **new** BufferedReader(fileReader);

File outputFile = **new** File(***outputFileName***);

FileWriter fileWriter = **new** FileWriter (outputFile);

BufferedWriter buffWriter = **new** BufferedWriter(fileWriter);

File erroredFile = **new** File(***eeroredFile***);

FileWriter fileErrWriter = **new** FileWriter (erroredFile);

BufferedWriter buffErrWriter = **new** BufferedWriter(fileErrWriter);

String record = **null**;

**while** ((record = br.readLine()) != **null**) {

String[] field = record.split(",");

**if**(field[3].length()==10)

{

buffWriter.write(field[3]);

}

**else**{

buffErrWriter.write(field[3]);

}

}

br.close();

buffWriter.close();

buffErrWriter.close();

}

}

1. **Reading and Writing to a file Using Map**

**package** file;

**import** java.io.BufferedReader;

**import** java.io.BufferedWriter;

**import** java.io.\*;

**import** java.io.FileReader;

**import** java.io.FileWriter;

**import** java.io.IOException;

**import** java.util.Map.Entry;

**import** java.util.\*;

**public** **class** FileUsgMap {

**private** **static** **final** String ***inputFileName*** = "C://Users/ragini.kulkarni/Desktop/NewFile.txt";

**private** **static** **final** String ***outputFileName*** = "C://Users/ragini.kulkarni/Desktop/MapOut.txt";

**private** **static** **final** String ***eroredFileName*** = "C://Users/ragini.kulkarni/Desktop/errorCollectfile.txt";

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

File inputFile = **new** File(***inputFileName***);

FileReader fileReader = **new** FileReader(inputFile);

BufferedReader br = **new** BufferedReader(fileReader);

File erroredFile = **new** File(***eroredFileName***);

FileWriter fileErrWriter = **new** FileWriter(erroredFile);

BufferedWriter buffErrWriter = **new** BufferedWriter(fileErrWriter);

String record = br.readLine();

Map<String, String> listOfNames = **new** HashMap<String, String>();

// TreeMap<String,String> listOfNames = new TreeMap<String,String>();

**while** ((record = br.readLine()) != **null**) {

String[] field = record.split(",");

// if (field[2].length() == 10)

listOfNames.put("Name:" + field[0] + " " + "Email:" + field[1] + " " + "phonenum:" + field[2], **null**);

// for (Map.Entry names : listOfNames.entrySet())

// System.out.println(names);

// for (String names : listOfNames) //{

// System.out.println(names);

// Collections.sort(listOfNames);

// buffWriter.write(field[0]+" "+field[1]+" "+field[2]);

// buffWriter.newLine();

// }

/\*

\* //} else { listOfNames.add("phonenum:" + field[2]);

\* Collections.sort(listOfNames); for (String names : listOfNames) {

\* System.out.println(names); buffErrWriter.write(names);

\* buffErrWriter.newLine(); }

\*/

}

/\* Unsorted List \*/

/\*

\* System.out.println("Before Sorting:"); for (String names :

\* listOfNames) { System.out.println(names); }

\*/

/\* Sort statement \*/

// Collections.sort(listOfNames);

/\* Sorted List \*/

System.***out***.println("\nSorted Map......");

Map<String, String> treeMap = **new** TreeMap<String, String>(listOfNames);

String names = *printMap*(treeMap);

// System.out.println(names);

// buffWriter.write(names);

// buffWriter.newLine();

/\*

\* System.out.println("After Sorting:"); for (Map.Entry names :

\* listOfNames.entrySet()) { System.out.println(names);

\* buffWriter.write(names); //buffWriter.newLine(); }

\*/

br.close();

// buffWriter.close();

// buffErrWriter.close();

// buffWriter.close();

}

**public** **static** <K, V> String printMap(Map<K, V> map) **throws** IOException {

File outputFile = **new** File(***outputFileName***);

FileWriter fileWriter = **new** FileWriter(outputFile);

BufferedWriter buffWriter = **new** BufferedWriter(fileWriter);

// **TODO** Auto-generated method stub

String result = " ";

**for** (Map.Entry<K, V> entry : map.entrySet()) {

System.***out***.println(entry.getKey());

result = (String) entry.getKey();

buffWriter.write(result);

buffWriter.newLine();

}

buffWriter.close();

**return** result;

}

}

1. **Will not print At least two largest element**

**package** sorting;

**import** java.util.Arrays;

**public** **class** AtleastTwogreatestElemnt {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

**int** a[] = { 3, 11, 8, 7, 9 };

**int** n = a.length;

**for** (**int** i = n - 1; i > 0; i--) {

**for** (**int** j = 0; j < i; j++) {

**if** (a[j] > a[j + 1]) {

**int** temp = a[j];

a[j] = a[j + 1];

a[j + 1] = temp;

}

}

}

**for**(**int** i=0;i<n-2;i++)

{

System.***out***.println(a[i]);

}

}

O/P : 3,7,8

1. **Will print true if the elements are sorted else false**

**package** sorting;

**import** java.util.Arrays;

**public** **class** CheckforSorting {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

**boolean** status = **true**;

**int** a[] = { 2, 5, 1, 7, 4, 3 };

**int** old[]=a.clone();

**if** (a.length != old.length) {

status = **false**;

} **else** {

Arrays.*sort*(a);

status = Arrays.*equals*(a, old);

}

**if** (status) {

System.***out***.println("1");

} **else** {

System.***out***.println("0");

}

}

}

O/P : 0

1. **Test Anagrams**

**package** string;

**import** java.util.Arrays;

**public** **class** TestAnagrams {

**static** **void** isAnagram(String str1, String str2) {

String s1 = str1.replaceAll("\\s", "");

String s2 = str2.replaceAll("\\s", "");

**boolean** status = **true**;

**if** (s1.length() != s2.length()) {

status = **false**;

} **else** {

**char**[] ArrayS1 = s1.toLowerCase().toCharArray();

**char**[] ArrayS2 = s2.toLowerCase().toCharArray();

Arrays.*sort*(ArrayS1);

Arrays.*sort*(ArrayS2);

status = Arrays.*equals*(ArrayS1, ArrayS2);

}

**if** (status) {

System.***out***.println(s1 + " and " + s2 + " are anagrams");

} **else** {

System.***out***.println(s1 + " and " + s2 + " are not anagrams");

}

}

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

*isAnagram*("Keep", "Peek");

*isAnagram*("Mother In Law", "Hitler Woman");

}

}

O/P : Keep and Peek are anagrams

MotherInLaw and HitlerWoman are anagrams

1. **Delete Alternate letters**

**package** string;

**public** **class** TestDeleteAlternate {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

String s1="ragini";

**int** n=s1.length();

**for**(**int** i=0;i<n;i++)

{ **if**( i % 2 == 0)

{

**char** sn= s1.charAt(i);

System.***out***.print(sn);

}

}

}

}

O/P : rgn

1. **Reverse And Appending**

**package** string;

**import** java.util.Scanner;

**public** **class** TestReverseAndAppending {

**public** **static** **void** main(String[] args) {

//String s1 = "ragini"; //String s2 = "kulkarni";

String reverse = "";

Scanner s=**new** Scanner(System.***in***);

**int** tc=s.nextInt();

**while**(tc > 0){

String s1=s.next();

String s2=s.next();

String s3 = (s1 + s2);

**int** n = s3.length();

**for** (**int** i = n - 1; i >= 0; i--) {

reverse = reverse + s3.charAt(i);

}

System.***out***.println(reverse);

}

}

}

O/P : 2

ragini

kulkarni

inraklukinigar

1. **ToLowerCase Alternate letters**

**package** string;

**public** **class** TestTolowerCase {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

String s1="RAGINI";

StringBuilder sb=**new** StringBuilder();

//char c[]=s1.toCharArray();

**for**(**int** i=0;i<s1.length();i++)

{

**char** c =s1.charAt(i);

**if**(i % 2 == 0)

{

System.***out***.print(String.*valueOf*(c).toLowerCase());

}

**else**

{

System.***out***.print(String.*valueOf*(c).toUpperCase());

}

}

}

}

O/P : rAgInI

1. **Remove Space**

**package** testDay1;

**public** **class** RemoveSpaces {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

String s="ragini kulkarni muddadaga"; String s1;

s1=s.trim().replaceAll(" ","");

//also replaceAll("\\s","");

System.***out***.println(s1);

}

}

O/P : raginikulkarnimuddadaga

1. **Second Largest**

package testDay1;

import java.util.Arrays;

import java.util.Scanner;

public class SecondLargest {

public static void main(String[] args) {

// TODO Auto-generated method stub

Scanner s = new Scanner(System.in);

int tc = s.nextInt();

for (int i = 0; i < tc; i++) {

int arrCount = s.nextInt();

{

int arr[] = new int[arrCount];

for (int j = 0; j < arrCount; j++) {

arr[j] = s.nextInt();

}

Arrays.sort(arr);

System.out.println("large" + arr[arrCount - 2]);

}

}

}

}

O/P : 5

4 2 6 1 7

large6

1. **Next Greatest Number in the list**

package testDay1;

import java.util.Map;

import java.util.Stack;

import java.util.TreeMap;

public class TestNextGreatestNum {

public static void main(String[] args) {

// TODO Auto-generated method stub

int a[]={3,8,5,9,2,10};

int n=a.length;

new TestNextGreatestNum().printNextGreater(a,n);

}

public void printNextGreater(int[] arr,int size)

{

Map<Integer,Integer> data=new TreeMap<Integer,Integer>();

Stack<Integer> stack=new Stack<Integer>();

int curr;

stack.add(arr[0]);

Boolean flag;

for(int i=0;i<size;i++)

{

data.put(arr[i],-1);

}

for(int i=1;i<size;i++)

{

flag=true;

while(!stack.empty() && flag)

{

curr=stack.peek();

if(curr<arr[i])

{

stack.pop();

data.put(curr,arr[i]);

}

else

{

flag=false;

}

}

stack.add(arr[i]);

}

//Printing all pairs

for(int num:arr)

{

System.out.println(num+" "+data.get(num));

}

}

}

O/P : 3 8 8 9 5 9 9 10 2 10 10 -1