**MIDTERM CODING QUESTIONS**

**PROGRAM 1: REVERSE STRING**

**import** java.util.ArrayList;

**public** **class** midtermCoding {

**public** **static** **void** reverse(ArrayList<Integer> list)

{

System.***out***.println("Before reverse" +list);

**int** i=0;

**int** j = list.size() - 1;

**while**(i<j)

{ *swap*(i,j,list);

i++;

j--;

}

System.***out***.println("After reverse" +list);

}

**public** **static** **void** swap(**int** i,**int** j,ArrayList<Integer> list )

{

**int** temp = list.get(i);

list.set(i,list.get(j));

list.set(j,temp);

}

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

{

ArrayList<Integer> al = **new** ArrayList<Integer>();

al.add(1);

al.add(2);

al.add(3);

al.add(4);

al.add(5);

*reverse*(al);

}

}

**PROGRAM 2: GENERATE STRINGS**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class generateStrings {

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

System.out.println("PROGRAM 2: GENERATE STRINGS");

BufferedReader bufReader = new BufferedReader(new InputStreamReader(System.in));

System.out.println("Enter First String: ");

String firstStr = bufReader.readLine();

System.out.println("Enter Second String: ");

String secondStr = bufReader.readLine();

String returnedString = generateSeq(firstStr, secondStr);

System.out.println("Returned String : " + returnedString.trim());

}

public static String generateSeq(String firstStr, String secondStr) throws IOException {

String bigString = firstStr;

String smallString = secondStr;

String returnedString = "";

if (firstStr.length() <= secondStr.length()) {

bigString = firstStr;

smallString = secondStr;

} else {

bigString = secondStr;

smallString = firstStr;

}

for (int i = 0; i < bigString.length(); i++) {

for (int j = 0; j < smallString.length(); j++) {

if (!returnedString.contains("" + bigString.charAt(i) + smallString.charAt(j) + "")) {

returnedString = returnedString + bigString.charAt(i) + smallString.charAt(j) + " ";

}

}

}

return returnedString;

}

}

**PROGRAM 3: DUPLICATE NUMBER IN AN ARRAY**

**public** **class** duplicateNumberinArray {

**public** **static** **int** singleNumber(**int**[] nums) {

**int** x = 0;

**for** (**int** a : nums) {

x = x ^ a;

}

**return** x;

}

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

{

**int** a[] = {2,2,5,4,5,6,4};

System.***out***.println("The duplicate string in the array is: " +*singleNumber*(a));

}}

**QUESTION 4:**

import java.util.Arrays;

import java.util.HashMap;

import java.util.Map;

public class addingIndex {

public static int[] twoSum(int[] nums, int target) {

Map<Integer,Integer> map = new HashMap<Integer,Integer>();

int[] result = new int[2];

for(int i=0;i<nums.length;i++){

if(map.containsKey(nums[i])){

int num = map.get(nums[i]);

result[0] = num;

result[1] = i;

}

else{

map.put(target-nums[i], i);

}

}

System.out.println("Indices of the two numbers after adding to the target: " +Arrays.toString(result));

return result;

}

public static void main(String args[])

{

int[] nums = {2,7,11,15};

twoSum(nums, 13);

}}

**PROGRAM 5: PASCAL TRIANGLE**

**import** java.util.Scanner;

**public** **class** pascalTriangle {

**public** **static** **void** print(**int** n) {

**int**[] previousRow;

**int**[] currentRow = {1};

*printArray*(currentRow);

previousRow = currentRow;

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

currentRow = **new** **int**[i];

currentRow[0] = 1;

currentRow[i - 1] = 1;

**for** (**int** j = 0; j <= i - 3; j++) {

currentRow[j + 1] = previousRow[j] + previousRow[j + 1];

}

*printArray*(currentRow);

previousRow = currentRow;

}

}

**public** **static** **void** printArray(**int**[] array) {

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

System.***out***.print(array[i] + " ");

}

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

}

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

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

System.***out***.print("Enter the row number upto which Pascal's triangle has to be printed: ");

**int** row = scanner.nextInt();

*print*(row);

scanner.close();

}

}