**Task to do**

**Friday – 24/11/2023**

**Week 3**

1. Write a Java Program to convert from normal numbers to Roman Numerals.

The Romans were a clever bunch. They conquered most of Europe and ruled it for

hundreds of years. They invented concrete and straight roads and even bikinis. One

thing they never discovered though was the number zero. This made writing and dating

extensive histories of their exploits slightly more challenging, but the system of numbers

they came up with is still in use today. For example the BBC uses Roman numerals to

date their programs.

The Romans wrote numbers using letters - I, V, X, L, C, D, M. (notice these letters have

lots of straight lines and are hence easy to hack into stone tablets).

1 => I

10 => X

7 => VII

The maximum number supported by this notation is 3,999. (The Romans themselves

didn't tend to go any higher)

example of 1990.

In Roman numerals 1990 is MCMXC:

1000=M 900=CM 90=XC

2008 is written as MMVIII:

2000=MM 8=VIII

ANSWER:

*public class ConvertToRoman {*

*private static final int[] VALUES = { 1000, 900, 500, 400, 100, 90, 50, 40, 10, 9, 5, 4, 1 };*

*private static final String[] SYMBOLS = { "M", "CM", "D", "CD", "C", "XC", "L", "XL", "X", "IX", "V", "IV", "I" };*

*public static String toRoman(int number) {*

*if (number < 1 || number > 3999) {*

*throw new IllegalArgumentException("Input should be between 1 and 3999");*

*}*

*StringBuilder roman = new StringBuilder();*

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

*while (number >= VALUES[i]) {*

*number -= VALUES[i];*

*roman.append(SYMBOLS[i]);*

*}*

*}*

*return roman.toString();*

*}*

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

*int number1 = 1990;*

*System.out.println("1990 in Roman numerals is: " + toRoman(number1));*

*int number2 = 2008;*

*System.out.println("2008 in Roman numerals is: " + toRoman(number2));*

*}*

*}*

2. Write a Java Program convert a number into a string that contains raindrop sounds

corresponding to certain potential factors. A factor is a number that evenly divides into another

number, leaving no remainder.

ANSWER:

*public class RaindropSounds {*

*public static String convert(int number) {*

*StringBuilder result = new StringBuilder();*

*if (number % 3 == 0) {*

*result.append("Pling");*

*}*

*if (number % 5 == 0) {*

*result.append("Plang");*

*}*

*if (number % 7 == 0) {*

*result.append("Plong");*

*}*

*// If none of the factors (3, 5, 7) are present, append the number itself*

*if (result.length() == 0) {*

*result.append(number);*

*}*

*return result.toString();*

*}*

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

*int number1 = 28;*

*System.out.println("Result for 28: " + convert(number1));*

*int number2 = 30;*

*System.out.println("Result for 30: " + convert(number2));*

*int number3 = 34;*

*System.out.println("Result for 34: " + convert(number3));*

*}*

*}*

3. Write a Java Program to check if 2 strings given are Anagrams of one another.

ANSWER:

*import java.util.Arrays;*

*public class AnagramChecker {*

*public static boolean areAnagrams(String str1, String str2) {*

*str1 = str1.replaceAll("\\s", "").toLowerCase();*

*str2 = str2.replaceAll("\\s", "").toLowerCase();*

*if (str1.length() != str2.length()) {*

*return false;*

*}*

*char[] charArray1 = str1.toCharArray();*

*char[] charArray2 = str2.toCharArray();*

*Arrays.sort(charArray1);*

*Arrays.sort(charArray2);*

*return Arrays.equals(charArray1, charArray2);*

*}*

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

*String str1 = "stops";*

*String str2 = "spots";*

*if (areAnagrams(str1, str2)) {*

*System.out.println("The strings '" + str1 + "' and '" + str2 + "' are anagrams.");*

*} else {*

*System.out.println("The strings '" + str1 + "' and '" + str2 + "' are not anagrams.");*

*}*

*}*

*}*

4. Write a Java code to decode the colors of a resistance into the Value based on the color bands.

ANSWER:

*import java.util.HashMap;*

*import java.util.Map;*

*public class ResistorColorDecoder {*

*public static int decodeResistorColors(String[] colors) {*

*Map<String, Integer> colorValues = getColorValues();*

*int value = 0;*

*for (int i = 0; i < 2 && i < colors.length; i++) {*

*if (colorValues.containsKey(colors[i])) {*

*value = value \* 10 + colorValues.get(colors[i]);*

*} else {*

*System.out.println("Invalid color: " + colors[i]);*

*}*

*}*

*return value;*

*}*

*private static Map<String, Integer> getColorValues() {*

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

*colorValues.put("Black", 0);*

*colorValues.put("Brown", 1);*

*colorValues.put("Red", 2);*

*colorValues.put("Orange", 3);*

*colorValues.put("Yellow", 4);*

*colorValues.put("Green", 5);*

*colorValues.put("Blue", 6);*

*return colorValues;*

*}*

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

*String[] resistorColors = {"Brown", "Green", "Blue", "Red"};*

*int decodedValue = decodeResistorColors(resistorColors);*

*System.out.println("Decoded resistor value: " + decodedValue);*

*}*

*}*