|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task1 { |
|  | public static void main(String[] args) { |
|  | Scanner input = new Scanner(System.in); |
|  | System.out.printf("Print in array length: "); |
|  | int number = Integer.parseInt(input.nextLine()); |
|  | System.out.printf("Bell's number is: %s\n", bell(number)); |
|  | } |
|  |  |
|  |  |
|  | public static int bell(int number) { |
|  | int[][] arr = new int[number + 1][number + 1]; |
|  | arr[0][0] = 1; |
|  | for (int i = 1; i <= number; i++) { |
|  | arr[i][0] = arr[i - 1][i - 1]; |
|  | for (int j = 1; j <= i; j++) { |
|  | arr[i][j] = arr[i - 1][j - 1] + arr[i][j - 1]; |
|  | } |
|  | } |
|  | return arr[number][0]; |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task2 { |
|  | public static void main(String[] args) { |
|  | Scanner input = new Scanner(System.in); |
|  | System.out.printf("Print in word/sentence to translate: "); |
|  | String word = input.nextLine(); |
|  | System.out.printf("Translate is: %s\n", translate(word)); |
|  | } |
|  |  |
|  |  |
|  | public static String translate(String word) { |
|  | String[] words = word.split(" "); |
|  | String result = ""; |
|  | String vowels = "EeAaUuIiOoYy"; |
|  | String charactes = "!@#$%%^&\*()<>,./?;:'\"[{}]\_-+="; |
|  | String stringStart = ""; |
|  | WORDS: |
|  | for (String w: words) { |
|  | for (char c: vowels.toCharArray()) { |
|  | if (w.charAt(0) == c) { |
|  | w += "yay"; |
|  | result += w + " "; |
|  | continue WORDS; |
|  | } |
|  | } |
|  | stringStart = ""; |
|  | CHARS: |
|  | for (char c: w.toCharArray()) { |
|  | for (char ch: vowels.toCharArray()) { |
|  | if (c == ch) break CHARS; |
|  | } |
|  | stringStart += String.valueOf(c); |
|  | } |
|  | result += w.replace(stringStart, "") + stringStart + "ay" + " "; |
|  | for (char c: w.toCharArray()) { |
|  | for (char ch: charactes.toCharArray()) { |
|  | if (c == ch) result = result.trim().replace(String.valueOf(c), "") + c + " "; |
|  | } |
|  | } |
|  | } |
|  | return result.trim(); |
|  | } |
|  | } |

|  |
| --- |
| import java.util.regex.\*; |
|  | import java.util.\*; |
|  |  |
|  |  |
|  | public class Task3 { |
|  | public static String rgb = "rgb\\(([0-9]|[1-9][0-9]|[1-2][0-9][0-9])\\,([0-9]|[1-9][0-9]|[1-2][0-9][0-9])\\,([0-9]|[1-9][0-9]|[1-2][0-9][0-9])\\)"; |
|  | public static String rgba = "rgba\\(([0-9]|[1-9][0-9]|[1-2][0-9][0-9])\\,([0-9]|[1-9][0-9]|[1-2][0-9][0-9])\\,([0-9]|[1-9][0-9]|[1-2][0-9][0-9])\\,[0-1]\\)"; |
|  | public static void main(String[] args) { |
|  | Scanner input = new Scanner(System.in); |
|  | System.out.printf("Print in rgb/rgba: "); |
|  | String color = input.nextLine(); |
|  | System.out.printf("Result is: %s\n", validColor(color)); |
|  | } |
|  |  |
|  |  |
|  | public static boolean validColor(String color) { |
|  | if (Pattern.matches(rgb, color)) return true; |
|  | if (Pattern.matches(rgba, color)) return true; |
|  | return false; |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task4 { |
|  | public static void main(String[] args) { |
|  | Scanner input = new Scanner(System.in); |
|  | System.out.printf("Print in url: "); |
|  | String url = input.nextLine(); |
|  | System.out.printf("Print in params separated by space: "); |
|  | String[] params = input.nextLine().split(" "); |
|  | if (params.length > 0) System.out.printf("Result is: %s\n", stripUrlParams(url, params)); |
|  | else System.out.printf("Result is: %s\n", stripUrlParams(url)); |
|  | } |
|  |  |
|  |  |
|  | public static String stripUrlParams(String url, String... args) { |
|  | String result = ""; |
|  | Map<String, String> params = new HashMap<String, String>(); |
|  | if (url.split("\\?").length == 1) return url; |
|  | String[] urlparams = url.split("\\?")[1].split("\\&"); |
|  | PARAMS: |
|  | for (String param: urlparams) { |
|  | String key = param.split("\\=")[0]; |
|  | String value = param.split("\\=")[1]; |
|  | for (String arg: args) { |
|  | if (arg.equals(key)) continue PARAMS; |
|  | } |
|  | params.put(key, value); |
|  | } |
|  | for (Map.Entry<String, String> entry: params.entrySet()) { |
|  | result += entry.getKey() + "=" + entry.getValue() + " "; |
|  | } |
|  | return url.split("\\?")[0] + "?" + result.trim().replace(" ", "&"); |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task6 { |
|  | public static void main(String[] args) { |
|  | Scanner input = new Scanner(System.in); |
|  | System.out.printf("Print in number: "); |
|  | Integer number = Integer.parseInt(input.nextLine()); |
|  | System.out.println(ulam(number)); |
|  | } |
|  |  |
|  |  |
|  | public static Integer ulam(Integer n) { |
|  | List<Integer> ulams = new ArrayList<Integer>(); |
|  | ulams.add(1); |
|  | ulams.add(2); |
|  | int next = ulams.get(ulams.size() - 1) + 1; |
|  | int count; |
|  | while (ulams.size() != n) { |
|  | count = 0; |
|  | for (int i = 0; i < ulams.size() - 1; i++) { |
|  | for (int j = i + 1; j < ulams.size(); j++) { |
|  | if (ulams.get(i) + ulams.get(j) == next) count++; |
|  | if (count > 1) break; |
|  | } |
|  | if (count > 1) break; |
|  | } |
|  | if (count == 1) ulams.add(next); |
|  | next++; |
|  | } |
|  | return ulams.get(n - 1); |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task7 { |
|  | public static void main(String[] args) { |
|  | Scanner input = new Scanner(System.in); |
|  | System.out.printf("Print in string: "); |
|  | String string = input.nextLine(); |
|  | System.out.println(longestNonrepeatingSubstring(string)); |
|  | } |
|  |  |
|  |  |
|  | public static String longestNonrepeatingSubstring(String string) { |
|  | ArrayList<String> uniques = new ArrayList<String>(); |
|  | String currentString = ""; |
|  | String currentChar; |
|  | int indexMax = 0; |
|  | for (char c: string.toCharArray()) { |
|  | currentChar = String.valueOf(c); |
|  | if (currentString.indexOf(currentChar) != -1) { |
|  | uniques.add(currentString); |
|  | currentString = currentChar; |
|  | continue; |
|  | } |
|  | currentString += currentChar; |
|  | } |
|  | if (currentString.length() != 0) uniques.add(currentString); |
|  | for (int i = 0; i < uniques.size(); i++) { |
|  | if (uniques.get(indexMax).length() < uniques.get(i).length()) { |
|  | indexMax = i; |
|  | } |
|  | } |
|  | return uniques.get(indexMax); |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task8 { |
|  | public static void main(String[] args) { |
|  | Scanner input = new Scanner(System.in); |
|  | System.out.printf("Print in number: "); |
|  | int number = Integer.parseInt(input.nextLine()); |
|  | System.out.println(convertToRoman(number)); |
|  | } |
|  |  |
|  |  |
|  | public static String convertToRoman(int number) { |
|  | String[] belowTen = new String[] {"" /\*0\*/, "I", "II", "III", "IV", "V", "VI", "VII", "VIII", "IX"}; |
|  | String[] hunderedToThousand = new String[] {"" /\*0\*/, "С", "СС", "ССС", "CD", "D", "DC", "DCC", "DCCC", "CM"}; |
|  | String[] tenToHundered = new String[] {"" /\*0\*/, "X", "XX", "XXX", "XL", "L", "LX", "LXX", "LXXX", "XC"}; |
|  | String result = ""; |
|  | if (number > 3999) return "This number is too big... sorry......"; |
|  | if (number > 999) { |
|  | for (int i = 0; i < number / 1000; i++) result += "M"; |
|  | number = number % 1000; |
|  | } |
|  | if (number / 100 > 0) { |
|  | result += hunderedToThousand[number / 100]; |
|  | number = number % 100; |
|  | } |
|  | if (number / 10 > 0) { |
|  | result += tenToHundered[number / 10]; |
|  | number = number % 10; |
|  | } |
|  | if (number > 0) result += belowTen[number]; |
|  | return result; |
|  | } |
|  | } |

|  |
| --- |
| import javax.script.ScriptEngineManager; |
|  | import javax.script.ScriptEngine; |
|  | import javax.script.ScriptException; |
|  | import java.util.\*; |
|  |  |
|  |  |
|  | public class Task9 { |
|  | public static void main(String[] args) throws ScriptException { |
|  | Scanner input = new Scanner(System.in); |
|  | System.out.printf("Print in formula: "); |
|  | String inputFormula = input.nextLine(); |
|  | System.out.println(formula(inputFormula)); |
|  | } |
|  |  |
|  |  |
|  | public static boolean formula(String input) throws ScriptException { |
|  | ScriptEngineManager mgr = new ScriptEngineManager(); |
|  | ScriptEngine engine = mgr.getEngineByName("JavaScript"); |
|  | ArrayList<String> formulas = new ArrayList<>(Arrays.asList(input.split("="))); |
|  | double result1; |
|  | double result2; |
|  | for (int i = 0; i < formulas.size() - 1; i++) { |
|  | result1 = Double.parseDouble(engine.eval(formulas.get(i)).toString()); |
|  | result2 = Double.parseDouble(engine.eval(formulas.get(i + 1)).toString()); |
|  | if (result1 != result2) return false; |
|  | } |
|  | return true; |
|  | } |
|  | } |