Задачи модуль 4

по дисциплине

“Объектно-ориентированное программирование”

Выполнила студентка

группы БФИ1901

Киселева Анна

Москва 2020

**6/6**

1. Число Белла - это количество способов, которыми массив из n элементов может быть разбит на непустые подмножества. Создайте функцию, которая принимает число n и возвращает соответствующее число Белла.

Пример:

bell(1) ➞ 1

// sampleArr = [1]

// possiblePartitions = [[[1]]]

bell(2) ➞ 2

// sampleArr = [1, 2]

// possiblePartitions = [[[1, 2]], [[1], [2]]]

bell(3) ➞ 5

// sampleArr = [1, 2, 3]

// possiblePartitions = [[[1, 2, 3]], [[1, 2], [3]], [[1], [2, 3]], [[1, 3], [2]], [[1], [2], [3]]]

import java.util.Arrays;

public class Task6 {

/\*1. Число Белла - это количество способов, которыми массив из n элементов может

быть разбит на непустые подмножества. Создайте функцию, которая принимает

число n и возвращает соответствующее число Белла.\*/

public static int bell(int num) {

int[][] mass = new int[num + 1][num + 1];

int sum = 0;

mass[0][0] = 1;

mass[num][num] = 1;

for (int n = 1; n <= num; n++) {

for (int i = 1; i < num; i++) {

mass[n][i] = mass[n - 1][i - 1] + i \* mass[n - 1][i];

}

}

for (int i = 0; i <= num; i++) sum += mass[num][i];

return sum;

}

1. В «поросячей латыни» (свинский латинский) есть два очень простых правила:

– Если слово начинается с согласного, переместите первую букву (буквы) слова до гласного до конца слова и добавьте «ay» в конец.

have ➞ avehay

cram ➞ amcray

take ➞ aketay

cat ➞ atcay

shrimp ➞ impshray

trebuchet ➞ ebuchettray

* Если слово начинается с гласной, добавьте "yay" в конце слова.

ate ➞ ateyay

apple ➞ appleyay

oaken ➞ oakenyay

eagle ➞ eagleyay

Напишите две функции, чтобы сделать переводчик с английского на свинский латинский. Первая функция translateWord (word) получает слово на английском и возвращает это слово, переведенное на латинский язык. Вторая функция translateSentence (предложение) берет английское предложение и возвращает это предложение, переведенное на латинский язык.

Пример:

translateWord("flag") ➞ "agflay"

translateWord("Apple") ➞ "Appleyay"

translateWord("button") ➞ "uttonbay"

translateWord("") ➞ ""

translateSentence("I like to eat honey waffles.") ➞ "Iyay ikelay otay eatyay oneyhay afflesway."

translateSentence("Do you think it is going to rain today?") ➞ "Oday youyay inkthay ityay isyay oinggay otay ainray odaytay?"

Примечание:

– Регулярные выражения помогут вам не исказить пунктуацию в предложении.

– Если исходное слово или предложение начинается с заглавной буквы, перевод должен сохранить свой регистр

/\* 2. В «поросячей латыни» (свинский латинский) есть два очень простых правила:

– Если слово начинается с согласного, переместите первую букву (буквы) слова до

гласного до конца слова и добавьте «ay» в конец.\*/

public static String translateWord(String word){

if (word.matches("[aeiouy]")) {

word += "yay";

}

else {

String newWord = word.split("[aeiouy]")[0];

word = word.replaceFirst(newWord,"")+newWord+"ay";

}

return word;

}

public static String translateSentence(String word){

String vow = "aeiouyAEIOUY";

String[] newword = word.substring(0, word.length()-1).split(" ");

word = "";

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

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

if (vow.indexOf(newword[i].charAt(j)) != -1) {

word += newword[i] + "yay ";

break;

}

else {

String newWord = newword[i].split("[aeiyouAEIYOU]")[0];

word += newword[i].replaceFirst(newWord, "") + newWord + "ay ";

break;

}

}

}

return word + ".";

}

1. Учитывая параметры RGB (A) CSS, определите, является ли формат принимаемых значений допустимым или нет. Создайте функцию, которая принимает строку (например, " rgb(0, 0, 0)") и возвращает true, если ее формат правильный, в противном случае возвращает false.

Пример:

validColor("rgb(0,0,0)") ➞ true

validColor("rgb(0,,0)") ➞ false

validColor("rgb(255,256,255)") ➞ false

validColor("rgba(0,0,0,0.123456789)") ➞ true

/\* 3. Учитывая параметры RGB (A) CSS, определите, является ли формат принимаемых

значений допустимым или нет. Создайте функцию, которая принимает строку

(например, " rgb(0, 0, 0)") и возвращает true, если ее формат правильный, в

противном случае возвращает false.\*/

public static boolean validColor (String rgb){

int num = rgb.indexOf('(');

if (rgb.contains(" "))

return false;

String [] newrgb = rgb.substring(rgb.indexOf('(')+1,rgb.indexOf(')')).split(",");

double [] n = new double [num];

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

n[i] = Double.parseDouble(newrgb[i]);

System.out.println(newrgb[i]);

}

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

if (n[i] >= 0 && n[i] <= 255 && num == newrgb.length)

return true;

else

return false;

}

return false;

}

1. Создайте функцию, которая принимает URL (строку), удаляет дублирующиеся параметры запроса и параметры, указанные во втором аргументе (который будет необязательным массивом).

Пример:

stripUrlParams("https://edabit.com?a=1&b=2&a=2") ➞ "https://edabit.com?a=2&b=2"

stripUrlParams("https://edabit.com?a=1&b=2&a=2", ["b"]) ➞ "https://edabit.com?a=2"

stripUrlParams("https://edabit.com", ["b"]) ➞ "https://edabit.com"

Примечание:

– Второй аргумент paramsToStrip является необязательным.

– paramsToStrip может содержать несколько параметров.

– Если есть повторяющиеся параметры запроса с разными значениями, используйте значение последнего встречающегося параметра (см. Примеры № 1 и № 2 выше).

/\* 4. Создайте функцию, которая принимает URL (строку), удаляет дублирующиеся

параметры запроса и параметры, указанные во втором аргументе (который будет

необязательным массивом).\*/

// https://edabit.com?a=1&b=2&a=2

public static String stripUrlParams(String url, String ... parameters){

String str = " ";

if (!url.contains("?"))

return url;

else if(url.indexOf("?")!= url.length()){

str = url.substring(url.indexOf("?")+1);

url = url.substring(0, url.indexOf("?"));

}

}

1. Напишите функцию, которая извлекает три самых длинных слова из заголовка газеты и преобразует их в хэштеги. Если несколько слов одинаковой длины, найдите слово, которое встречается первым.

Пример:

getHashTags("How the Avocado Became the Fruit of the Global Trade")

➞ ["#avocado", "#became", "#global"]

getHashTags("Why You Will Probably Pay More for Your Christmas Tree This Year")

➞ ["#christmas", "#probably", "#will"]

getHashTags("Hey Parents, Surprise, Fruit Juice Is Not Fruit")

➞ ["#surprise", "#parents", "#fruit"]

getHashTags("Visualizing Science")

➞ ["#visualizing", "#science"]

Примечание:

– Если заголовок содержит менее 3 слов, просто расположите слова в заголовке по длине в порядке убывания (см. Пример №4).

– Пунктуация не считается с длиной слова.

/\* 5. Напишите функцию, которая извлекает три самых длинных слова из заголовка

газеты и преобразует их в хэштеги. Если несколько слов одинаковой длины,

найдите слово, которое встречается первым.\*/

public static String [] getHashTags(String str){

String[] hashtags = new String[]{""};

return hashtags;

}

1. Последовательность Улама начинается с:

ulam = [1, 2]

Следующее число в последовательности - это наименьшее положительное число, равное сумме двух разных чисел (которые уже есть в последовательности) ровно одним способом. Тривиально, это 3, так как в стартовой последовательности есть только 2 числа.

ulam = [1, 2, 3]

Следующее число 4, которое является суммой 3 + 1. 4 также равно 2 + 2, но это уравнение не учитывается, так как 2 добавления должны быть различны.

ulam = [1, 2, 3, 4]

Следующее число не может быть 5, так как 5 = 1 + 4, но также и 5 = 2 + 3. Должен быть только один способ сделать число Улама из 2 различных добавлений, найденных в последовательности. Следующее число 6 (2 + 4). Есть 2 способа сделать 7 (1 + 6 или 3 + 4), поэтому следующий - 8 (2 + 6). И так далее.

ulam = [1, 2, 3, 4, 6, 8, 11, 13, 16, 18, 26, 28, 36, 38, 47, 48, 53, …]

Создайте функцию, которая принимает число n и возвращает n-е число в последовательности Улама.

Пример:

ulam(4) ➞ 4

ulam(9) ➞ 16

ulam(206) ➞ 1856

/\* 6.Создайте функцию, которая принимает число n и возвращает n-е число в

последовательности Улама.\*/

public static int ulma ( int n){

return n;

}

1. Напишите функцию, которая возвращает самую длинную неповторяющуюся подстроку для строкового ввода.

Пример:

longestNonrepeatingSubstring("abcabcbb") ➞ "abc"

longestNonrepeatingSubstring("aaaaaa") ➞ "a"

longestNonrepeatingSubstring("abcde") ➞ "abcde"

longestNonrepeatingSubstring("abcda") ➞ "abcd"

Примечание:

– Если несколько подстрок связаны по длине, верните ту, которая возникает первой.

– Бонус: можете ли вы решить эту проблему в линейном времени?

1. Создайте функцию, которая принимает арабское число и преобразует его в римское число.

Пример:

convertToRoman(2) ➞ "II"

convertToRoman(12) ➞ "XII"

convertToRoman(16) ➞ "XVI"

Примечание:

– Все римские цифры должны быть возвращены в верхнем регистре.

– Самое большое число, которое может быть представлено в этой нотации, - 3,999.

1. Создайте функцию, которая принимает строку и возвращает true или false в зависимости от того, является ли формула правильной или нет.

Пример:

formula("6 \* 4 = 24") ➞ true

formula("18 / 17 = 2") ➞ false

formula("16 \* 10 = 160 = 14 + 120") ➞ false

/\* 9. Создайте функцию, которая принимает строку и возвращает true или false в

зависимости от того, является ли формула правильной или нет. \*/

public static boolean formula (String formula){

int ans = Integer.parseInt(formula.substring(formula.indexOf('=')));

switch (formula){

}

return false;

}

1. Число может не быть палиндромом, но его потомком может быть. Прямой потомок числа создается путем суммирования каждой пары соседних цифр, чтобы создать цифры следующего числа.

Например, 123312 – это не палиндром, а его следующий потомок 363, где: 3 = 1 + 2; 6 = 3 + 3; 3 = 1 + 2.

Создайте функцию, которая возвращает значение true, если само число является палиндромом или любой из его потомков вплоть до 2 цифр (однозначное число - тривиально палиндром).

Пример:

palindromedescendant(11211230) ➞ true

// 11211230 ➞ 2333 ➞ 56 ➞ 11

palindromeDescendant(13001120) ➞ true

// 13001120 ➞ 4022 ➞ 44

palindromeDescendant(23336014) ➞ true

// 23336014 ➞ 5665

palindromeDescendant(11) ➞ true

// Number itself is a palindrome.

Примечание:

– Числа всегда будут иметь четное число цифр.