Министерство цифрового развития, связи и массовых коммуникаций Государственное образовательного учреждение высшего образования

Ордена Трудового Красного Знамени

«Московский технический университет связи и информатики»

Задачи для самостоятельного решения

по дисциплине «Структура и алгоритмы обработки данных»

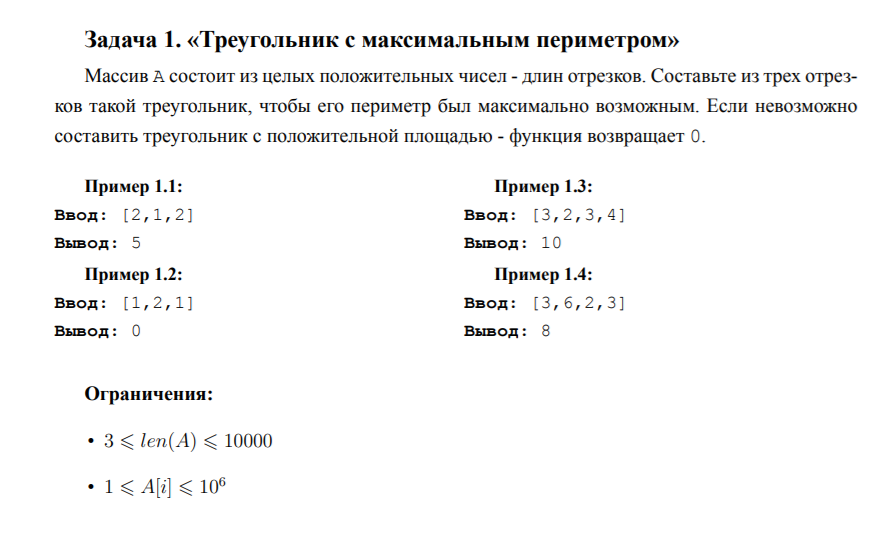
Выполнил студент группы БВТ1905:

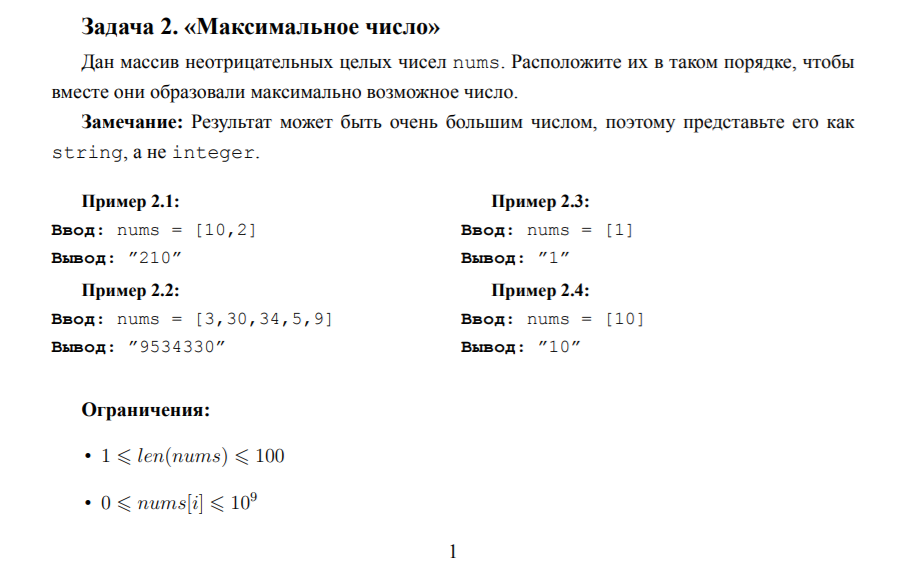
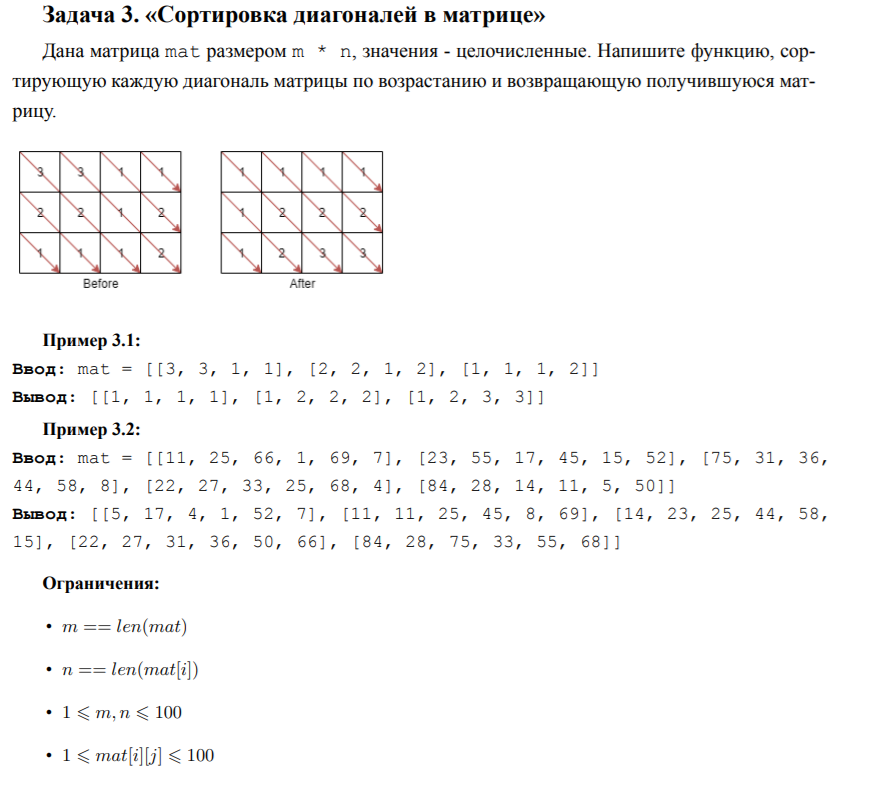
Прокудин Е. В.

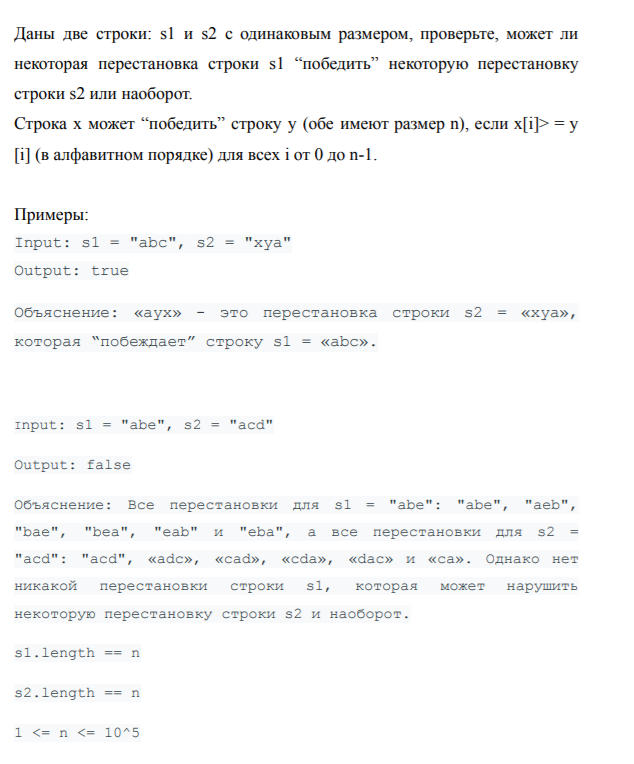
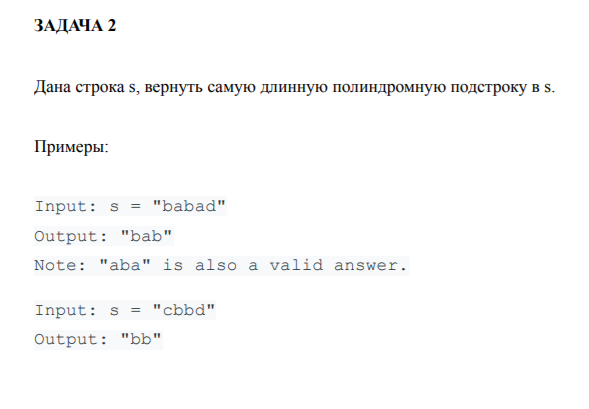
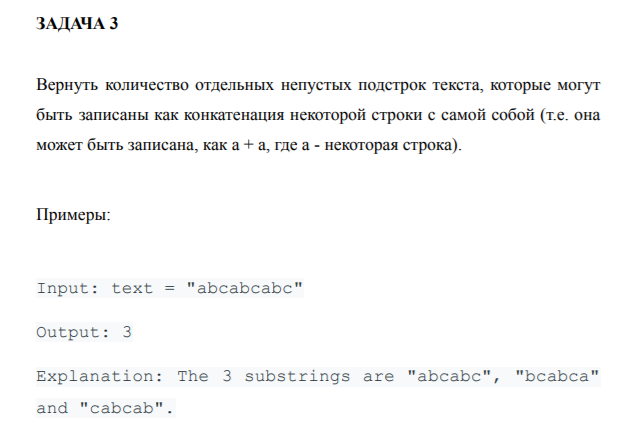
Москва

2021

# Задание



**Задача 1.**

В Тридевятом Царстве уже много лет как установилась конституционная монархия. В парламент Тридевятого царства входят 26 партий, обозначаемых строчными буквами английского алфавита от «a» до «z». Заседание парламента в соответствии с регламентом проходит по следующей схеме:

* В начале заседания никаких законопроектов на повестке дня не стоит.
* Любая партия (в том числе и та, которая внесла текущий законопроект) в любой момент времени может внести на обсуждение новый законопроект.
* При вносе партией нового законопроекта обсуждение предыдущего приостанавливается и начинается обсуждение нового. После того, как парламентарии голосуют за новый законопроект, обсуждение законопроекта, прерванного им, продолжается.
* Дискуссия по любому законопроекту обязана завершиться голосованием по этому законопроекту.
* В конце заседания не должно остаться внесённых законопроектов, дискуссия по которым не завершена.

Вам дана запись о событиях на сессии парламента. Каждое событие является либо внесением нового законопроекта, либо голосованием за какой-то законопроект, причём в обоих случаях известен номер партии, этот законопроект предложившей. События даны в том порядке, в котором они происходили. Проверьте, может ли данная запись о событиях соответствовать какому-либо заседанию, удовлетворяющему порядку проведения, описанному выше.

### Входные данные

Первая строка входных данных содержит одно целое число *K* – количество строк в записи сессии (1 ≤ *K* ≤ 1000). Каждая строка соответствует одному из двух событий:

* Add *x* – партия x внесла на голосование законопроект.
* Vote *x* – прошло голосование за документ, внесённый партией *x*.

Здесь *x* – строчная буква английского алфавита от «a» до «z», задающая партию.

### Вывод

Выведите «Yes», если существует корректный порядок проведения заседания, который мог привести к такой записи, и «No», если ни при каком корректном порядке проведения заседания данная запись появиться не могла.

## Задача 2. Правильный подмногоугольник

Задан правильный *N*-угольник. Требуется выбрать наименьшее количество его вершин, которые также образуют правильный многоугольник.

**Входные данные**

Входные данные содержат одно целое число *N* (3 ≤ *N* ≤ 1012).

**Вывод**

Выведите одно число — наименьшее количество вершин заданного многоугольника, которые образуют правильный многоугольник.

На форуме, на котором обсуждаются задачи олимпиад по информатике, ввели следующий аналог капчи. Участнику выдаётся строка из *N* десятичных цифр (без ведущих нулей). В качестве ответа требуется ввести такое основание системы счисления *B*, что в этой системе счисления выданная запись будет соответствовать составному числу (назовем его *D*), а также число *X*, большее 1 и меньшее *D*, являющееся делителем *D*.

При этом *B* и *X* не должны превосходить 109.

По заданной строке десятичных цифр найдите любую пару чисел *B* и *X*, удовлетворяющую ограничениям, или ответьте, что решения в заданных ограничениях не существует.

### Входные данные

Входные данные состоят из непустой строки длиной до 3 ⋅ 106 символов, составленной из цифр от 0 до 9 и не начинающейся с 0.

### Вывод

Если решение существует, выведите два числа – основание системы счисления *B* и делитель *X*, записанные в десятичной системе счисления. Оба числа должны удовлетворять ограничениям 2 ≤ *B*, *X* ≤ 109. Если решения не существует, выведите –1.

# Код программы

function generateArray(lenght){

    let arr = [];

    for(let i = 0; i <lenght; i++){

        arr[i] = 0 + Math.floor(Math.random() \* (Math.pow(10,6) - 0 + 1));

    }

    return arr;

}

function thriaAngle(array){

    let per = 0, maxA, maxB,maxC;

    for (let i = 0; i < array.length - 2; i++){

        maxA = array[i];

        for (let j = 1; j < array.length - 1; j++){

            maxB = array[j];

            for (let k = 2; k < array.length ; k++){

                maxC = array[k];

                if(i != j && j != k && i != k){

                    if ((maxA + maxB > maxC) && (maxB + maxC > maxA) && (maxA + maxC > maxB)){

                        console.log(maxA , maxB, maxC);

                        if(per<maxA + maxB + maxC){

                            per = maxA + maxB + maxC;

                        }

                    }

                }

            }

        }

     }

    return per;

}

let array = generateArray(4);

console.log(array);

console.log(thriaAngle(array));

let nums = [3,30,34,5,954,1,2972,3,574,5,6,724,8,9,10,11,12,20,30,33,90,900,1000,354,99,1111,355,324]

// let nums = [0,0,0,1];

function maxNum(array){

    let check = array.every(element => element == 0);

    if(check){

        return 0;

    }

    return array

        .map((item) => item.toString())

        .sort((str1,str2) => (str2+str1) - (str1+str2))

        .join('');

}

console.log(maxNum(nums));

let mat = [[3, 3, 5, 6], [2, 2, 1, 8], [1, 1, 1, 2]]

mat = [[3, 3, 5, 6], [2, 2, 1, 8], [1, 1, 1, 2],[2, 2, 1, 8],[2, 2, 1, 8]]

mat = [[11, 25, 66, 1, 69, 7], [23, 55, 17, 45, 15, 52], [75, 31, 36, 44, 58, 8], [22, 27, 33, 25, 68, 4], [84, 28, 14, 11, 5, 50]]

mat = [[3, 3, 5, 6], [2, 2, 1, 8], [1, 1, 1, 2],[2, 2, 1, 8]]

let rowLen = mat[0].length, columnLen = mat.length

mat.forEach((e)=>{

    console.log(e);

})

console.log('sorting')

let buff = []

let triangleDia = Math.min(rowLen, columnLen ) - 1

let countMainDia = Math.abs(rowLen-columnLen)+1

let lenMainDia = Math.min(rowLen, columnLen )

function triangleUp(){

    for(let i = 0; i < triangleDia; i++){

        //console.log(triangleDia);

        for(let j = 0; j < i+1; j++ ){

            //console.log(i,j, rowLen-1-i+j, mat[j][rowLen-1-i+j])

            buff.push(mat[j][rowLen-1-i+j])

        }

        //console.log('buff',buff );

        buff.sort((a,b)=>{return a-b})

        for(let j = 0; j < i+1; j++ ){

            //console.log(j,rowLen-1-i, i,buff[j])

            mat[j][rowLen-1-i+j] = buff[j]

        }

        buff = []

    }

}

triangleUp()

function diaMain(){

    //console.log(countMainDia);

    for(let i = 0; i < countMainDia; i++){

        if(rowLen>columnLen){

            for(let j = 0; j < lenMainDia; j++){

                buff.push(mat[j][j+i])

            }

            buff.sort((a,b)=>{return a-b})

            for(let j = 0; j < lenMainDia; j++ ){

                mat[j][j+i] = buff[j]

            }

            buff = []

        }else{

            for(let j = 0; j < lenMainDia; j++){

                buff.push(mat[j+i][j])

            }

            buff.sort((a,b)=>{return a-b})

            for(let j = 0; j < lenMainDia; j++ ){

                mat[j+i][j] = buff[j]

            }

            buff = []

        }

    }

}

diaMain()

function triangleDown(){

    for(let i = 0; i < triangleDia; i++){

        //console.log(triangleDia);

        for(let j = 0; j < i+1; j++ ){

            buff.push(mat[columnLen-1-i+j][j])

        }

        //console.log('buff',buff );

        buff.sort((a,b)=>{return a-b})

        for(let j = 0; j < i+1; j++ ){

            //console.log(j,rowLen-1-i, i,buff[j])

            mat[columnLen-1-i+j][j] = buff[j]

        }

        buff = []

    }

}

triangleDown()

mat.forEach((e)=>{

    console.log(e);

})

function findWin(str1,str2){

    if(str1.length === 0 || str2.length === 0){

        return `it is empty`;

    }else if (str1.length !== str2.length){

        return `Can't compare it`;

    }

   let arrChar1 = str1.split('').sort();

   let arrChar2 = str2.split('').sort();

    console.log(arrChar1)

    console.log(arrChar2)

   if((arrChar1.every((el, idx) => el <= arrChar2[idx])) || (arrChar2.every((el, idx) => el <= arrChar1[idx]))) {

    return true;

   }

    return false;

}

let str1 = `abe`;

let str2 = 'acd';

console.log(findWin(str1,str2));

function isPolindrome(string){

    return string === string.split('').reverse().join('');

}

function largestPolindrome(string){

    if(string.length ===0){

        return `it is empty`;

    }

    let largest ='';

    for(let i = 0; i<string.length; i++){

        for(let j = 0; j<string.length; j++){

        let substr = string.slice(i,j);

            if(isPolindrome(substr)){

                if(largest.length<substr.length){

                    largest = substr;

                }

            }

        }

    }

    if(largest.length === 0){

        return `Haven't polindrome`;

    }

    return largest;

}

let string = 'babaaabaaa';

console.log(largestPolindrome(string));

const isConcat = (string, subString) => {

    if (subString.length === 0) {

        return false

    }

    for (let i = 0; i < string.length; i += subString.length)

        if (string.slice(i, i + subString.length) !== subString)

            return false

    return true

}

const largestConcat = (string) => {

    let largest = ''

    for (let i = 0; i < string.length; i++) {

        for (let j = 0; j < string.length; j++) {

            const subString = string.slice(i, j)

            console.log(subString)

            if (isConcat(string, subString)) {

                if (largest.length < subString.length) {

                    largest = subString

                }

            }

        }

    }

    return largest

}

largestConcat('abcabcabc')

let inputData = ['add a','add b','add a','vote a','vote b','vote a']

function slow(){

    let el = buff[buff.length-1]

    for (let j = buff.length-1; j > 0 ; j--) {

        buff[j] = buff[j-1]

    }

    buff[0] = el

}

let buff = []

function check(){

    if(inputData.length % 2 != 0){

        return 'No'

    }

    else{

        for(let i = 0; i< inputData.length; i++){

            if(inputData[i].split(' ')[0] == 'add'){

                buff.push(inputData[i].split(' ')[1])

            }else if(inputData[i].split(' ')[0] == 'vote'){

                if(buff[buff.length-1] == inputData[i].split(' ')[1] ){

                    slow()

                }else{

                    return 'No'

                }

            }

        }

        return 'Yes'

    }

}

let answ = check()

console.log(answ)

let number = 90

function kuku(){

    let i = 3

    while(number % i != 0){

        i++

    }

    console.log(i)

}

kuku()

let num = 100

if(num != 1){

    console.log('Система счисления : ' , num, 'число в системе 10 делитель число 2');

}

**Вывод**

На рисунке 1 представлен результат работы программы составления наибольшего числа



Рисунок - 1 Результат работы программы составления наибольшего числа

На рисунке 2 представлен результат работы программы поиска палиндрома



Рисунок - 2 Результат работы программы поиска палиндрома

На рисунке 3 представлен результат работы программы проверки голосований

Рисунок - 3 Результат работы программы проверки голосований

В ходе выполнеия данных заданий я применил на практике знания полученные в течении курса, разработал программы для решения заданий. Внедрение полученного за курс материала помогло оптимизировать мои алгоритмы.