**Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации**

**Ордена Трудового Красного Знамени Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра «Математическая кибернетика и информационные технологии»

Курсовая работа

Вариант № 1

Выполнил:

студент группы БФИ2001

Степанов Д.А.

Проверил: Симонов С.Е.

Москва 2022

**Оглавление**

Цель3

Задача 1.4

Задача 2.5

Задача 3.6

Задача 4.8

Задача 5.10

Задача 6.11

Задача 7.13

Задача 8.14

Задача 9.16

Задача 10.19

Вывод22

**Цель**

Целью курсовой работы является выполнения поставленных задач, описание хода решения, описания алгоритмов решения задач на языке Python и демонстрация результатов работы.

**Задача 1**

**Условие:**

Леша не любит скучать. Поэтому, когда ему скучно, он придумывает игры. Как-то раз Леша придумал следующую игру.

Задана последовательность a, состоящая из n целых чисел. Игрок может сделать несколько ходов. За один ход игрок может выбрать некоторый элемент последовательности (обозначим выбранный элемент ak) и удалить его, при этом из последователости также удаляются все элементы, равные ak + 1 и ak - 1. Описанный ход приносит игроку ak очков.

Леша максималист и поэтому хочет набрать как можно больше очков. Какое максимальное количество очков он сможет набрать?

Входные данные В первой строке задано целое число n (1 ≤ n ≤ 105) — количество элементов последовательности. Во второй строке записаны n целых чисел a1, a2, ..., an (1 ≤ ai ≤ 105) — элементы последовательности.

Выходные данные Выведите целое число — максимальное количество очков, которые может набрать Леша.

**Примечание:**  
Рассмотрим третий тестовый пример. В этом тестовом примере нужно действовать так.

Первоначально нужно выбрать любой элемент, равный 2. Тогда последовательность станет равна [2, 2, 2, 2]. Далее, делаем еще 4 хода, на каждом ходу выбираем любой элемент, равный 2. Итого мы заработали 10 очков.

**Ход решения:**

В этой задаче необходимо максимизировать сумму набранных очков. Первым делом нужно создать массив всех чисел d[x], где х – это количество чисел в этом массиве. Дальше считаем по функцииям:

*f*(*i*) = *max*(*f*(*i* - 1), *f*(*i* - 2) +  d[x]·*i*), 2 ≤ *i* ≤ *n*;

*f*(1) =  d[1];

*f*(0) = 0;

Ответ будет содержаться в *f*(*n*).

**Листинг 1 – Решение задачи №1**

d = 9

c = "1 2 1 3 2 2 2 2 3"

def task1(d,c):

d=[0]\*100

for x in map(int, c.split()):

d[x]+=x

a=b=0

for i in d:

a,b=max(a,i+b),a

return(a)

task1(d,c)

**Результат работы:**

Вывод: 10

**Задача 2**

**Условие:**

Дана строка s. Требуется определить, существуют ли в данной строке s две непересекающиеся подстроки "AB" и "BA" (подстроки могут идти в любом порядке).

Входные данные\ На вход подаётся строка s длиной от 1 до 105 символов, состоящая из заглавных букв латинского алфавита.\

Выходные данные\ Выведите "YES" (без кавычек), если строка s содержит две непересекающиеся подстроки "AB" и "BA", и "NO" иначе.

Примеры\ входные данные\ ABA\ выходные данные\ NO\ входные данные\ BACFAB\ выходные данные\ YES\ входные данные\ AXBYBXA\ выходные данные\ NO

**Ход решения:**

Проверим две возможности — когда подстрока "AB" идет первее "BA" и наоборот. Проверять можно следующим образом: найти самое первое вхождение "AB" в исходной строке и рассмотреть подстроки длины 2 правее. Если среди них встретилась подстрока "BA" — ответ "YES". Аналогично проверяется второй случай. Если оба варианта не выполнены, ответ "NO".

**Листинг 2 – Решение задачи №2**

s = "BACFAB"

def task2(s):

u=s.find('AB')

v=s.find('BA')

if (u+1and s.find('BA',u+2)+1) or (v+1and s.find('AB',v+2)+1):

return("YES")

else:

return("NO")

task2(s)

**Результат работы:**

YES

**Задача 3**

**Условие:**

Кефа решил отпраздновать свой первый крупный заработок походом в ресторан.

Он живет возле необычного парка. Парк представляет из себя подвешенное дерево из n вершин c корнем в вершине 1. В вершине 1 также находится дом Кефы. К сожалению для нашего героя, в парке также находятся коты. Кефа уже выяснил номера вершин, в которых находятся коты.

В листовых вершинах парка находятся рестораны. Кефа хочет выбрать ресторан, в который он пойдет, но, к сожалению, он очень боится котов, поэтому он ни за что не пойдёт в ресторан, на пути к которому от его дома найдётся более m подряд идущих вершин с котами.

Ваша задача — помочь Кефе посчитать количество ресторанов, в которые он может сходить.

Входные данные  
В первой строке записаны два целых числа n и m (2 ≤ n ≤ 105, 1 ≤ m ≤ n) — количество вершин дерева и максимальное количество подряд идущих вершин с котами, которое способен перенести Кефа.

Во второй строке содержится n целых чисел a1, a2, ..., an, где каждое ai либо равняется 0 (тогда в вершине i нет кота), либо равняется 1 (тогда в вершине i есть кот).

В следующих n - 1 строках записаны ребра дерева в формате «xi yi»(без кавычек) (1 ≤ xi, yi ≤ n, xi ≠ yi), где xi и yi — вершины дерева, соединенные очередным ребром.

Гарантируется, что данный набор рёбер задаёт дерево.

**Ход решения:**

Будем спускаться по дереву от корня, поддерживая дополнительный параметр *k*  — количество встреченных подряд вершин с котами. Если *k* превысило *m*, то выходим. Тогда ответ — это количество листьев, до которых мы смогли дойти.

**Листинг 3 – Решение задачи №3**

M=lambda:map(int,input().split())

def task3(M):

n,m=M()

\*c,=M()

t=[[]for i in range(n)]

v=[0]\*n

for i in range(n-1):

x,y=M()

t[x-1].append(y-1)

t[y-1].append(x-1)

a=i=0

q=[(0,0)]

while i<len(q):

x,N=q[i]

v[x]=1

if c[x]+N<=m:

L=1

for y in t[x]:

if not v[y]:

L=0

q.append((y,c[x]\*(c[x]+N)))

if L:

a+=1

i+=1

return(a)

task3(M)

**Результат работы:**

Вывод:2

**Задача 4**

**Условие:**

Дано целое неотрицательное число n, запись которого состоит из не более, чем 100 цифр, и не содержит ведущих незначащих нулей.

Нужно определить, можно ли в данном числе вычеркнуть некоторое (возможно нулевое) количество цифр так, чтобы полученное после вычеркивания цифр число содержало хотя бы одну цифру, было неотрицательным, не имело ведущих незначащих нулей и делилось на 8. После вычеркивания переставлять цифры запрещается.

Если решение существует, необходимо вывести его.

Входные данные.

В единственной строке входных данных содержится целое неотрицательное число n. Запись числа n не содержит ведущих незначащих нулей, и ее длина не превосходит 100 цифр.

Выходные данные.

Выведите "NO" (без кавычек), если искомого способа вычеркнуть некоторые цифры из числа n не существует.

Иначе выведите "YES" в первой строке и число, получившееся в результате вычеркивания некоторых цифр числа n, во второй строке. Выведенное число должно делиться на 8.

Если возможных ответов несколько, требуется вывести любой из них.

**Ход решения:**

Число делится на 8 тогда и только тогда, когда число, образованное его последними тремя цифрами, делится на 8. Достаточно проверять на делимость на 8 все однозначные, двухзначные и трехзначные числа, образуемые из исходного вычеркиванием цифр. Это можно сделать за *O*(*n*3) вложенными циклами по цифрам числа, где *n* — количество цифр числа.

**Листинг 4 – Решение задачи №4**

**import** re

n = "3454"

def task4(n):

for i in range(0, 1000, 8):

if re.search('.\*'.join(list(str(i))), n):

return('YES ' + str(i))

break

return('NO')

task4(n)

**Результат работы:**

Ввод:

3454

Вывод:

YES 344

**Задача 5**

**Условие:**

Маленькая девочка Сьюзи каждый день слушает сказки перед сном. Сегодняшняя сказка была про дровосеков, поэтому маленькая девочка сразу же начала представлять себе, как дровосеки рубят деревья, и в её мыслях родилась ситуация, описанная ниже.

Есть n деревьев, расположенных вдоль дороги в точках с координатами x1, x2, ... , xn. Каждое дерево имеет свою высоту hi. Дерево можно срубить и повалить влево или вправо. Тогда оно будет занимать отрезки [xi - hi, xi] и [xi;xi + hi] соответственно. Пока дерево не срублено, оно занимает точку с координатами xi. Дерево можно повалить, если на отрезке, который оно должно занимать после сваливания, нет ни одной занятой точки. Дровосеки хотят заготовить как можно большее число деревьев, поэтому Сьюзи стало интересно, какое наибольшее количество деревьев можно повалить.

Входные данные.

В первой строке находится целое число n (1 ≤ n ≤ 105) — количество деревьев.

В последующих n строках находятся пары целых чисел xi, hi (1 ≤ xi, hi ≤ 109) — координата и высота і-го дерева.

Пары заданы в порядке возрастания xi. Никакие два дерева не находятся в точке с одинаковой координатой.

Выходные данные.

Требуется вывести одно число — максимальное количество деревьев, которое можно срубить по указанным правилам.

**Ход решения:**

Повалим самое левое дерево влево. После этого, попробуем повалить следующее дерево. Если мы можем повалить его влево, сделаем это. Если не можем, тогда пробуем повалить вправо. Если это возможно, делаем это. Последний шаг справедлив, потому что сваливание некоторого дерева вправо может помешать только сваливанию следующего дерева. Так что мы можем "обменять" одно дерево на другое, не ухудшив ответа.

**Листинг 5 – Решение задачи №5**

n = int(input())

l=[list(map(int,input().split())) for i in range(n)]

def task5(n,l):

s=2

for i in range(1,n-1):

x,h=l[i]

if x-h > l[i-1][0]:

s+=1

elif x+h < l[i+1][0]:

s+=1

l[i][0]+=h

return(s if n>1 else 1)

task5(n,l)

**Результат работы:**

Ввод:

5

1 2

2 1

5 10

10 9

19 1

Вывод: 3

**Задача 6**

**Условие:**

У вас есть массив 𝑎 длины 𝑛. Для каждого положительного числа 𝑥 в течение 𝑥-й секунды вы собираетесь выполнить следующую операцию:

Выберите несколько различных индексов 𝑖1,𝑖2,…,𝑖𝑘, которые находятся в диапазоне от 1 до 𝑛 включительно, и добавьте 2𝑥−1 к каждой соответствующей позиции в 𝑎. Формально 𝑎𝑖𝑗:=𝑎𝑖𝑗+2𝑥−1 для 𝑗=1,2,…,𝑘. Обратите внимание, что вы можете также не выбрать ни одного индекса. Вы должны сделать 𝑎 неубывающим как можно быстрее. Найдите наименьшее число 𝑇 такое, что вы можете сделать массив неубывающим не позднее, чем через 𝑇 секунд.

Массив 𝑎 называется неубывающим, если и только если 𝑎1≤𝑎2≤…≤𝑎𝑛.

Вы должны ответить на 𝑡 независимых тестовых случаев.

Входные данные.

Первая строка содержит одно целое число 𝑡 (1≤𝑡≤104) — количество тестовых случаев.

Первая строка каждого тестового случая содержит одно целое число 𝑛 (1≤𝑛≤105) — длину массива 𝑎. Гарантируется, что сумма значений 𝑛 по всем тестовым случаям не превышает 105.

Вторая строка каждого теста содержит 𝑛 целых чисел 𝑎1,𝑎2,…,𝑎𝑛 (−109≤𝑎𝑖≤109).

Выходные данные.

Для каждого тестового примера выведите минимальное количество секунд, за которое вы можете сделать 𝑎 неубывающим.

**Листинг 6 – Решение задачи**

import math

def task6(n,a):

b=0

for i in range(n-1):

if a[i]>a[i+1]:

b=max(b,int(math.log2(a[i]-a[i+1]))+1)

a[i+1]=a[i]

return(b)

t=int(input())

out = []

for \_ in range(t):

n=int(input())

a=list(map(int,input().split()))

out.append(task6(n,a))

print(out)

**Результат работы:**

Ввод:

3

4

1 7 6 5

5

1 2 3 4 5

2

0 -4

Вывод

[2, 0, 3]

**Задача 7**

**Условие:**

Кефа хочет отметить свой первый крупный заработок походом в ресторан. Однако ему нужна компания.

У Кефы есть n друзей, каждый из которых согласится пойти в ресторан, если Кефа попросит. Каждый друг характеризуется количеством денег у него и степенью дружбы с Кефой. Наш попугай не хочет, чтобы какой-то друг почувствовал себя бедным по сравнению с кем-то другим в компании (Кефа не в счет). Друг чувствует себя бедным, если в компании есть кто-то, у кого денег хотя бы на d единиц больше, чем у него. Также Кефа хочет, чтобы суммарная степень дружбы членов компании была максимальной. Помогите ему пригласить оптимальную компанию!

Входные данные.

Первая строка ввода содержит два целых числа, разделенных пробелом, n и d (1 ≤ n ≤ 105, ) — количество друзей у Кефы и минимальная разница денег, приводящая к тому, что человек чувствует себя бедным.

В последующих n строках даны описания друзей Кефы, в (i + 1)-й строке содержится описание i-го друга вида mi, si (0 ≤ mi, si ≤ 109) — количество денег и степень дружбы с Кефой соответственно.

Выходные данные.

Выведите максимальную суммарную степень дружбы, которой можно добиться.

**Ход решения:**

Заметим, что если в массиве есть две пересекающиеся непрерывные неубывающие подпоследовательности, то их можно объединить в одну. Поэтому можно просто проходиться по массиву слева направо. Если текущую подпоследовательность можно продолжить с помощью *i*-го элемента, то делаем это, иначе начинаем новую. Ответом будет максимальная из всех найденных подпоследовательностей.

**Листинг 7 – Решение задачи №7**

**def** task7():

n,d=map(int,input().split())

l=sorted([[\*map(int,input().split())]for \_ in' '\*n])

def task7(n,d,l):

j=a=A=0

for i in range(n):

a+=l[i][1]

while l[i][0]-l[j][0]>=d:

a-=l[j][1]

j+=1

A=max(A,a)

return(A)

task7(n,d,l)

**Результат работы:**

Ввод:

5 100

0 7

11 32

99 10

46 8

87 54

Вывод:

111

**Задача 8**

**Условие:**

Вам задана массив 𝑎 длины 𝑛, состоящий из нулей. Вы выполняете 𝑛 действий с этим массивом: в течение 𝑖-го действия происходит следующая последовательность операций:

Выбирается максимальный по длине подмассив (последовательный подотрезок), состоящий только из нулей, среди всех таких отрезков выбирается самый левый; Пусть этот отрезок равен [𝑙;𝑟]. Если 𝑟−𝑙+1 нечетно (не делится на 2), то присваивается 𝑎[𝑙+𝑟2]:=𝑖 (где 𝑖 — номер текущего действия), иначе (если 𝑟−𝑙+1 четно) присваивается 𝑎[𝑙+𝑟−12]:=𝑖. Рассмотрим массив 𝑎 длины 5 (изачально 𝑎=[0,0,0,0,0]). Тогда он меняется следующим образом:

Сначала мы выбираем отрезок [1;5] и присваиваем 𝑎[3]:=1, таким образом 𝑎 становится равен [0,0,1,0,0]; затем мы выбираем отрезок [1;2] и присваиваем 𝑎[1]:=2, таким образом 𝑎 становится равен [2,0,1,0,0]; затем мы выбираем отрезок [4;5] и присваиваем 𝑎[4]:=3, таким образом 𝑎 становится равен [2,0,1,3,0]; затем мы выбираем отрезок [2;2] и присваиваем 𝑎[2]:=4, таким образом 𝑎 становится равен [2,4,1,3,0]; и наконец мы выбираем отрезок [5;5] и присваиваем 𝑎[5]:=5, таким образом 𝑎 становится равен [2,4,1,3,5]. Ваша задача — найти массив 𝑎 длины 𝑛 после выполнения всех 𝑛 действий. Заметьте, что ответ существует и единственен.

Вам необходимо ответить на 𝑡 независимых наборов тестовых данных.

Входные данные.

Первая строка входных данных содержит одно целое число 𝑡 (1≤𝑡≤104) — количество наборов тестовых данных. Затем следуют 𝑡 наборов тестовых данных.

Единственная строка набора тестовых данных содержит одно целое число 𝑛 (1≤𝑛≤2⋅105) — длину 𝑎.

Гарантируется, что сумма 𝑛 по всем наборам тестовых данных не превосходит 2⋅105 (∑𝑛≤2⋅105).

Выходные данные.

Для каждого набора тестовых данных выведите ответ — массив 𝑎 длины 𝑛 после выполнения 𝑛 действий, описанных в условии задачи. Заметьте, что ответ существует и единственен.

**Листинг 8 – Решение задачи 8**

def task8(l,r):

if l>r:return

m=(l+r)//2

a[m]=(l-r,m)

task8(l,m-1)

task8(m+1,r)

for \_ in range(int(input())):

n=int(input())+1

a=b=[0]\*n

task8(1,n-1)

for i,j in enumerate(sorted(a[1:n])):

b[j[1]]=i+1

print("Строка массива: ", \*b[1:n])

**Результат работы:**

Ввод:

6

1

2

3

4

5

6

Вывод:

1

12

213

3124

24135

341526

**Задача 9**

**Условие:**

Девочка очень любит задачи про запросы на массиве.

Однажды ей попалась довольно известная задача: дан массив из 𝑛 элементов (элементы массива проиндексированы от 1); также есть 𝑞 запросов, каждый из которых задается парой целых чисел 𝑙𝑖, 𝑟𝑖 (1≤𝑙𝑖≤𝑟𝑖≤𝑛). Для каждого запроса необходимо найти сумму всех элементов массива с индексами от 𝑙𝑖 до 𝑟𝑖 включительно.

Такая задача показалась Девочке довольно скучной. Она решила, что перед тем, как отвечать на запросы, она перемешает элементы массива, причем так, чтобы сумма ответов на все запросы была максимально возможной. Ваша задача — найти значение этой максимальной суммы.

Входные данные.

Первая строка содержит два целых числа 𝑛 (1≤𝑛≤2⋅105) и 𝑞 (1≤𝑞≤2⋅105), разделенных пробелом — количество элементов в массиве и количество запросов, соответственно.

Следующая строка содержит 𝑛 целых чисел 𝑎𝑖 (1≤𝑎𝑖≤2⋅105), разделенных пробелами — элементы массива.

Каждая из следующих 𝑞 строк содержит два целых числа 𝑙𝑖 и 𝑟𝑖 (1≤𝑙𝑖≤𝑟𝑖≤𝑛), разделенных пробелом — 𝑖-й запрос.

Выходные данные.

В единственной строке выведите целое число — максимальную сумму ответов на запросы после перемешивания элементов массива.

**Ход решения:**

Для каждой ячейки массива посчитаем количество запросов, которые ее покрывают. Утверждается, что ячейке, которую покрывает большее количество запросов, нужно ставить в соответствие большее число. Более формально: пусть *b* — массив, в *i*-ой ячейке которого записано количество запросов, покрывающих эту ячейку, *а* – входной массив. Отсортируем эти массивы. Утверждается, что ответ в таком случае равен https://espresso.codeforces.com/3a28d112ee9ff8fe01ca906f5f4d9537dc489337.png

Докажем это утверждение. Рассмотрим какие-то индексы *i* < *j*, а так же соответсвующие им элементы *a*[*i*], *a*[*j*] , *b*[*i*], *b*[*j*] (*a*[*i*] ≤ *a*[*j*], *b*[*i*] ≤ *b*[*j*]). Эти элементы вносят в ответ следующую величину: *a*[*i*]·*b*[*i*] + *a*[*j*]·*b*[*j*]. Поменяем элементы *a*[*i*] и a$[j]$ местами. Теперь эти элементы вносят в ответ величину *a*[*i*]·*b*[*j*] + *a*[*j*]·*b*[*i*]. Рассмотрим разницу между полученными значениями:

*a*[*i*]·*b*[*j*] + *a*[*j*]·*b*[*i*] - *a*[*i*]·*b*[*i*] - *a*[*j*]·*b*[*j*] = *b*[*j*]·(*a*[*i*] - *a*[*j*]) + *b*[*i*]·(*a*[*j*] - *a*[*i*]) = (*b*[*j*] - *b*[*i*])·(*a*[*i*] - *a*[*j*]) ≤ 0.

Таким образом, перестановка двух любых элементов приводит к неувеличению суммарного результата, а значит — начальный порядок является оптимальным.

Теперь нам нужно быстро научиться считать массив *b*.

Для этого можно использовать разные структуры данных, поддерживающие модификацию на отрезке (дерево отрезков, декартово дерево и т.д). Однако, существует гораздо более простой метод.

Создадим некоторый массив *d*. При поступлении запроса *li*, *ri* увеличим элементы массива *d*[*li*] на 1, и уменьшим значение элемента *d*[*ri* + 1] на 1. Таким хитрым образом мы добавляем 1 всем ячейкам от *li* до *ri* включительно. После этого необходимо пробежаться по массиву *d*, накапливая соответствующий результат для *b*[*i*].

Теперь, зная массив *b*, можно легко узнать ответ.

**Листинг 9 – Решение задачи 9**

intin=lambda:map(int,input().split())

Ain=lambda:list(map(int,input().split()))

def task9(intin,Ain):

n,q=intin()

a=Ain()

p=[0]\*n

for i in range(q):

x,y=intin()

if y-1>=0: p[y-1]+=1

if x-2>=0: p[x-2]-=1

i=n-2

while i>=0:

p[i]+=p[i+1]

i-=1

p.sort()

a.sort()

ans=0

i=0

while i<n:

ans+=a[i]\*p[i]

i+=1

return(ans)

task9(intin,Ain)

**Результат работы:**

Ввод:

5 3

5 2 4 1 3

1 5

2 3

2 3

Вывод:

33

**Задача 10**

**Условие:**

Дано бесконечное множество, сгенерированное следующим образом:

1.1 принадлежит множеству.

2.Если 𝑥 принадлежит множеству, то 𝑥⋅𝑎 и 𝑥+𝑏 также принадлежат множеству.

Например, если 𝑎=3 и 𝑏=6, то первые пять элементов множества равны:

1, 3 (1 лежит в множестве, поэтому 1⋅𝑎=3 лежит в множестве), 7 (1 лежит в множестве, поэтому 1+𝑏=7 лежит в множестве), 9 (3 лежит в множестве, поэтому 3⋅𝑎=9 лежит в множестве), 13 (7 лежит в множестве, поэтому 7+𝑏=13 лежит в множестве). Даны положительные целые числа 𝑎, 𝑏, 𝑛. Выясните, принадлежит ли 𝑛 множеству.

Входные данные.

Каждый тест содержит несколько наборов входных данных. Первая строка содержит одно целое число 𝑡 (1≤𝑡≤105) – количество наборов входных данных. Далее следует описание наборов входных данных.

Строка, описывающая набор входных данных содержит три целых числа 𝑛, 𝑎, 𝑏 (1≤𝑛,𝑎,𝑏≤109), разделенных пробелом.

Выходные данные.

Для каждого набора входных данных выведите «Yes», если 𝑛 принадлежит множеству, и «No» иначе. Вы можете выводить каждую букву в любом регистре (строчную или заглавную).

Входные данные.

5

24 3 5

10 3 6

2345

1 4

19260817 394 485

19260817 233 264

Выходные данные.

Yes

No

Yes

No

Yes

В первом наборе 24 генерируется следующим образом:

1 лежит в множестве, поэтому 3 и 6 лежат в множестве; 3 лежит в множестве, поэтому 9 и 8 лежат в множестве; 8 лежит в множестве, поэтому 24 и 13 лежат в множестве. Таким образом, 24 принадлежит множеству.

Первые пять элементов множества из второго набора входных данных описаны в условии. Можно заметить, что 10 не в их числе.

**Листинг 10 – Решение задачи 10**

def task9(n,a,b):

for k in range(31):

if n < a \*\* k:

return("NO")

break

if n % b == (a \*\* k) % b:

return("YES")

break

for \_ in range(int(input())):

n, a, b = map(int, input().split())

print(task9(n,a,b))

**Результат работы:**

Ввод:

5

24 3 5

10 3 6

2345 1 4

19260817 394 485

19260817 233 264

Вывод:

YES

NO

YES

NO

YES

**Вывод**

В ходе выполнения курсовой работой были освоены навыки решения алгоритмических задач.