**Отчёт по практической работе**

Беспечалов Артём 2023-ФГиИБ-ПИ-1б

1. На вход подаётся натуральное число. Вывести все квадраты чисел, которые меньше этого числа.

Входные данные: 20

Выходные данные: 1 4 9 16

Код:

n = int(int(input()) \*\* 0.5) + 1

for i in range(1, n):

print(i \*\* 2)

Кол-во операций: int(n \*\* 0.5) + 1  
Сложность алгоритма: O(n)

1. На вход подаётся последовательность чисел, оканчивающаяся нулём. Найти сумму всех чисел этой последовательности.

Входные данные: 1 2 3 4 5 6 7 8 9 0

Выходные данные: 45

Код:

a = int(input())

sum1 = 0

while a != 0:

sum1 += a

a = int(input())

print(sum1)

print(f'Количество операций: {c}')

Кол-во операций: (n — 1) \* 3 + 2  
Сложность алгоритма: O(n)

1. На вход подаётся последовательность чисел, оканчивающаяся нулём. Найти второе максимальное число последовательности.

Входные данные: 1 2 3 4 5 6 7 8 9 0

Выходные данные: 8

Код:

max1 = 1

max2 = 0

n = int(input())

while n != 0:

if n >= max1:

max2, max1 = max1, n

elif n >= max2:

max2 = n

n = int(input())

print(max2)

Кол-во операций: (n — 1) \* 4 + 4

Сложность алгоритма: O(n)

1. На вход подаётся последовательность чисел, заканчивающаяся нулём. Найти длину максимальной подпоследовательности, состоящей из подряд идущих одинаковых чисел. Если такой подпоследовательности нет, вывести 1.

Входные данные: 2 2 2 3 5 3 5 5 5 5 0

Выходные данные: 4

Код:

max1, max2, last, n = 0, 1, int(input()), int(input())

while n != 0:

if last == n:

max2 += 1

elif max2 > max1:

max1, max2 = max2, 1

else:

max2 = 1

last, n = n, int(input())

if max2 > max1:

max1 = max2

print(max1)

Кол-во операций: 54  
Сложность алгоритма: O(n)

1. На вход подаётся последовательность чисел, оканчивающаяся нулём. Найти наименьшее расстояние между локальными максимумами. Первое и последнее числа локальными максимумами не являются. Если локальных максимумов нет, вывести 0.

Входные данные: 2 3 4 3 4 5 6 5 6 5 0

Выходные данные: 2

Код:

n = int(input())

arr = []

ind = []

while n != 0:

arr.append(n)

n = int(input())

for i in range(1, len(arr) - 1):

if arr[i] > arr[i - 1] and arr[i] > arr[i + 1]:

ind.append(i)

arr.clear()

for i in range(len(ind) - 1):

arr.append(abs(ind[i] - ind[i + 1]))

if arr:

print(min(arr))

else:

print(0)

Кол-во операций: 72  
Сложность алгоритма: O(n)

1. Дан массив. Переставить местами соседние элементы. Если кол-во элементов нечётное, то последний элемент остаётся на своем месте.

Входные данные: 5

1 2 3 4 5

Выходные данные: 2 1 4 3 5

Код:

n = int(input('Введите длину: '))

arr = [int(input()) for i in range(n)]

for i in range(0, n - 1, 2):

arr[i], arr[i + 1] = arr[i + 1], arr[i]

print(arr)

Кол-во операций: 3 + (n // 2) \* 5 + (n - 2) \* ((n - 1) % 2) + (n % 2) \* (n - 1)  
Сложность алгоритма: O(n)

1. Дан список с упорядоченными по неубыванию элементами. Найти количество уникальных значений.

Входные данные: 7

1 2 2 2 3 3 5

Выходные данные: 4

Код:

n = int(input('Введите длину: '))

arr = [int(input())]

for i in range(n - 1):

a = int(input())

if a < arr[i]:

print('Вы ввели число меньше предыдущего')

while True:

b = int(input(f'Введите число больше или равно {arr[i]}: '))

if b >= arr[i]:

arr.append(b)

break

else:

arr.append(a)

kol = 1

for i in range(1, len(arr)):

if arr[i] > arr[i - 1]:

kol += 1

print(kol)

Кол-во операций: 42  
Сложность алгоритма: O(n)

1. Дан размер массива, дан массив. Вывести элементы, которые встречаются один раз.

Входные данные: 6

1 1 2 2 3 4

Выходные: 3 4

Код:

n = int(input('Введите длину: '))

arr = [int(input()) for i in range(n)]

for i in range(n):

if arr[i] not in arr[i + 1:] and arr[i] not in arr[:i]:

print(arr[i])

print(arr)

Кол-во операций: 6 + (n - 1) \* 5  
Сложность алгоритма: O(n\*\*2)