

Федеральное государственное автономное образовательное  
учреждение высшего образования  
Университет ИТМО

Факультет инфокоммуникационных технологий

## **Алгоритмы и структуры данных:**

**Отчёт по лабораторной работе №2: Сортировка  
слиянием, метод декомпозиции**

Выполнил:  
**Бочкарь Артём Артёмович**

Группа: **K32392**

Преподаватели:  
**Артамонова В. Е.**

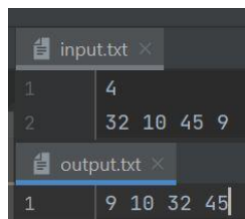
Санкт-Петербург 2023 г.

## Задача №1: Сортировка слиянием

В первом задании от нас требовалось написать сортировку слиянием. В первой строке входного файла содержится число  $n$  ( $1 \leq n \leq 103$ ) – число элементов в массиве. Во второй строке находится  $n$  различных целых чисел:

```
f = open('input.txt')
n = int(f.readline())
mass = list(map(int, f.readline().split()))
def Merge_sort(mass):
    if len(mass) <= 1:
        return mass
    fin_sort = list()
    i, j = 0, 0
    mid = len(mass) // 2
    l_mass, r_mass = mass[:mid], mass[mid:]
    Merge_sort(l_mass)
    Merge_sort(r_mass)
    while j < len(r_mass) and i < len(l_mass):
        if l_mass[i] > r_mass[j]:
            fin_sort.append(r_mass[j])
            j += 1
        else:
            fin_sort.append(l_mass[i])
            i += 1
    while j < len(r_mass):
        fin_sort.append(r_mass[j])
        j += 1
    while i < len(l_mass):
        fin_sort.append(l_mass[i])
        i += 1
    for i in range(len(mass)):
        mass[i] = fin_sort[i]
    return mass
mass = Merge_sort(mass)
mass = ' '.join(map(str, mass))
f = open('output.txt', 'w')
f.write(mass)
```

Тесты:



input.txt	
1	4
2	32 10 45 9
output.txt	
1	9 10 32 45

## Задача №3: Число инверсий

В третьем задании нужно было подсчитать количество инверсий. В первой строке входного файла содержится число  $n$  ( $1 \leq n \leq 105$ ) – число элементов в массиве. Во второй строке находятся  $n$  различных целых чисел:

```
f = open('input.txt')
n = int(f.readline())
mass = list(map(int, f.readline().split())) count = 0
def Merge_sort(mass):
    if len(mass) <= 1:
        return mass
    global count
    fin_sort = list()
    i, j = 0, 0
    mid = len(mass) // 2
    l_mass, r_mass = mass[:mid], mass[mid:]
    Merge_sort(l_mass)
    Merge_sort(r_mass)
    while j < len(r_mass) and i < len(l_mass):
        if l_mass[i] > r_mass[j]:
            count += (len(l_mass) - i)
            fin_sort.append(r_mass[j])
            j += 1
        else:
            fin_sort.append(l_mass[i])
            i += 1
    while j < len(r_mass):
        fin_sort.append(r_mass[j])
        j += 1
    while i < len(l_mass):
        fin_sort.append(l_mass[i])
        i += 1
    for i in range(len(mass)):
        mass[i] = fin_sort[i]
    return mass
Merge_sort(mass)
f = open('output.txt', 'w')
f.write(str(count))
```

Тесты:

input.txt	
1	4
2	32 10 45 9
output.txt	
1	4

## Задача №4: Бинарный поиск

В четвертом задании нужно было написать алгоритм бинарного поиска. В первой строке входного файла содержится число  $n$  ( $1 \leq n \leq 10^5$ ) – число элементов в массиве, и последовательность  $a_0 < a_1 < \dots < a_{n-1}$  из  $n$  различных положительных целых чисел в порядке возрастания,  $1 \leq a_i \leq 10^9$  для всех  $0 \leq i \leq n$ . Следующая строка содержит число  $k$ ,  $1 \leq k \leq 10^5$  и  $k$  положительных целых чисел  $b_0, \dots, b_{k-1}$ ,  $1 \leq b_j \leq 10^9$  для всех  $0 \leq j < k$ :

```
n = int(f.readline())
mass = list(map(int, f.readline().split())) m = int(f.readline())
mass2 = list(map(int, f.readline().split()))
for fin in mass2:
    h=0
    g=m-1
    fin = -1
    while h <= g:
        mid = int((h + g) / 2)
        if (mass[mid] > fin):
            g = mid - 1
        else:
            h = mid + 1
            fin = mid
    if (mass[fin] != fin):
        f.write('-1 ')
    else:
        f.write(str(fin) + ' ')

f = open('output.txt', 'w')
```

## Задача №5: Представитель большинства

В пятом задании нужно было написать алгоритм, который бы искал элемент, который повторяется в массиве наибольшее число раз. В первой строке входного файла содержится число  $n$  ( $1 \leq n \leq 10^5$ ) – число элементов в массиве. Во второй строке находятся  $n$  положительных целых чисел:

```
f = open('input.txt')
n = int(f.readline())
mass = list(map(int, f.readline().split()))
def match(element, mass):
    if len(mass) > 1:
        mid = len(mass) // 2
        r_mass = mass[mid:]
        l_mass = mass[:mid]
        return match(element, l_mass) + match(element, r_mass)
    else:
        return int(element == mass[0])

def run(mass):
    for element in mass:
        if match(element, mass) > len(mass) / 2:
            return 1
    return 0

f = open('output.txt', 'w')
f.write(str(run(mass)))
```

Тесты:

input.txt ×	
2	1 2 2 2 13
3	
output.txt ×	
1	1

## Задача №7: Поиск максимального подмассива за минимальное время

В этой задаче нужно найти максимальный подмассив за минимально возможное время:

```
f = open('input.txt')
n = int(f.readline())
mass = list(int(c) for c in f.readline().split())
def find_max_subarray(mass):
    length = len(mass)
    cur_sum = mass[0]
    max_sum = cur_sum
    for i in range(1, len(mass)):
        if cur_sum + mass[i] > mass[i]:
            cur_sum = cur_sum + mass[i]

        else:
            cur_sum = mass[i]

        if cur_sum > max_sum:
            max_sum = cur_sum

    return max_sum

max_sum = find_max_subarray(mass)
f = open('output.txt', 'w')
f.write(str(max_sum))
```

Тесты:

input.txt ×	
1	5
2	1 -2 29 20 13
output.txt ×	
1	62