Т2Ч4

Фролов АА, 1-к, ИВТ-2

4.3

Задание 1.

```
// Входные данные: массив A[1,2,3,4,...,n-1]
// Выходные данные: медиана массива A

Сортировка массива A
med = A[(n-1)/2]
```

Сортировка: $O(n \log n)$
Доступ к элементу: $O(1)$

Итог: $O(n \log n)$

Задание 2.

```
// Входные данные: массив A[1,2,3,4,...,n-1]
// Выходные данные: |x-y|

func()
    Сортировка массива A
    if A[i] == A[n-1]
        return 0
    i = 0 // iter
    while i < n-1
        if A[i] != A[i+1]
            return |A[i]-A[i+1]|
        i++
    return 0
```

Сортировка: $O(n \log n)$
Поиск разницы: $O(n)$
Итог: $O(nlogn)$

Сложность метода грубой силы: $O(n^2)$

Вывод: метод преобразования - лучше

Задание 3.

Грубая сила

```
// Входные данные: массив(мн-во) A[1,2,3,4,...,n-1] массив(мн-во) B[1,2,3,4,...,m-1]
// Выходные данные: массив(мн-во) C[] со схожими элементами

for i in A:
    for j in B:
        if A[i] == B[j]:
            C.добавить(A[i])
```

Эффективность: $O(n * m)$

Преобразование

```
// Входные данные: массив(мн-во) A[1,2,3,4,...,n-1] массив(мн-во) B[1,2,3,4,...,m-1]
// Выходные данные: массив(мн-во) C[] со схожими элементами

Сортировка A
Сортировка B

i = 0 //iter A
j = 0 //iter B

while i < n and j < m
    if A[i] == B[j]
        C.добавить(A[i])
    if A[i] > B[j]
        j++
    else
        i++
```

Сортировки: $O(n \log n)$ $и$ $O(m \log m)$
Поиск пересечений: O(n+m)
Итог: $O(n \log n + m \log m + n + m) = O(n \log n + m \log m)$

Задание 4.

Алгоритм:

```
// Входные данные: массив A[1,2,3,4,...,n-1]
// Выходные данные: min и max
Сортировка A
min = A[0]
max = A[n-1]
```

Сортировка: $O(n \log n)$
Доступ к элементам: $O(1)$

Итог: $O(n \log n)$

Метод	Эффекивность
Грубая сила	O(n)
Декомпозиция	O(n)
Преобразование	$O(n \log n)$

Задание 5.

```
// Множество dots(dot1, dot2, ... dotn) n>=3
// dot(x,y)

Функция build_polygon(список точек):
    Если количество точек меньше 3:
        Выдать ошибку "Нужно минимум 3 точки"

    Выбрать первую точку p0 из списка точек

    Определить функцию сортировки по углу относительно p0:
        Для точки p:
            dx = координата x точки p − координата x точки p0
            dy = координата y точки p − координата y точки p0
            Угол = вычислить арктангенс от (dy, dx)
            Вернуть угол

    Отсортировать все точки кроме p0 по функции сортировки угла

    Вернуть список из:
        p0 + отсортированные точки + p0 (чтобы замкнуть полигон)
```

Всегда ли имеет решение: да, если $n \geq 3$ и точки не лежат на одной прямой.
Всегда ли решение единственно: Нет.

Задание 6.

```
// Массив A[] Число S
// Ответ да или нет

создать пустое мн-во остатки
for num in A
    остаток = S-num
    Если остаток есть в остатки
```

```
        Вывести("да")
        Остановить
    Добавить num в остатки
Вывести("Нет")
```

Эффективность: $O(n)$

Задание 7.

```
Функция НайтиАнаграммы(список_слов):
    Создать словарь анаграммы: ключ – строка, значение – список строк

    Для каждого слова в список_слов:
        Перевести слово в нижний регистр
        Отсортировать буквы слова в алфавитном порядке → ключ
        Добавить исходное слово в список по этому ключу в словаре

    Создать результирующий словарь
    Для каждой пары (ключ, список_слов) в словаре анаграммы:
        Если длина список_слов > 1:
            Добавить пару в результирующий словарь

    Вернуть результирующий словарь
```

4.8

Задание 1.

1. Найти значения логарифмов от A и B в таблице
2. Сложить эти значения
3. Найти антилогарифм от суммы

Задание 2.

1. Проверить все комбинации из 3 точек, которые могут быть вершинами треугольника.
2. Для текущего треугольника ABCABC проверить, что все остальные точки лежат внутри него.
3. Учёт граничных случаев
   Коллинеарные точки: Если A,B,C лежат на одной прямой, такой треугольник не рассматривается.
   Точки на границе: Если $S_{ABP} + S_{BCP} + S_{CAP} = S_{ABC}$, точка P может лежать на стороне треугольника. В зависимости от условия, можно считать её лежащей "внутри".
4. Возврат результат
   Если хотя бы для одного треугольника все точки лежат внутри,

возвращаем **"Да"**
Если ни для одного треугольника условие не выполнилось, возвращаем **"Нет"**.

Задание 3.

1. **Рёберный граф (Line Graph):**
   - Строится новый граф L(G), где:
     - **Вершины** соответствуют **рёбрам** исходного графа G.
     - **Рёбра** в L(G) соединяют вершины, если соответствующие им рёбра в G **имеют общую вершину** (т.е. смежны).
2. **Пример:**
   - Пусть исходный граф G имеет рёбра:
     - e1=(A,B),
     - e2=(B,C)
   - В рёберном графе L(G) вершины e1 и e2 соединены, потому что рёбра e1 и e2 в G имеют общую вершину B.
3. **Раскраска рёбер → Раскраска вершин:**
   - Раскрасить рёбра G так, чтобы смежные рёбра были разного цвета — это **то же самое**, что раскрасить вершины L(G) так, чтобы смежные вершины были разного цвета.
   - Минимальное число цветов для рёберной раскраски G равно хроматическому числу L(G).