МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Отчет по выполнению практического задания №1.2

Тема: ТЕОРЕТИЧЕСКИЙ МЕТОД ОЦЕНКИ СЛОЖНОСТИ ИТЕРАТИВНОГО АЛГОРИТМА. КОРРЕКТНОСТЬ ИТЕРАТИВНОГО АЛГОРИТМА

Дисциплина: Структуры и алгоритмы обработки данных

Выполнил студент student
группа group

Москва 2025

Цель работы: актуализация знаний и приобретение практических умений и навыков по определению вычислительной сложности алгоритмов (эмпирический подход).

Задание 1

1.1

алгоритм delFirstMetod:

a) Формулы функции роста

Худший случай:

Внешний цикл выполняется n раз.

Внутренний цикл выполняется 1 … n – 1 раз

$$\boldsymbol{T(n) = \frac{(n+1)n}{2} = O(n^2)}$$

Лучший случай:

Внешний цикл выполняется n раз.

Внутренний цикл не выполняется.

$$\boldsymbol{T(n) = n = O(n)}$$

b) Количество критических операций

n	Худший случай	Лучший случай
10	55	10
100	5050	100
1000	500500	1000
10000	50005000	10000

c) Порядок роста:

Худший случай: $O(n^2)$

Лучший случай: O(n)

Алгоритм delOtherMetod:

a) Формулы функции роста

Худший случай:

Цикл for выполняется n раз.

$\boldsymbol{T(n) = n = O(n)}$

Лучший случай:

Цикл for выполняется n раз.

$\boldsymbol{T(n) = 2n = O(n)}$

b) Количество критических операций

n	Худший случай	Лучший случай
10	10	20
100	100	200
1000	1000	2000
10000	10000	20000

c) Порядок роста:

Худший случай: O(n)

Лучший случай: O(n)

1.2 Сопоставление теоретических и эмпирических результатов

Сводная таблица результатов (1 алгоритм)

	Все элементы ключевые		**Все элементы неключевые**	
n	**$T_т$**	**$T_п$**	**$T_т$**	**$T_п$**
10	55	55	10	10
100	5050	5050	100	100
1000	500500	500500	1000	1000
10000	50005000	50005000	10000	10000

Сводная таблица результатов (2 алгоритм)

	Все элементы ключевые		**Все элементы неключевые**	
n	**$T_т$**	**$T_п$**	**$T_т$**	**$T_п$**
10	10	10	20	20
100	100	100	200	200
1000	1000	1000	1000	2000
10000	10000	10000	10000	20000

2 Сравнение результатов

Результаты полученные теоретически совпадают с теми, что были получены эмпирически.

3 Ёмкостная сложность

Оба алгоритма имеют ёмкостную сложность O(1) (используют константную доп. память)

4 Вывод об эффективности алгоритмов

Вывод: алгоритм delOtherMetod гораздо эффективнее при сложности O(n) в худшем случае, в то время, как алгоритм delFirstMetod имеет сложность $O(n^2)$.

Задание 2

1 Псевдокод сортировки пузырьком и формулы функции роста

Сортировка пузырьком (псевдокод)

```
bubbleSort(arr, n) {
   for i ← 1 to n-1 do
      for j ← 0 to n-i-1 do
         if arr[j] > arr[j+1] then
            swap(arr[j], arr[j+1])
         endif
      od
   od
}
```

Худший случай:

Внешний цикл выполняется n – 1 раз

Внутренний цикл выполняется n – i раз

$$\boldsymbol{T(n) = \frac{(n-1)3n}{2} + \frac{(n-1)n}{2}}$$

Лучший случай:
Внешний цикл выполняется n – 1 раз
Внутренний цикл выполняется n – i раз

$$\boldsymbol{T(n) = \frac{(n-1)n}{2}}$$

2 Сопоставление теоретических и эмпирических результатов

	Уже упорядочен		**Обратная упорядоченность**	
n	**$T_т$**	**$T_п$**	**$T_т$**	**$T_п$**
10	45	45	180	180
100	4950	4950	19800	19800
1000	499500	499500	1998000	1998000
10000	49995000	49995000	199980000	199980000

3 Сравнение результатов
Результаты полученные теоретически совпадают с теми, что были получены эмпирически.
4 Ёмкостная сложность
Алгоритм имеет ёмкостную сложность O(1) (используют константную доп. память).
5 Вывод об эмпирической вычислительной сложности алгоритмов
Вывод: сортировка пузырьком имеет сложность $O(n^2)$ в лучшем и худшем случаях.

Задание 3

Инвариант
1) Формулировка инварианта цикла:
   После k итераций цикла первые j – 1 j−1 элементов массива x содержат все элементы, не равные key, из первых k элементов исходного массива.
2) Доказательство истинности инварианта:
   - **Инициализация:** Перед первой итерацией j = 1, k = 0. Инвариант выполняется, т.к нет элементов.
   - **Сохранение**: На каждой итерации:
     - Если x[i] ≠ key, элемент копируется в x[j] и j увеличивается.
     - Если x[i]= key элемент пропускается.
     - Инвариант сохраняется.
   - **Завершение:** После n итераций первые j−1 элементов содержат все элементы, не равные key.

2) Доказательство конечности:
Цикл выполняется ровно n раз, так как i увеличивается от 1 до n.
Область неопределенности: n − i уменьшается на каждой итерации, гарантируя завершение цикла.

ЛИТЕРАТУРА:

1. Бхаргава А. Грокаем алгоритмы, 2-е изд. – СПб: Питер, 2024. – 352 с.

2. Вирт Н. Алгоритмы + структуры данных = программы. – М.: Мир, 1985. – 406 с.

3. Кнут Д.Э. Искусство программирования, том 3. Сортировка и поиск, 2-е изд. – М.: ООО «И.Д. Вильямс», 2018. – 832 с.

4. Седжвик Р. Фундаментальные алгоритмы на C++. Анализ/Структуры данных/Сортировка/Поиск. – К.: Издательство «Диасофт», 2001. – 688 с.

5. Алгоритмы – всё об алгоритмах / Хабр [Электронный ресурс]. URL: https://habr.com/ru/hub/algorithms/ (дата обращения 05.02.2025).