САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

ФАКУЛЬТЕТ ИНФОКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ

Отчет по лабораторной работе № 2

по курсу «Алгоритмы и структуры данных»

Тема: Сортировка слиянием. Метод декомпозиции

Вариант 2

Выполнила:

Азизова Наида Элимизовна

К3140

Проверил(а):

Афанасьев А. В.

Санкт-Петербург

2024 г.

Содержание отчёта

[Задания по варианту 3](#_Toc180252111)

[Задача №1. Сортировка слиянием 3](#_Toc180252112)

[Задача №2. Сортировка слиянием + 6](#_Toc180252113)

[Задача №4. Бинарный поиск 9](#_Toc180252114)

[Вывод 11](#_Toc180252115)

# Задания по варианту

## Задача №1. Сортировка слиянием

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание

Решение:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Текстовое объяснение решения:

1. Импортируем модули time и tracemalloc;
2. Функция merge\_list выполняет слияние двух отсортированных списков. Принимает два массива: a и b. Их нужно слить в один список с сохранением сортировки, для этого мы вводим третий список с, который сейчас пустой и определяем длины наших списков a и b (N, M). Затем создаем цикл, пока не дойдем до конца либо 1-го 2-го списка и проверяем если i-ый элемент 1-го списка меньше либо рано j-ый элемент 2-го списка, то в список с добавляем i-ый элемент из 1-го списка и счетчик у 1-го списка увеличиваем на единицу, иначе добавляем в список с j-ый элемент из 2-го списка и счетчик у 2-го списка увеличиваем на единицу. После этого оставшиеся элементы, которые мы не добавили в нашем цикле, добавляем их в переменную с и функция возвращает готовый список;
3. Функция split\_and\_merge – это функция деления списка и слияния списков в общий отсортированный список. Производим деление массива на два (примерно равной длины). И сравниваем: если длина 1-го списка больше 1, то делим дальше, и если длина 2-го списка больше 1, то делим дальше и возвращаем слияние двух отсортированных списков в один (merge\_list(mas1, mas2));
4. Открываем файл input.txt, в котором содержатся входные данные. Считываем строку и приводим число n к целочисленному виду. Считываем следующую строку и преобразовываем ее в список целых чисел (mas). И закрываем файл;
5. Открываем файл output.txt и записываем в него полученный результат (отсортированный список). И закрываем файл;
6. В конце выводим время (из начального времени вычитаем конечное) и затраченную память.

Результат работы кода на примерах из текста задачи:

Ввод:

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Вывод:

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Время выполнения и затраченная память:

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Вывод по задаче: Я научилась осуществлять сортировку слиянием и узнала время ее выполнения и затраты памяти.

## Задача №2. Сортировка слиянием +

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание

Решение:

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Текстовое объяснение решения:

1. Импортируем модули time и tracemalloc;
2. Функция merge\_list принимает несколько параметров: a, b – два отсортированных списка, а left, mid, right – индексы, которые указывают на границы слияния и output\_file. Создаем пустой список с для хранения результата слияния, определяем длины наших списков a и b (N, M), i,j – индексы для прохода по нашим спискам. Далее цикл продолжается, пока есть элементы в обоих списках и сравниваем текущие элементы из a и b, и меньший добавляем в с. Затем если один из списков исчерпан, то оставшиеся элементы другого списка добавляются в с; If, Il – индексы начала и конца слияния (с учетом 1-индексации), Vf, Vl – значение первого и последнего элементов, результирующего массива с; записываем промежуточные результаты в output\_file.
3. Функция split\_and\_merge рекурсивно делит массив на подмассивы и сортирует их. Если массив состоит из 1-го элемента, он уже отсортирован, и возвращается этот элемент. Затем массив делится на две половины: mas1 и mas2, которые сортируются рекурсивно. Отсортированные подмассивы mas1 и mas2 сливаются с помощью функции merge\_list.
4. Открываем файл input.txt, в котором содержатся входные данные. Считываем строку и приводим число n к целочисленному виду. Считываем следующую строку и преобразовываем ее в список целых чисел (mas). И закрываем файл;
5. Открываем файл output.txt и записываем в него полученный результат. И закрываем файл;
6. В конце выводим время (из начального времени вычитаем конечное) и затраченную память.

Результат работы кода на примерах задачи:

Ввод:

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Вывод:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Время выполнения и затраченная память:

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Вывод по задаче: Я научилась осуществлять сортировку слиянием + с выводом информации о каждом слиянии, включая индексы и значения граничных элементов и узнала время ее выполнения и затраты памяти.

## Задача №4. Бинарный поиск

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Решение:

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

Текстовое объяснение решения:

1. Импортируем модули time и tracemalloc;
2. Функция binary\_search принимает на вход элемент для поиска, индексы начала и конца списка и сам список. В цикле вычисляем серединный индекс и проверяем элемент на данной позиции. Если равен нужному – возвращает индекс, меньше – делаем верхней границей значение mid, больше – делаем нижней границей значение mid + 1. Если такого элемента нет, то выводим -1;
3. В Функции main открываем файл input.txt и считываем из него длину списка, сам список, длину списка элементов для поиска и список элементов для поиска. В переменную answ записываем результат работы функции binary\_search для каждого элемента из списка spis\_e.
4. Открываем файл output.txt и записываем в него полученный результат. И закрываем файл;
5. В конце выводим время (из начального времени вычитаем конечное) и затраченную память.

Результат работы кода на примерах задачи:

Ввод:

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Вывод:

Изображение выглядит как текст, Шрифт, снимок экрана, число

Автоматически созданное описание

Время выполнения и затраченная память:

Изображение выглядит как текст, Шрифт, снимок экрана, линия

Автоматически созданное описание

Вывод по задаче: Я научилась осуществлять бинарный поиск и узнала время его выполнения и затраты памяти.

# Вывод

В этой лабораторной работе я научилась осуществлять сортировку слиянием, сортировку слиянием + и бинарный поиск. Также я проверила время работы алгоритмов и память, которая тратится на их выполнение.