Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Российский химико-технологический университет имени Д.И. Менделеева»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

Выполнил студент группы КС-36: Золотухин А.А.
Ссылка на репозиторий: https://github.com/MUCTR-IKT-CPP/ZolotukhinAA_36_ALG
Принял: Крашенников Роман Сергеевич
Дата сдачи: 03.03.2025

Москва
2025

Оглавление

Описание задачи

В лабораторной работе предлагается изучить альтернативные первой лабораторной работы сортировки, которые обладают меньшей асимптотической сложностью и сравнить их с результатами предыдущей лабораторной работы.

Используя предыдущий код посерийного выполнения алгоритма сортировки и измерения времени требуется реализовать метод пирамидальной сортировки.

Задание:

- Реализовать проведения тестирования алгоритма сериями расчётов для измерения параметров времени.

  За один расчёт выполняются следующие операции:

  1. Генерируется массив случайных значений;
  2. Запоминается время начала расчёта алгоритма сортировки;
  3. Выполняется алгоритм сортировки
  4. Вычисляется время, затраченное на сортировку: текущее время - время начала;
  5. Сохраняется время для одной попытки.

     После этого расчёт повторяется до окончания серии.

     - Алгоритм вычисляется 8 сериями по 20 раз за серию;
     - Алгоритм в каждой серии вычисляется для массива размером M (1000,2000, 4000, 8000, 16000, 32000, 64000, 128000);
     - Массив заполняется значениями чисел с плавающей точкой в интервале от -1 до 1;
     - Для серии запоминаются все времена, которые были замерены.

- По полученным данным времени построить графики зависимости времени от числа элементов в массиве:

  1. Совмещенный график наихудшего времени выполнения сортировки и сложности алгоритма, указанной в нотации O большое;

     Для построения графика вычисляется O большое для каждого размера массива. При этом при вычислении функции O(c * g(N)) подбирается такая константа c, чтобы при значении >1000 график O(N) был выше графика наихудшего случая, но второй график на его фоне не превращался в прямую линию.
  2. Совмещенный график среднего, наихудшего и наилучшего времени исполнения;

3. Совмещённый график средней, наилучшей и наихудшей глубины рекурсии;
4. Совмещённый график среднего по серии количество вызовов функции построения кучи и количества вызовов внутренней функции;
5. График среднего процентного соотношения вызовов внутренней функции к общему вызову функции.

- По результатам расчётов оформляется отчёт по предоставленной форме, в отчете:

1. Приводится описание алгоритма;
2. Приводится описание выполнения задачи (описание кода и специфических элементов реализации);
3. Приводятся выводы (Графики и их анализ). Требуется ответить на вопрос о поведении алгоритма, изученного в процессе выполнения лабораторной работы и зафиксировать его особенности.

Описание метода/модели

Пирамидальная сортировка (или, Сортировка кучей) - это метод сортировки на основе сравнения, основанный на двоичной куче данных. При сортировке кучей мы используем двоичную кучу, чтобы быстро находить и перемещать максимальный элемент за *O(logN)* вместо *O(N)* и, следовательно, достигать временной сложности *O(NlogN)*. Ход алгоритма:

1. Переставляем элементы массива так, чтобы они образовывали максимальную кучу;
2. Повторяем следующие шаги до тех пор, пока куча не будет содержать только один элемент:
   (a) Меняем местами корневой элемент кучи с последним элементом кучи;
   (b) Удаляем последний элемент кучи;
   (c) Складываем в кучу остальные элементы кучи.
3. Получаем отсортированный массив.

Анализ сложности пирамидальной сортировки:

- **Лучший вариант: O(N)**, если массив уже отсортирован;
- **Средний вариант: O($N(logN)$)**, если массив упорядочен случайным образом;
- **Наихудший вариант: O($N(logN)$)**, если массив находится в обратном порядке, где N - количество элементов в массиве.

Преимущества:

- Эффективная временная сложность;
- Использование памяти может быть минимальным;
- Простота.

Недостатки:

- Дорогостоящая, так как константы выше по сравнению с сортировкой слиянием;
- Неэффективен из-за высоких констант во временной сложности.

Выполнение задачи

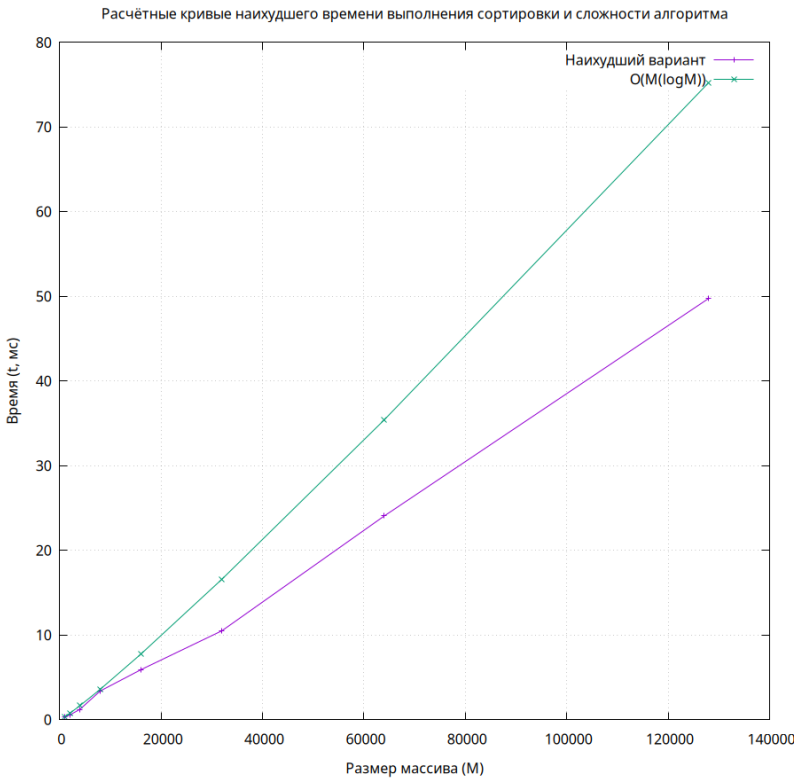
Алгоритм пирамидальной сортировки реализован на языке *C++*. Построение графиков проводить с помощью программы *GNUplot*.

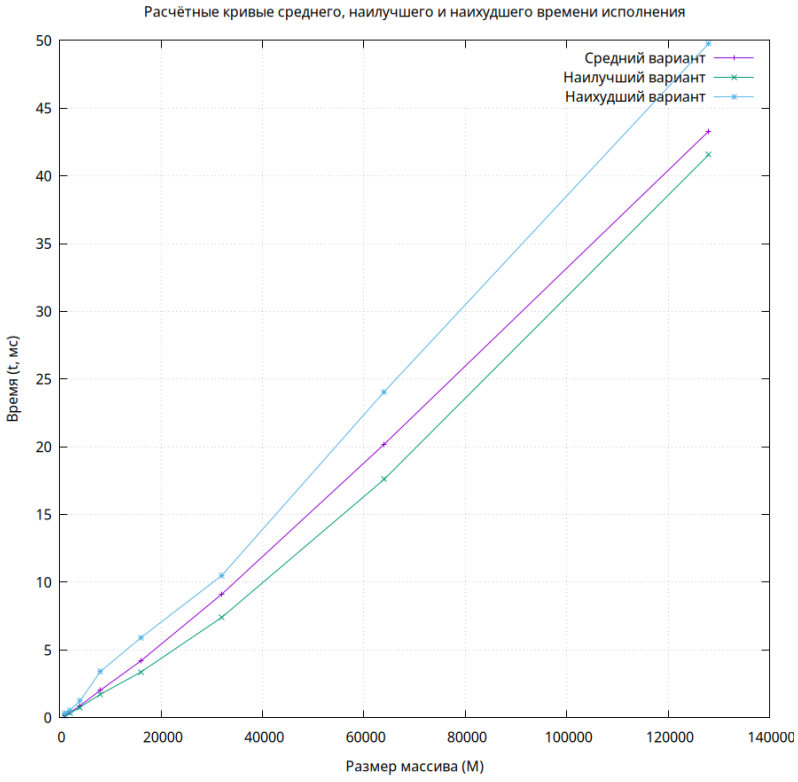
"main" функция работает с циклом, в ходе которого производится расчёт минимального, максимального и среднего времени на сортировку массива размером *M*. Каждая серия просчитывается по 20 раз. В итоге получаются данные, выведенные в определенные файлы, с помощью которых впоследствии строятся графики.

```
1
2
```

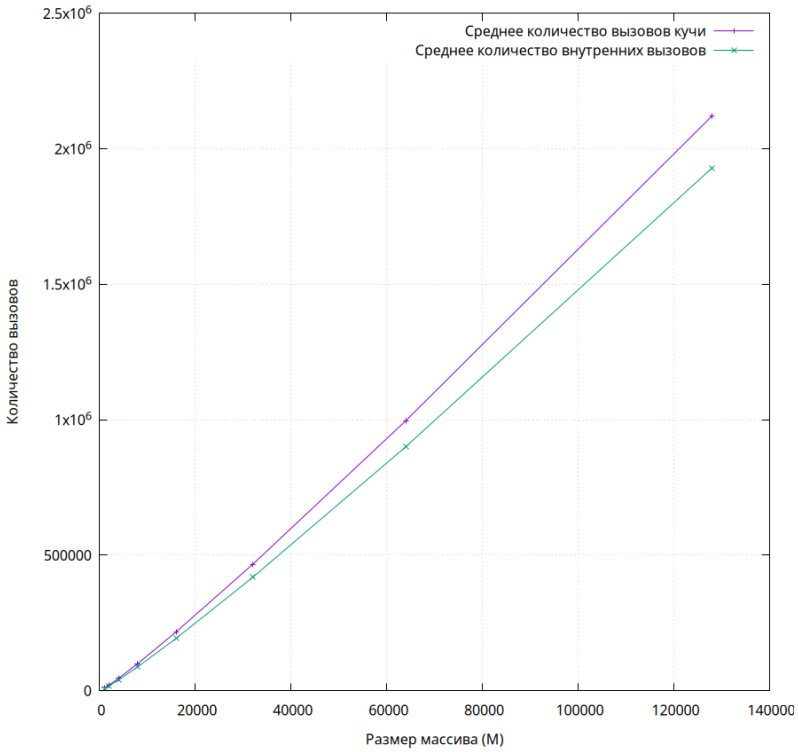
"generationArray" функция принимает два аргумента: array - массив, size - размер массива. Формирует массив размера size, который заполняется случайными числами с плавающей точкой от -1 до 1.

```
1    void generationArray(double array[], int size) {
2      std::random_device rd;
3      std::mt19937 engine(rd());
4      std::uniform_real_distribution<double> gen(-1.0, 1.0);
5
6      for (int i = 0; i < size; i++)
7        array[i] = gen(engine);
8    }
9
```

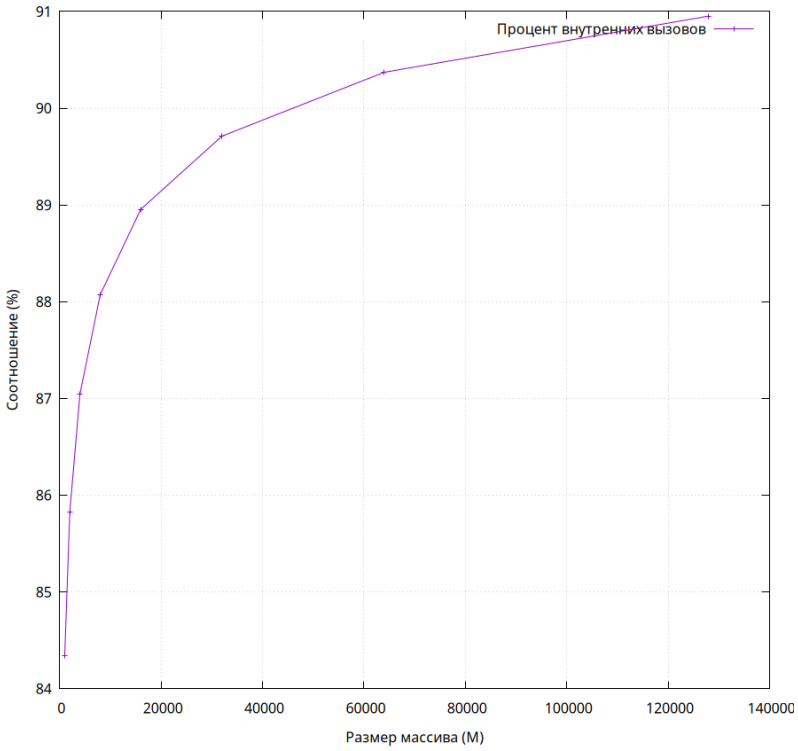

Расчётные кривые наихудшего времени выполнения сортировки и сложности алгоритма
Наихудший вариант
O(M(logM))
Время (t, мс)
Размер массива (M)
0
10
20
30
40
50
60
70
80
0
20000
40000
60000
80000
100000
120000
140000


Расчётные кривые среднего, наилучшего и наихудшего времени исполнения
Средний вариант
Наилучший вариант
Наихудший вариант
Время (t, мс)
Размер массива (M)
0
5
10
15
20
25
30
35
40
45
50
0
20000
40000
60000
80000
100000
120000
140000

асчётные кривые среднего по серии количества вызовов функции построения кучи и количества вызовов внутрен



Расчётная кривая процентного соотношения вызовов внутренней функции к общему вызову функции



Выводы