**Отчёт по лабораторной работе по курсу «Анализ и разработка алгоритмов»**

**Тема: «Сравнение различных подходов к хешированию»**

Вариант №3.

Выполнил студент группы 381703-4м Четвериков Антон

Целью данной работы является сравнение различных методов борьбы с коллизиями:

1. Метод открытой адресации;
2. Метод кукушки;
3. Также необходимо сделать сравнения со стандартными средствами языка С++: std::map, std::hash\_map.

В качестве хеш-функций были использованы функции из универсальных семейств.

Входные данные – случайные натуральные числа.

Программа должна измерять:

1. Время вставки;
2. Время удаления;
3. Время поиска.

Вывод:

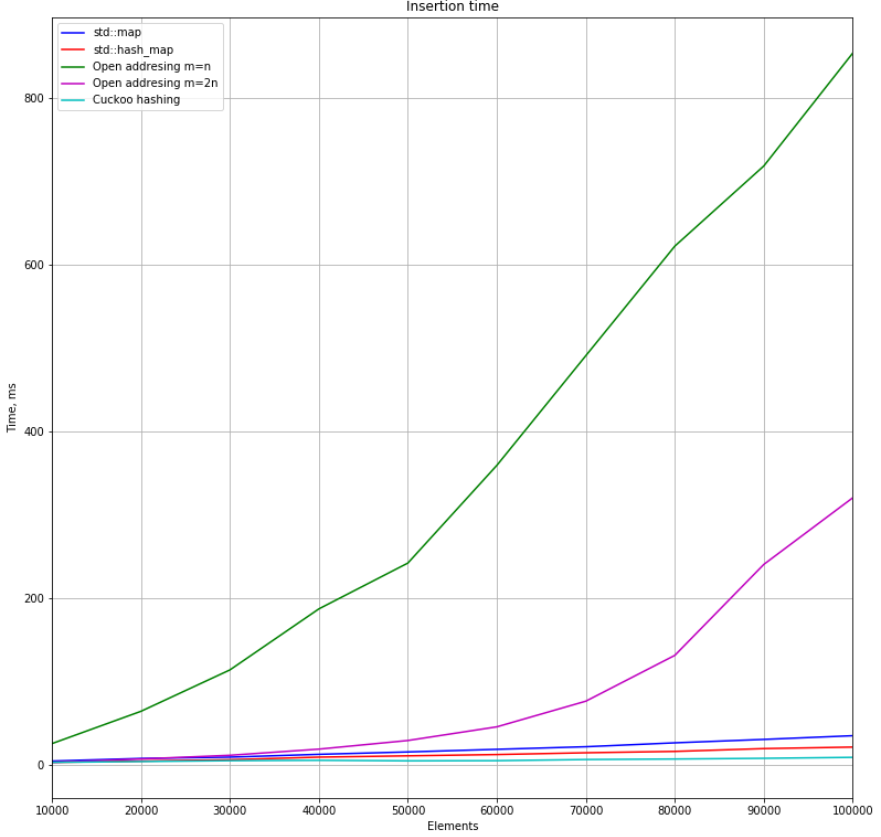
График зависимости скорости вставки от количества элементов в таблице;

График зависимости скорости удаления от количества элементов в таблице;

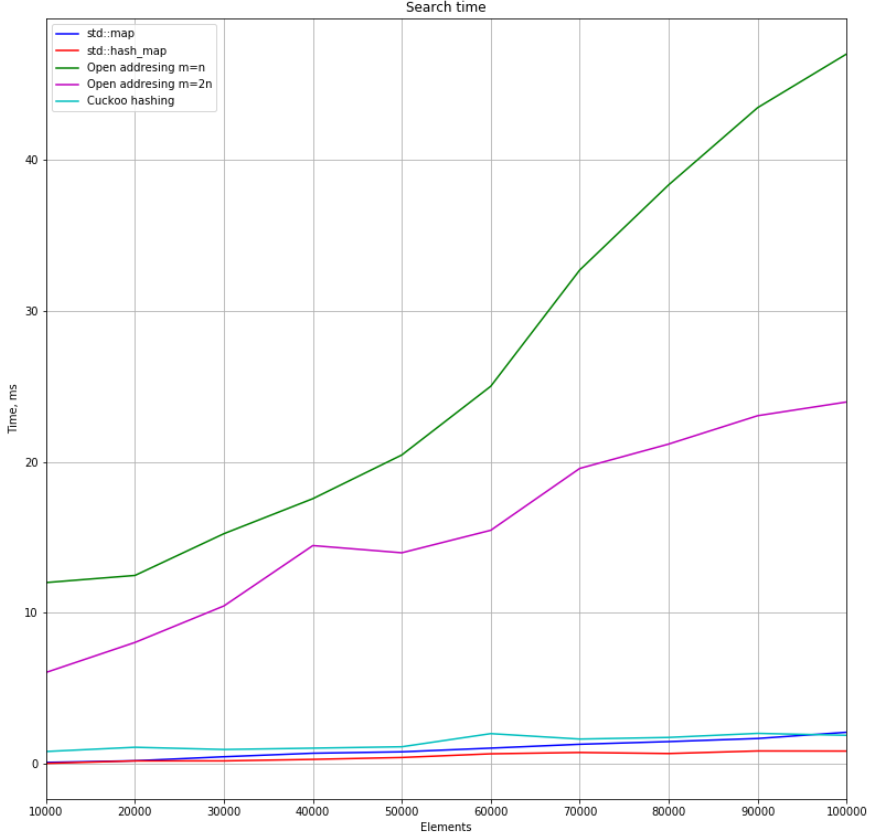
График зависимости скорости поиска от количества элементов в таблице;

Эксперименты проводились в релизной версии программы, для получения более объективных результатов измерение времени выполнения операций проводилось несколько раз и вычислялось среднее значение времени.

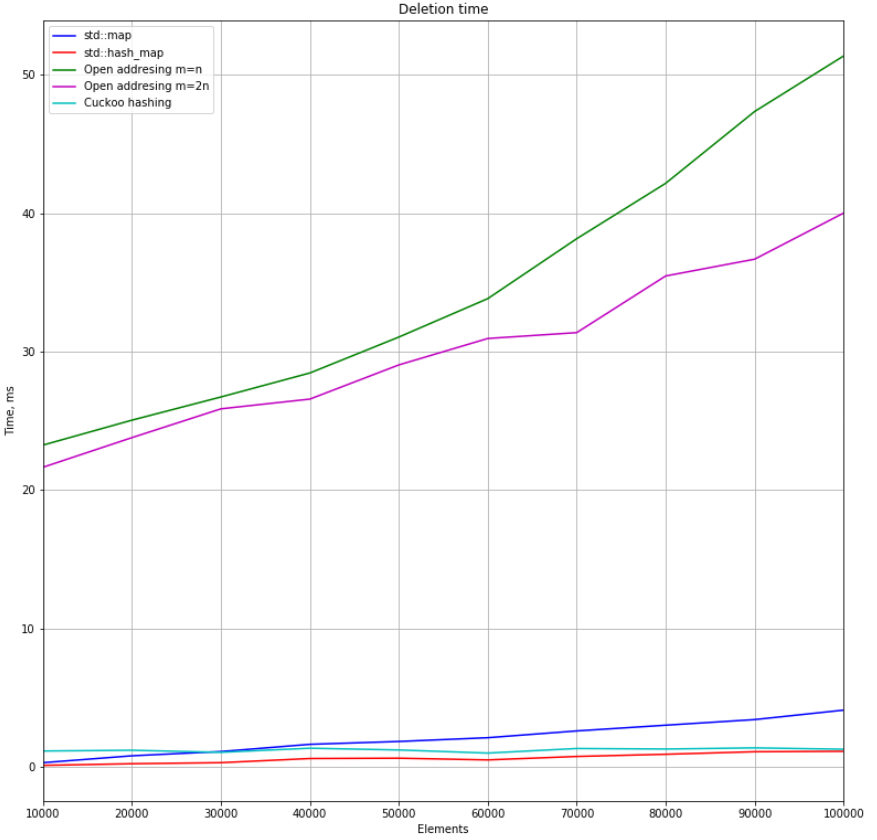
Полученные результаты вручную заносились в текстовый файл, для построения графиков использована оболочка Jupyter Notebook.

График зависимости скорости вставки от количества элементов в таблице.

Хэширование методом кукушки, std::map, std::hash\_map показывают сходные результаты, при этом значения для std::map, std::hash\_map незначительно растут с увеличением числа элементов. Метод открытой адресации показывает сильную зависимость времени вставки от числа элементов, увеличение выделяемой памяти позволяет снизить затраты по времени.

График зависимости скорости поиска от количества элементов в таблице.

Хэширование методом кукушки, std::map, std::hash\_map показывают сходные результаты. Значения, полученные для метода кукушки, фактически константные, так как с увеличением числа элементов их рост не наблюдается. Метод открытой адресации показывает зависимость времени поиска от числа элементов, увеличение выделяемой памяти позволяет снизить затраты по времени.

График зависимости скорости удаления от количества элементов в таблице.

Хэширование методом кукушки и std::hash\_map показывают сходные результаты, std::map с увеличением числа элементов затрачивает по сравнению с ними больше времени. Значения, полученные для std::map и std::hash\_map, растут в зависимости от количества элементов. Значения, полученные для метода кукушки, фактически константные, так как с увеличением числа элементов их рост не наблюдается. Метод открытой адресации показывает зависимость времени удаления от числа элементов, увеличение выделяемой памяти позволяет несколько снизить затраты по времени.