**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра МКИТ

Отчет по лабораторной № 2

по дисциплине

«Структуры и алгоритмы обработки данных»

Выполнил: студент группы

БВТ1904

Игнатенко Ярослав Игоревич

Руководитель:

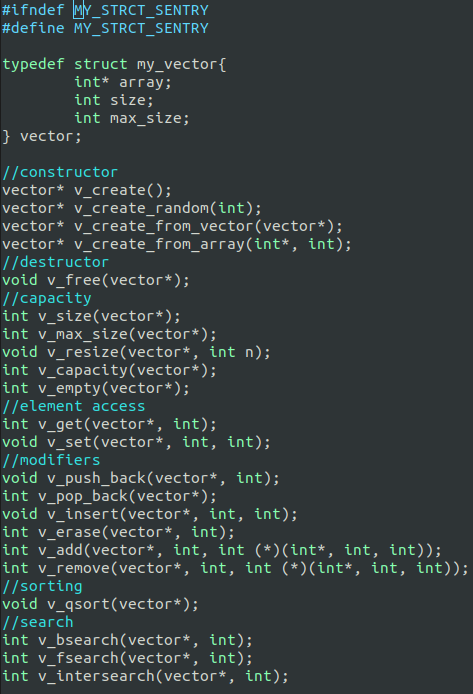
Павликов А.Е.

Москва 2021

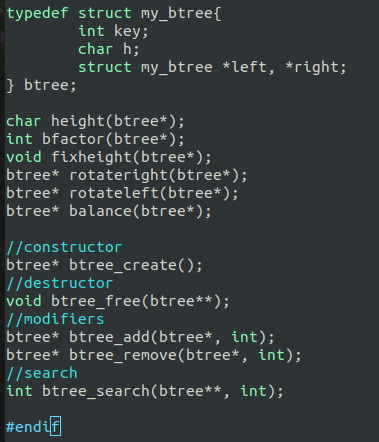
# **Цель работы**

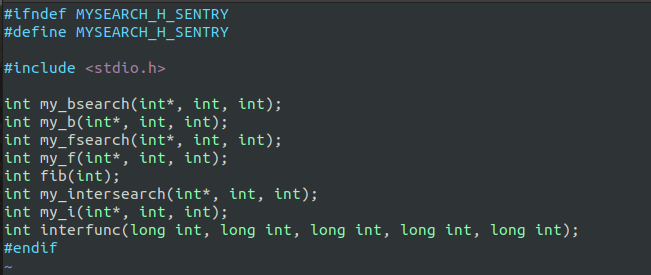
# Изучение алгоритмов поиска и сравнение времени их выполнения. Изучение рехеширования и сравнение времени их выполнения. Для каждого варианта нужно реализовать добавление, поиск, и удаление.

**Определение динамического массива vector**

****

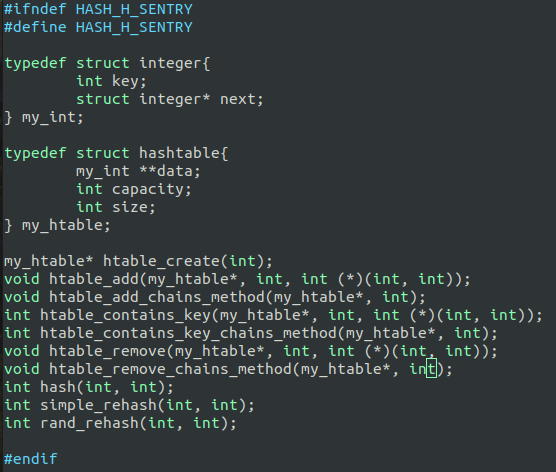
**Определение AVL-дерева**

**Определение методов поиска**

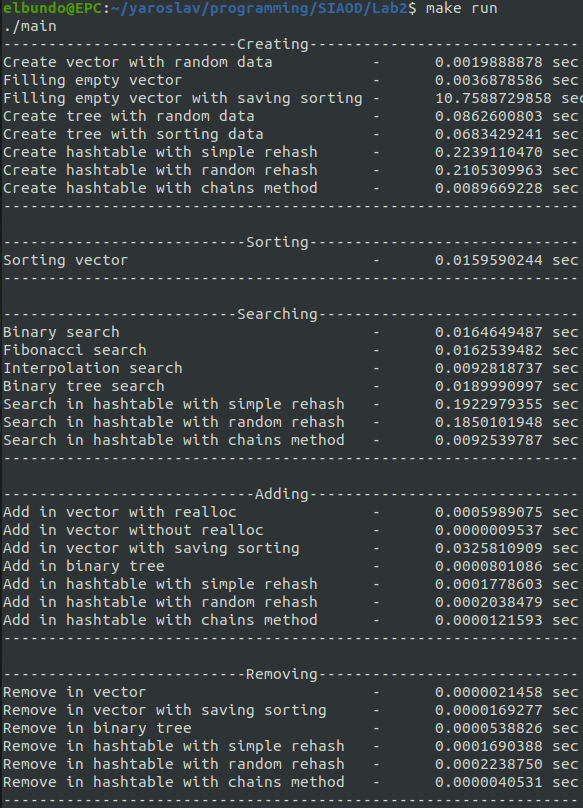
**my\_bsearch, my\_fsearch, my\_intersearh — функции поиска, которые возвращают индекс найденного элемента или -1 если элемент отсутствует.**

**my\_b, my\_f, my\_i — функции поиска, которые возвращают индекс найденного элемента если он есть и позицию где должен был бы стоять этот элемент, если элемент отсутствует. Используется в добавлении нового элемента в vector с сохранением упорядоченности.**

**Определение хеш таблиц**

**Функция htable\_add, htable\_contains\_key и htable\_remove принимают указатель на функцию метода рехеширования, в данном случае либо простое либо с использованием случайных чисел.**

**Результат работы**

****

**Создание**:

Создание вектора на 100000 элементов и заполение его случайными числами

Создание пустого вектора и добавление в него 100000 элементов

Создание пустого вектора и добавление в него 100000 элементов с сохранением упорядоченности

Заполнение бинарного дерева случайными числами

Заполнение бинарного дерева числа по возрастанию

Создание хеш таблицы на 300000 и заполнение 100000 элементами с использованием простого рехеширования

Создание хеш таблицы на 300000 и заполнение 100000 элементами с использованием рехеширования с случайными числами

Создание хеш таблицы на 300000 и заполнение 100000 элементами с использованием метода цепочек

**Сортировка вектора**

**Поиск**:



В поиске идет перебор всех элементов вектора и поиск этих элементов в самом векторе. Для хеш таблиц выполяется функция contains\_key.

**Добавление:**

Сначала создается вектор на 100 элементов и каждый элемент вспомогательного вектора добавляется во все структуры данных с сохранением упорядоченности.4

**Удаление:**

Из каждой структуры данных удаляется каждый элемент, который был добавлен в прошлом шаге с сохранением упорядоченности

**Вывод**

По результатам выполнения программы я сделал вывод, что при равномерном распределении данных в массиве интерполяционный поиск сравним по скорости с поиском методом цепочек в хеш таблице. Самое быстрое добавление без сохранения упорядоченности получилось у вектора без перевыделения памяти. А с сохранением упорядоченности выигрывает хеш таблица с методом цепочек.