Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

ордена Трудового Красного Знамени федеральное государственное

бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра “Математическая кибернетика и информационные технологии”

Лабораторные работы по дисциплине

«Структуры и алгоритмы обработки данных»

Лабораторная работа №2

«Методы поиска»

Выполнила студентка группы БСТ2002

Сергеева А. А.

Проверил: Аршинов Е.А.

Москва 2022 г.

# Цель работы

Реализовать методы поиска в соответсвии с заданием. Организовать генерацию начального набора случайных данных. Для всех вариантов добавить реализацию добавления, поиска и удаления элементов. Оценить время работы каждого алгоритма поиска и сравнить его со стандартной функцией поиска, используемой в выбранном вами языке программирования.

Задание

|  |  |  |  |
| --- | --- | --- | --- |
| Бинарный поиск | Бинарное дерево | Фибоначчи | Интерполяционный |
| Простое хеширование | Рехеширование с помощью псевдослучайных чисел | | Метод цепочек |

# Ход работы

## Бинарный поиск

Бинарный поиск возможен только в отсортированном массиве. На каждом шаге в массиве выбираются левая и правая границы (на первом шаге левой границей является элемент с нулевым индексом, а правой – последний элемент). По границам находится середина, и алгоритм проверяет является ли данный элемент искомым, а если нет, то проверяет больше он или меньше искомого. В случае, если искомый элемент меньше серединного, то мы сдвигаем правую границу на текущую середину, в противном случае – левую. Так алгоритм идет по массиву, и он завершит свою работу в случае, если искомый элемент найден или если левая граница стала больше правой, что значит, что искомого элемента в массиве нет.

## Бинарное дерево

Элементы в бинарном дереве расположены следующим образом: каждый из элементов может иметь один родительский элемент и два дочерних. Элемент находящийся слева меньше текущего, а элемент справа – больше. Таким образом, для того чтобы найти элемент, нужно сравнить его с текущим и в случае, если он окажется меньше, то следующий элемент алгоритм ищет слева, в противном случае – справа. В конце искомый элемент либо будет найден, либо текущий элемент, не оказавшийся искомым, не будет иметь дочерних элементов.

## Фибоначчи

Поиск Фибоначчи возможен только в отсортированном массиве. С искомым элементом сравниваются элементы массива, находящиеся в массиве с индексами, соответствующими числам ряда Фибоначчи. Если текущий элемент меньше искомого, то алгоритм возвращается к элементу, находящемуся по индексу предыдущего числа Фибоначчи. И ряд Фибоначчи сбрасывается к началу.

## Интерполяционный

Интерполяционный поиск так же возможен только в отсортированном массиве. Данный алгоритм работает так же, как и бинарный поиск. Но вместо серединного элемента, элемент берется по формуле.

## Простое хеширование

В простом хешировании используется функция, вычисляющая хеш по заданному значению. Если у двух значений хэш совпадает, то последние замещает первое. Таким образом, для того чтобы найти конкретное значение, необходимо посчитать его хеш и в общем массиве искать значение по индексу, вычисленному хэш-функций.

## Рехеширование с помощью псевдослучайных чисел

Рехеширование с помощью псевдослучайных чисел происходит в случае, если программе нужно сохранить значение, но на месте, рассчитанном хэш-функцией уже есть какое-то значение. В таком случае используется функция, считающая новый хэш на основе старого хэша и простых чисел.

## Метод цепочек

В методе цепочек место хранения можно представить как массив, каждый элемент которого – связный список. Получается, что элементы с одинаковым хэшем хранятся в связном списке, и для добавления еще одного необходимо изменить ссылку последнего на данный момент с None на новый элемент.

# Выводы

После проделанной работы были изучены методы поиска и хеширования.