Министерство высшего образования и науки Российской Федерации

Федеральное государственное образовательное автономное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет» (ПНИПУ)**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

ОТЧЁТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ

Дисциплина: Информатика

Тема: Интерполяционный поиск

Выполнил

Студент группы РИС-22-2б

Худеньких В.Д.

Проверил

доц. кафедры ИТАС

Полякова О.А.

Пермь 2023

**Содержание**

Введение **3**

Разработка программы вычисления значений функции с использованием языка программирования C++**9**

Заключение**11**

**Введение**

В современном мире большое значение имеет эффективная обработка больших объемов данных. Одним из задач, которые являются важными для этого, является поиск элемента в отсортированном массиве. Для эффективного решения этой задачи можно использовать интерполяционный поиск, который основывается на интерполяции значений для примерного расчета позиции искомого элемента.

Цель данной лабораторной работы – разработка программы на языке С++, решающей задачу интерполяционного поиска элемента в отсортированном массиве. Основными задачами работы являются изучение метода интерполяционного поиска, разработка алгоритма и реализация программы, а также тестирование ее на различных наборах данных.

В данном отчете будет представлено описание методики решения задачи интерполяционного поиска, разработка программы на языке С++, протестированной на различных наборах данных, а также приведен вывод о результатах выполненной работы.

**Разработка программы с использованием языка программирования C++.**

Интерполяционный поиск: теоретическое обоснование

В основе интерполяционного поиска лежит не линейный, а более точный метод поиска элемента в упорядоченном массиве. Алгоритм основывается на оценке интервала, в котором находится искомый элемент, путем использования формулы интерполяции. Формула интерполяции позволяет оценить значение элемента по имеющимся в массиве данным, что дает возможность примерно вычислить его позицию и сократить количество сравнений при поиске.

Основным преимуществом интерполяционного метода является уменьшение количества сравнений элементов, так как он позволяет уменьшить количество проверок тех элементов, которые расположены далеко друг от друга.

Разработка алгоритма программы

Для реализации интерполяционного поиска необходимо проанализировать массив данных и вычислить интерполяционное значение, которое дает возможность оценить позицию искомого элемента. Для этого используется следующая формула:

pos = low + ((key - arr[low]) / (arr[high] - arr[low])) \* (high - low)

где pos – примерное местоположение элемента в массиве, key – значение ключа для поиска, arr[low] и arr[high] – значения первого и последнего элементов в интервале нахождения искомого элемента.

После определения примерной позиции искомого элемента в массиве производится проверка на соответствие ключа. Если значение ключа совпадает с найденным, то поиск завершается успешно. Если нет, то проверяем, в каком из интервалов, определенных на предыдущем шаге, находится искомый элемент. Если элемент находится в левом интервале, то поиск продолжается от первого элемента до позиции pos - 1, если элемент находится в правом интервале, то поиск продолжается от позиции pos + 1 до последнего элемента.

Реализация программы

Программа на языке С++ была написана в Code::Blocks. Программа принимает на вход упорядоченный массив элементов, определяет значение ключа и выполняет поиск интерполяционным методом.

Тестирование программы

Программа была протестирована на нескольких наборах данных разной сложности. Тесты позволили подтвердить правильность работы алгоритма и его эффективность.

**Заключение**

Интерполяционный поиск – это метод, который позволяет уменьшить время поиска элемента в упорядоченном массиве. Благодаря использованию формулы интерполяции можно оценить позицию искомого элемента, что сокращает количество проверок и уменьшает время выполнения алгоритма. Разработанная программа на языке С++ показала высокую эффективность при поиске элементов в больших массивах данных.

**ПРИЛОЖЕНИЕ А**

**Блок-схемы программ**

