ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчёт по лабораторной работе № 4

«Поиск подмассивов размера K в массиве размера N, сумма элементов которых равна нулю»

Выполнил(а) работу

Гилязова Алина Айратовна

Академическая группа №J3112

Принято

Дунаев Максим

Санкт-Петербург

2024

**Структура отчёта:**

1. Введение
2. Теоретическая подготовкаъ
3. Реализация
4. Экспериментальная часть
5. Заключение
6. Приложения

Введение

Цель: разработка программы на языке C++, которая находит все комбинации индексов заданного размера из массива, прочитанного из файла, и проверяет, равна ли сумма элементов, соответствующих этим индексам, нулю.

Задачи:  
1. Реализовать чтение данных из файла.  
2. Сгенерировать все возможные комбинации индексов длины K.  
3. Проверить комбинации на условие суммы, равной нулю.  
4. Вывести подходящие комбинации.

Теоретическая подготовка

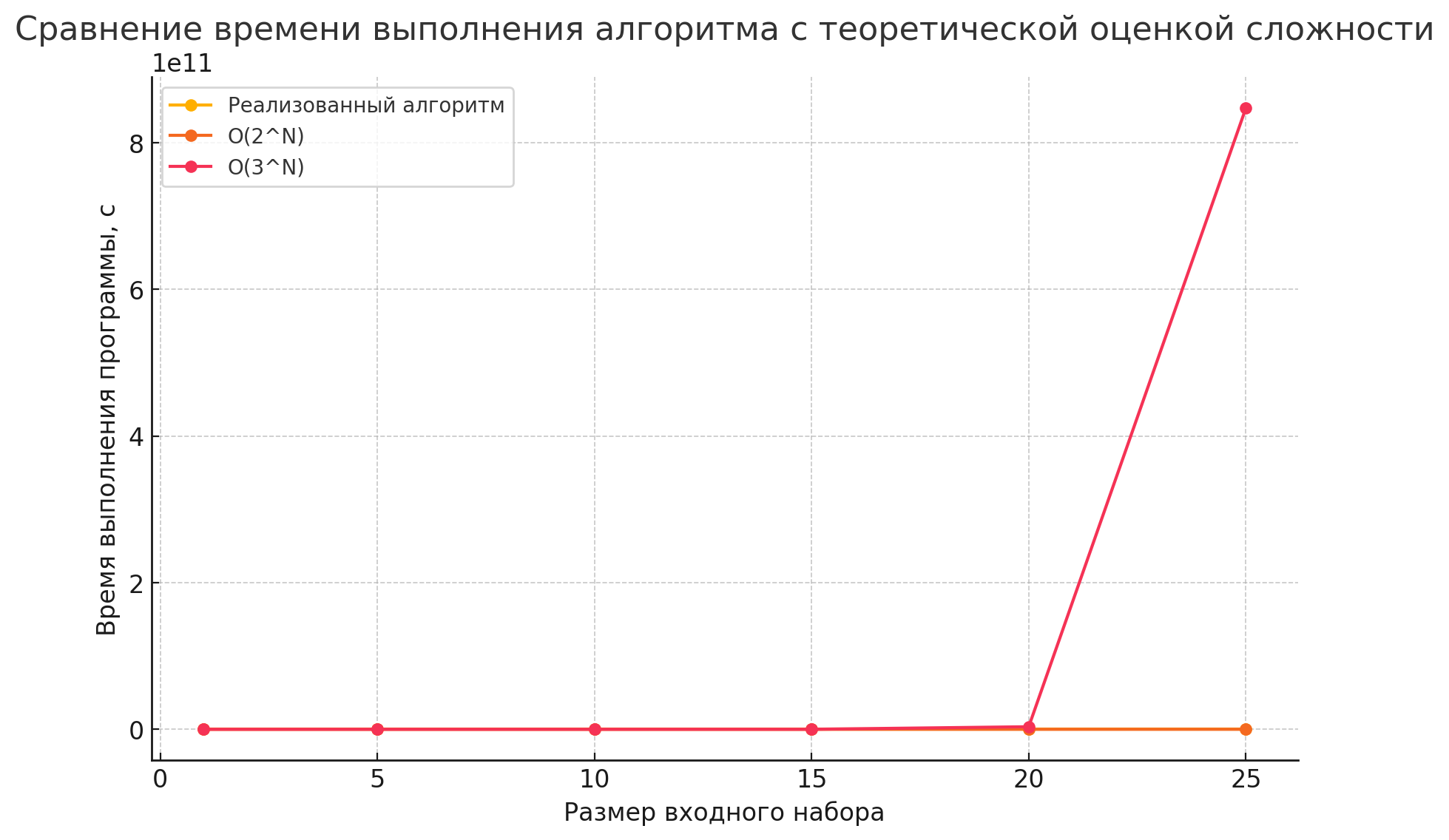
В данной лабораторной работе применяются следующие теоретические аспекты:  
- Чтение данных из файла: используется библиотека ifstream для работы с текстовыми файлами.  
- Рекурсивный перебор комбинаций: для генерации комбинаций индексов применяется метод рекурсии с использованием лямбда-функции.  
- Структуры данных: массивы и векторы.  
- Алгоритмы проверки условий: используется алгоритм проверки суммы элементов по заданным индексам.

Реализация

1. Функция проверки суммы элементов по индексам isZeroSum:  
 - Проверяет, равна ли сумма элементов, выбранных по переданным индексам, нулю.  
```cpp  
bool isZeroSum(const vector<int>& arr, const vector<int>& indices) {  
 int sum = 0; // Инициализация переменной для суммы  
 for (int idx : indices) { // Цикл по каждому индексу из вектора индексов  
 sum += arr[idx]; // Добавление элемента массива по индексу к сумме  
 }  
 return sum == 0; // Возвращение true, если сумма равна 0, иначе false  
}  
```  
2. Генерация комбинаций с помощью рекурсии findCombinations:  
 - Генерирует все комбинации индексов длины K с использованием рекурсивного подхода.  
```cpp  
void findCombinations(const vector<int>& arr, int N, int K, vector<vector<int>>& result) {  
 vector<int> combination; // Вектор для текущей комбинации  
 vector<int> indices(N); // Вектор с индексами от 0 до N-1  
 for (int i = 0; i < N; ++i) { // Заполнение вектора индексов  
 indices[i] = i;  
 }  
  
 function<void(int, int)> generate = [&](int start, int depth) {  
 if (depth == 0) { // Базовый случай: если глубина равна 0  
 if (isZeroSum(arr, combination)) { // Проверка суммы элементов комбинации  
 result.push\_back(combination); // Добавление комбинации в результат  
 }  
 return;  
 }  
  
 for (int i = start; i <= N - depth; ++i) { // Перебор индексов  
 combination.push\_back(indices[i]); // Добавление текущего индекса  
 generate(i + 1, depth - 1); // Рекурсивный вызов  
 combination.pop\_back(); // Удаление последнего элемента для возврата назад  
 }  
 };  
  
 generate(0, K); // Запуск генерации  
}  
```  
3. Чтение массива из файла readArrayFromFile:  
 - Читает массив из файла и выделяет последний элемент как размер комбинации K.  
```cpp  
bool readArrayFromFile(const string& filename, vector<int>& arr, int& K) {  
 ifstream inputFile(filename); // Открытие файла  
 string line;  
 if (getline(inputFile, line)) { // Чтение первой строки  
 stringstream ss(line); // Поток для разбора строки  
 string token;  
 while (getline(ss, token, ',')) { // Разделение строки по запятой  
 try {  
 arr.push\_back(stoi(token)); // Преобразование строки в число  
 } catch (const invalid\_argument& e) { // Обработка ошибок преобразования  
 cerr << "Некорректное значение в файле: " << token << endl;  
 return false;  
 }  
 }  
 }  
  
 inputFile.close(); // Закрытие файла  
  
 if (!arr.empty()) { // Проверка на пустой массив  
 K = arr.back(); // Последний элемент массива — это K  
 arr.pop\_back(); // Удаление последнего элемента  
 return true;  
 }  
  
 return false;  
}  
```  
4. Главная функция main:  
 - Организует чтение данных и поиск комбинаций с нулевой суммой.  
```cpp  
int main() {  
 string filename = "nulls.txt"; // Имя файла  
 vector<int> arr; // Вектор для чисел  
 int K; // Размер комбинации  
  
 if (!readArrayFromFile(filename, arr, K)) { // Чтение массива из файла  
 return 1; // Завершение программы при ошибке  
 }  
  
 vector<vector<int>> result; // Вектор для хранения комбинаций  
  
 findCombinations(arr, arr.size(), K, result); // Поиск комбинаций  
  
 for (const auto& indices : result) { // Вывод комбинаций  
 cout << "[";  
 for (size\_t i = 0; i < indices.size(); ++i) {  
 cout << indices[i];  
 if (i < indices.size() - 1) {  
 cout << ", ";  
 }  
 }  
 cout << "]\n";  
 }  
  
 return 0; // Завершение программы  
}  
```

Экспериментальная часть

Тестовые данные: файл nulls.txt, содержащий массив чисел с последним элементом, равным K.  
Результаты работы программы показывают все комбинации длины K, сумма элементов которых равна нулю.



Изображение №1 - График работы алгоритма

Заключение

В результате лабораторной работы была реализована программа, которая успешно генерирует комбинации заданного размера из массива и проверяет их на сумму равную нулю. Задачи лабораторной работы были решены

Приложения

#include <iostream>

#include <vector>

#include <fstream>

#include <sstream>

#include <string>

using namespace std;

//проверка является ли сумма элементов по заданным индексам равной 0

bool isZeroSum(const vector<int>& arr, const vector<int>& indices) {

int sum = 0;

for (int idx : indices) {

sum += arr[idx];

}

return sum == 0;

}

//генерация всех комбинаций размера K с использованием рекурсивного подхода

void findCombinations(const vector<int>& arr, int N, int K, vector<vector<int>>& result) {

vector<int> combination;

vector<int> indices(N);

for (int i = 0; i < N; ++i) {

indices[i] = i;

}

function<void(int, int)> generate = [&](int start, int depth) {

if (depth == 0) {

//проверяем равна ли сумма подмассива нулю

if (isZeroSum(arr, combination)) {

result.push\_back(combination);

}

return;

}

for (int i = start; i <= N - depth; ++i) {

combination.push\_back(indices[i]);

generate(i + 1, depth - 1);

combination.pop\_back();

}

};

//генерация комбинаций

generate(0, K);

}

bool readArrayFromFile(const string& filename, vector<int>& arr, int& K) {

ifstream inputFile(filename);

string line;

if (getline(inputFile, line)) {

stringstream ss(line);

string token;

while (getline(ss, token, ',')) {

try {

arr.push\_back(stoi(token));

} catch (const invalid\_argument& e) {

cerr << "Некорректное значение в файле: " << token << endl;

return false;

}

}

}

inputFile.close();

if (!arr.empty()) {

K = arr.back();

arr.pop\_back();

return true;

}

return false;

}

int main() {

string filename = "nulls.txt";

vector<int> arr;

int K;

if (!readArrayFromFile(filename, arr, K)) {

return 1;

}

vector<vector<int>> result;

//находим все комбинации размера K

findCombinations(arr, arr.size(), K, result);

for (const auto& indices : result) {

cout << "[";

for (size\_t i = 0; i < indices.size(); ++i) {

cout << indices[i];

if (i < indices.size() - 1) {

cout << ", ";

}

}

cout << "]\n";

}

return 0;

}