

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Санкт-Петербургский государственный технологический институт (технический университет)»

Дисциплина: «Программирование»

Отчёт по лабораторной работе № 3

**Лабораторная работа №3. Методы хэширования**

**Выполнил студент группы №485:**

Зобнин Илья Михайлович

**Проверили:**

Иван Григорьевич Корниенко

Алексей Константинович Федин

Санкт-Петербург

2019

# Постановка задачи

Необходимо составить программу для поиска по хэшам данных. В модуле поиска предусмотреть реализацию обработки случая, при котором хэш-коды различных данных совпадают. Дана таблица текстовой базы данных записями: фамилия; имя; отчество. Произвести хэширование вместе трёх полей и поиск по запросу “Фамилия Имя Отчество”.

# Исходные данные

В качестве исходных данных программа использует вводимый пользователем путь к базе данных.

# Особые ситуации

* Если пользователь при указании пути к файлу использует запрещённые имена, например: con, aux и т.д., программа попросит ввести путь заново.
* Если пользователь укажет путь к файлу, к которому программа не может получить доступ из-за недостатка прав, она попросит ввести путь заново.
* Если пользователь укажет путь к несуществующему файлу, из которого программа должна получить данные, она попросит ввести путь заново.
* Если пользователь при сохранении результата работы программы укажет путь к уже существующему файлу, программа спросит, хочет ли пользователь перезаписать этот файл. В случае отказа программа попросит ввести путь заново.

# Математические методы и алгоритмы решения задач

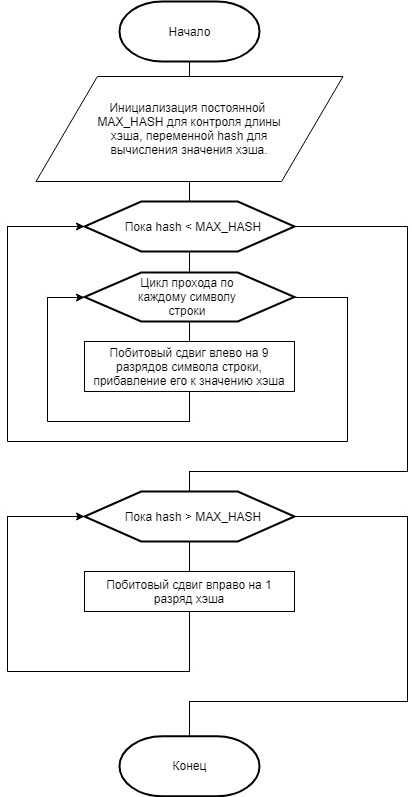


Рисунок 1 – метод хэширования



Рисунок 2 – класс Str, содержащий в себе строку и её номер в базе данных



Рисунок 3 – функция поиска строки в базе данных

# Форматы представления данных

Программа использует следующие переменные:

Таблица 1 – Переменные, используемы в программе

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Описание** |
| running | bool | Определяет запущена ли программа |
| hash | int | Значение хэша строки |
| i | size\_t | Итерация номера строки в базе данных |
| path | string | Путь к базе данных |
| file | ifstream | Открытие базы данных |
| strMap | multimap<int, string> | Структура данных для сохранения пары ключ-значение (хэш-строка) |
| it | multimap iterator | Итератор прохода по структуре данных strMap |
| str | string | Считывание строк из базы данных |
| duplicates | list<Str> | Хранение строк, равных по хэшу со строкой, введённой пользователем |
| choice | bool | Хочет ли пользователь сохранить результат работы программы в файл |
| userInput | int | Проверка на валидность введённых пользователем данных |
| isPathValid | bool | Проверка на валидность введённого пути |

Для задания максимального и минимального размера массива, а также обозначения максимального пункта меню используются следующие константы:

Таблица 2 – Константы, используемы в программе

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя** | **Тип** | **Значение** | **Описание** |
| INT\_MIN | const int | -2147483647 | Минимальное целое число |
| INT\_MAX | const int | 2147483647 | Максимальное целое число |
| MAX\_HASH | const int | 9999 | Максимальный размер хэша |

# Структура программы

В силу большого количества функций программа разделена на 4 исполняемых модуля, из которых один является основным и отвечает за запуск программы и содержит меню, двое оставшихся содержат в себе функции, необходимые для работы программы. Последний файл отвечает за предкомпиляцию и сокращает время сборки программы с 8,2 до 3,6-х секунд.

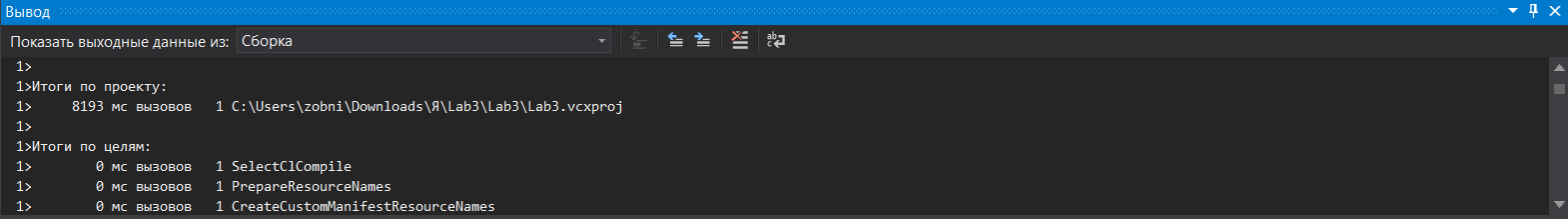


Рисунок 8 – время сборки программы с выключенным предкомпилированным заголовком

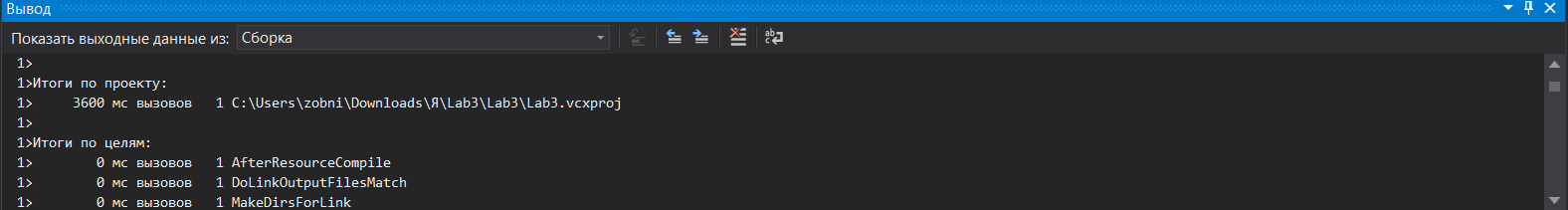


Рисунок 9 – время сборки программы с включённым предкомпилированным заголовком

Модуль Lab3:

Таблица 3 – Функции, составляющие модуль Lab3

|  |  |
| --- | --- |
| **Имя** | **Описание** |
| main | Начало программы |
| Menu | Отображение меню программы |

Модуль Hash:

Таблица 4 – Функции, составляющие модуль Hash

|  |  |
| --- | --- |
| **Имя** | **Описание** |
| Hash | Вычисление хэша строки |
| Str | Занесение в объект класса строку и её номер |
| Search | Поиск введённой пользователем строки в базе данных |

Модуль FileCheckAndColorSet:

Таблица 5 – Функции, составляющие модуль FileCheckAndColorSet

|  |  |
| --- | --- |
| **Имя** | **Описание** |
| SetColor | Изменение цвета текста, выводимого в консоль |
| GetUint | Проверка на валидность данных вводимых пользователем типа unsigned int |
| GetBool | Проверка на валидность данных вводимых пользователем типа bool |
| GetFilePath | Проверка на валидность введённого пути к базе данных |
| OutputDataInFile | Вывод результата работы программы в файл |

# Описание хода выполнения лабораторной работы

* В ходе лабораторной работы было создано решение (Lab3) в интегрированной среде разработки Microsoft Visual Studio C++ 2017. В нём был создан проект.
* При выполнении самого задания было принято решение сначала подсчитывать кол-во элементов в базе данных и после создавать динамический массив соответствующего размера и занесение строк и хэшей в этот массив, что предполагало два раза проходить по базе данных. Это ухудшило бы быстродействие программы и затруднило бы поиск хэша. Поэтому было решено в дальнейшем использовать библиотеки STL и структуру multimap, в частности.
* В созданном проекте нужно было включить все библиотеки, использованные в программе, в предкомпилированный заголовок pch.h для её более быстрой сборки.
* При работе программы с файлами нужно было добавить проверки на валидность имени файла, а также на то, создан ли файл или нет при сохранении.
* При получении пользовательского ввода необходимо было добавить проверку на его соответствие предполагаемому типу данных и условиям выбора.
* Перед повторением программы необходимо очищать память для того, чтобы не возникало непредвиденных ошибок, связанных с заполнением областей памяти старыми данными.

# Результат работы программы

В результате работы программа выводит пронумерованные строки базы данных и их хэши, а также найденные совпадения и их количество.

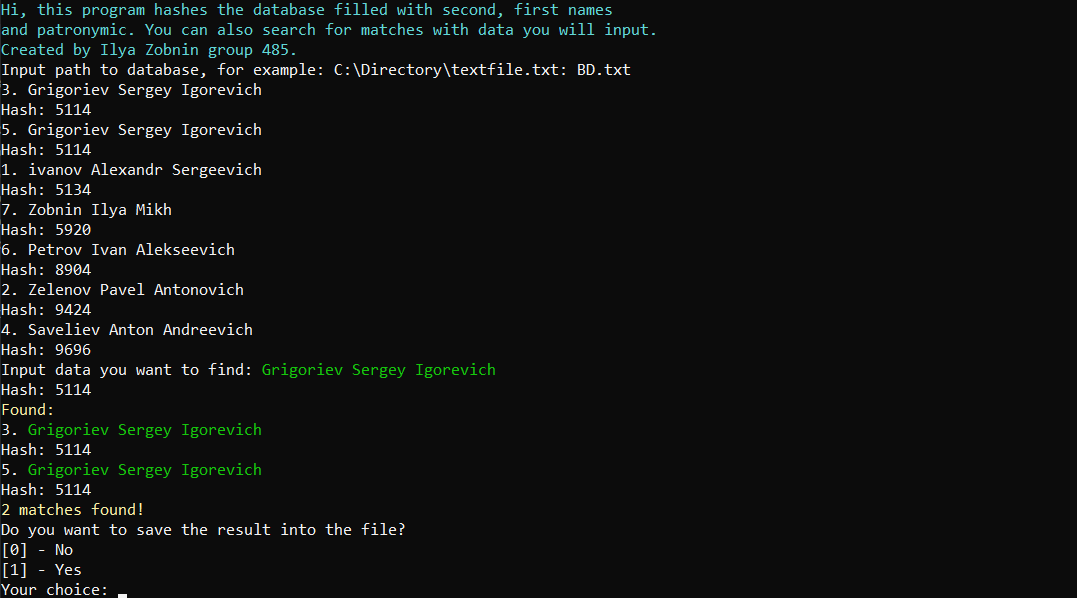


Рисунок 4 – Поиск и вывод двух совпадений в базе данных

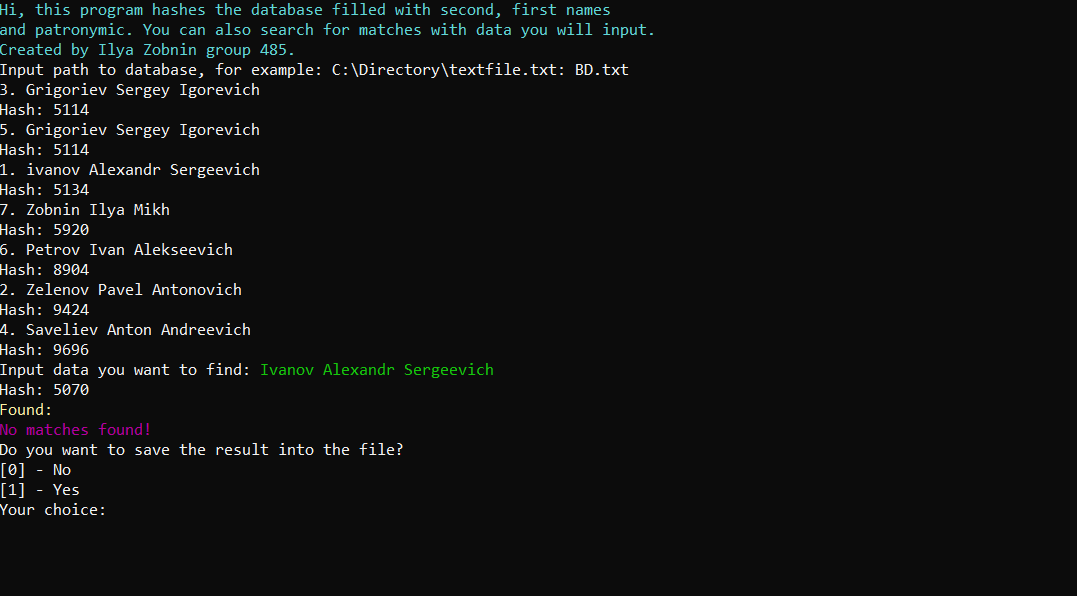


Рисунок 5 – Поиск и вывод сообщения о том, что совпадений в базе данных не найдено

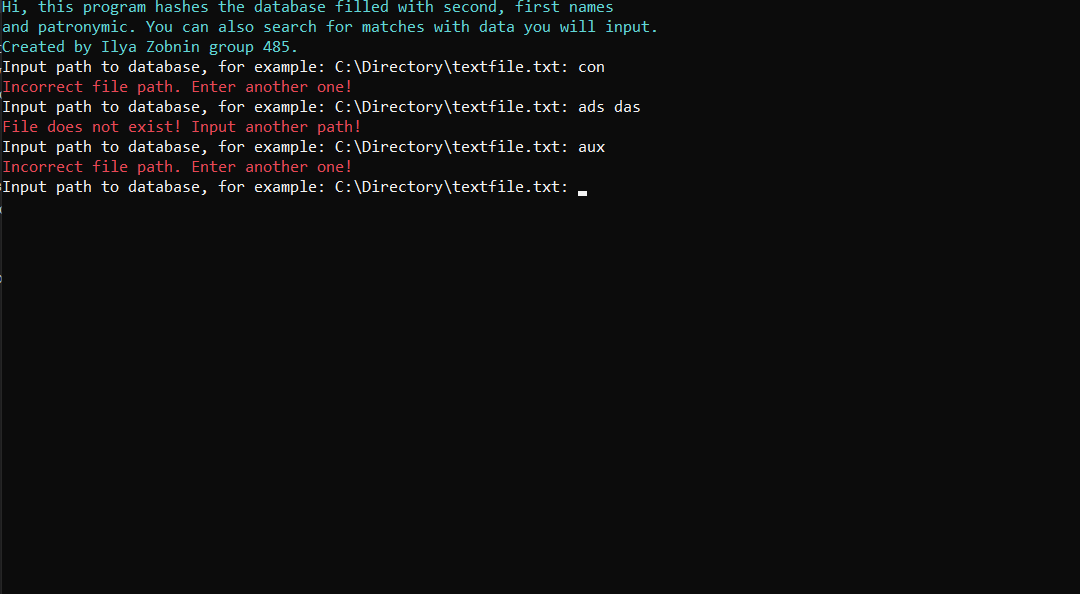


Рисунок 6 – Проверка путей к базе данных

# Текст программы

# [--- Начало программы ---]

**// pch.cpp**

**// Лабораторная работа №3.**

**// Студент группы 485, Зобнин Илья Михайлович. 2019 год**

#include "pch.h"

**// pch.h**

**// Лабораторная работа №3.**

**// Студент группы 485, Зобнин Илья Михайлович. 2019 год**

#pragma once

#include <iostream>

#include <fstream>

#include <Windows.h>

#include <map>

#include <list>

#include <string>

#include <filesystem>

using namespace std;

using namespace std::experimental::filesystem;

**// Hash.h**

**// Лабораторная работа №3.**

**// Студент группы 485, Зобнин Илья Михайлович. 2019 год**

#pragma once

#include "pch.h"

void Search();

int Hash(string str);

class Str

{

public:

string str;

size\_t iterator;

int hash;

Str(const size\_t iterator, const string str);

friend int Hash(string str);

};

**// FileCheckAndColorSet.h**

**// Лабораторная работа №3.**

**// Студент группы 485, Зобнин Илья Михайлович. 2019 год**

#pragma once

#include "pch.h"

#include "Hash.h"

enum { blue = 9, green, azure, red, purple, yellow, white };

void SetColor(int color);

string GetFilePath();

int GetUint();

bool GetBool();

void OutputDataInFile(list <Str> match, int matches, string str, int hash);

**// FileCheckAndColorSet.cpp**

**// Лабораторная работа №3.**

**// Студент группы 485, Зобнин Илья Михайлович. 2019 год**

#include "pch.h"

#include "Hash.h"

enum { blue = 9, green, azure, red, purple, yellow, white };

void SetColor(const int color) {//функция для изменения цвета текста в консоли

SetConsoleTextAttribute(GetStdHandle(STD\_OUTPUT\_HANDLE), color);

}

int GetUint() { // Проверка ввода целого числа

int userInput;

cin >> userInput;//ввод пользователем необходимых данных

while (cin.fail() || userInput < 0) {//цикл пока ввод данных не соответствует заданному типу

SetColor(red);

cout << "Try again: " << endl;

SetColor(white);

cin.clear();//обнуление cin.fail

cin.ignore(INT\_MAX, '\n');//игнорирование введённых данных

cin >> userInput;//повторный ввод переменной

}

cin.ignore(INT\_MAX, '\n');

return userInput;

}

bool GetBool() { // Проверка ввода целого числа

bool userInput;

cin >> userInput;//ввод пользователем необходимых данных

while (cin.fail()) {//цикл пока ввод данных не соответствует заданному типу

SetColor(red);

cout << "Try again: " << endl;

SetColor(white);

cin.clear();//обнуление cin.fail

cin.ignore(INT\_MAX, '\n');//игнорирование введённых данных

cin >> userInput;//повторный ввод переменной

}

cin.ignore(INT\_MAX, '\n');

return userInput;

}

string GetFilePath() {

setlocale(LC\_ALL, "ru");

string filePath;

bool isPathValid = false;

do {

cout << "Input path to database, for example: C:\\Directory\\textfile.txt: ";

cin >> filePath;

if (!ifstream(filePath)) {//если файла не существует

SetColor(red);

cout << "File does not exist! Input another path!" << endl;

SetColor(white);

cin.ignore(INT\_MAX, '\n');

continue;

}

if (!is\_regular\_file(filePath)) {//проверка на валидность имени (защита от aux, com и т.д.)

SetColor(red);

cout << "Incorrect file path. Enter another one!" << endl;

SetColor(white);

cin.ignore(INT\_MAX, '\n');

continue;

}

ifstream myFile(filePath);

if (!myFile) {//если нет доступа к файлу

SetColor(red);

cout << "Access denied! Enter another path!" << endl;

SetColor(white);

myFile.close();

continue;

}

myFile.close();

isPathValid = true;

} while (!isPathValid);

return filePath;

}

void OutputDataInFile(list <Str> match, int matches, string str, int hash) {

string filePath;

bool isDataSaved = false;

do {

cout << "Input path to file, for example: C:\\Directory\\textfile.txt: " << endl;

cin >> filePath;

if (ifstream(filePath))//проверка на существование файла

if (!is\_regular\_file(filePath)) {//проверка на запрещённые имена (aux, com и т.д.)

SetColor(red);

cout << "Incorrect file path. Enter another one!" << endl;

SetColor(white);

cin.ignore(INT\_MAX, '\n');

continue;

}

else {//хочет ли пользователь перезаписать содержимое

SetColor(red);

cout << "File already exists! Do you want to erase all data and write your array in it?" << endl;

SetColor(white);

cout << "[0] - No" << endl << "[1] - Yes" << endl << "Your choice:";

bool isAnother = GetBool();

if (!isAnother)

continue;

}

ofstream myFile(filePath);

if (!myFile) {//проверка на доступ к уже существующему файлу

SetColor(red);

cout << "Access denied! Enter another path!" << endl;

SetColor(white);

myFile.close();

continue;

}

list<Str>::iterator it = match.begin();

myFile << "You searched for: " << str << endl << "Hash: " << hash << endl

<< "Found: " << endl;

if (!matches)

myFile << "No matches found!";

else {

for (it = match.begin(); it != match.end(); it++)

myFile << it->iterator << ". " << it->str << endl << "Hash: " << it->hash << endl;

if (matches == 1)

myFile << matches << " match found";

else

myFile << matches << " matches found";

}

SetColor(green);

cout << "Saved successfully!" << endl;

SetColor(white);

isDataSaved = true;

} while (!isDataSaved);

match.clear();

}

**// Hash.cpp**

**// Лабораторная работа №3.**

**// Студент группы 485, Зобнин Илья Михайлович. 2019 год**

#include "pch.h"

#include "FileCheckAndColorSet.h"

const int MAX\_HASH = 9999;

int Hash(string str) {

size\_t hash = 0;

while (hash < MAX\_HASH)

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

hash += (str[i] << 9);

}

while (hash > MAX\_HASH)

hash = (hash >> 1);

return hash;

}

Str::Str(const size\_t iterator, const string str) {

this->iterator = iterator;

this->str = str;

this->hash = Hash(str);

}

void Search() {

string path = GetFilePath(), str;

ifstream file(path);

multimap<int, Str> strMap;

size\_t i = 0;

while (!file.eof()) {

getline(file, str);

if (str == "\n" || str == " " || str == "\r")

continue;

i++;

Str strIn(i, str);

strMap.emplace(strIn.hash, strIn);

}

file.close();

cin.ignore(INT\_MAX, '\n');

for (auto it = strMap.begin(); it != strMap.end(); it++)

cout << it->second.iterator << ". " << it->second.str << endl << "Hash: " << it->first << endl;

cout << "Input data you want to find: ";

SetColor(green);

getline(cin, str);

SetColor(white);

int hash = Hash(str);

cout << "Hash: " << hash << endl;

list <Str> duplicates;

auto it = strMap.find(hash);

size\_t matchCounter = 0;

SetColor(yellow);

cout << "Found:" << endl;

SetColor(white);

if (it != strMap.end()) {

while (it->first == hash) {

Str duplicate(it->second);

duplicates.emplace\_back(duplicate);

it++;

}

for (auto it = duplicates.begin(); it != duplicates.end(); it++)

if (it->str == str) {

matchCounter++;

cout << it->iterator << ". ";

SetColor(green);

cout << str << endl;

SetColor(white);

cout << "Hash: " << hash << endl;

}

}

if (matchCounter) {

SetColor(yellow);

if (matchCounter == 1)

cout << matchCounter << " match found!" << endl;

else

cout << matchCounter << " matches found!" << endl;

SetColor(white);

}

else {

SetColor(purple);

cout << "No matches found!" << endl;

SetColor(white);

}

cout << "Do you want to save the result into the file?" << endl << "[0] - No" << endl << "[1] - Yes" << endl << "Your choice: ";

bool choice = GetBool();

if (choice)

OutputDataInFile(duplicates, matchCounter, str, hash);

strMap.clear();

duplicates.clear();

}

**// Lab3.cpp**

**// Лабораторная работа №3.**

**// Студент группы 485, Зобнин Илья Михайлович. 2019 год**

#include "pch.h"

#include "FileCheckAndColorSet.h"

#include "Hash.h"

enum { close = 0, running };

bool Menu() {

cout << "[0] - Exit the program" << endl << "[1] - Keep running the program" << endl << "Your choice: ";

bool menu = GetBool();

return menu;

}

void main() {

SetColor(azure);

cout << "Hi, this program hashes the database filled with second, first names" << endl << "and patronymic."

<< " You can also search for matches with data you will input." << endl << "Created by Ilya Zobnin group 485." << endl;

SetColor(white);

bool running = true;

while (running) {

Search();

running = Menu();

}

}

**[--- Конец программы ---]**