МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

**Лабораторная работа №9**

по дисциплине: ООП

тема Использование стандартной библиотеки шаблонов STL»

Выполнил: студент группы ВТ-222

Маркевич Александр Александрович

Белгород 2024 г.

**Цель работы:** знакомство со стандартной библиотекой шаблонов в С++; получение навыков использования классов контейнеров, итераторов, алгоритмов.

**Вариант 2.**

Разработать программное обеспечение для решения следующей задачи: организовать поиск по html странице. Построить словарь всех слов, встречающихся на заданной html страницы в следующем виде map <key, value>, где value представляет собой объект класса с поля количества встречаемости слов и ссылки на них, key искомое слово. организовать неточный поиск, использовать ::iterator. Выполнить слияние нескольких map.

Файл HTML:

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>9</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<h1>Слово 1</h1>

<h1>Слово 2</h1>

<h1>Слово 2</h1>

<h1>Слово 3</h1>

<h1>Слово 3</h1>

<h1>Слово 3</h1>

</body>

</html>

Код программы:

#include <iostream>

#include <map>

#include <string>

#include <fstream>

#include <vector>

using namespace std;

#define FILENAME "index.txt"

class value

{

private:

int count;

string name;

public:

value() : count(0), name("") {}

value(int count, string name)

{

this->count = count;

this->name = name;

}

int getCount()

{

return this->count;

}

string getName()

{

return this->name;

}

string \*getNameAdress()

{

return &this->name;

}

void incrementCount(int incrementValue = 1)

{

count += incrementValue;

}

};

class FileReader

{

protected:

string line;

ifstream in;

public:

FileReader(const string &filename, vector<string> &storage)

{

in.open(filename);

if (in.is\_open())

{

while (getline(in, line))

{

storage.push\_back(line);

}

}

in.close();

}

};

class Parser

{

private:

string key;

map<string, value> result;

int count = 0;

public:

map<string, value> parserFile(const vector<string> &storage)

{

bool found = false;

for (int i = 0; i < storage.size(); i++)

{

for (int j = 0; j < storage[i].size(); j++)

{

if (storage[i].substr(j, 4) == "<h1>")

{

found = true;

for (int z = j + 4; z < storage[i].size(); z++)

{

if (storage[i][z] != '<')

{

key += storage[i][z];

}

if (key != "" && storage[i][z] == '<')

{

if (result[key].getName() != "")

{

count += 1;

}

else

{

count = 1;

}

result[key] = value(count, key);

key = "";

break;

}

}

}

}

}

if (!found)

{

cerr << "No <h1> tag found!" << endl;

}

return result;

}

};

class MapProcessing

{

private:

map<string, value> result;

void printMap(map<string, value>::iterator it)

{

cout << "Key: " << it->first << "\n"

<< "Count: " << it->second.getCount() << "\n"

<< "Adress: " << it->second.getNameAdress() << "\n"

<< endl;

}

public:

MapProcessing(map<string, value> stories)

{

this->result = stories;

}

void getMap()

{

for (map<string, value>::iterator it = result.begin(); it != result.end(); ++it)

{

printMap(it);

}

}

void search(string key)

{

for (map<string, value>::iterator it = result.begin(); it != result.end(); ++it)

{

if (it->first == key)

{

printMap(it);

return;

}

}

cout << "Key not found!" << endl;

}

void mergingPairs(string key1, string key2)

{

if (result.find(key1) != result.end() && result.find(key2) != result.end())

{

result[key1].incrementCount(result[key2].getCount());

result.erase(key2);

cout << "Pair merged" << endl;

return;

}

cout << "Keys not found!" << endl;

}

};

int main()

{

vector<string> storage;

FileReader reader(FILENAME, storage);

Parser parser = Parser();

MapProcessing processor = MapProcessing(parser.parserFile(storage));

processor.getMap();

processor.search("Сидоров");

processor.search("Слово 2");

processor.mergingPairs("Сидоров", "Слово 2");

processor.mergingPairs("Слово 2", "Слово 3");

processor.getMap();

return 0;

}

Вывод программы:

Key: Слово 1

Count: 1

Adress: 0x109bebc

Key: Слово 2

Count: 2

Adress: 0x109be1c

Key: Слово 3

Count: 3

Adress: 0x109be6c

Key not found!

Key: Слово 2

Count: 2

Adress: 0x109be1c

Keys not found!

Pair merged

Key: Слово 1

Count: 1

Adress: 0x109bebc

Key: Слово 2

Count: 5

Adress: 0x109be1c

**Вывод**. знакомство со стандартной библиотекой шаблонов в С++; получение навыков использования классов контейнеров, итераторов, алгоритмов.