|  |  |
| --- | --- |
| *voenmeh* | МИНОБРНАУКИ РОССИИ  федеральное государственное бюджетное образовательное учреждение  высшего образования  **«Балтийский государственный технический университет «ВОЕНМЕХ» им. Д.Ф. Устинова»**  **(БГТУ «ВОЕНМЕХ» им. Д.Ф. Устинова»)** |
| БГТУ.СМК-Ф-4.2-К5-02 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Факультет |  | О |  | Естественнонаучный |
|  |  | шифр |  | наименование |
| Кафедра |  | О7 |  | Информационные системы и программная инженерия |
|  |  | шифр |  | наименование |
| Дисциплина |  | Информационные технологии и программирование | | |

|  |
| --- |
| Практическая работа №4 |
| STL |
| Вариант №25 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Выполнил студент группы | | | |  | И924Б |
| Фокин В. К. | | | | | |
| Фамилия И.О. | | | | | |
| **Преподаватель:** | | | | | |
| Гладевич А. А. | |  |  | | |
| Фамилия И.О. Подпись | | | | | |
| Оценка |  | | | |  |
| «\_\_\_\_\_» |  | | | | 2023г. |

САНКТ-ПЕТЕРБУРГ

2023 г.

**Постановка задачи**

Дан файл с английскими словами, разделёнными знаком '$'. Используя std::getline или regex составить множество (std::set) из уникальных слов в файле. Слова хранить в std::string. Также составить словарь (std::map) по следующему принципу: Ключ - число, значение - частота встречи в файле слов, из указанного числа букв Сгенерировать список из 100 структур со следующими полями: Название - случайное слово из множества Основной параметр - используя std::advance получить значения трёх случайных элементов из словаря и перемножить их Вторичный параметр - квадрат значения из словаря, связанного с названием(в случае буквы брать первую букву названия) Используя функции из библиотеки algorithms Отсортировать список по убыванию основного параметра Используя std::copy\_if, создать новый список, содержащий только: Элементы, содержащие максимальное значение вторичного параметра Сгенерировать на основе полученного отфильтрованного списка файл формата json. Добавить возможность загрузки файла формата json, создания на основе данных в нём списка, перетасовки (shuffle) его и отображения полученного результата.

**Текст программы**

programm.cpp

#include <iostream>

#include <fstream>

#include <cstdlib>

#include <ctime>

#include <iomanip>

#include <algorithm>

#include <set>

#include <map>

#include <vector>

#include <string>

#include <random>

#include <iterator>

#include <Windows.h>

#include "include/nlohmann/json.hpp"

using namespace std;

struct Struct {

string name;

int mainPar;

int auxPar;

};

bool compareMain(const Struct& a, const Struct& b) {

return a.mainPar > b.mainPar; //сравниваем вторичные параметры в убывающем порядке

}

int main() {

SetConsoleOutputCP(1251);

srand(time(nullptr));

ifstream file("C:\\it'n'pr\\it4\\programm\\25.txt");

if (!file.is\_open()) {

cout << "Ошибка при открытии файла" << endl;

return 1;

}

//создание множества

set<string> words;

string line1;

while (getline(file, line1, '$')) {

words.insert(line1);

}

cout << "Множество создано" << endl;

file.close();

//создание словаря

map<int, int> frequency;

map<int, int> count;

string line2;

file.open("C:\\it'n'pr\\it4\\programm\\25.txt");

while (getline(file, line2, '$')) {

int length = line2.length();

if (frequency.find(length) == frequency.end()) {

frequency[length] = 1;

}

else {

frequency[length]++;

}

}

cout << "Словарь создан" << endl;

for (const auto& word : words) {

int length = word.length();

if (count.find(length) == count.end()) {

count[length] = 1;

}

else {

count[length]++;

}

}

//создание списка структур

vector<Struct> List;

for (int i = 0; i < 100; i++) {

Struct strct;

//выбор случайного слова из множества

int randInd = rand() % words.size();

auto it = next(words.begin(), randInd);

strct.name = \*it;

//получение трех случайных элементов из словаря

auto freqSize = frequency.size();

auto randomOffset1 = rand() % freqSize;

auto randomOffset2 = rand() % freqSize;

auto randomOffset3 = rand() % freqSize;

auto it1 = frequency.begin();

auto it2 = frequency.begin();

auto it3 = frequency.begin();

advance(it1, randomOffset1);

advance(it2, randomOffset2);

advance(it3, randomOffset3);

//вычисление основного параметра

strct.mainPar = it1->second \* it2->second \* it3->second;

//вычисление вторичного параметра

int length = strct.name.length();

strct.auxPar = count[length] \* count[length];

List.push\_back(strct);

sort(List.begin(), List.end(), compareMain);

}

cout << "-------------------------------------------------" << endl;

cout << "| Список, отсортированный по убыванию |" << endl;

cout << "| основного параметра |" << endl;

cout << "-------------------------------------------------" << endl;

cout << "| Название | Основной | Вторичный |" << endl;

cout << "-------------------------------------------------" << endl;

for (const auto& strct : List) {

cout << "| " << left << setw(16) << strct.name << "| " << left << setw(11) << strct.mainPar << "| " << left << setw(9) << strct.auxPar << "|" << endl;

}

cout << "-------------------------------------------------" << endl;

system("pause");

int maxAuxPar = List[0].auxPar;

for (const auto& strct : List) {

if (strct.auxPar > maxAuxPar) {

maxAuxPar = strct.auxPar;

}

}

vector<Struct> newList;

copy\_if(List.begin(), List.end(), back\_inserter(newList), [maxAuxPar](const Struct& strct) {

return strct.auxPar == maxAuxPar; });

cout << "-------------------------------------------------" << endl;

cout << "| Новый список, содержащий максимальные |" << endl;

cout << "| значения вторичного параметра |" << endl;

cout << "-------------------------------------------------" << endl;

cout << "| Название | Основной | Вторичный |" << endl;

cout << "-------------------------------------------------" << endl;

for (const auto& strct : newList) {

cout << "| " << left << setw(16) << strct.name << "| " << left << setw(11) << strct.mainPar << "| " << left << setw(9) << strct.auxPar << "|" << endl;

}

cout << "-------------------------------------------------" << endl;

system("pause");

nlohmann::json jsonObj;

for (const auto& strct : newList) {

nlohmann::json item;

item["name"] = strct.name;

item["mainPar"] = strct.mainPar;

item["auxPar"] = strct.auxPar;

jsonObj.push\_back(item);

}

ofstream File("output.json");

if (!File.is\_open()) {

cout << "Ошибка при открытии файла для записи" << endl;

return 1;

}

File << setw(4) << jsonObj << endl;

File.close();

cout << "Файл output.json успешно создан" << endl;

//загрузка файла формата JSON

ifstream inputFile("output.json");

if (!inputFile.is\_open()) {

cout << "Ошибка при открытии файла" << endl;

return 1;

}

//чтение JSON-объекта из файла

nlohmann::json jsonobj;

inputFile >> jsonobj;

inputFile.close();

//создание списка на основе данных из JSON-объекта

vector<Struct> list;

for (const auto& item : jsonobj) {

Struct strct;

strct.name = item["name"];

strct.mainPar = item["mainPar"];

strct.auxPar = item["auxPar"];

list.push\_back(strct);

}

//перетасовка

random\_device rd;

default\_random\_engine generator(rd());

shuffle(list.begin(), list.end(), generator);

//вывод

cout << "-------------------------------------------------" << endl;

cout << "| Перетасованный список |" << endl;

cout << "-------------------------------------------------" << endl;

cout << "| Название | Основной | Вторичный |" << endl;

cout << "-------------------------------------------------" << endl;

for (const auto& strct : list) {

cout << "| " << left << setw(16) << strct.name << "| " << left << setw(11) << strct.mainPar << "| " << left << setw(9) << strct.auxPar << "|" << endl;

}

cout << "-------------------------------------------------" << endl;

return 0;

}

**Результат работы программы**

При запуске программы создается множество и словарь, затем создается список из 100 структур и сортируется по убыванию основного параметра, что показано на рисунках 1, 2 и 3.

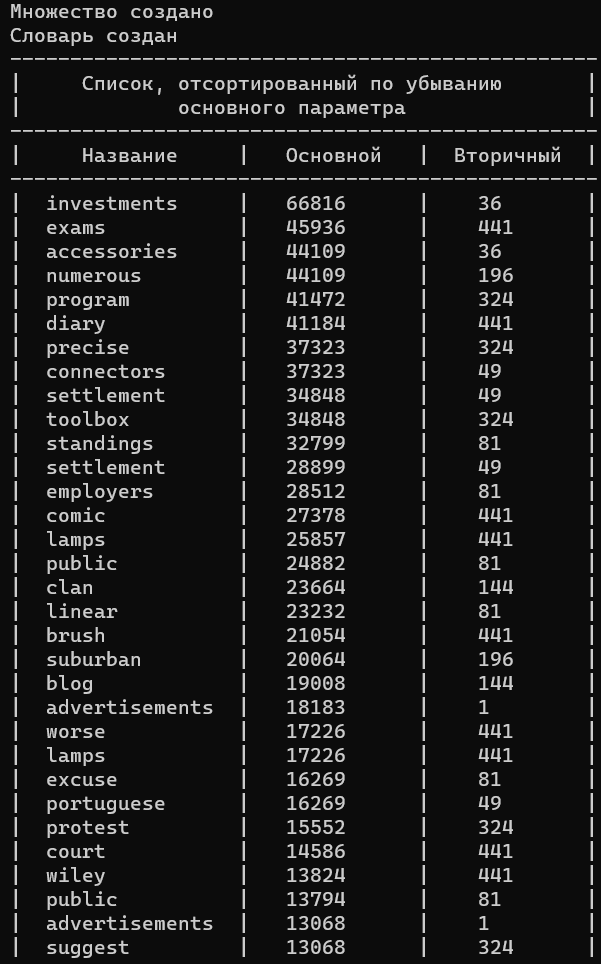


Рисунок 1 − Запуск программы

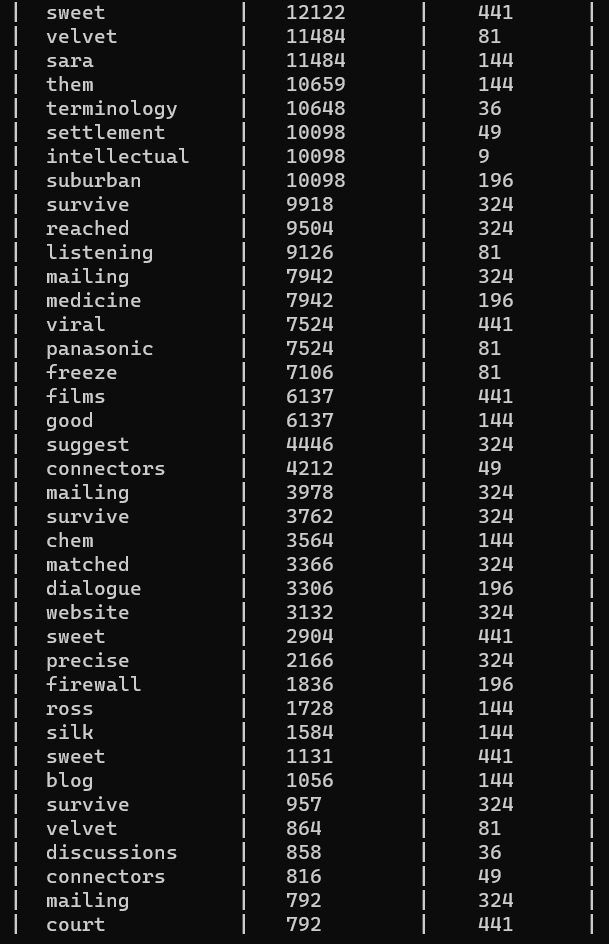


Рисунок 2 − Запуск программы

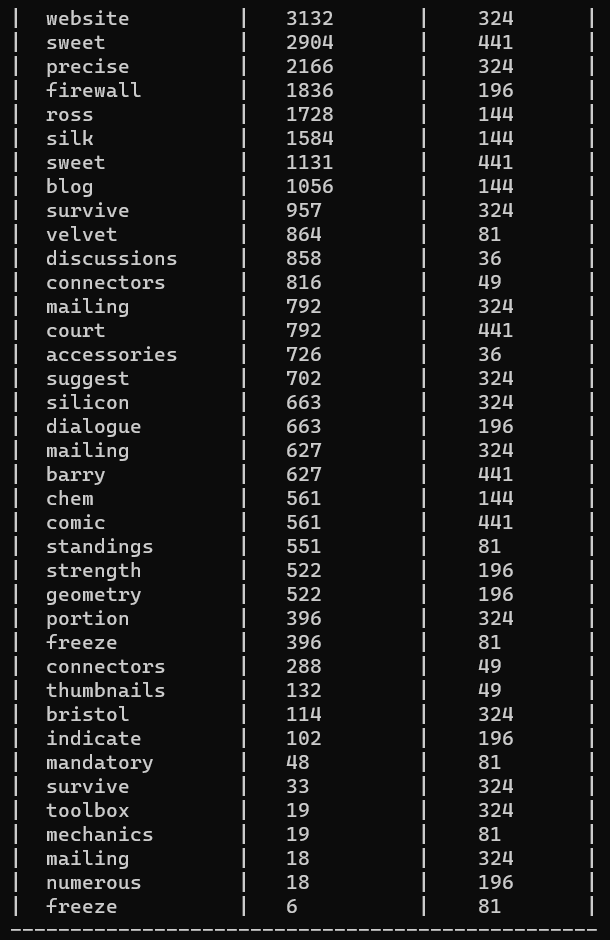


Рисунок 3 − Запуск программы

Далее выводится новый список, содержащий максимальные значения вторичного параметра, что показано на рисунке 4.

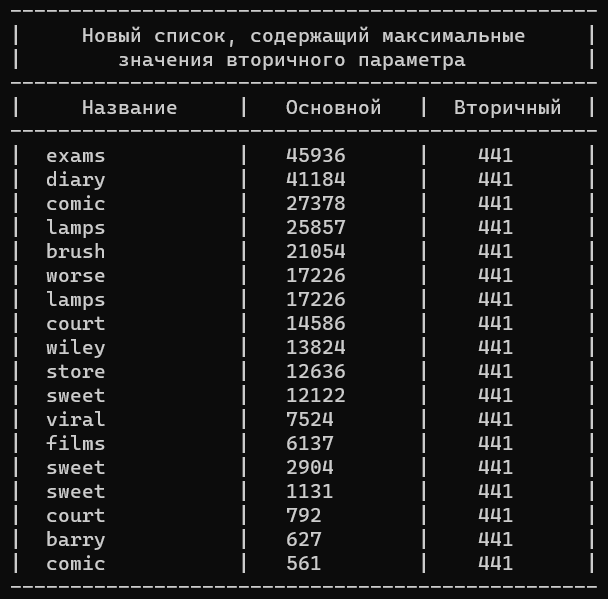


Рисунок 4 − Вывод нового списка

Затем на основе полученного списка создается файл формата .json, этот файл загружается в программу и список перетасовывается. В конце программа выводит перетасованный список, что показано на рисунке 5.

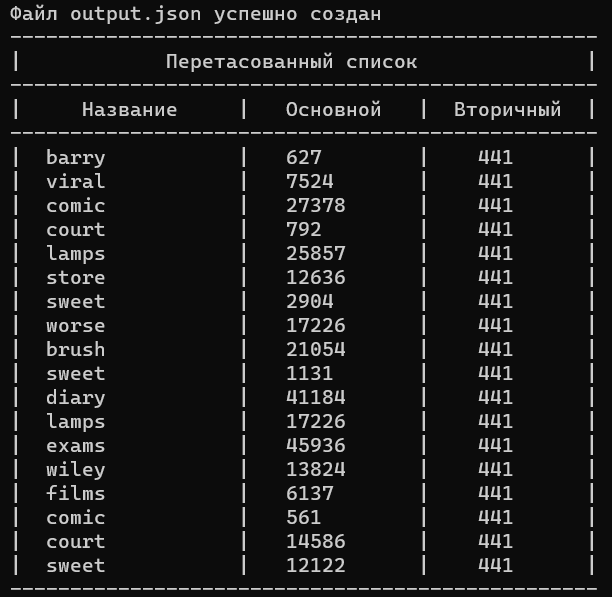


Рисунок 5 − Вывод перетасованного списка