Министерство образования и науки Российской Федерации

Новосибирский государственный технический университет

Кафедра прикладной математики

Лабораторная работа №1

По дисциплине: «Сравнительный анализ языков программирования»

Факультет: ПМИ

Группа: ПМ-43

Студенты: Иост А.В.

Мошкина А.Д.

Преподаватель: Рояк М.Э.

Ступаков И.М.

1. Вариант: 5

Новосибирск

2017

## Цель работы

Разработать программу для решения поставленной задачи на языке С++, удовлетворяющую следующим требованиям:

* Программа должна использовать для ввода-вывода потоки STL
* Программа не должна содержать собственных реализаций стандартных алгоритмов и структур данных, а использовать существующие в STL
* Размер каждой подпрограммы не должен превышать 10 строк

Протестировать разработанную программу. Исследовать асимптотические свойства разработанной программы на системе тестов с возрастающей размерностью.

## Задание

Задан большой текст (книга). Для наиболее часто встречающегося слова найти N наиболее часто встречающихся пар слов.

**Алгоритм решения**

1.Читаем книгу из файла.

2.Для каждого уникального слова книги находим его частоту встречаемости и берём максимальное по частоте, последнее слово – искомое (ключевое слово).

3.Для данного слова находим уникальные пары слов, содержащие данное слово, считаем их количество в тексте и сортируем по убыванию, первая пара – искомая.

4.Вывести все первые N наиболее встречающихся пар слов, содержащих ключевое слово.

**Текст программы**

#include <fstream>

#include <string>

#include <map>

#include <vector>

#include <functional>

#include <ctime>

#include <algorithm>

using namespace std;

void input(vector <string>& book)

{

ifstream fin("numb\_texts.txt");

int numb;

fin >> numb;

fin.close();

switch (numb)

{

case 1: fin.open("text1.txt"); break;

case 2: fin.open("text2.txt"); break;

case 3: fin.open("text3.txt"); break;

case 4: fin.open("text4.txt"); break;

case 5: fin.open("text5.txt"); break;

default: fin.open("text1.txt");

}

string word;

//запись книги

while (fin >> word)

book.push\_back(word);

fin.close();

}

//поиск наиболее часто встречающегося слова

pair<string,int> find\_often\_word(const vector <string>& book, multimap <int, string>& repeat\_words)

{

map <string, int> repeat;

//проход по всей книге

for (auto i : book)repeat[i]++;

pair<string, int> res("", 0);

for (auto p : repeat)if (p.second > res.second)res = p;

return res;

}

//поиск наиболее часто встречающихся пар слов

void find\_pairs(multimap <int, pair<string, string>, greater<>>& often\_pair, vector <string>& book, pair<string, int> word)

{

map <pair<string, string>, int> pairs;

if (\*begin(book) == word.first)

pairs[make\_pair(word.first, \*(begin(book) + 1))]++;

if (\*(end(book)-1) == word.first)

pairs[make\_pair(\*(end(book) -2),word.first)]++;

//проход по всей книге

for (auto i = begin(book)+1; i < end(book)-1; i++)

//если выбранное слово совпадает с ключевым

if (\*i == word.first) {

pairs[make\_pair(word.first, \*(i+1))]++;

pairs[make\_pair(\*(i - 1),word.first)]++;

}

//проходя по всему словарю меняем местами ключ и значение

for (auto i : pairs)

often\_pair.insert({ i.second, i.first });

}

void output(pair<string, int> word, multimap <int, pair<string, string>, greater<>> often\_pair, float result\_time)

{

int n, j = 0;

ifstream fin("n.txt");

ofstream fout("output.txt");

fin >> n;

fin.close();

fout << "Most often words" << endl << word.first << '\t' << word.second << endl << endl;

fout << "Most often pairs of words and numb of their" << endl;

for (auto i = begin(often\_pair); i != end(often\_pair) && j < n; j++, i++)

fout << i->second.first << ' ' << i->second.second << '\t' << i->first << endl;

fout << endl << "time" << endl << result\_time << endl;

fout.close();

}

void main()

{

vector <string> book;

multimap <int, string> repeat\_words;

multimap <int, pair<string, string>, greater<>> often\_pair;

input(book);

double beg\_time = clock();

pair<string,int> word = find\_often\_word(book, repeat\_words);

find\_pairs(often\_pair, book, word);

double end\_time = clock();

double result\_time = (end\_time - beg\_time)/\* / CLOCKS\_PER\_SEC\*/;

output(word, often\_pair, result\_time);

book.clear();

repeat\_words.clear();

often\_pair.clear();

}

**Генератор текста**

#include <fstream>

#include <string>

#include <vector>

#include <ctime>

#include <algorithm>

using namespace std;

void input(vector <char>& letters)

{

ifstream fin("words.txt");

char letter;

while (fin >> letter)

letters.push\_back(letter);

fin.close();

}

void generator(vector <char> letters)

{

long int n;

int numb\_letter;

string word;

vector <string> book;

ifstream fin("n.txt");

fin >> n;

fin >> numb\_letter;

fin.close();

ofstream fout("text.txt");

word.resize(numb\_letter, ' ');

book.resize(n);

// автоматическая рандомизацияр генератора случайных чисел

//позволяет каждый раз получать рандомные числа, отличные от предыдущего запуска

srand(0);

for (int i = 0; i < n; ) {

for (int k = 0; k < numb\_letter; k++) {

int j = 0 + rand() % (letters.size() - 1);

word[k] = letters[j];

}

int s = 1 + rand() % (letters.size());

for (int k = 0; k < s && i < n; k++) {

book[i] = word;

i++;

}

}//перемешивает слова в тексте

random\_shuffle(book.begin(), book.end());

for (int i = 0; i < n; i++)

if (i % 10 == 0 && i) fout << book[i] << endl;

else fout << book[i] << ' ';

fout.close();

}

void main()

{

vector <char> letters;

input(letters);

generator(letters);

}

**Исследования**

*Влияние количества слов с тексте на время выполнения программы*

*Все слова текста состоят из 5 букв.*

|  |  |  |
| --- | --- | --- |
| Количество слов в тексте | Время выполнения, мс | tn/t(n-1) |
| 1 000 | 1 | - |
| 10 000 | 3 | 3,00 |
| 100 000 | 32 | 10,67 |
| 1 000 000 | 545 | 17,03 |
| 10 000 000 | 11864 | 21,77 |

*Влияние количества букв в слове на время выполнения программы*

*Для исследования был взят текст на 100 000 слов.*

|  |  |  |
| --- | --- | --- |
| Количество букв в слове | Время выполнения, мс | tn/t(n-1) |
| 5 | 33 | - |
| 10 | 33 | 1,00 |
| 15 | 38 | 1,15 |
| 20 | 51 | 1,34 |
| 30 | 52 | 1,02 |

*Влияние количества повторений слов с тексте на время выполнения программы*

*Для исследования был взят текст на 100 000 слов, каждое слово из 5 букв.*

|  |  |  |
| --- | --- | --- |
| Количество повторений слова в тексте | Время выполнения, мс | tn/t(n-1) |
| 100 | 35 | - |
| 1 000 | 32 | 0,91 |
| 5 000 | 39 | 1,22 |
| 10 000 | 46 | 1,18 |
| 50 000 | 77 | 1,67 |

**Вывод**

Исходя из результатов исследования, можно сделать вывод о том, что происходит нечто странное, что мы понять не в силах. Этакая компьютерная магия.