**Задание 1.** **Используя контейнер list реализовать программу – словарь.**

Основные возможности:

1. поиск слова;
2. добавление, удаление, редактирование слов;
3. чтение и запись словаря в файл.

**Задание 2. Программа «Статистика»**

Подсчитать, сколько раз каждое слово встречается в заданном тексте. Результат записать в контейнер **map<TKey, TValue>**. Текс использовать из приложения 1. Вывести статистику по тексту в виде таблицы (рисунок 1).

**Приложение 1.**

Вот дом, Который построил Джек. А это пшеница, Которая в темном чулане хранится В доме, Который построил Джек. А это веселая птица-синица, Которая часто ворует пшеницу, Которая в темном чулане хранится В доме, Который построил Джек.

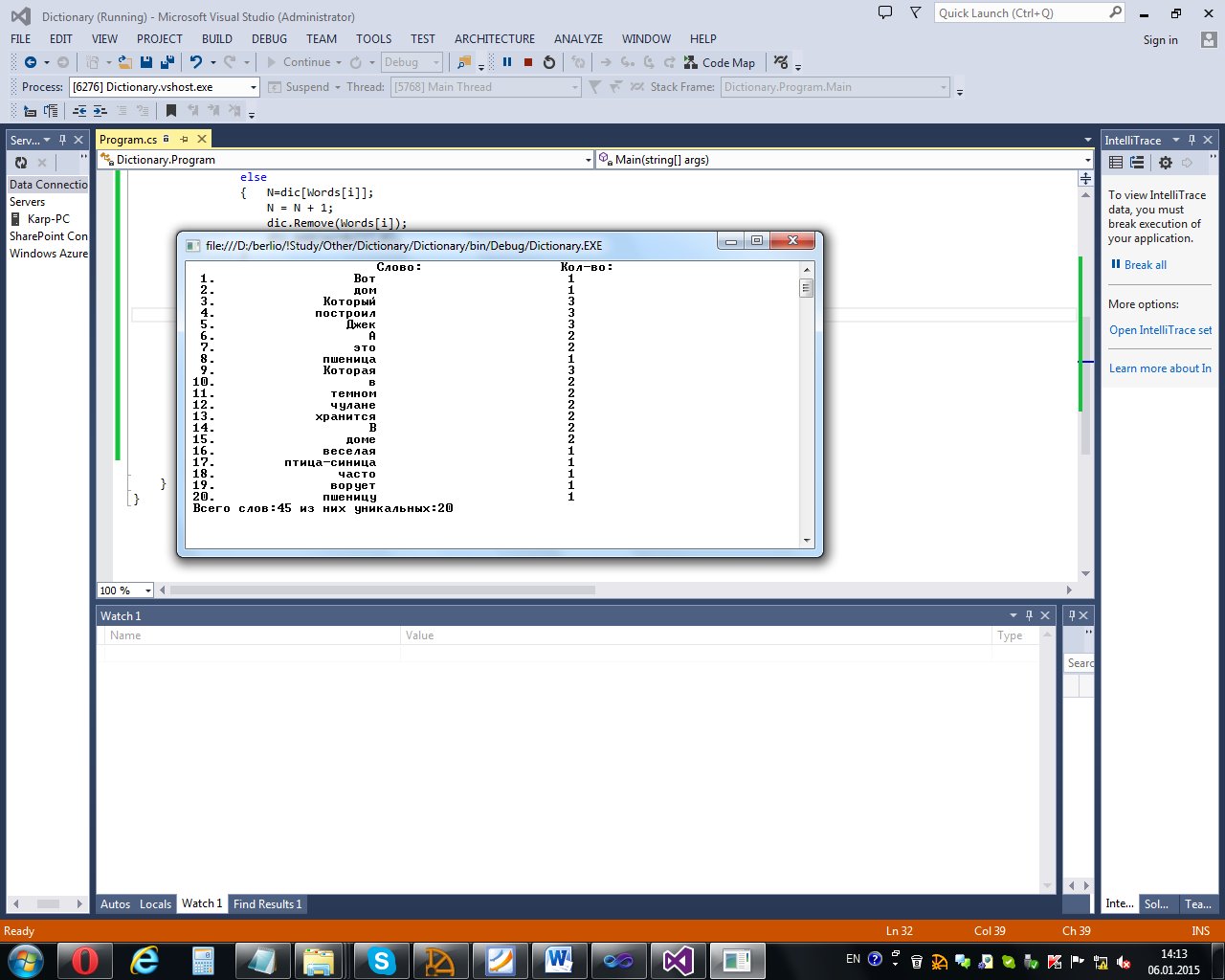


Рисунок 1.

Завдання 1

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include<string>

#include<list>

#include<Windows.h>

#include<fstream>

using namespace std;

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

list<string>dictionary;

string word;

fstream f;

int a;

bool b;

cout << "Додати слово ->1, видалити слово ->2, редагувати слово ->3, знайти слово ->4, зчитати з файлу ->5, зберегти у файл ->6, показати весь список ->7, вихід ->0\n";

cin >> a;

while (a)

{

switch (a)

{

case 1:

cout << "Введіть слово ->";

cin >> word;

dictionary.push\_back(word);

break;

case 2:

cout << "Введіть слово для видалення ->";

cin >> word;

b = false;

for (auto iter = dictionary.begin(); iter != dictionary.end(); iter++)

{

if (\*iter == word)

{

dictionary.erase(iter);

b = true;

break;

}

}

if (!b)

cout << "Не знайдено!\n";

break;

case 3:

cout << "Введіть слово для редагування ->";

cin >> word;

b = false;

for (auto iter = dictionary.begin(); iter != dictionary.end(); iter++)

{

if (\*iter == word)

{

cout << "Введіть відредаговане слово ->";

cin >> word;

\*iter = word;

b = true;

break;

}

}

if (!b)

cout << "Не знайдено!\n";

break;

case 4:

cout << "Введіть слово для пошуку ->";

cin >> word;

b = false;

for (auto iter = dictionary.begin(); iter != dictionary.end(); iter++)

{

if (\*iter == word)

{

cout << "Слово знайдено!\n";

b = true;

break;

}

}

if (!b)

cout << "Не знайдено!\n";

break;

case 5:

f.open("E:\\TEST.txt", ios::in);

while (f>>word)

{

dictionary.push\_back(word);

}

f.close();

cout << "Успішно зчитано і додано до словника\n";

break;

case 6:

f.open("E:\\TEST.txt", ios::out);

for (auto iter = dictionary.begin(); iter != dictionary.end(); iter++)

{

f << \*iter << endl;

}

f.close();

cout << "Успішно збережено у файл\n";

break;

case 7:

for (auto iter = dictionary.begin(); iter != dictionary.end(); iter++)

{

cout << \*iter << endl;

}

break;

}

cout << "Додати слово ->1, видалити слово ->2, редагувати слово ->3, знайти слово ->4, зчитати з файлу ->5, зберегти у файл ->6, показати весь список ->7, вихід ->0\n";

cin >> a;

}

return 0;

}

Завдання 2

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include<string>

#include<map>

#include<Windows.h>

using namespace std;

void parseString(const string& txt, map<string,int>& tree)

{

char\* copy = new char[txt.size() + 1];

strcpy(copy, txt.c\_str());

char separator[] = "\n ,!.?;:";

char\* pstr;

pstr = strtok(copy, separator);

while (pstr != NULL)

{

auto w = tree.find(pstr);

if (w != tree.end())

w->second++;

else

tree[pstr] = 1;

pstr = strtok(NULL, separator);

}

delete[]copy;

}

void Static(map<string, int>& tree)

{

int words = 0, uniq\_words = 0, i=1;

cout << "-------------------------------------------\n";

cout << " Слово Кількість\n";

for (auto iter = tree.begin(); iter != tree.end(); iter++)

{

cout << i << ". " << iter->first << "\t\t" << iter->second << endl;

cout << "-------------------------------------------\n";

words += iter->second;

if (iter->second == 1)

uniq\_words++;

i++;

}

cout << "Всього слів: " << words << " з них унікальних: "<<uniq\_words<<endl;

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

map<string,int> tree;

string a;

cout << "Введіть текст:\n";

getline(cin,a);

parseString(a, tree);

Static(tree);

return 0;

}