Московский государственный технический

университет им. Н.Э. Баумана

Курс «Основы информатики»

Отчёт по лабораторной работе №9

Ввод, сортировка и двоичный поиск в массиве структур.

Проверил:

Преподаватель каф. ИУ5

Аксёнова М.В.

Подпись и дата:

Факультет «Информатика и система управления»

Кафедра ИУ5 «Система обработки информации и управления»

Выполнил:

Студент группы ИУ5-13Б

Пермяков Дмитрий Кириллович

Подпись и дата:

**Постановка задачи.**

Англо-русский словарь построен в виде массива структур Dictionary. Структура содержит английское слово и соответствующее ему русское слово. Максимальный размер словаря – 100 пар слов.

Разработать программу, которая:

* обеспечивает формирование словаря (добавление и удаление записей);
* записывает словарь, отсортированный по английским значениям слов, в файл;
* обеспечивает просмотр словаря;
* выполняет перевод слов с английского на русский, используя для поиска слова в словаре метод двоичного поиска в отсортированном массиве;
* выполняет перевод слов с русского на английский, используя для поиска слов в словаре метод перебора;

Программа должна обеспечивать диалог с помощью меню.

Начальное число слов в словаре равно 10.

Для исключения проблем, связанных с вводом кириллицы, вводите русские слова латинскими буквами, например: kot (cat), sobaka(dog) и т.п.

Переменные: d\* words- массив структур, size\_t size- кол-во слов в словаре, char word[30]- введённое слово, bool flag- помогает найти слово бинарным писком, char\* str- вспомогательный массив, size\_t number\_of\_menu- ввыдёный пункт меню с клавиатуры

Функции:

void Add- вводим слово на англ и рус и закидываем в структурный массив

void Delete- вводим слово, которое хотим удалить из словаря. Находим его индекс в структурном массиве и заменяем его следующим словом

void TranslateToRus- сортирую массив, нахожу индекс введённого слова на англ бинарным поиском и вывожу структурный массив с этим индексом на русский

void TranslateToEng- аналогичен TranslateToRus, только используем метод перебора

void Print- сортируем массив и печатаем, используя цикл for

void FillFile- аналогичен Print, только текст передаём в файл

size\_t read- используем char\* str- вспомогательный массив, в который записываются слово из блокнота, чтобы определить длину слова и скопировать в структурный массив words.eng и words.rus

size\_t Menu- меню для пользователя

void dict\_sort- сортируем массив words.eng благодаря функции strcmp

**Код программы**

**Заголовки:**

#include <iostream>

#include <fstream>

#include <cstring>

#include <algorithm>

#include <Windows.h>

using namespace std;

struct d

{

char\* eng, \* rus;

};

void Add(d\* words, size\_t& size);

void Delete(d\* words, size\_t& size);

void TranslateToRus(d\* words, size\_t size);

void TranslateToEng(d\* words, size\_t size);

void Print(d\* words, size\_t size);

void FillFile(d\* words, size\_t size);

size\_t read(d\* words);

size\_t Menu();

void dict\_sort(d\* dict, size\_t size);

Функции:

#include "Header.h"

void Add(d\* words, size\_t& size)

{

char word[100];

cout << "Enter an Eglish word: ";

cout << "\x1b[35m";

cin >> word;

cout << "\x1b[0m";

words[size].eng = new char[strlen(word) + 1];

copy(word, word + strlen(word) + 1, words[size].eng);

cout << "Enter a Russian word: ";

cout << "\x1b[35m";

cin >> word;

cout << "\x1b[0m";

words[size].rus = new char[strlen(word) + 1];

copy(word, word + strlen(word) + 1, words[size].rus);

++size;

cout << endl;

}

void Delete(d\* words, size\_t& size)

{

dict\_sort(words, size);

bool flag = false;

char word[30];

cout << "Enter the english word for deletion: ";

cout << "\x1b[35m";

cin >> word;

cout << "\x1b[0m";

size\_t l = 0;

size\_t r = size;

size\_t middle;

while (l <= r)

{

middle = (l + r) / 2;

if (strcmp(words[middle].eng, word) == 0)

{

flag = true;

break;

}

if (strcmp(words[middle].eng, word) > 0)

r = middle - 1;

else

l = middle + 1;

}

if (flag)

{

delete[] words[middle].eng;

delete[] words[middle].rus;

for (size\_t i = middle; i < size - 1; i++)

{

words[i].eng = words[i + 1].eng;

words[i].rus = words[i + 1].rus;

}

--size;

}

else

cout << "\x1b[31mNot found\x1b[0m\n\n";

}

void TranslateToRus(d\* words, size\_t size)

{

dict\_sort(words, size);

bool flag = false;

char word[30];

cout << "Enter the english word for the translation: ";

cout << "\x1b[35m";

cin >> word;

cout << "\x1b[0m";

size\_t l = 0;

size\_t r = size;

size\_t middle;

while (l <= r)

{

middle = (l + r) / 2;

if (strcmp(words[middle].eng, word) == 0)

{

flag = true;

break;

}

if (strcmp(words[middle].eng, word) > 0)

r = middle - 1;

else

l = middle + 1;

}

if (flag)

cout << "\x1b[36mTranslation: " << words[middle].rus << "\x1b[0m\n\n";

else

cout << "\x1b[31mNot found\x1b[0m\n\n";

}

void TranslateToEng(d\* words, size\_t size)

{

char word[30];

cout << "Enter the russian word for the translation: ";

cout << "\x1b[35m";

cin >> word;

cout << "\x1b[0m";

bool flag = false;

for (size\_t i = 0; i < size; i++)

{

if (strcmp(words[i].rus, word) == 0)

{

cout << "\x1b[36mTranslation: " << words[i].eng << "\x1b[0m\n\n";

flag = true;

break;

}

}

if (!flag)

cout << "\x1b[31mNot found\x1b[0m\n\n";

}

void Print(d\* words, size\_t size)

{

dict\_sort(words, size);

cout << "OUR DICTIONARY: \x1b[36m\n\n";

for (size\_t i = 0; i < size; ++i)

cout << " " << words[i].eng << " - " << words[i].rus << endl;

cout << "\x1b[0m\n";

}

void FillFile(d\* words, size\_t size)

{

ofstream fin("dictionary.txt");

dict\_sort(words, size);

for (size\_t i = 0; i < size; ++i)

{

if (i != size - 1)

fin << words[i].eng << " - " << words[i].rus << endl;

else

fin << words[i].eng << " - " << words[i].rus;

}

}

size\_t read(d\* words)

{

setlocale(0, "rus");

ifstream f("dictionary.txt");

size\_t size = 0;

while (!f.eof())

{

char\* str = new char[100];

f >> str;

// создаем массив для англ слова

words[size].eng = new char[strlen(str) + 1];

copy(str, str + strlen(str) + 1, words[size].eng);

f >> str;

f >> str;

// создаем массив для англ слова

words[size].rus = new char[strlen(str) + 1];

copy(str, str + strlen(str) + 1, words[size].rus);

++size;

}

return size;

}

size\_t Menu()

{

cout << "Меню:\n";

cout << "1) добавление слов в словарь\n";

cout << "2) удаление слов из словаря\n";

cout << "3) перевод слов с английского на русский\n";

cout << "4) перевод слов с русского на английский\n";

cout << "5) просмотр словаря\n";

cout << "6) вывод словаря в файл\n";

cout << "7) выход\n";

cout << "Введите пункт меню: ";

size\_t number\_of\_menu;

cout << "\x1b[35m";

cin >> number\_of\_menu;

cout << "\x1b[0m";

cin.get();

cout << '\n';

return number\_of\_menu;

}

void dict\_sort(d\* words, size\_t size)

{

for (size\_t i = 1; i < size; i++)

{

for (size\_t i2 = 0; i2 < size - i; i2++)

{

if (strcmp(words[i2].eng, words[i2 + 1].eng) > 0)

swap(words[i2].eng, words[i2+1].eng);

}

}

}

MAIN:

#include "Header.h"

int main(){

SetConsoleCP(1251); SetConsoleOutputCP(1251);

setlocale(0, "rus");

d\* words = new d[100];

size\_t size = read(words);

while (true)

{

switch (Menu())

{

case 1:

Add(words, size);

break;

case 2:

Delete(words, size);

break;

case 3:

TranslateToRus(words, size);

break;

case 4:

TranslateToEng(words, size);

break;

case 5:

Print(words, size);

break;

case 6:

FillFile(words, size);

break;

case 7:

delete[] words;

return 0;

default:

cout << "\x1b[31mНадо вводить число от 1 до 7.\x1b[0m\n";

break;

}

}

}

**Тестирование:**

|  |  |
| --- | --- |
|  |  |