КУРСОВАЯ РАБОТА
Алгоритмы работы со словарями по дисциплине “Алгоритмы и структуры данных”

Выполнил Степанов Г. Ж.
студент гр. 3530904-90005

Руководитель Самочадина Т. Н.
старший преподаватель

Англо-русский словарь. Бинарное дерево поиска

Постановка задачи

Для разрабатываемого словаря реализовать основные операции:

INSERT (ключ, значение) –добавить запись с указанным ключом и значением

SEARCH (ключ)-найти запись с указанным ключом

DELETE (ключ)-удалить запись с указанным ключом

Предусмотреть обработку и инициализацию исключительных ситуаций, связанных, например, с проверкой значения полей перед инициализацией и присваиванием.

Программа должна быть написана в соответствии со стандартом программирования: C++ ProgrammingStyleGuidelines(http://geosoft.no/development/cppstyle.html).

Тесты должны учитывать как допустимые, так и не допустимые последовательности входных данных.

Разработать и реализовать алгоритм работы с англо-русским словарем, реализованным как бинарное дерево поиска.

Узел бинарного дерева поиска должен содержать:

Ключ –английское слово,

Информационная часть –ссылка на список, содержащий переводы английского слова, отсортированные по алфавиту (переводов слова может быть несколько).

1. Описание алгоритма решения и используемых структур данных

Разработан класс List со специальным набором методов, реализует двусвязный список для информационной части (ссылка на список, содержащий переводы английского слова, отсортированные по алфавиту (переводов слова может быть несколько))

Разработан класс EnglishRussianDictionary со специальным набором методов, реализующий словарь на основе бинарного дерева поиска с ключевой структурой.

Разработано консольное приложение на основе специальных методов класса EnglishRussianDictionary, предоставляющее пользователю набор из 6 функций для работы со словарем.

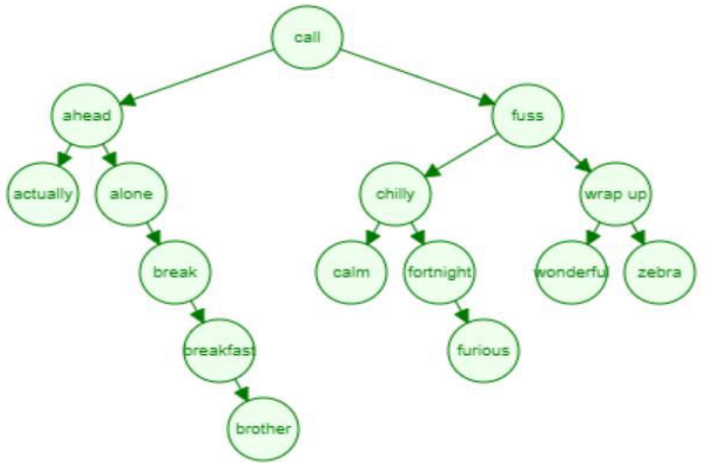
Класс List

```h
#ifndef DOUBLE_DIRECTION_LIST_HPP
#define DOUBLE_DIRECTION_LIST_HPP

#include <iostream>

class List
{
    class Node
    {
    public:
        std::string data_;
        Node* next_;
        Node* prev_;

        Node() :
            data_(""),
            next_(nullptr),
            prev_(nullptr)
        {
        }
    };

    Node* head_;
    std::size_t size_;

public:
    List();
    ~List();
    std::size_t getSize() const;
    List& deleteNode(const std::string data);
    List& operator+=(const std::string data);
    friend std::ostream& operator<<(std::ostream& out, const List& List);

private:
    Node* searchNodeAndReturn(const std::string data) const;
};

#endif
```

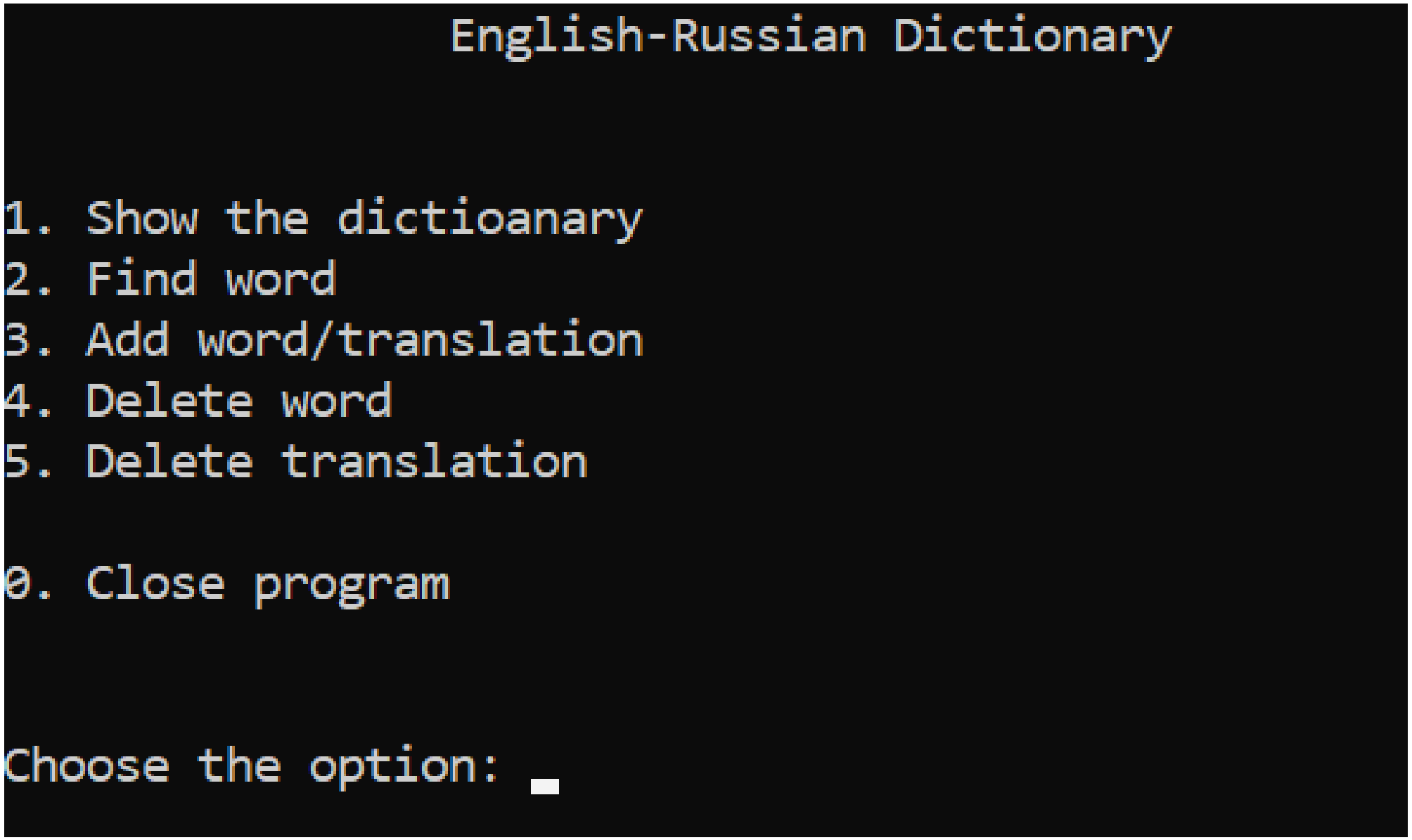
```
call - звать, навещать, называть
ahead - вперёд, впереди
actually - на самом деле, фактически
alone - один, одинокий
break - ломать, разбивать
breakfast - завтрак
brother - брат
fuss - волноваться, суетиться
chilly - зябко, прохладный, холодно
calm - спокойный
fortnight - две недели
furious - взбешённый
wrap up - кутаться
wonderful - замечательный
zebra - зебра
```

Класс EnglishRussianDictionary

```cpp
#ifndef BINARY_SEARCH_TREE_HPP
#define BINARY_SEARCH_TREE_HPP

#include <iostream>
#include "double_direction_list.hpp"

class EnglishRussianDictionary
{
    class Word
    {
    public:
        std::string key_;
        List translation_;
        Word* left_;
        Word* right_;
        Word* p_;

        Word() :
            key_(""),
            translation_(List()),
            left_(nullptr),
            right_(nullptr),
            p_(nullptr)
        {
        }
    };

    Word* root_;
    std::size_t size_;

public:
    EnglishRussianDictionary();
    ~EnglishRussianDictionary();
    void insertWord(const std::string key, const std::string translation);
    void printWord(const std::string key) const;
    void deleteWord(const std::string key);
    void deleteTranslation(const std::string key, const std::string translation);
    void print() const;
    bool checkWord(const std::string word, const std::string language) const;

private:
    Word* searchWordAndReturn(const std::string key) const;
    Word* searchSuccessorAndReturn(const std::string key) const;
    void deleteDictionary(Word* word);
    void printWordsStartingWith(Word* word) const;
};

#endif
```



Консольное приложение



2. Анализ алгоритма

EnglishRussianDictionary::Word* EnglishRussianDictionary::searchWordAndReturn(const std::string key) const - сложность поиска O(log(n));

void EnglishRussianDictionary::insertWord(const std::string key, const std::string translation) - сложность вставки O(log(n));

void EnglishRussianDictionary::deleteWord(const std::string key) - сложность удаления аппроксимирована к O(log(n));

Источник: https://sohabr.net/habr/post/442352/

3-4. Описание спецификации программы и самой программы

```
call - звать, навещать, называть
ahead - вперёд, впереди
actually - на самом деле, фактически
alone - один, одинокий
break - ломать, разбивать
breakfast - завтрак
brother - брат
fuss - волноваться, суетиться
chilly - зябко, прохладный, холодно
calm - спокойный
fortnight - две недели
furious - взбешённый
wrap up - кутаться
wonderful - замечательный
zebra - зебра
```

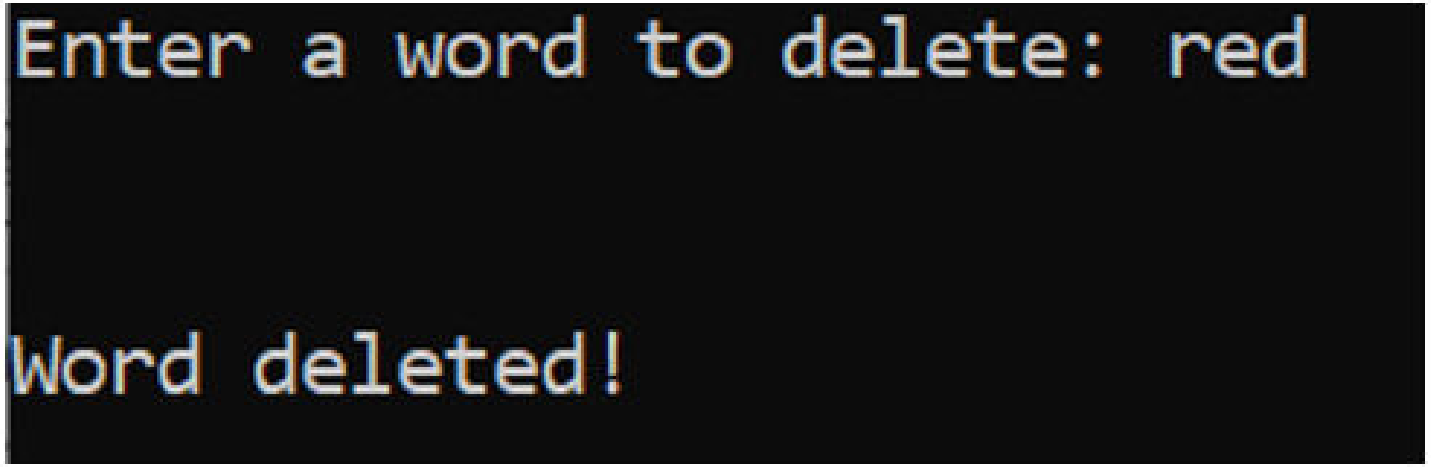
Функция 1. Show the dictionary

```
Enter a search word: call


Result: call - звать, навещать, называть
```

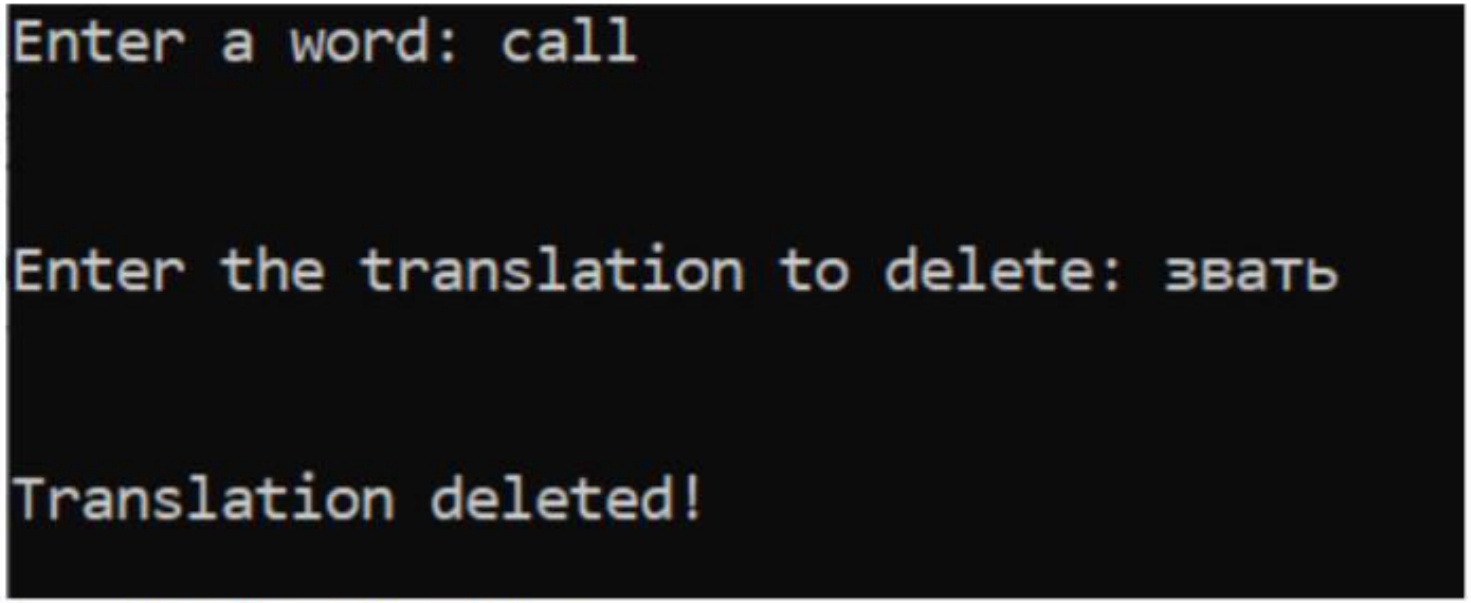
Функция 2. Find word

```
Enter a word to add: red


Enter the translation: красный


Word and translation added!
```

Функция 3. Add word/translation



Функция 4. Delete word



Функция 5. Delete translation

Заключение

В ходе работы мною были подробно изучены и реализованы в коде следующие понятия: двусвязный линейный список, бинарное дерево поиска, словарь. С помощью данной теории мною была разработана программа, реализующая англо-русский словарь.