Руководство программиста по программе «convertHtml»

Программа предназначена для создания HTML файлов, содержащие весь текст из файла с текстом, в тексте жирным и наклонным шрифтом выделены слова, которые есть в файле словаря.

Весь код программы выполнен на языке C++ в среде Linux (Ubuntu 14.04) и скомпилирован в программе Code::Blocks IDE.

Для запуска программы в средах отличных от Linux необходимо перекомпиляция исходных файлов.

Запуск программы происходит из консоли, первый аргумент — словарь, второй — текстовый файл для обработки, третий — по необходимости задает количество строк в выходных HTML файлах (по умолчанию = 100).

Далее приведен код из файлов программы:

1) main.cpp

```
#include "header.h"

int countStringHtml = 100;   //кол-во строк в выходном файле HTML

//Программа ConvertHtml
int main(int argc, char* argv[]) {
    try {
        Check ch(argc, argv);   //проверка введенных аргументов с консоли
    }
    catch (int e) {
        if (e==0) { cout << "Ошибка: в командной строке должно быть 2 или 3 аргумента, читайте документацию" << endl;  return -1; }
        if (e==1) { cout << "Ошибка: первый и второй файл д.б. с расширением .txt" << endl; return -1; }
        if (e==2) { cout << "3-й аргумент д.б. числом" << endl; return -1; }
    }
    set<string> setDictionary;  //множество для хранения словаря
    try {
        LoadDictionary dictionary(argv[1]);   //создаем объект словарь, передавая ссылку по аргументу
        setDictionary = dictionary.getDictionary(); //загружаем словарь
    }
    catch (int e) {
        if (e==0) { cout << "Ошибка, файл больше 2 Mb";  return -1; }
        if (e==1) { cout << "Ошибка, в словаре больше 100 000 строк"; return -1; }
    }
    catch (...) {
        cout << "Что-то пошло не так" ; return -1;
    }
    try {
        ConvertToHtml convert(setDictionary, argv[2]); //создаем объект convert, передавая ссылку по аргументу
        convert.modifyText();  //изменям текст из файла
        convert.buildHtml();   //генерируем HTML файлы
    }
```

```
    catch (int e) {
        if (e==0) { cout << "Ошибка, файл больше 2 Mb";  return -1; }
    }
     catch (...) {
        cout << "Что-то пошло не так" ; return -1;
    }
    cout << "Файлы HTML успешно сгенерированы" << endl;
    return 0;
}
```

2) header.h

```
#ifndef HEADER_H
#define HEADER_H

#include <set>
#include <vector>
#include <string>
#include <fstream>
#include <iostream>
#include <cctype>   // для функции isdigit
#include <cstdlib>  // для функции atoi
#include <sstream>

using namespace std;

const int LENGTH_FILE = 2097152;  //размер файла 2Mb в байтах

//Класс для загрузки словаря
class LoadDictionary {
    const static int LENGTH_STRING = 100000; //кол-во заданных строк, не более
    const char* fileName;   //имя файла словаря
public:
     LoadDictionary (const char* fileName);
     void chekMemorySize();
     set<string> getDictionary();
};

//Класс для генерации HTML файлов
class ConvertToHtml {
    set<string> setDict;
    vector<string> modText;
    const char* dict;
    const char* text;
public:
    ConvertToHtml(set<string> setDictionary, const char* text);
    void modifyText();
    void buildHtml();
    string Replace(string a,string b,string c);
    void chekMemorySize();
};

//Класс проверки введенных аргументов с консоли
class Check {
    const char* dict;
    const char* text;
    const char* valueHtml;
public:
    Check(int argc, char* argv[]);
    void checking();
};
```

```
#endif
```

3) function.cpp

```
#include "header.h"

extern int countStringHtml;

////В конструкторе проверяем введенные аргументы с консоли
Check::Check(int argc, char* argv[]) {
         if ((!(argc==3))&&(!(argc==4))) {
       throw 0;      //выбрасываем Exception
   }
   else if (argc==3) {
       this->dict = argv[1];
       this->text = argv[2];
       string buf1(dict);
       string buf2(text);
       buf1 = buf1.substr((buf1.length()-4), 4);
       buf2 = buf2.substr((buf2.length()-4), 4);
       if ((!(buf1==".txt"))||(!(buf2==".txt"))) {
          throw 1;      //выбрасываем Exception
       }
   }
    else if (argc==4) {
       valueHtml = argv[3];
       if (isdigit(valueHtml[0])) {    // если символы строки число
          countStringHtml = atoi (valueHtml); //изменяем кол-во строк в выходном файле
       }
       else throw 2;    //выбрасываем Exception
   }
}

//Конструктор класса LoadDictionary
LoadDictionary::LoadDictionary (const char* fileName) {
   this->fileName = fileName;
   chekMemorySize();
}

//Проверка размера памяти файла словаря
 void LoadDictionary::chekMemorySize() {
   fstream file(fileName);
   int size = 0;
   file.seekg (0, std::ios::end);
   size = file.tellg();
   file.close();
   if (size > LENGTH_FILE) {
       throw 0;      //выбрасываем Exception, если размер файла больше заданного
   }
 }

//Загрузка словаря
set<string> LoadDictionary::getDictionary() {
   set<string> setDictionary;  //множество Set
   string str;
   int stringValue = 0;    //кол-во строк в словаре
   ifstream in(fileName);
   if (in.is_open()) {
       while (!in.eof()) {
          getline (in, str);
          if (str.length()!=0) {
```

```
            setDictionary.insert(str);  //заполняем Set словарем
            stringValue++;
            if (stringValue > LENGTH_STRING) {
                throw 1;            //выбрасываем Exception, если кол-во строк больше
заданного
            }
        }
    }
    return setDictionary;
    in.close();
  }
  else cout << "Unable to open file";
  return setDictionary;
}

//Конструктор класса ConvertToHtml
ConvertToHtml::ConvertToHtml(set<string> setDictionary, const char* text) {
  this->setDict = setDictionary;
  this->text = text;
}

//Изменение текста из файла
void ConvertToHtml::modifyText() {
  string str, buf;
  ifstream in(text);
  if (in.is_open()) {
    while (!in.eof()) {
      getline(in, str); //читаем текст по строчно
      if (str.length()!=0) {
        set<string>::iterator it;
        string modStr;
        for (it=setDict.begin(); it!=setDict.end(); it++) {
          buf = *it;
          string buffer = " ";
          buffer.insert(1,buf); //добавим пробел к заменяемому слову
          string str1 = " <i><b>";
          string str2 = "</b></i> ";
          str1.insert(str1.length(), buf);
          str1.insert(str1.length(), str2);
          string strBuf = Replace(str, buffer, str1); //изменяем строку
          if (strBuf.length()>modStr.length()) {
            modStr = strBuf;
          }
        }
        modText.push_back(modStr);
      }
    }
    in.close();
  }
   else cout << "Unable to open file";
}

//Генерация HTML файлов
void ConvertToHtml::buildHtml() {
  const char * TEXT_BEGIN = "<!DOCTYPE html> \n<html>\n<head>\n<meta
charset=\"UTF-8\">\n<title>ConvertToHTML</title>\n</head>\n<body>";
      const char * TEXT_END = "</body>\n</html>";
      const char * fileName = "convertHTML.html";
  int currentCountStringHtml = 0;
  int countFile = 1;

      FILE * file = fopen(fileName, "w");
```

```
    if (file) { // если есть доступ к файлу,
        fputs(TEXT_BEGIN, file); // и записываем в файл HTML код

        vector<string>::iterator it;
        for (it=modText.begin(); it!=modText.end(); it++) {
            if (currentCountStringHtml==countStringHtml) { //если число строк в создаваевом
файле равно заданному создаем новый файл
                fputs(TEXT_END, file);
                fclose(file);
                countFile++;    //счетчик кол-ва выходных файлов
                ostringstream convert;  //stream used for the conversion
                convert << countFile;
                string number = convert.str();
                string name = "convertHTML_";
                name.insert(name.length(), number);
                name.insert(name.length(), ".html");
                const char * fileName = name.c_str();
                file = fopen(fileName, "w");
                fputs(TEXT_BEGIN, file);
                currentCountStringHtml=0;
            }
            string str = *it;
            str.insert(str.length(), "</br>");
            const char * c = str.c_str();
            fputs(c, file);
            currentCountStringHtml++;
        }
        fputs(TEXT_END, file);  // и записываем в файл HTML код
    }
    else cout << "Unable to open file" << endl;
    fclose(file);
}

//Замена слов по словарю в строке
string ConvertToHtml::Replace(string a,string b,string c) {
    int pos;
    do {
        pos = a.find(b);
        if (pos!=-1)  a.replace(pos, b.length(), c);
    }
    while (pos!=-1);
    return a;
}

//Проверка размера памяти файла с текстом
 void ConvertToHtml::chekMemorySize() {
    fstream file(text);
    int size = 0;
    file.seekg (0, std::ios::end);
    size = file.tellg();
    file.close();
    if (size > LENGTH_FILE) {
        throw 0;      //выбрасываем Exception, если размер файла больше заданного
    }
 }
```