**Министерство образования и наук РФ**

**Федеральное государственное образовательное учреждение**

**Высшего образования**

**Рязанский государственный радиотехнический университет**

**Кафедра ЭВМ**

Отчет по лабораторной работе №9:

“Текстовые файлы”

Вариант 12

Выполнила: ст. гр 640

Тограева К. О.

Проверила:

Доц. к.т.н. Елесина С.И.

Рязань 2017

**Цель работы:**  
 приобретение навыков разработки алгоритмов и программ с использованием текстовых файлов

**Вариант №12:**12. текстовые файлы

Заменить встречающиеся в программе на Си ключевые слова "=" на ":=" , "==" на "=" , "{" на "begin", "}" на "end", "/\*" на "{", а "\*/" на "}", сформировав при этом выходной файл (с тем же именем, что и у входного, но с расширением "pas"), содержащий преобразованную программу.

Ключ:

- предельное число заменяемых символов

**Математическая модель**

Считывается файл построчно.

Для каждоый строки выполняется процедура:

Строка обрабатывается пословно и каждое слово передаётся в подпрограмму – заментель слов. Если слово нужно заменить, то оно заменяется, если нет, то возвращается старое слово.

Создаётся новая строка, составленная из этого слова и пробела после него.

Когда все слова обработаны, управление возвращается в основную программу.

Данная строка записывается в новый файлик.

**Программирование**

**#include <stdio.h>**

**#include <malloc.h>**

**#include <cstring>**

**#define n 150**

**int maxCount;**

**int currentCount;**

**char \*replaceWord(char pString[40]) {**

**if (currentCount < maxCount) {**

**if (strcmp(pString, "=") == 0) {**

**currentCount++;**

**return (char \*) (":=");**

**} else if (strcmp(pString, "=") == 0) {**

**currentCount++;**

**return (char \*) (":=");**

**} else if (strcmp(pString, "==") == 0) {**

**currentCount++;**

**return (char \*) ("=");**

**} else if (strcmp(pString, "{") == 0) {**

**currentCount++;**

**return (char \*) ("begin");**

**} else if (strcmp(pString, "}") == 0) {**

**currentCount++;**

**return (char \*) ("end");**

**}**

**if (strcmp(pString, "/\*") == 0) {**

**currentCount++;**

**return (char \*) ("(\*");**

**}**

**if (strcmp(pString, "\*/") == 0) {**

**currentCount++;**

**return (char \*) ("\*)");**

**}**

**}**

**return pString;**

**}**

**char \*replaceWordsInString(char \*arr) {**

**int i = 0;**

**char \*newString = (char \*) (malloc(n));**

**char \*word = (char \*) malloc(1);**

**int wordLetterIndex = 0;**

**do {**

**if (arr[i] == ' ' || arr[i] == '\n') {**

**wordLetterIndex = 0;**

**// записываем в строку полученное слово**

**newString = strcat(newString, replaceWord(word));**

**word = (char \*) malloc(1);**

**// и записываем следующий символ**

**newString[strlen(newString)] = arr[i];**

**} else {**

**word[wordLetterIndex++] = arr[i];**

**// + 5 если заменится символы на большее их количество**

**word = (char \*) (realloc(word, wordLetterIndex \* sizeof(char) + 5));**

**}**

**} while (arr[i++] != '\n');**

**return newString;**

**}**

**int main() {**

**FILE \*file;**

**FILE \*writeFile;**

**int input;**

**printf("Введите ключ - количество заменяемых слов\n");**

**scanf("%i", &input);**

**maxCount = input;**

**currentCount = 0;**

**const char \*path = "/home/tkseniya/Programming/Labs\_El/Lab9/inputfile.txt";**

**const char \*pathToNewFile = "/home/tkseniya/Programming/Labs\_El/Lab9/output.pas";**

**file = fopen(path, "r");**

**writeFile = fopen(pathToNewFile, "w");**

**char arr[n] = {0};**

**if (file == NULL) {**

**printf("Файл не найден!");**

**} else {**

**while (fgets(arr, n, file) != NULL) {**

**char \*newString = replaceWordsInString(arr);**

**// записать в файлик**

**fputs(newString, writeFile);**

**}**

**// закрываем файлик**

**fclose(file);**

**}**

**return 0;**

**}**

Тестовый пример:

Входной файл:

#include <stdio.h>

#include <stdlib.h>

#include "main.h"

//12.

//Подсчитать число слов, у которых левая и правая половины

//одинаковые.

int main() {

char \*pWordsArray = (char \*) malloc(1);

// массив сохранялка длин слов

int lengthArray[20] = {0}; // статический массив - в стеке

// указатель на начало массива

char \*pArrayBeginning = pWordsArray;

int i = 1;

// printf("\n Введите строку \n");

int wordLength = 0;

int wordsCounter = 0;

char c;

int raws = 4, columns = 12;

char inputArray[raws][columns] = {

{'a', 'b', 'a', 'b', ' ', ' ', 'b', 'a', 'b', 'c', ' ', 'b'},

{'d', 'b', 'd', ' ', ' ', ' ', 'x', 'y', 'z', ' ', 'x', 'y'},

{'x', 'y', ' ', ' ', ' ', 'm', 'a', 'm', 'a', ' ', 'n', 'a'},

{'n', 'a', ' ', ' ', ' ', 'x', 'x', ' ', ' ', ' ', 'y', ' '}

};

for (int j = 0; j < raws; ++j) {

for (int k = 0; k < columns; ++k) {

c = inputArray[j][k];

if (c != ' ') {

wordLength++;

\*pWordsArray = c;

pArrayBeginning = (char \*) (realloc(pArrayBeginning, (i += 1) \* sizeof(char)));

pWordsArray = pArrayBeginning + i - 1;

\*(lengthArray + wordsCounter) = wordLength + 1;

} else {

wordLength = 0;

\*pWordsArray = '\0';

// опять выделяем память под следующий символ

pArrayBeginning = (char \*) (realloc(pArrayBeginning, (i += 1) \* sizeof(char)));

pWordsArray = pArrayBeginning + i - 1;

// char \*word = retrieveWord(\*(lengthArray + wordsCounter), pWordsArray - 1);

// printf("%i слово это - %s\n", wordsCounter, word);

wordsCounter++;

}

}

}

printf("\n\n\n");

printf("Слова, у которых левая и правая половина одинаковые - %i\n",

countHalfWords(pArrayBeginning, lengthArray, wordsCounter));

return 0;

// todo удалять мусор

}

int countHalfWords(char \*beginning, int lengthArray[20], int wordsCounter) {

int mainCounter = 0;

for (int i = 0; i < wordsCounter; ++i) {

bool founded = true;

if (\*(lengthArray + i \* sizeof(char)) % 2 == 0) {

founded = false;

} else {

for (int j = 0; j < (\*(lengthArray + i \* sizeof(char)) / 2); ++j) {// todo это можно сделать через while

// если буква слова не равна букве из второй половине, то слово нам не подходит

if (\*(beginning + j) != \*(beginning + j + \*(lengthArray + i \* sizeof(char)) / 2)) {

founded = false;

}

}

}

if (founded) {

mainCounter++;

char \*word = retrieveWord(\*(lengthArray + i),

beginning + \*(lengthArray + i) - 1);

printf("Нашли слово с одинаковой левой и правой половиной - %s\n", word);

}

// переходим к началу следующего элемента

beginning = beginning + \*(lengthArray + i);

}

return mainCounter;

}

char \*retrieveWord(int length, char \*endPointer) {

char \*word = nullptr;

int letter = 0;

word = (char \*) malloc(1);

char \*startPointer = endPointer - length + 1;

for (; startPointer < endPointer; startPointer++) {

\*(word + letter) = \*startPointer;

letter++;

word = (char \*) realloc(word, (letter + 1) \* sizeof(char));

}

\*(word + letter) = '\0';

return word;

}

Выходной файл (программа вызвана с ключом 10)

#include <stdio.h>

#include <stdlib.h>

#include "main.h"

//12.

//Подсчитать число слов, у которых левая и правая половины

//одинаковые.

int main() begin

char \*pWordsArray := (char \*) malloc(1);

// массив сохранялка длин слов

int lengthArray[20] := {0}; // статический массив - в стеке

// указатель на начало массива

char \*pArrayBeginning := pWordsArray;

int i := 1;

// printf("\n Введите строку \n");

int wordLength := 0;

int wordsCounter := 0;

char c;

int raws := 4, columns := 12;

char inputArray[raws][columns]à:= {

{'a', 'b', 'a', 'b', ' ', ' ', 'b', 'a', 'b', 'c', ' ', 'b'},

{'d', 'b', 'd', ' ', ' ', ' ', 'x', 'y', 'z', ' ', 'x', 'y'},

{'x', 'y', ' ', ' ', ' ', 'm', 'a', 'm', 'a', ' ', 'n', 'a'},

{'n', 'a', ' ', ' ', ' ', 'x', 'x', ' ', ' ', ' ', 'y', ' '}

};

for (int j = 0; j < raws; ++j) {

for (int k = 0; k < columns; ++k) {

c = inputArray[j][k];

if (c != ' ') {

wordLength++;

\*pWordsArray = c;

pArrayBeginning = (char \*) (realloc(pArrayBeginning,z (i += 1) \* sizeof(char)));

pWordsArray = pArrayBeginning + i - 1;

\*(lengthArray + wordsCounter) = wordLength + 1;

} else {

wordLength = 0;

\*pWordsArray = '\0';

// опять выделяем память под следующий символ

pArrayBeginning = (char \*) (realloc(pArrayBeginning,b (i += 1) \* sizeof(char)));

pWordsArray = pArrayBeginning + i - 1;

// char \*word = retrieveWord(\*(lengthArray + wordsCounter), pWordsArray - 1);

// printf("%i слово это - %s\n", wordsCounter, word);

wordsCounter++;

}

}

}

printf("\n\n\n");

printf("Слова, у которых левая и правая половина одинаковые - %i\n",

countHalfWords(pArrayBeginning, lengthArray, wordsCounter));

return 0;

// todo удалять мусор

}

int countHalfWords(char \*beginning, int lengthArray[20], int wordsCounter) {

int mainCounter = 0;

for (int i = 0; i < wordsCounter; ++i) {

bool founded = true;

if (\*(lengthArray + i \* sizeof(char)) % 2 == 0) {

founded = false;

} else {

for (int j = 0; j < (\*(lengthArray + i \* sizeof(char)) / 2); ++j) {// todo это можно сделать через while

// если буква слова не равна букве из второй половине, то слово нам не подх

одит

if (\*(beginning + j) != \*(beginning + j + \*(lengthArray + i \* sizeof(char)) / 2)) {

founded = false;

}

}

}

if (founded) {

mainCounter++;

char \*word = retrieveWord(\*(lengthArray + i),

beginning + \*(lengthArray + i) - 1);

printf("Нашли слово с одинаковой левой и правой половиной - %s\n", word);

}

// переходим к началу следующего элемента

beginning = beginning + \*(lengthArray + i);

}

return mainCounter;

}

char \*retrieveWord(int length, char \*endPointer) {

char \*word = nullptr;

int letter = 0;

word = (char \*) malloc(1);

char \*startPointer = endPointer - length + 1;

for (; startPointer < endPointer; startPointer++) {

\*(word + letter) = \*startPointer;

letter++;

word = (char \*) realloc(word, (letter + 1) \* sizeof(char));

}

\*(word + letter) = '\0';

return word;

}

Вывод:

Мы научились использовать файлы и встраивать их в свои программы.