Федеральное агентство связи

Сибирский государственный университет телекоммуникация и информатики

КУРСОВАЯ РАБОТА

по дисциплине «Программирование»

ВАРИАНТ 12

Выполнил: студент группы ИП-715

Алексеев С.В.

Проверили:

доцент кафедры ПМиК Перцев И. В.

ассистент кафедры ПМиК Дьячкова И. С.

Новосибирск 2018

**Содержание:**

1. Введение
2. Описание алгоритма
3. Код программы на С++
4. Результат работы программы
5. **Введение**

***Задача: считать из текстового файла слова в двусвязный список и затем удалить из него симметричные слова – палиндромы (например - АЛЛА). Вывести элементы списка на экран до и после удаления палиндромов.***

1. **Описание алгоритма**

Описываем структуру, содержащую ссылки на себе подобные структуры и поле для хранения слова. Создаём объект типа FILE, вспомогательный массив символов. Проверяем можно ли открыть текстовый файл, если можно — открываем. Объявляем ссылки типа вышеописанной структуры для создания двусвязного списка. Создаём двусвязный список. Функция fgets выдаёт строки, записывая их во вспомогательный массив по очереди, одновременно с этим происходит выделение отдельных слов и увеличение двусвязного списка.

Выводим полученные слова на экран.

Далее идёт удаление палиндромов. В цикле for проходим по буквам с индексами от нуля до половины длины слова. Сравниваем с буквами второй половины слова (сравниваем буквы с равноудалёнными от середины индексами). В случае нахождения палиндрома удаляем содержащую его структуру из списка.

Выводим список на экран. Очищаем память.

1. **Код программы на С++**

#include "stdafx.h"

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <string.h>

#pragma warning(disable: 4996)

typedef struct list {

list \*next, \*prev;

char str[20];

} List;

int main()

{

setlocale(LC\_ALL, "RUS");

FILE \*file;

char str[100];

if (file = fopen("words.txt", "rt"));

else perror("Error");

printf("Программа прочитает файл words.txt,\n");

printf("Найдет в нем слова палиндромы и удалит их.\n\n");

printf("\t\tТекст\n\n");

List \*head, \*tail, \*point;

head = new List;

head->next = nullptr;

head->prev = nullptr;

strcpy(head->str, "");

tail = head;

point = new List;

point->next = nullptr;

point->prev = tail;

tail->next = point;

tail = point;

int i, j=0;

while (fgets(str, 100, file)) {

printf("%s", str);

for (i = 0; i < strlen(str); i++) {

if ((str[i] == ' ' || str[i] == '\n') && (str[i-1] != ' ')) {

if (!feof(file)) {

point->str[j] = '\0';

point = new List;

point->next = nullptr;

point->prev = tail;

tail->next = point;

tail = point;

j = 0;

}

}

else {

point->str[j++]=str[i];

}

}

}

fclose(file);

printf("before:\n");

point = head;

while (point->next) {

printf("%s ", point->str);

point = point->next;

}

int flag;

point = head->next;

while (point->next) {

flag = 1;

for (i = 0; i < strlen(point->str)/2; i++) {

if (point->str[i] == point->str[strlen(point->str)-1- i]);

else flag = 0;

}

if (flag) {

tail = point;

point->prev->next = point->next;

point->next->prev = point->prev;

point = point->next;

delete tail;

} else point = point->next;

}

printf("\n\nAfter:\n");

point = head;

while (point->next) {

printf("%s ", point->str);

point = point->next;

}

point = tail = head;

while (point) {

point = tail->next;

delete tail;

tail = point;

}

head = nullptr;

return 0;

}