**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра математического обеспечения и применения ЭВМ**

отчет

**по лабораторной работе №4**

**по дисциплине «Программирование»**

Тема: Линейные списки.

|  |  |  |
| --- | --- | --- |
| Студент гр. 6303 |  | Ильяшук Д.И. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2016

**Цель работы:** Создать двунаправленный список музыкальных композиций MusicalComposition и api (application programming interface - в данном случае набор функций) для работы со списком.

**Ход работы:**

**Задание:**

Создайте двунаправленный список музыкальных композиций MusicalComposition и **api** (***a****pplication****p****rogramming****i****nterface - в данном случае набор функций*) для работы со списком.

Структура элемента списка (тип - MusicalComposition)

* name - строка неизвестной длины (гарантируется, что длина не может быть больше 80 символов), название композиции.
* author - строка неизвестной длины (гарантируется, что длина не может быть больше 80 символов), автор композиции/музыкальная группа.
* year - целое число, год создания.

Функция для создания элемента списка (тип элемента MusicalComposition)

* MusicalComposition\* createMusicalComposition(char\* name, char\* author, int year)

Функции для работы со списком:

* MusicalComposition\* createMusicalCompositionList(char\*\* array\_names, char\*\* array\_authors, int\* array\_years, int n); // создает список музыкальных композиций MusicalCompositionList, в котором:
  + ***n****- длина массивов****array\_names****,****array\_authors****,****array\_years****.*
  + поле **name** первого элемента списка соответствует первому элементу списка array\_names (**array\_names[0]**).
  + поле **author** первого элемента списка соответствует первому элементу списка array\_authors (**array\_authors[0]**).
  + поле **year** первого элемента списка соответствует первому элементу списка array\_authors (**array\_years[0]**).

*Аналогично для второго, третьего, ...****n-1****-го элемента массива.*

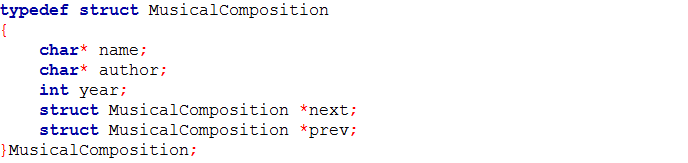
*! длина массивов****array\_names, array\_authors, array\_years****одинаковая и равна n, это проверять не требуется.*

*Функция возвращает указатель на первый элемент списка.*

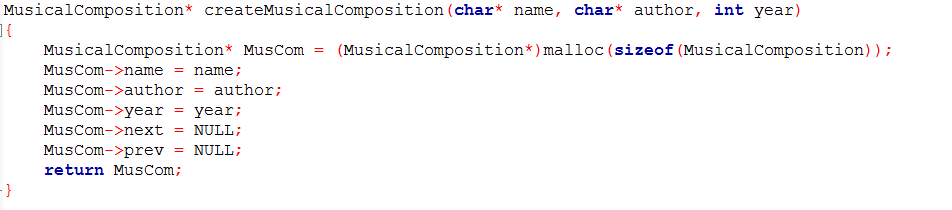
* void push(MusicalComposition\*  head, MusicalComposition\* element); // добавляет **element**  в конец списка **musical\_composition\_list**
* void removeEl (MusicalComposition\*  head, char\* name\_for\_remove); // удаляет элемент **element** списка, у которого значение **name** равно значению  **name\_for\_remove**
* int count(MusicalComposition\*  head); //возвращает количество элементов списка
* void print\_names(MusicalComposition\*  head); //Выводит названия композиций

**Описание функций:**

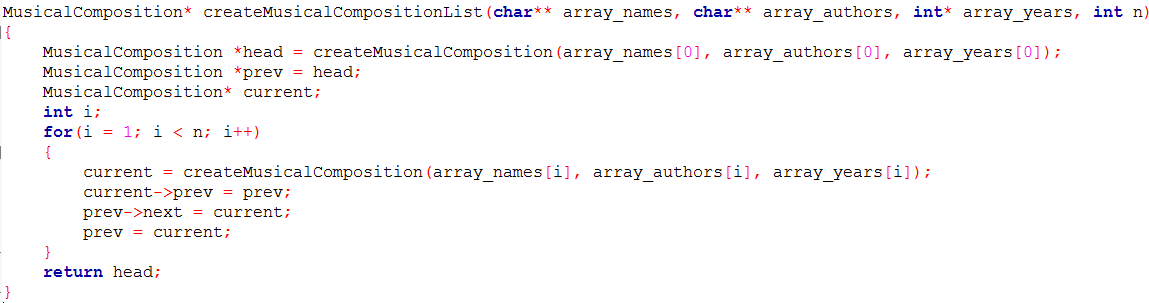
1. Была описана структура **MusicalComposition**. При помощи оператора **typedef** типу **struct MusicalComposition** было присвоено название **MusicalComposition**.

****

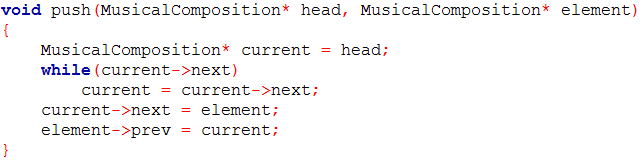
1. Затем была написана функция **createMusicalComposition**, которая создает структуру, она получает на вход значения, которые должна хранить структура, затем выделяется память для структуры, которая потом заполняется.

****

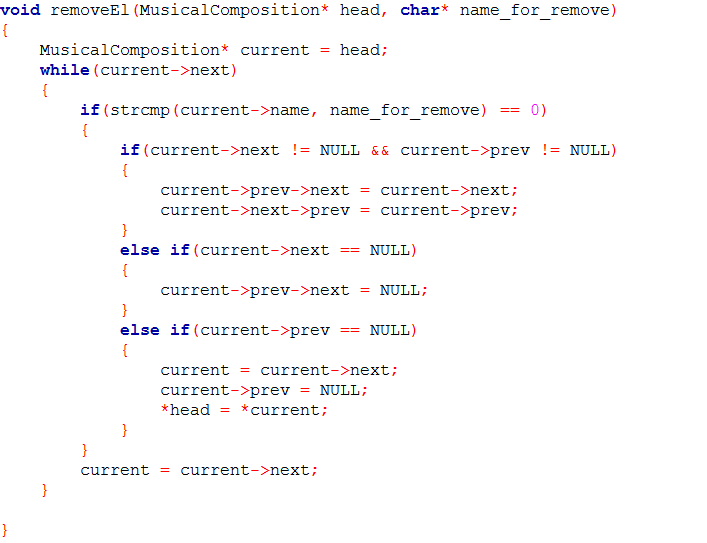
1. Затем была написана функция **createMusicalCompositionList**, которая создает двусвязный список музыкальных композиций. На вход она принимает три массива параметров музыкальных композиций. Возвращает она корень созданного списка.

****

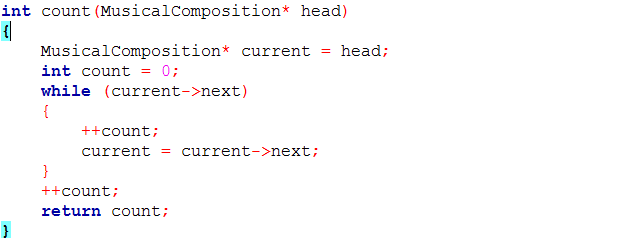
1. Далее была написана функция **push**, которая добавляет композицию в конец списка. На вход она принимает корень списка и указатель на новый элемент.

****

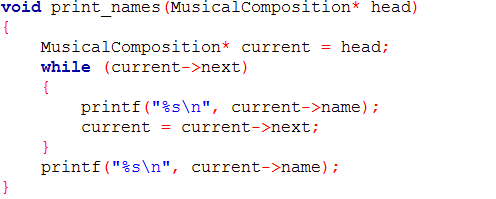
1. Затем была написана функция **removeEl**, которая удаляет из списка все композиции с заданным названием. На вход она принимает корень списка и название композиции, которую нужно удалить.

****

1. Далее была написана функция **count**, которая считает количество композиций в списке, на вход она принимает корень списка, а возвращает число композиций в списке.

****

1. Затем была написана функция **print\_names,** которая выводит на экран названия композиций в списке. Функция получает на вход корень списка.

****

Код программы и отчет выложены в ветку ilyashuk\_lr4 на github.

**Вывод:** В ходе лабораторной работы получены навыки работы с указателями и двунаправленными линейными списками языка Си, которые успешно применены на практике при создании программы.