**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №4**

**по дисциплине «Программирование»**

Тема: **Линейные списки.**

|  |  |  |
| --- | --- | --- |
| Студент гр. 6304 |  | Иванов В.С. |
| Преподаватель |  | Кринкин К.В. |

Санкт-Петербург

2016

**Оглавление**

[**Цель работы** 3](#_Toc469512893)

[**Задача** 3](#_Toc469512894)

[**Содержание** 4](#_Toc469512895)

[1. Структура элемента списка (тип - MusicalComposition) 4](#_Toc469512896)

[2. Функция для создания элемента списка (тип элемента MusicalComposition) 4](#_Toc469512897)

[3. Функция для создания списка( MusicalComposition\* createMusicalCompositionList(char\*\* array\_names, char\*\* array\_authors, int\* array\_years, int n) 4](#_Toc469512898)

[4. Функция для добавления элемента в конец списка( void push (MusicalComposition\*  head, MusicalComposition\* element)) 4](#_Toc469512899)

[5. Функция для удаления нужного элемента( void removeEl (MusicalComposition\*  head, char\* name\_for\_remove)) 5](#_Toc469512900)

[6. Функция подсчёта кол-ва элементов(int count(MusicalComposition\*  head)) 5](#_Toc469512901)

[7. Функция печати названий композиций( void print\_names(MusicalComposition\*  head) 5](#_Toc469512902)

[**Вывод** 5](#_Toc469512903)

# **Цель работы**

Создание программы для использования линейного списка и работы с ним

# **Задача**

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.*

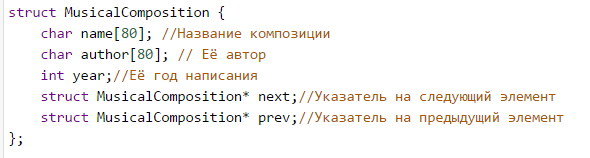
*Функция возвращает указатель на первый элемент списка.*

* 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

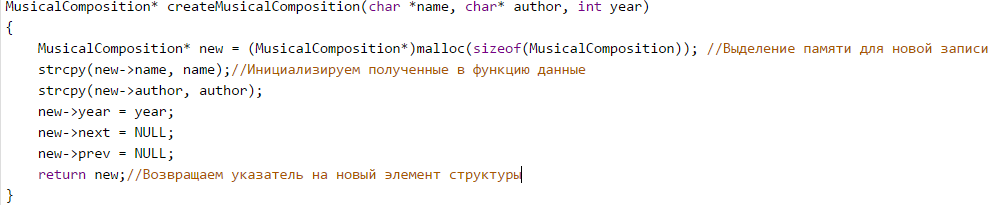
В функции main написана некоторая последовательность вызова команд для проверки работы вашего списка.

# **Содержание**

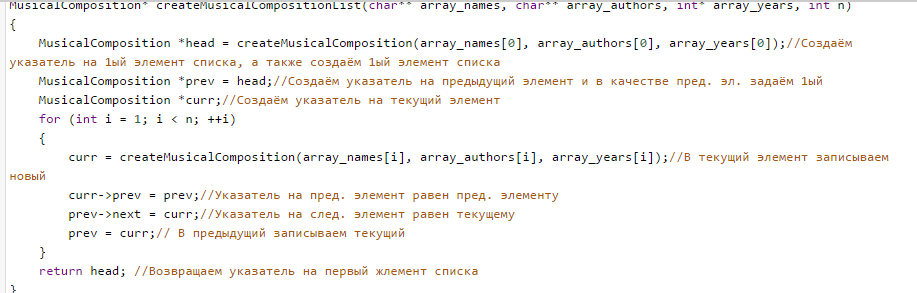
## Структура элемента списка (тип - MusicalComposition)



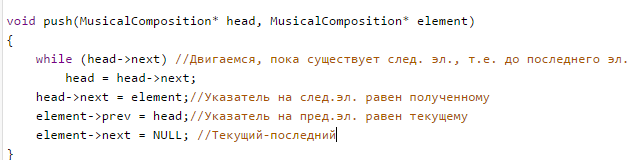
## Функция для создания элемента списка (тип элемента MusicalComposition)



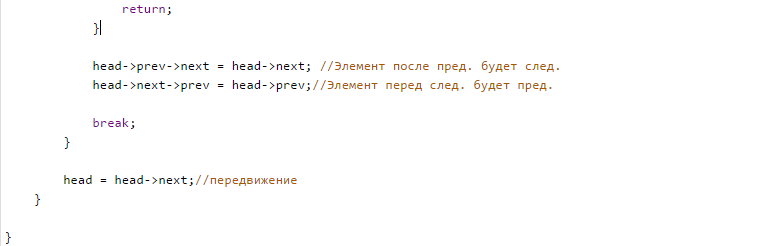
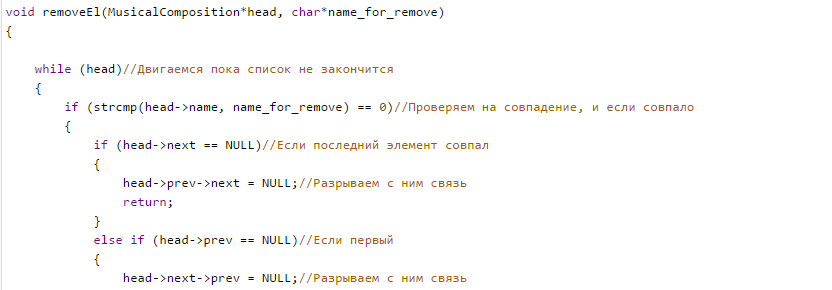
## Функция для создания списка( MusicalComposition\* createMusicalCompositionList(char\*\* array\_names, char\*\* array\_authors, int\* array\_years, int n)



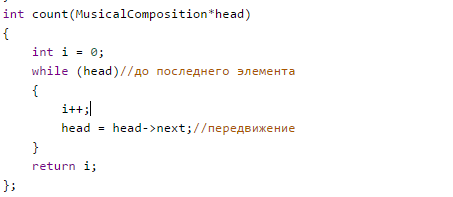
## Функция для добавления элемента в конец списка( void push (MusicalComposition\*  head, MusicalComposition\* element))



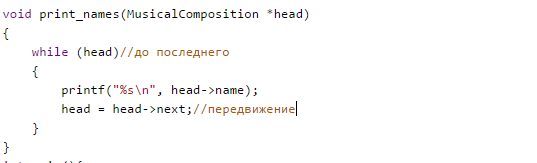
## Функция для удаления нужного элемента( void removeEl (MusicalComposition\*  head, char\* name\_for\_remove))



## Функция подсчёта кол-ва элементов(int count(MusicalComposition\*  head))



## Функция печати названий композиций( void print\_names(MusicalComposition\*  head)



# **Вывод**

Выполнив данную лабораторную работу, мы освоили и закрепили на практике написание программы с применением структур в качестве элементов сложных типов представления данных таких, как одно- и дву- направленные линейные списки, а также некоторые функции работы с ней.