**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра математического обеспечения и применения ЭВМ**

**КУРСОВАЯ РАБОТА**

**по дисциплине «Программирование»**

Тема: Линейные списки

|  |  |  |
| --- | --- | --- |
| Студент |  | Шевченко Д.В. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2016

**ЗАДАНИЕ**

**на курсовую работу**

|  |  |  |
| --- | --- | --- |
| Студент Шевченко Д.В. | | |
| Группа 6303 | | |
| Тема работы: Линейные списки | | |
|  | | |
| Содержание пояснительной записки:   * Содержание * Введение * Описание функций, необходимых для работы с двунаправленным линейным списком * Описание функций сортировки списка * Примеры работы программы * Заключение * Список использованных источников * Приложение А. Исходный код программы | | |
|  | | |
| Дата выдачи задания: 02.12.2016 | | |
| Дата сдачи реферата: 21.12.2016 | | |
| Дата защиты реферата: 21.12.2016 | | |
| Студент |  | Шевченко Д.В. |
| Преподаватель |  | Берленко Т.А. |

**Аннотация**

В данной работе была разработана программа на языке программирования C, которая позволяет работать с набором функций, отвечающих засписок музыкальных композиций. Для функционирования списка были созданы и описаны необходимые функции, позволяющие добавлять, удалять, сортировать и выводить элементы списка и их количество в консоль, а также описана структура элемента списка. Помимо этого, была проведена работа над оптимизацией исходного кода программы для ускорения ее быстродействия и оптимального использования памяти и ресурсов клиента. Приведено полное описание исходного кода.

**содержание**

|  |  |  |
| --- | --- | --- |
|  | Введение | 4 |
| 1. | Описание функций, необходимых для работы с двунаправленным линейным списком | 6 |
| 1.1. | Созданиеэлемента | 6 |
| 1.2. | Создание двунаправленного списка связанных элементов | 6 |
| 1.3. | Добавление элемента в конец списка | 6 |
| 1.4. | Удаление элементов из списка | 7 |
| 1.5. | Подсчет количества элементов в списке | 7 |
| 1.6. | Вывод элементов списка в консоль | 7 |
| 2. | Функция удаления нечетных элементов. | 8 |
|  | Примеры работы программы | 10 |
|  | Заключение | 12 |
|  | Список использованных источников | 13 |
|  | Приложение А. Исходный код программы | 14 |

**введение**

Необходимо создать 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)

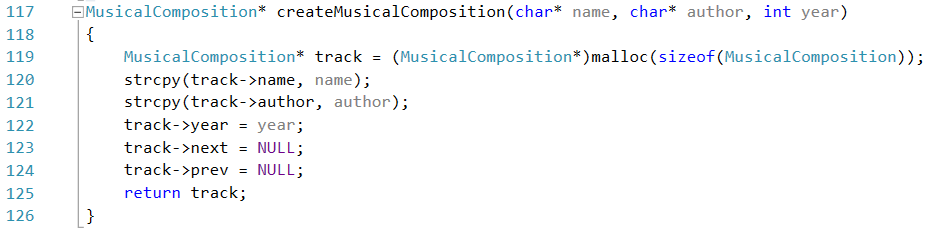
* name - строка неизвестной длины (гарантируется, что длина не может быть больше 80 символов), название композиции.
* author - строка неизвестной длины (гарантируется, что длина не может быть больше 80 символов), автор композиции/музыкальная группа.
* year - целое число, год создания.

Функции, необходимые для работы программы

* Создание элемента типа MusicalComposition
* Создание двунаправленного списка связанных элементов типа MusicalComposition
* Добавление элемента в конец списка
* Удаление определенного элемента (или элементов) из списка
* Подсчет количества элементов в списке
* Вывод элементов списка в консоль
* Сортировка элементов в списке
* Удаление всех нечётных элементов списка

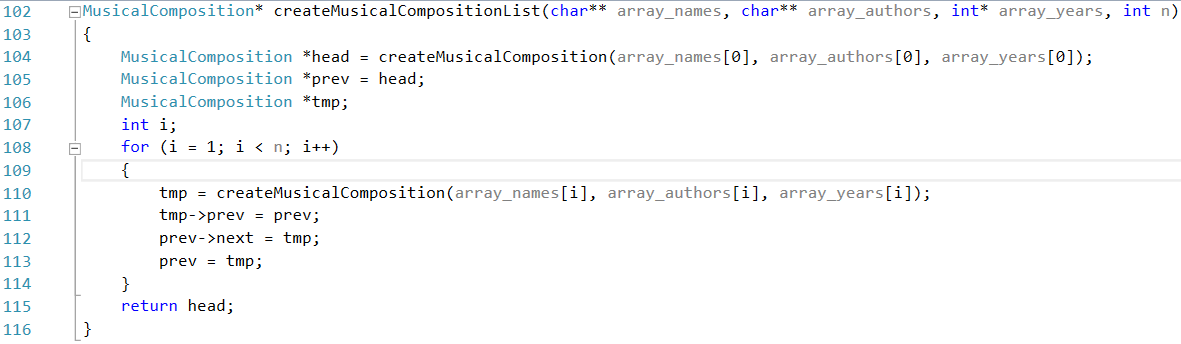
**1. функции для работы с двунаправленным линейным списком**

* 1. Функция для создания элемента списка (тип элемента MusicalComposition).

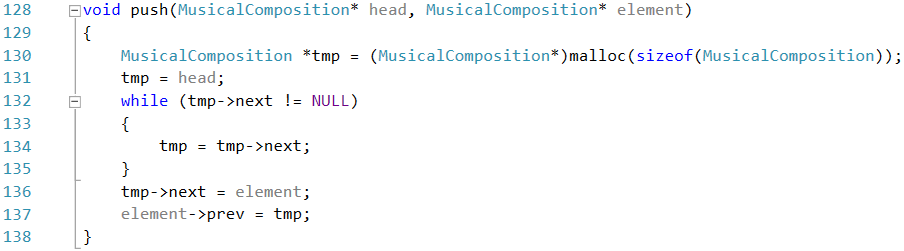


Данной функции передают три параметра: название, имя автора и год создания композиции; которые она записывает в новый элемент списка, для которого резервирует память с помощью функции malloc. Функция возвращает указатель на созданный элемент списка.

* 1. Функция для создания списка музыкальных композиций MusicalCompositionList.

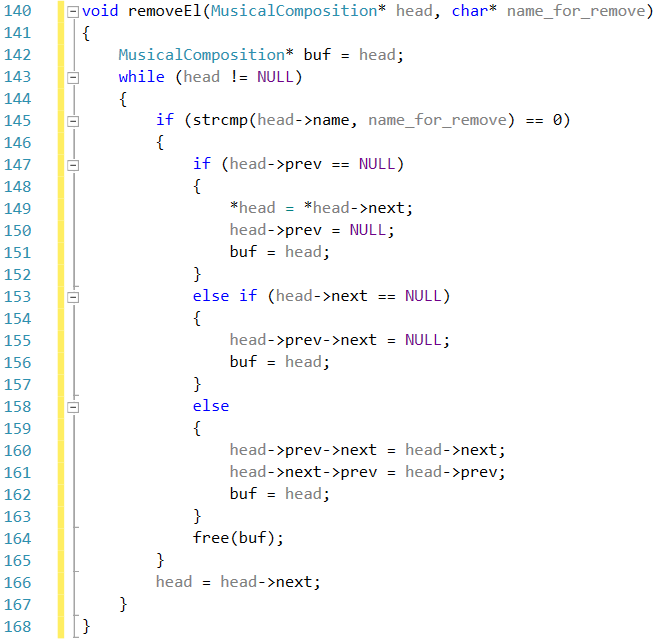


Функция получает количество существующих композиций три массива: массив названий, массив авторов и массив годов созданий; и формирует линейный двунаправленный список.

* 1. Функция для добавления нового элемента в конец списка. 

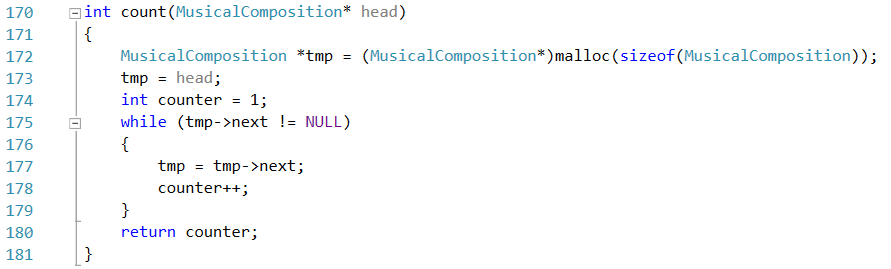
Данной функции передается указатель на начало линейного списка и элемент, который необходимо добавить. Функция перемещается по списку, пока не доходит до последнего элемента, а затем соединяет последний и новый элементы списка посредством указателей.

* 1. Функция для удаления элемент списка, с определённым названием композиции.

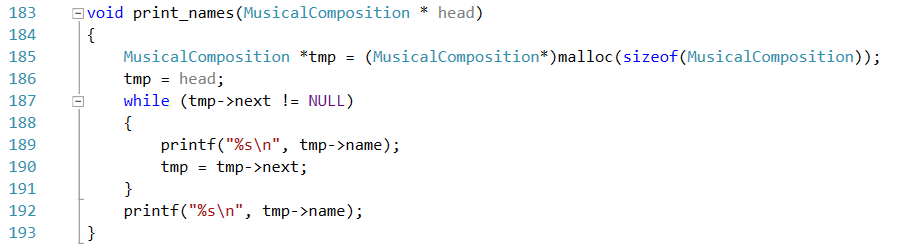


Функция получает указатель на начало списка и название композиции, которую нужно удалить. Перемещаясь по списку, она сверяет название композиции с переданной строкой. При совпадении значений функция связывает предыдущий и следующий элементы посредством указателей, тем самым удаляя найденный элемент из списка.

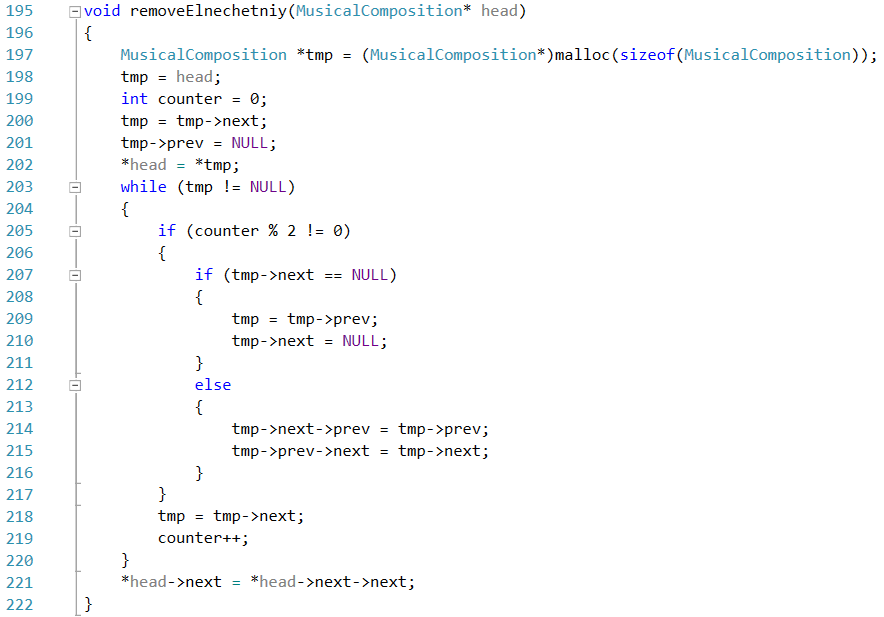
* 1. Функция для подсчёта количества элементов списка.

Функция получает указатель на первый элемент списка, и пробегая весь список, увеличивает счётчик counter на каждом элементе списка.

* 1. Функция для выведения названия композиций.

Данной функции передается указатель на первый элемент списка. Перебирая все элементы, функция печатает названия каждой композиции.

1. **Функция удаления нечетных элементов.**

Функция получает указатель на начало списка. Перемещаясь по списку, она удаляет элементы с нечетным индексом. В каждом случае функция связывает предыдущий и следующий элементы посредством указателей, тем самым удаляя найденный элемент из списка.

**примеры работы программы:**

|  |  |  |
| --- | --- | --- |
| № | ВВОд | вЫВОД |
| 1 | 6 // количество элементов списка  After Midnight // элементы для формирования списка  Blink-182  2011  Creep  Radiohead  1993  Master of Puppets  Metallica  1984  Numb  Linkin Park  2003  Hurt  Johny Cash  2003  Everlong  Foo Fighters  1995  Snuff // элемент для добавления в список  Slipknot  2008  Numb //имя элемента для удаления из списка | After Midnight Blink-182 2011//первый элемент списка  6 //количество элементов до добавления  7 //количество элементов после добавления  After Midnight //список названий композиций  Creep //после добавления элемента  Master of Puppets  Hurt  Everlong  Snuff  Numb  6 //количество элементов после удаления  After Midnight  Creep //список элементов после удаления  Master of Puppets  Hurt  Everlong  Snuff  3 // количество элементов после удаления нечетных элементов  // Список элементов после удаления  Creep // Нечетных элементов  Hurt  Snuff |
| 2 | 4 // количество элементов для формирования списка  Name1 // элементы для формирования списка  Lastname1  1  Name2  Lastname2  2  Name3  Lastname3  3  Name4  Lastname4  4  Addname //элемент для добавления  Addlastname  5  Addname //имя элемента для удаления | Name1 Lastname1 1 // первый элемент списка  4 // количество элементов до добавления  5 // количество элементов после добавления  Name1 // имена элементов списка после добавления  Name2  Name3  Name4  Addname  4 // количество элементов после удаления элемента  Name1 // имена элементов списка после удаления  Name2  Name3  Name4  2//количество элементов после удаления нечётных эл.  Name2 // имена элементов после удаления  Name4 |
| 3 | 5 // количество элементов для формирования списка  Danil // элементы для формирования списка  Ilyashuk  1998  Nikita  Vaganov  1998  Dmitriy  Ivanov  1999  Vyacheslav  Ekvas  1998  Danil  Shevchenko  Anton //элемент для добавления  Kuligin  1997  Danil // элемент для удаления | Danil Ilyashuk 1998 // первый элемент списка  5 // количество элементов до добавления  6 // количество элементов после добавления  Danil//имена элементов списка после добавления  Nikita  Dmitriy  Vyacheslav  Danil  Anton  4 // количество элементов после удаления элемента  Nikita // имена элементов списка после удаления  Dmitriy  Vyacheslav  Anton  2//количество элементов после удаления нечётных эл.  Dmitriy // имена элементов после удаления  Anton |

**заключение:**

При выполнении курсовой работы, были освоены и закреплены на практике навыки написания программы с применением структур в качестве элементов сложных типов представления данных таких, как двунаправленные линейные списки, а также API для работы с ними, в частности, функции вставки, удаления, подсчета и вывода элементов списка.