**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 |  | Пискунов Я.А. |
| Преподаватель |  | Кринкин К.В. |

Санкт-Петербург

2016

**Цель:**

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

**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

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

**Ход работы:**

* 1. **Код программы:**

**#include <stdlib.h>**

**#include <stdio.h>**

**#include <string.h>**

**#include <stddef.h>**

// Описание структуры MusicalComposition

**typedef struct MusicalComposition {**

**char name[80];**

**char author[80];**

**int year;**

**struct MusicalComposition\* prev;**

**struct MusicalComposition\* next;**

**} MusicalComposition;**

// Создание структуры MusicalComposition

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

**MusicalComposition\* ptr = (MusicalComposition\*)malloc(sizeof(MusicalComposition));**

**ptr->prev = NULL;**

**ptr->next = NULL;**

**strcpy(ptr->name, name);**

**strcpy(ptr->author, author);**

**ptr->year = year;**

**return ptr;**

**}**

// Функции для работы со списком MusicalComposition(прототипы)

**MusicalComposition\* createMusicalCompositionList(char\*\* array\_names, char\*\* array\_authors, int\* array\_years, int n);**

**void push(MusicalComposition\* head, MusicalComposition\* element);**

**void removeEl(MusicalComposition\* head, char\* name\_for\_remove);**

**int count(MusicalComposition\* head);**

**void print\_names(MusicalComposition\* head);**

//Функция создания плейлиста

**MusicalComposition\* createMusicalCompositionList(char\*\* array\_names, char\*\* array\_authors, int\* array\_years, int n){**

**MusicalComposition\* temp = createMusicalComposition(array\_names[0], array\_authors[0], array\_years[0]);**

**MusicalComposition\* head = temp;**

**for (int i=1; i<n; i++)**

**{**

**temp->next = createMusicalComposition(array\_names[i], array\_authors[i], array\_years[i]);**

**temp->next->prev = temp;**

**temp = temp->next;**

**}**

**return head;**

**}**

//Функция добавления элемента в конец списка

**void push(MusicalComposition\* head, MusicalComposition\* element){**

**while (head->next!=NULL)**

**head = head->next;**

**head->next = element;**

**element->prev = head;**

**}**

//Функция подсчета количества элементов в списке

**int count(MusicalComposition\* head){**

**int number = 0;**

**while (head!=NULL){**

**head = head->next;**

**number++;**

**}**

**return number;**

**}**

// функция вывода имен композиций на печать

**void print\_names(MusicalComposition\* head){**

**while (head!=NULL){**

**printf("%s\n", head->name);**

**head = head->next;**

**}**

**}**

//Функция удаления элемента

**void removeEl(MusicalComposition\* head, char\* name\_for\_remove){**

**while (strcmp(head->name, name\_for\_remove)!=0){**

**head = head->next;**

**}**

**while ((strcmp(name\_for\_remove, head->name)!=0) && (head!=NULL)){**

**head = head->next;**

**}**

**if ((head->prev!=NULL)&&(head->next!=NULL)){**

**head->prev->next = head->next;**

**head->next->prev = head->prev;**

**}**

**else if (head->next == NULL){**

**head->prev->next = NULL;**

**}**

**else if(head->prev == NULL){**

**while(head->next!=NULL){**

**strcpy(head->name,head->next->name);**

**strcpy(head->author, head->next->author);**

**head->year = head->next->year;**

**}**

**head->prev->next = NULL;**

**head->prev = NULL;**

**}**

**}**

* 1. **Вывод:**

В ходе данной работы успешно создан двунаправленный список музыкальных композиций «MusicalComposition». Изучены основные способы работы со структурами: создание, описание и объединение в списки.