МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ по лабораторной работе №2

по дисциплине «Программирование»

Тема: Линейные списки

Студентка гр. 1304 Виноградова М.О.

Преподаватель Чайка К.В.

Санкт-Петербург
2022

Цель работы.

Написать программу в соответствии с условием задачи.

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.

Функцию main менять не нужно.

Выполнение работы.

Функции:

createMusicalComposition – функция принимает значения необходимые для одного элемента списка и возвращает указатель на одну переменную типа MusicalComposition с переданными ранее значениями.

push – функция принимает указатель на головной(первый) элемент списка и указатель на элемент, который надо добавить в конец списка. Функция ничего не возвращает, элемент добавляется путем перезаписи последнего элемента(в значении next будет указан элемент).

createMusicalCompositionList – функция принимает три массива (массив композиций, массив авторов, массив годов) и количество элементов в массивах, функция создает список из элементов типа MusicalComposition с переданными значениями (“голова” списка создается со значениями равными первым значениям трех массивов, остальные элементы добавляются в список в цикле с помощью функции push). Функция возвращает указатель на первый элемент списка.

removeEl – В функцию передаются указатель на первый элемент списка и название композиции из списка, которое необходимо удалить. После нахождения элемента с заданным значением, он удаляется путем перезаписи указателей на предыдущий и следующий элементы. Функция ничего не возвращает.

count – в функцию передается указатель на первый элемент списка. С помощью цикла while подсчитывается количество элементов в данном списке. Функция возвращает число элементов.

print_names – в функцию передается указатель на первый элемент списка. С помощью цикла while функция выводит названия композиций. Функция ничего не возвращает.

Разработанный программный код см. в приложении А.

Тестирование.

Результаты тестирования представлены в табл. 1.

Таблица 1 – Результаты тестирования

№ п/п	Входные данные	Выходные данные	Комментарии

1.	7 Fields of Gold Sting 1993 In the Army Now Status Quo 1986 Mixed Emotions The Rolling Stones 1989 Billie Jean Michael Jackson 1983 Seek and Destroy Metallica 1982 Wicked Game Chris Isaak 1989 Points of Authority Linkin Park 2000 Sonne Rammstein 2001 Points of Authority	Fields of Gold Sting 1993 7 8 Fields of Gold In the Army Now Mixed Emotions Billie Jean Seek and Destroy Wicked Game Sonne 7	Программа работает корректно

Вывод.

В соответствии с условием задачи была реализована программа.

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>


// Описание структуры MusicalComposition
typedef struct MusicalComposition{
   char * name;
   char * author;
   int year;
   struct MusicalComposition * next;
   struct MusicalComposition * previous;
```

```
} MusicalComposition;

// Создание структуры MusicalComposition

MusicalComposition* createMusicalComposition(char* name, char* autor,int year){
   struct MusicalComposition *music= malloc(sizeof ( MusicalComposition));
   music->name=name;
   music->author=autor;
   music->year=year;
   music->next=NULL;
   music->previous=NULL;
   return music;
}
void erraseList(MusicalComposition *head){
   while (head){
      MusicalComposition *tmp=head->next;
      free(head);
      head=tmp;
   }
}
void push(MusicalComposition* head, MusicalComposition* element){

   //element->next=NULL;
   while (head->next){
      head=head->next;
   }
   head->next=element;
   element->previous=head;

}
// Функции для работы со списком MusicalComposition

MusicalComposition* createMusicalCompositionList(char** array_names, char** array_authors, int*
array_years, int n){
   MusicalComposition *mus= createMusicalComposition(array_names[0],array_authors[0],array_years[0]);
   for(int i=1;i<n;i++){
      push(mus, createMusicalComposition(array_names[i],array_authors[i],array_years[i]));
   }
   return mus;
}



void removeEl(MusicalComposition* head, char* name_for_remove){
   MusicalComposition *now_next;
   MusicalComposition *now_prev;

   while (head!=NULL){
      if(strcmp(head->name,name_for_remove)==0){
```

```
        now_prev=head->previous;
        now_next=head->next;

        if(now_prev!= NULL){

           now_prev->next=now_next;
        }

        now_next->previous=now_prev;


        return;
      }

        head=head->next;

   }


}

int count(MusicalComposition* head){
   int k=0;
   while (head!=NULL){
      head=head->next;
      k++;
   }
   //if(k==1) return 0;
   return k;
}

void print_names(MusicalComposition* head){
   while (head!=NULL){
      printf("%s\n",head->name);
      head = head->next;
   }

}



int main(){
   int length;
   scanf("%d\n", &length);

   char** names = (char**)malloc(sizeof(char*)*length);
   char** authors = (char**)malloc(sizeof(char*)*length);
   int* years = (int*)malloc(sizeof(int)*length);
```

```
    for (int i=0;i<length;i++)
    {
        char name[80];
        char author[80];

        fgets(name, 80, stdin);
        fgets(author, 80, stdin);
        fscanf(stdin, "%d\n", &years[i]);

        (*strstr(name,"\n"))=0;
        (*strstr(author,"\n"))=0;

        names[i] = (char*)malloc(sizeof(char*) * (strlen(name)+1));
        authors[i] = (char*)malloc(sizeof(char*) * (strlen(author)+1));

        strcpy(names[i], name);
        strcpy(authors[i], author);

    }
    MusicalComposition* head = createMusicalCompositionList(names, authors, years, length);
    char name_for_push[80];
    char author_for_push[80];
    int year_for_push;

    char name_for_remove[80];

    fgets(name_for_push, 80, stdin);
    fgets(author_for_push, 80, stdin);
    fscanf(stdin, "%d\n", &year_for_push);
    (*strstr(name_for_push,"\n"))=0;
    (*strstr(author_for_push,"\n"))=0;

    MusicalComposition* element_for_push = createMusicalComposition(name_for_push, author_for_push,
year_for_push);

    fgets(name_for_remove, 80, stdin);
    (*strstr(name_for_remove,"\n"))=0;

    printf("%s %s %d\n", head->name, head->author, head->year);
    int k = count(head);

    printf("%d\n", k);
    push(head, element_for_push);

    k = count(head);
    printf("%d\n", k);

    removeEl(head, name_for_remove);
    print_names(head);
```

```
    k = count(head);
    printf("%d\n", k);

    for (int i=0;i<length;i++){
        free(names[i]);
        free(authors[i]);
    }
    free(names);
    free(authors);
    free(years);

    return 0;

}
```