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

Студент гр. 3344 _____ Анахин Е.Д.

Преподаватель _____ Глазунов С.А.

Санкт-Петербург
2024

Цель работы

Изучение структуры данных двусвязный список и получение опыта работы с ними в языке программирования C.

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

Был создан прототип структуры MusicalComposition, в котором были добавлены поля: n, name, author, year, *next, *prev.

Далее была создана функция для создания объектов createMusicalComposition, которая принимает на вход name, author, year.

Затем была создана функция для создания списка объектов MusicalComposition — createMusicalCompositionList, которая создает двусвязный список из объектов MusicalComposition.

Далее были созданы функции для работы со списком музыкальных композиций:

- push — меняет указатель крайнего элемента списка на новую структуру
- removeEl — Меняет указатели двух соседних (при наличии) элементов на другие структуры
- count — выводит количество элементов списка
- print_names — выводит названия всех структур в списке

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	Корректно
2.	3 Floods Pantera 1996 Points of Authority Linkin Park 2000 Seek and Destroy Metallica 1982 Angel of Death Slayer 1986	Floods Pantera 1996 3 4 Floods Seek and Destroy Angel of Death 3	Корректно

	Points of Authority		

Выводы

Был получен работы с двусвязными списка в языке программирования C.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

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

typedef struct MusicalComposition {
    int n;
    char name[80];
    char author[80];
    int year;
    struct MusicalComposition* next;
    struct MusicalComposition* prev;
} MusicalComposition;

MusicalComposition*   createMusicalComposition(char*   name,   char*
author, int year) {
                              MusicalComposition*      musicItem      =
malloc(sizeof(MusicalComposition));
    if (musicItem == NULL) {
        return NULL;
    }
    musicItem->n = 1;
    strcpy(musicItem->author, author);
    strcpy(musicItem->name, name);
    musicItem->year = year;
    musicItem->next = NULL;
    musicItem->prev = NULL;
    return musicItem;
}

MusicalComposition*             createMusicalCompositionList(char**
array_names, char** array_authors, int* array_years, int n) {
    MusicalComposition* firstItem = NULL;
    MusicalComposition* prevItem = NULL;
    for (int i = 0; i < n; i++) {
                                  MusicalComposition*    thisItem    =
createMusicalComposition(array_names[i],               array_authors[i],
array_years[i]);
        if (thisItem == NULL) {
            while (firstItem != NULL) {
                MusicalComposition* temp = firstItem;
                firstItem = firstItem->next;
                free(temp);
            }
            return NULL;
        }
        if (prevItem != NULL) {
            prevItem->next = thisItem;
            thisItem->prev = prevItem;
        } else {
            firstItem = thisItem;
```

```
        }
        prevItem = thisItem;
    }
    return firstItem;
}

void push(MusicalComposition* head, MusicalComposition* element) {
    MusicalComposition* this = head;
    while (this->next != NULL) {
        this = this->next;
    }
    this->next = element;
    element->prev = this;
}

void removeEl(MusicalComposition* head, char* name_for_remove) {
    MusicalComposition* this = head;
    while (this != NULL) {
        if (strcmp(this->name, name_for_remove) == 0) {
            if (this->prev != NULL) {
                this->prev->next = this->next;
            }
            if (this->next != NULL) {
                this->next->prev = this->prev;
            }
            free(this);
            return;
        }
        this = this->next;
    }
}

int count(MusicalComposition* head) {
    int amount = 0;
    MusicalComposition* this = head;
    while (this != NULL) {
        amount++;
        this = this->next;
    }
    return amount;
}

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

}
```