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. 3342 _____ Иванов Д. М.

Преподаватель _____ Глазунов С. А.

Санкт-Петербург
2024

Цель работы

Изучить двунаправленные списки и их реализацию на язые программирования C. Реализовать двунаправленный список с несколькими полями и написать для работы с ним api.

Задание

Создайте двунаправленный список музыкальных композиций MusicalComposition и api (application programming interface - 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:

1) 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]).

2)void push(MusicalComposition* head, MusicalComposition* element); // добавляет element в конец списка musical_composition_list

3)void removeEl (MusicalComposition* head, char* name_for_remove); // удаляет элемент element списка, у которого значение name равно значению name_for_remove

4)int count(MusicalComposition* head); //возвращает количество элементов списка

5)void print_names(MusicalComposition* head); //Выводит названия композиций.

Выполнение работы

Необходимо было создать структуру двунаправленного списка, содержащего следующие поля: char* name(навзание композиции), char* author(автор композиции), int year(год создания), struct MusicalComposition* next(указатель на следующую композицию), struct MusicalComposition* prev(указатель на прошлую).

Дальше идет написание функций для этого списка:

1) MusicalComposition* createMusicalComposition(char* name, char* author,int year) – создание элемента списка. Выделяется память, заполняются поля через функцию -> и данный элемент возвращается функцией.

2) MusicalComposition* createMusicalCompositionList(char** array_names, char** array_authors, int* array_years, int n) - создает список музыкальных композиций. Для начала создается “голова” списка из первого элемента. Потом циклом происходит обход поданного списка, создается новая структура и связывается с остальными элементами через поля next и prev.

3) void push(MusicalComposition* head, MusicalComposition* element) – добавления элемента в конец. Идет цикл до того момента, когда элемент будет ссылаться на NULL. Затем эта структура через next добавляется в список.

4) void removeEl(MusicalComposition* head, char* name_for_remove) - удаление элемента по названию композиции. Циклом находим нужный элемент. Освобождаем память и меняем свяь в списке через поля next и prev.

5) int count(MusicalComposition* head) – подсчет количества элементов в списке. Циклом происходит обход элементов до ссылки на NULL и увеличение счетчика.

6) void print_names(MusicalComposition* head) – вывод всех композиций через цикл до ссылки next на NULL.

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	Верный вывод

Выводы

Была разработана программа, создающая днунаправленный список из музыкальных композиций и выполняющая с ним определенные функции. Изучена работа с линейными списками, со структурами и реализация их на языке Си.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.c

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


// Описание структуры MusicalComposition
typedef struct MusicalComposition{
    char* name;
    char* author;
    int year;
    struct MusicalComposition* next;
    struct MusicalComposition* prev;
}MusicalComposition;

void memory_error(){
    fprintf(stderr, "Error with memory allocation!");
    exit(1);
}

// Создание структуры MusicalComposition

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

// Функции для работы со списком MusicalComposition

MusicalComposition*  createMusicalCompositionList(char**  array_names,
char** array_authors, int* array_years, int n){
    MusicalComposition* head = createMusicalComposition(array_names[0],
array_authors[0], array_years[0]);
    MusicalComposition* tmp = head;
    for (int i = 1; i < n; i++){
        MusicalComposition*                     new                     =
createMusicalComposition(array_names[i],              array_authors[i],
array_years[i]);
        new->prev = tmp;
        tmp->next = new;
        tmp = new;
    }
    return head;
```

```
}

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

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

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

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

}
```