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. 0382 _____ Здобнова К.Д.

Преподаватель _____ Берленко Т.А.

Санкт-Петербург

2021

Цель работы.

Изучить структуру данных «списки», реализовать двунаправленный список на языке Си.

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:

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

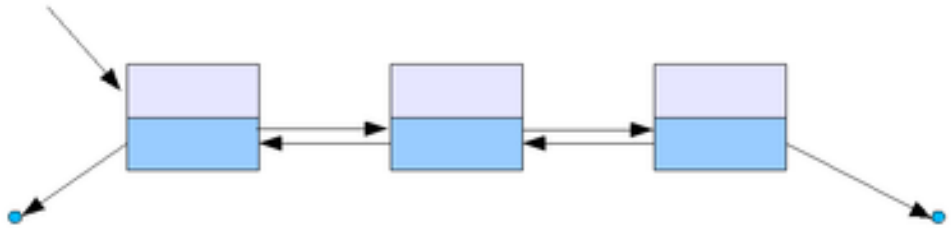
void print_names(MusicalComposition* head); //Выводит названия композиций

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

Основные теоритические сведения.

Список - некоторый упорядоченный набор элементов любой природы.

Линейный двунаправленный список - список, каждый элемент которого хранит помимо значения указатель на предыдущий и на следующий элемент. В последнем элементе указатель на следующий элемент равен NULL (также указатель на предыдущий элемент у первого элемента).



Выполнение работы.

Разработана структура MusicalComposition, с полями char name (название песни), char author (автор песни), int year (год создания); prev и next – указатели на предыдущий и следующий элемент соответственно.

Функция MusicalComposition* createMusicalComposition(char* name, char* author,int year) является конструктором экземпляра MusicalComposition. Принимает данные композиций и возвращается указатель на готовый экземпляр.

Функция MusicalComposition* createMusicalCompositionList(char** array_names, char** array_authors, int* array_years, int n) создает двунаправленный список из полученных масиивов и возвращает указатель на первый элемент head.

Функция void push(MusicalComposition* head, MusicalComposition* element) добавляет входной элемент в конец списка.

Функция void removeEl(MusicalComposition* head, char* name_for_remove) удаляет элемент, поле name которого равен name_for_remove.

Функция int count(MusicalComposition* head) считает количество элементов в списке.

Функция void print_names(MusicalComposition* head) печатает на экран значения поля name элементов.

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

Выводы.

Были изучены списки, реализован код для работы с двунаправленным списком.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: *main.c*

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

// Описание структуры MusicalComposition

struct MusicalComposition{
    char name[81];
    char author[81];
    int year;
    struct MusicalComposition* prev;
    struct MusicalComposition* next;
};

typedef struct MusicalComposition MusicalComposition;

// Создание структуры MusicalComposition

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

// Функции для работы со списком MusicalComposition

MusicalComposition* createMusicalCompositionList(char** array_names, char**
array_authors, int* array_years, int n){
    MusicalComposition* head = (struct
MusicalComposition*)malloc(sizeof(struct MusicalComposition));
    MusicalComposition* cur = head;
    MusicalComposition* prev = NULL;
    for(int i = 0; i < n; i++){
        cur->prev = prev;
```

```
        strcpy(cur->name, array_names[i]);
        strcpy(cur->author, array_authors[i]);
        cur->year = array_years[i];
        prev = cur;
        if (i < n - 1){
            cur = (struct MusicalComposition*)malloc(sizeof(struct
MusicalComposition));
            prev->next = cur;
        }
    }
    prev->next = NULL;
    return head;
}

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

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

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

void print_names(MusicalComposition* head){
```

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