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. 3384 _____ Рудаков А.Л.

Преподаватель _____ Глазунов С.А.

Санкт-Петербург
2024

Цель работы.

Изучить теорию работы с линейными списками и их использованием в языке С. Получить практические навыки путем написания программного кода.

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.

Функцию main менять не нужно.

Выполнение работы.

В начале была описана структура *MusicalComposition*, элементами которой являются *char* name* — название композиции, *char* author* — группа или автор композиции, *int year* — год выпуска композиции, s*truct MusicalComposition* next* — указатель на следующий элемент линейного списка, *struct MusicalComposition* previous* — указатель на предыдущий элемент линейного списка. Также использован *typedef* для замены *struct MusicalComposition* на *MusicalComposition*.

Далее написана функция *MusicalComposition* createMusicalComposition(char* name, char* author,int year)*, принимающая на вход указатель на название, указатель на автора и год выпуска. Далее создается указатель на структуру *MusicalComposition* и динамически выделяется память под неё. Далее полям структуры *Composition→name*, *Composition→author*,

Composition→year, присваиваются значения *name*, *author* и *year* соответственно. После чего при помощи оператора *return* возвращается указатель на данную структуру.

Далее написана функция *createMusicalCompositionList(char** array_names, char** array_authors, int* array_years, int n)*, принимающая на вход указатель на массив строк с названиями, указатель на массив строк с авторами, указатель на массив чисел с годами и количество элементов в массивах. Внутри инициализируется 3 указателя на структуру *Composition*, *PreviousComposition* и *BeginComposition*. Далее идет цикл, проходящий по всему массиву. Внутри при помощи функции *createMusicalComposition()* создается структура с аргументами *array_names[i]*, *array_authors[i]*, *array_years[i]* и ее указатель присваивается *Composition*. Далее, если это первый элемент, то *Composition→previous=NULL*, а также указатель *BeginComposition* присваивается указатель *Composition*. В противном случае *Composition→previous=PreviousComposition*, *Composition→previous→next=Composition*. После условного оператора идет присвоение указателю *PreviousComposition* указателя *Composition*. Далее *Composition→next=NULL*. И при помощи оператора *return* возвращается указатель на первую структуру *BeginComposition*.

Далее идет функция *void push(MusicalComposition* head, MusicalComposition* element)*, которая принимает на вход указатель на первый элемент линейного списка и указатель на элемент, который нужно добавить. Далее создается *MusicalComposition* tek* - указатель на текущий элемент списка, которому присваивается начальное значение *head*. После чего идет цикл *while*, благодаря которому происходит проход по списку до последнего элемента, после чего в поле *next* последнего элемента помещается указатель на новый элемент, в поле *previous* нового элемента помещается указатель на последний элемент списка, в поле *next* нового элемента записывается *NULL*.

Далее идет функция *void removeEl(MusicalComposition* head, char* name_for_remove)*, которая получает на вход указатель на первый элемент списка и название композиции, с которым нужно удалить элемент. После чего создается

указатель на текущий элемент списка *tek*, которому присваивается значение *head* и *tek2*, который в дальнейшем будет использован для удаления указателя на элемент. Далее идет цикл *while*, проходящий по всему списку. Далее идет сравнение названия, которое нужно удалить с названием текущего произведения. Если они совпадают и текущий элемент не является первым и последним, то *tek→previous→next=tek→next*, *tek→next→previous=tek→previous*. Если же этот элемент является последним, то *tek→previous→next=NULL*. Если элемент является первым, то *tek→next→previous=NULL*. Далее *tek2* присваивается указатель *tek*, *tek=tek→next*, указатель *tek2* очищается. Если же названия композиций разные, то *tek=tek→next*.

Далее идет функция *int count(MusicalComposition* head)*, получающая на вход указатель на первый элемент списка. Внутри инициализируется переменная *k*, изначальное значение которой равно 1. Далее идет цикл *while*, проходящий до конца списка, в котором переменная *k* увеличивается на 1. После чего значение *k* возвращается при помощи *return*.

После написана функция *void print_names(MusicalComposition* head)*, которая принимает на вход указатель на первый элемент списка. Внутри инициализируется указатель *tek*, которому передается значение указателя *head*. Далее идет цикл *while*, проходящий по всему списку, внутри которого на экран выводятся названия композиций текущих элементов списка.

Разработанный программный код см. в приложении А.

Тестирование см. в приложении Б.

Выводы.

Была изучена теория работы с линейными списками и их использования в языке С. В работе были созданы структуры, из которых был создан линейным список. В каждой структуре содержался указатель на предыдущий и следующий элементы списка.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main_lb2.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* author,int
year){
    MusicalComposition* Composition =
(MusicalComposition*)malloc(sizeof(MusicalComposition));
    Composition->name=name;
    Composition->author=author;
    Composition->year=year;
    return Composition;
}

// Функции для работы со списком MusicalComposition

MusicalComposition* createMusicalCompositionList(char** array_names,
char** array_authors, int* array_years, int n){
    MusicalComposition* Composition;
    MusicalComposition* PreviousComposition;
    MusicalComposition* BeginComposition;
    for (int i=0; i<n; i++){

Composition=createMusicalComposition(array_names[i],array_authors[i],arra
y_years[i]);
        if (i==0){
            Composition->previous=NULL;
            BeginComposition=Composition;
        } else{
            Composition->previous=PreviousComposition;
            Composition->previous->next=Composition;
        }
        PreviousComposition=Composition;
    }
    Composition->next=NULL;
    return BeginComposition;
}

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

```
    MusicalComposition* tek=head;
    while (tek->next!=NULL){
        tek=tek->next;
    }
    tek->next=element;
    element->next=NULL;
    element->previous=tek;
}

void removeEl(MusicalComposition* head, char* name_for_remove){
    MusicalComposition* tek=head;
    MusicalComposition* tek2;
    while (tek!=NULL){
        if (strcmp(tek->name,name_for_remove)==0){
            if (tek->previous!=NULL){
                if (tek->next!=NULL){
                    tek->previous->next=tek->next;
                    tek->next->previous=tek->previous;
                } else {
                    tek->previous->next=NULL;
                }
            } else if (tek->next!=NULL){
                tek->next->previous=NULL;
            }
            tek2=tek;
            tek=tek->next;
            free(tek2);
        } else{
            tek=tek->next;
        }
    }
}

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

void print_names(MusicalComposition* head){
    MusicalComposition* tek=head;
    while (tek!=NULL){
        puts(tek->name);
        tek=tek->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;

}
```

ПРИЛОЖЕНИЕ Б
ТЕСТИРОВАНИЕ

Таблица Б.1 - Примеры тестовых случаев

№ 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	Верно