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. 8383 _____ Дейнега В.Е.

Преподаватель _____ Берленко Т. А.

Санкт-Петербург
2019

Цель работы.

Реализовать двунаправленный линейный список и api (application programming interface - в данном случае набор функций) для работы с ним на языке программирования Си.

Основные теоретические положения.

Линейный однонаправленный список

Список - некоторый упорядоченный набор элементов любой природы.
Линейный однонаправленный (односвязный) список - список, каждый элемент которого хранит помимо значения указатель на следующий
элемент. В последнем элементе указатель на следующий элемент равен NULL (константа нулевого указателя).

Линейный двусвязный список

-базовая динамическая структура данных в информатике, состоящая из узлов, каждый из которых содержит как собственно данные, так и одну или две ссылки на следующий и/или предыдущий узел списка. Принципиальным преимуществом перед массивом является структурная гибкость: порядок элементов связного списка может не совпадать с порядком расположения элементов данных в памяти компьютера.

Каждый узел двунаправленного (двусвязного) линейного списка (ДЛС) содержит два поля указателей — на следующий и на предыдущий узлы. Указатель на предыдущий узел корня списка содержит нулевое значение. Указатель на следующий узел последнего узла также содержит нулевое значение.

Реализация:

1. Описана структура MusicalComposition, представляющая собой элемент списка, содержащая 2 строки и 2 ссылки (на следующий элемент списка и не предыдущий).

2.Описаны следующие функции:

2.1 MusicalComposition* createMusicalComposition(char* name, char* author, int year);

Функция выделяет память под новый элемент списка, заполняет поля данными и возвращает указатель на созданный элемент.

2.2 MusicalComposition* createMusicalCompositionList(char** array_names, char** array_authors, int* array_years, int n);

Функция создает двусвязный список из элементов, используя вышеописанную функцию, возвращает ссылку на голову списка.

2.3 void push(MusicalComposition* head, MusicalComposition* element);

Функция добавляет элемент в конец списка.

2.4 void removeEl (MusicalComposition* head, char* name_for_remove);

Функция ищет элемент с заданным именем, а затем удаляет его из списка.

2.5 int count(MusicalComposition* head);

Функция проходится по всему списку и возвращает количество элементов в нем.

2.6 void print_names(MusicalComposition* head);

Функция печатает поля name каждого элемента списка.

3. Функция main дана по условию.

Тестирование программы:

Ввод:

```
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

Вывод:
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

Выводы.

В ходе выполнения задания лабораторной работы были изучены линейный и двусвязный список, а также их реализация на языке программирования си.

ПРИЛОЖЕНИЕ

КОД ПРОГРАММЫ:

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


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


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


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 = createMusicalComposition(array_names[1],
array_authors[1], array_years[1]);
    head->next = tmp;
    tmp->prev = head;
    for (int i=2; i<n; i++){
        tmp->next = createMusicalComposition(array_names[i],
array_authors[i], array_years[i]);
        tmp->next->prev = tmp;
        tmp = tmp->next;
      }
    return head;
}


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

```
void removeEl (MusicalComposition*  head, char* name_for_remove){
    MusicalComposition* elem = head;
    while(strcmp(elem->name, name_for_remove)){
    elem = elem->next;
    }
    elem->next->prev = elem->prev;
    elem->prev->next = elem->next;
}


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


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


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;
 }
```