**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

**отчет**

**по лабораторной работе №4**

**по дисциплине «Программирование»**

**Тема: Линейные списки**

|  |  |  |
| --- | --- | --- |
| Студент гр. 6303 |  | Архипцев Е.Д. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2016

**Цель работы**

Научиться использовать линейные списки, создать api (application programming interface) для работы со списком.

**Ход работы**

Ниже приведен код программы:

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

typedef struct MusicalComposition

{

char\* name;

char\* author;

int year;

struct MusicalComposition \*next;

struct MusicalComposition \*prev;

}MusicalComposition;

MusicalComposition\* createMusicalComposition(char\* name, char\* author, int year)

{

MusicalComposition\* MusCom = (MusicalComposition\*)malloc(sizeof(MusicalComposition));

MusCom->name = name;

MusCom->author = author;

MusCom->year = year;

MusCom->next = NULL;

MusCom->prev = NULL;

return MusCom;

}

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 \*prev = head;

MusicalComposition\* current;

int i;

for(i = 1; i < n; i++)

{

current = createMusicalComposition(array\_names[i], array\_authors[i], array\_years[i]);

current->prev = prev;

prev->next = current;

prev = current;

}

return head;

}

void push(MusicalComposition\* head, MusicalComposition\* element)

{

MusicalComposition\* current = head;

while(current->next)

current = current->next;

current->next = element;

element->prev = current;

}

void removeEl(MusicalComposition\* head, char\* name\_for\_remove)

{

MusicalComposition\* current = head;

while(current->next)

{

if(strcmp(current->name, name\_for\_remove) == 0)

{

if(current->next != NULL && current->prev != NULL)

{

current->prev->next = current->next;

current->next->prev = current->prev;

}

else if(current->next == NULL)

{

current->prev->next = NULL;

}

else if(current->prev == NULL)

{

current = current->next;

current->prev = NULL;

\*head = \*current;

}

}

current = current->next;

}

}

int count(MusicalComposition\* head)

{

MusicalComposition\* current = head;

int count = 0;

while (current)

{

++count;

current = current->next;

}

return count;

}

void print\_names(MusicalComposition\* head)

{

MusicalComposition\* current = head;

while (current)

{

printf("%s\n", current->name);

current = current->next;

}

}

int main(){

int length, i;

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 (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);

return 0;

}

**Вывод**

В ходе лабораторной работы получены навыки работы с линейными списками, создан api(application programming interface) для работы со списком, с помощью которого реализована программа для работы со списком музыкальных композиций.