**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МО ЭВМ**

отчет

**по лабораторной работе №4**

**по дисциплине «Программирование»**

Тема: *Линейные списки*

|  |  |  |
| --- | --- | --- |
| Студент гр. 6303 |  | Кулигин А.И. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2016

**Цель работы:**научиться использовать линейные списки, создать **api (application programming interface)** для работы со списком.

**Ход работы:**

В ветке Kuligin\_lr4 репозитория pr-2016-6303 создал папку Lab4 с проектом, состоящим из 1 файла***lab.c***

Основная часть программы ***lab4*** вызывает функцию из набора **api**.

***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); // создаетсписокмузыкальныхкомпозиций MusicalCompositionList, вкотором:***
  + ***n - длинамассивов 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 вконецсписка musical\_composition\_list***
* ***void removeEl (MusicalComposition\* head, char\* name\_for\_remove); // удаляетэлемент element списка, укоторогозначение name равнозначениюname\_for\_remove***
* ***int count(MusicalComposition\* head); //возвращает количество элементов списка***

***void print\_names(MusicalComposition\* head); //Выводитназваниякомпозиций***

Ниже приведен код программы:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct MusicalComposition

{

char \*name;

char \*author;

int year;

int status;

struct MusicalComposition \*next, \*prev;

};

typedef struct MusicalComposition MusicalComposition;

// Функции для работы со списком MusicalComposition

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

{

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

MusComp->name = name;

MusComp->author = author;

MusComp->year = year;

MusComp->status = 1;

MusComp->prev = NULL;

MusComp->next = NULL;

return MusComp;

}

MusicalComposition \*createMusicalCompositionList()

{

int length;

scanf("%d", &length);

getchar();

char \*\*name = (char\*\*)malloc(sizeof(char\*)\*length);

char \*\*author = (char\*\*)malloc(sizeof(char\*)\*length);

int \*year = (int\*)malloc(sizeof(int)\*length);

for (int i = 0; i < length; ++i)

{

name[i] = (char\*)malloc(sizeof(char\*)\*81);

author[i] = (char\*)malloc(sizeof(char\*)\*81);

gets(name[i]);

gets(author[i]);

scanf("%d", &year[i]);

getchar();

}

MusicalComposition \*head = createMusicalComposition(name[0], author[0], year[0]);

MusicalComposition \*prev = head;

MusicalComposition \*curr;

for (int i = 1; i < length; ++i)

{

curr = createMusicalComposition(name[i], author[i], year[i]);

curr->prev = prev;

prev->next = curr;

prev = curr;

}

return head;

}

void push(MusicalComposition \*head) // добавляет element в конец списка musical\_composition\_list

{

char \*name\_for\_push = (char\*)malloc(sizeof(char)\*81);

char \*author\_for\_push = (char\*)malloc(sizeof(char)\*81);

int year\_for\_push;

gets(name\_for\_push);

gets(author\_for\_push);

scanf("%d", &year\_for\_push);

getchar();

MusicalComposition \*element\_for\_push = createMusicalComposition(name\_for\_push, author\_for\_push, year\_for\_push);

if ((head->status) == 0)

memcpy(head, element\_for\_push, sizeof(MusicalComposition));

else

{

while (head->next)

head = head->next;

element\_for\_push->prev = head;

head->next = element\_for\_push;

}

}

int count(MusicalComposition \*head)

{

int count = 0;

if ((head->status) == 0)

return 0;

else

while (head)

{

++count;

head = head->next;

}

return count;

}

void removeEl(MusicalComposition \*head)// удаляет элемент element списка, у которого значение name равно значению name\_for\_remove

{

char name\_for\_remove[81];

int amount = count(head);//возвращает количество элементов списка

MusicalComposition \*temp = head;

MusicalComposition \*curr;

gets(name\_for\_remove);

while (temp->next)

{

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

{

if ((temp->prev) == NULL)

{

temp = head->next;

memcpy(head, temp, sizeof(MusicalComposition));

head->prev = NULL;

free(temp);

temp = head;

}

else

{

temp->prev->next = temp->next;

temp->next->prev = temp->prev;

curr = temp->next;

free(temp);

temp = curr;

}

--amount;

}

else

temp = temp->next;

}

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

{

if (amount != 1)

{

temp->prev->next = NULL;

free(temp);

}

else

head->status = 0;

}

}

void print\_names(MusicalComposition \*head)//Выводит названия композиций

{

if (head->status)

while (head)

{

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

head = head->next;

}

}

int main()

{

//Функция createMusicalCompositionList всегда вызывается первой и всего один раз

MusicalComposition \*head = createMusicalCompositionList();

printf("%s %s %d\n", head->name, head->author, head->year);

printf("%d\n", count(head));

push(head);

printf("%d\n", count(head));

removeEl(head);

print\_names(head);

printf("%d\n", count(head));

return 0;

}

Созданная папка с файлом загружена в репозиторий на Github с помощью следующих команд:

* git add Lab4
* git commit –m Lab4
* git push origin fokin\_lr4

**Вывод:** в ходе лабораторной работы получены навыки работы со структурами и списками в языке Си, которые успешно применены на практике при создании программы ***lab4.c***.