**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 на С.

|  |  |  |
| --- | --- | --- |
| Студент гр. 6304 |  | Запевалов А.И. |
| Преподаватель |  |  |

Санкт-Петербург

2016

Оглавление

[1. Цель работы. 2](#_Toc470339632)

[2. Ход работы. 2](#_Toc470339633)

[Объявление структуры 2](#_Toc470339634)

[Создание списка 2](#_Toc470339635)

[Добавление элемента/элементов в конец списка 3](#_Toc470339636)

[Удаление элемента 3](#_Toc470339637)

[Подсчёт элементов и вывод 4](#_Toc470339638)

[Код программы полностью. 4](#_Toc470339639)

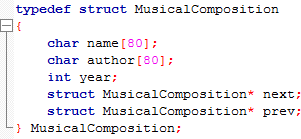
[3. Вывод. 8](#_Toc470339640)

1. **Цель работы.**

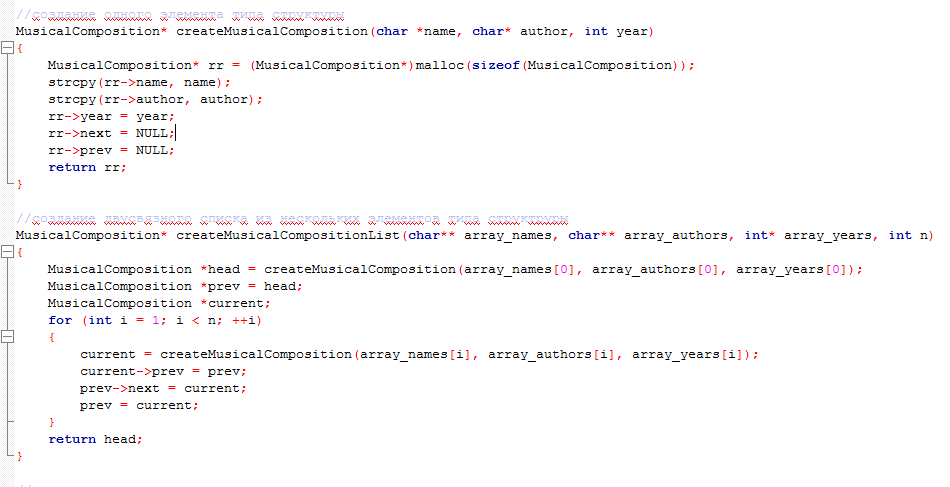
Научиться работать с двунаправленным списком: создавать, удалять и перемещать его элементы.

1. **Ход работы.**

**Объявление структуры**

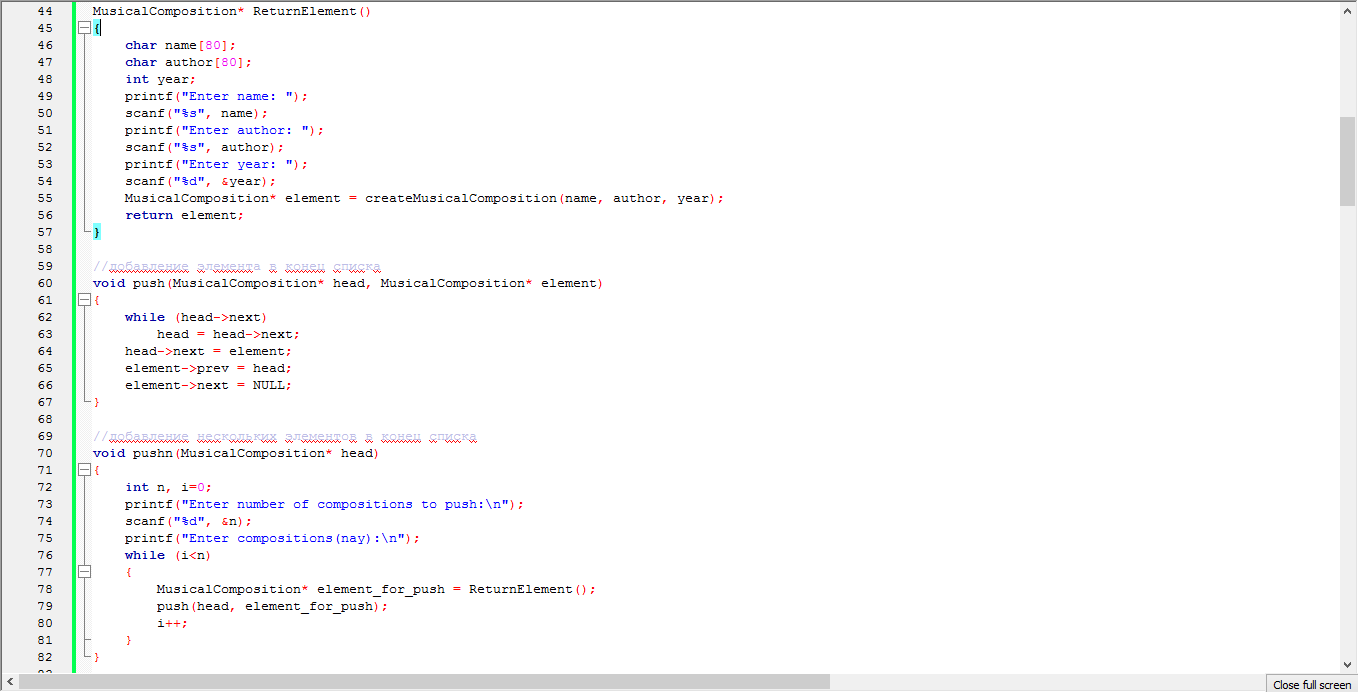


**Создание списка**



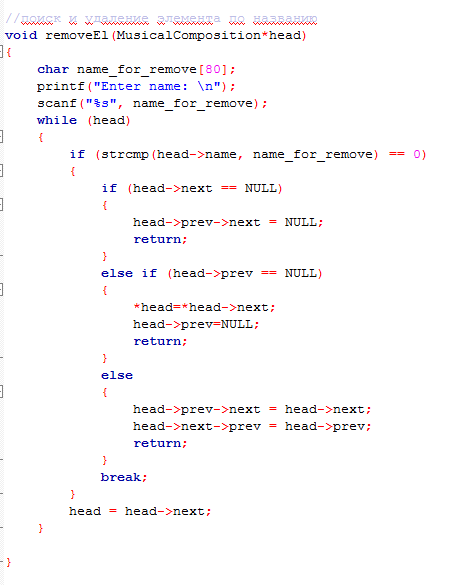
Функция createMusicalCompositionList получает на вход массивы, которые записывает в структуры. Сначала создается голова списка и в неё записываются первые значения массивов, а потом в head->next создается следующая структура, после чего происходит head = head->next. Так продолжается, пока не кончатся массивы.

**Добавление элемента/элементов в конец списка**

****

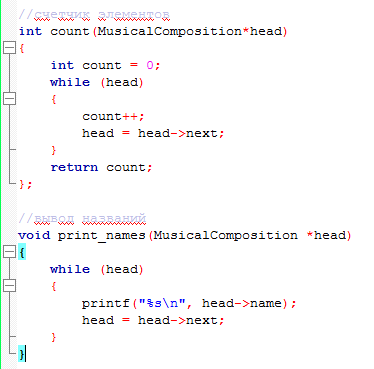
Происходит следующим образом – создается элемент типа структуры в функции ReturnElement, после чего добавляется в конец списка при помощи функции push. В случае с несколькими элементами предыдущие функции вызываются указанное в функции pushn количество раз.

**Удаление элемента**



В случае, если элемент не в конце, указатели предыдущего и следующего элементов «перебрасываются» через текущий, а если элемент в конце – указателю next в предпоследнем элементе присваивается NULL. Если же элемент начале, то указатель на головной элемент смещается на следующий элемент, его указателю на предыдущий элемент присваивается NULL.

**Подсчёт элементов и вывод**



Подсчет элементов происходит прибавлением 1 к переменной и перемещением вперед по списку до достижения конца, т.е. NULL pointer. Вывод на печать названия каждого элемента происходит аналогично.

**Код программы полностью.**

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

struct MusicalComposition {

char name[80];

char author[80];

int year;

struct MusicalComposition\* next;

struct MusicalComposition\* prev;

};

typedef struct MusicalComposition MusicalComposition;

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

{

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

strcpy(rr->name, name);

strcpy(rr->author, author);

rr->year = year;

rr->next = NULL;

rr->prev = NULL;

return rr;

}

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;

for (int 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)

{

while (head->next)

head = head->next;

head->next = element;

element->prev = head;

element->next = NULL;

}

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

{

while (head)

{

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

{

if (head->next == NULL)

{

head->prev->next = NULL;

return;

}

else if (head->prev == NULL)

{

head->next->prev = NULL;

return;

}

head->prev->next = head->next;

head->next->prev = head->prev;

break;

}

head = head->next;

}

}

int count(MusicalComposition\*head)

{

int count = 0;

while (head)

{

count++;

head = head->next;

}

return count;

};

void print\_names(MusicalComposition \*head)

{

while (head)

{

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

head = head->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);

return 0;

}

1. **Вывод.**

Мы научились работать со списком: создавать, добавлять, удалять, его элементы.