Государственный комитет Российской Федерации по телекоммуникациям

Сибирский государственный университет

телекоммуникаций и информатики

**КУРСОВАЯ РАБОТА**

**По дисциплине « Программирование на языках высокого уровня»**

**Вариант № 4**

Работу выполнил

студент Мгдесян К.А.

группы ПБЗ-59

**Новосибирск, 2015**

Задание:

Разработать программу для создания и работы с двусвязным списком, состоящим из структур. Для работы со списком создать меню со следующими пунктами:

1. Создание списка.

2. Просмотр списка.

3. Добавление в список новой записи.

4. Поиск и корректировка записи в списке.

5. Удаление записи из списка.

6. Сохранение списка в файле.

7. Загрузка списка из файла.

8. Выход.

Структура содержит название книги, ф.и.о. автора, год издания. Удалять из списка издания с годом меньше заданного. Изменять год издания по введённому названию книги. **Добавлять новые записи в конец списка.**

Оглавление

[Введение 4](#_Toc417403762)

[1. Постановка комплекса задач 5](#_Toc417403763)

[2. Блок-схема функционирования системы 7](#_Toc417403764)

[3. Блок-схема добавления записи 8](#_Toc417403765)

[4. Проектный раздел 9](#_Toc417403766)

[5. Результаты тестирования программы 15](#_Toc417403767)

[Список литературы 17](#_Toc417403768)

# Введение

Данная работа посвящена разработке программ на языке программирования высокого уровня Си. Целью курсовой работы является закрепление основ и углубление знаний приемов программирования на языке Си, получение практических навыков в создании программного продукта.

Задачей курсовой работы является разработка программы для создания и обработки двусвязного списка структур. Работа со списком должна осуществляться посредством меню.

# Постановка комплекса задач

Задача: Разработать программу для создания и работы с двусвязным списком, состоящим из структур. Для работы со списком создать меню со следующими пунктами:

1. Создание списка.

2. Просмотр списка.

3. Добавление в список новой записи.

4. Поиск и корректировка записи в списке.

5. Удаление записи из списка.

6. Сохранение списка в файле.

7. Загрузка списка из файла.

8. Выход.

Структура содержит название книги, ф.и.о. автора, год издания. Удалять книги заданного автора. Изменять год издания по введённому названию книги. **Добавлять новые записи перед последней**.

Структура – это набор элементов разных типов. Элементами структуры могут быть базовые типы, массивы, указатели, структуры и т.д. Элементы структуры вместе с их описанием называются полями. Над полями можно выполнять действия, допустимые для данных этого типа.

Список – это последовательность структур, каждая из которых содержит ссылку, связывающую её с другой структурой. Для организации списков используются структуры, состоящие из двух смысловых частей – информационной и дополнительной. Информационная часть содержит подлежащую обработке информацию, в дополнительной находятся указатели на последующую или предыдущую структуру списка. В двусвязном списке каждая структура содержит две ссылки: на предыдущую и последующую структуры. Таким образом, по списку можно перемещаться от начала к концу и от конца к началу. Для доступа к началу и концу списка должны быть известны их адреса, которые могут сохраняться в переменных типа указатель.

Задача может быть разбита на несколько подзадач:

* создание списка;
* вывод содержимого списка;
* добавление в список новых элементов;
* корректировка списка;
* удаление записи из списка;
* сохранение списка в файле;
* загрузка списка из файла;
* организация интерфейса.

Организацию интерфейса логично расположить в основной программе, а подзадачи работы со списком оформить в виде функций, вызываемых из основной программы в зависимости от действий пользователя.

# Блок-схема функционирования системы



### Блок-схема добавления записи



# Проектный раздел

При решении данной задачи интерфейс пользователя реализован в основной программе. При запуске на экран выводится меню, в котором предлагается выбрать действие из списка:

1. Создание списка.

2. Просмотр списка.

3. Добавление в список новой записи.

4. Поиск и корректировка записи в списке.

5. Удаление записи из списка.

6. Сохранение списка в файле.

7. Загрузка списка из файла.

8. Выход.

После выбора пользователем пункта меню вызывается соответствующая выбору функция либо, в случае выбора пункта 8 меню, выход из программы. После обработки выбора пользователя управление возвращается в основную программу.

Каждый элемент двусвязного списка содержит следующие поля:

* title – название книги;
* name – фамилия автора;
* year – год издания;
* prev – указатель на предыдущий элемент;
* next – указатель на следующий элемент.

Для работы со списком имеются следующие указатели:

* head – указатель на первый элемент списка;
* tail – указатель на последний элемент списка.

**Описание отдельных функций**

Функция Menu() выводит меню на экран.

Функция Clear() очищает список, т.е. удаляет все элементы списка из динамической памяти и обнуляет указатели на начало и конец списка.

Функция Print() выводит список на экран. Каждый элемент списка выводится в отдельной строке.

Функция Add() добавляет элемент перед последним элементом списка. Выделяет для элемента списка динамическую память, запрашивает ввод с клавиатуры всех полей элемента списка и настраивает связи между элементами списка.

Функция Change() изменяет год издания по названию книги. Запрашивает ввод названия книги и, в случае наличия такого названия в списке, запрашивает ввод года издания.

Функция Delete() удаляет из списка. Запрашивает ввод фамилии автора и, в случае наличия в списке изданий с заданным автором, удаляет запись. При этом при удалении элемента происходит перенастройка связей между элементами списка, соседними с удаляемым.

Функция Save() сохраняет список в файле inf.dat. Записи списка сохраняются в файле построчно.

Функция Load() загружает список из файла inf.dat. Перед загрузкой список уничтожается из динамической памяти.Исходный модуль программы

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <alloc.h>

#include <string.h>

const char FileName[] = "inf.dat"; // Имя файла

// Структура элемента двусвязного списка

struct Inf

{

char title[30]; // Название книги

char name[20]; // Фамилия автора

int year; // Год издания

Inf \*prev; // Указатель на предыдущий элемент

Inf \*next; // Указатель на следующий элемент

};

Inf \*head = NULL; // Указатель на первый элемент списка

Inf \*tail = NULL; // Указатель на последний элемент списка

// Выводит меню на экран

void Menu()

{

puts("1 - Создание списка");

puts("2 - Просмотр списка");

puts("3 - Добавление в список новой записи");

puts("4 - Поиск и корректировка записи в списке");

puts("5 - Удаление записи из списка");

puts("6 - Сохранение списка в файле");

puts("7 - Загрузка списка из файла");

puts("8 - Выход\n");

}

// Очищает список

void Clear()

{

Inf \*p = head;

while (p != NULL) {

Inf \*q = p;

p = p->next;

free(q); // Удаляем элемент списка из динамической памяти

}

head = NULL;

tail = NULL;

}

// Выводит список на экран

void Print()

{

Inf \*p = head;

while (p != NULL) { // Цикл по всем элементам списка

printf("%30s ", p->title);

printf("%20s ", p->name);

printf("%10d\n", p->year);

p = p->next;

}

}

// Добавляет новую запись перед последней

void Add()

{

Inf \*p = (Inf \*) malloc(sizeof(Inf)); // Выделяем динамическую память

printf("Название книги: "); // Вводим все поля элемента списка

scanf("%s", p->title);

printf("Фамилия автора: ");

scanf("%s", p->name);

printf("Год издания: ");

scanf("%d", &p->year);

p->prev = NULL;

p->next = NULL;

if (head == NULL) // Если список пустой

head = tail = p;

else if (head->next == NULL) { // Если одна запись в списке

p->next = head;

head->prev = p;

head = p;

}

else {

p->prev = tail->prev;

p->next = tail;

p->prev->next = p;

tail->prev = p;

}

}

// Изменяет год издания по названию книги

void Change()

{

char title[30];

printf("Название: "); // Вводим название книги

scanf("%s", title);

Inf \*p = head;

int found = 0;

while (p != NULL) { // Цикл по всем элементам списка

if (!strcmp(p->title, title)) {

printf("Год издания: ");

scanf("%d", &p->year);

found = 1;

break;

}

p = p->next;

}

if (found == 0)

puts("Книга не найдена");

else

puts("Год издания изменен");

}

// Удаляет из списка книги заданного автора

void Delete()

{

char name[20];

printf("Введите фамилию автора: ");

scanf("%s", name);

Inf \*p = head;

int found = 0;

while (p != NULL) {

if (!strcmp(p->name, name)) {

// Если удаляемый элемент является одновременно первым и последним

if (p->prev == NULL && p->next == NULL) {

free(p); // Удаляем элемент списка из динамической памяти

head = NULL;

tail = NULL;

break;

}

// Если удаляем первый элемент

else if (p->prev == NULL) {

Inf \*q = p;

p = p->next;

free(q); // Удаляем элемент списка из динамической памяти

p->prev = NULL;

head = p;

}

// Если удаляем последний элемент

else if (p->next == NULL) {

Inf \*q = p;

p = p->prev;

free(q); // Удаляем элемент списка из динамической памяти

p->next = NULL;

tail = p;

break;

}

// Если удаляем элемент из середины списка

else {

p->prev->next = p->next;

p->next->prev = p->prev;

Inf \*q = p;

p = p->next;

free(q); // Удаляем элемент списка из динамической памяти

}

found = 1;

}

else

p = p->next;

}

if (found == 0)

puts("Издания не найдены");

else

puts("Издания удалены");

}

// Сохраняет список в файле

void Save()

{

FILE \*f; // файл

f = fopen(FileName, "w"); // открываем файл для записи

Inf \*p = head;

while (p != NULL) { // Цикл по всем элементам списка

fprintf(f, "%s %s %d\n", p->title, p->name, p->year);

p = p->next;

}

fclose(f);

printf("Список сохранен в файле %s\n", FileName);

}

// Загружает список из файла

void Load()

{

Clear();

FILE \*f; // файл

f = fopen(FileName, "r"); // открываем файл для чтения

while (!feof(f)) {

Inf \*p = (Inf \*) malloc(sizeof(Inf)); // Выделяем динамическую память

fscanf(f, "%s", p->title); // Вводим все поля элемента списка

if (feof(f))

break;

fscanf(f, "%s", p->name);

fscanf(f, "%d", &p->year);

p->prev = NULL;

p->next = NULL;

if (head == NULL) // Если список пустой

head = tail = p;

else {

tail->next = p;

p->prev = tail;

tail = p;

}

}

fclose(f);

printf("Список загружен из файла %s\n", FileName);

}

void main()

{

clrscr(); // Очищаем экран

Menu();

char c;

while (1) {

c = getch();

switch(c) {

case '1': // Создание списка

Clear();

puts("Список создан");

break;

case '2': // Просмотр списка

puts("Список:");

Print();

break;

case '3': // Добавление в список новой записи

Add();

break;

case '4': // Поиск и корректировка записи в списке

Change();

break;

case '5': // Удаление записи из списка

Delete();

break;

case '6': // Сохранение списка в файле

Save();

break;

case '7': // Загрузка списка из файла

Load();

break;

case '8': // Выход

return;

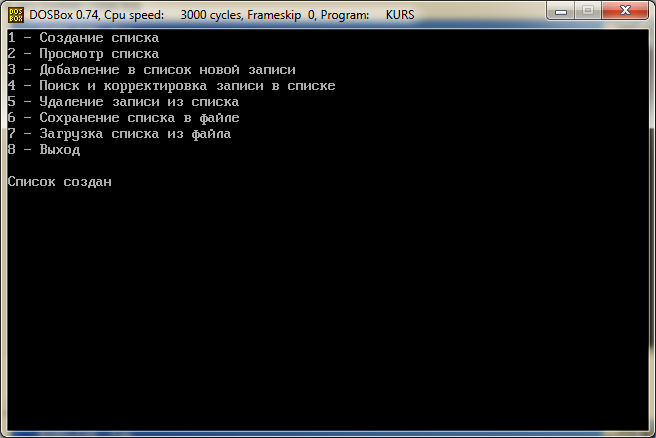
}

}

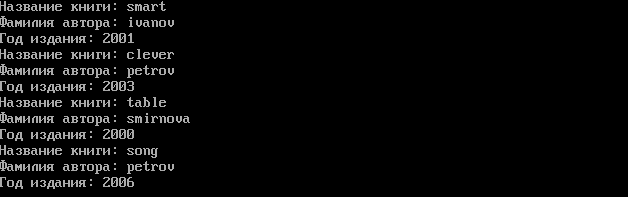
}

# Результаты тестирования программы

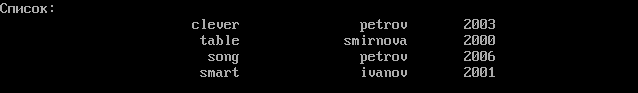
Создание списка



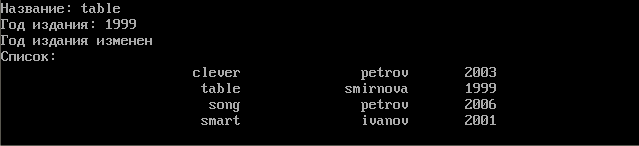
Добавление четырех элементов



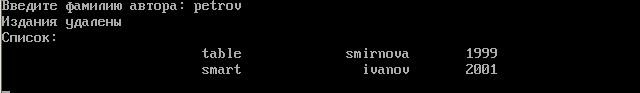
Просмотр списка



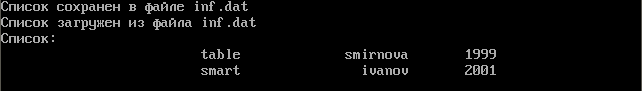
Корректировка списка



Удаление из списка



Сохрание в файле и загрузка из файла



# Список литературы

1. Подбельский В.В., Фомин С.С. Программирование на языке Си: Учеб.пособие. – 2-е доп. изд. – М.: Финансы и статистика, 2004.- 600с.:ил.
2. Шилдт Г.Самоучитель C++: Пер. с англ. — 3-е изд. — [BHV - Санкт - Петербург](http://www.ozon.ru/context/detail/id/856483/), 2006 г. — 688 с.