**Министерство науки и высшего образования РФ**

**Пензенский государственный университет**

**Кафедра «Вычислительная техника»**

**ОТЧЁТ**

По лабораторной работе №11

По дисциплине «Программирование»

На тему «Динамические структуры данных»

Вариант №9

**Выполнили студенты группы 24ВВВ2:**

Лисицын Н.С.

Кажичкин Д.А.

Железнов Д.В.

**Приняли:**

Юрова О.В.

Патунин Д.В.

**Пенза, 2024**

**Цель работы**

Изучение способов организации и описания динамических структур, приобретение навыков работы с ними.

**Лабораторное задание**

Данная лабораторная работа выполняется на основе задания к лабораторной работе №9 "Структуры".

1) Выбрать задание к лабораторной работе "Структуры" в соответствии с номером бригады.

2) Оформить структуру данных в виде динамического списка.

3) Разработать программу, осуществляющую:

а) ввод исходных данных в список, завершение ввода должно отмечаться специальным символом (в список должно быть введено не менее пяти записей);

б) вывод исходных данных на экран в виде таблицы;

в) перестановку местами первого и последнего элементов динамического списка;

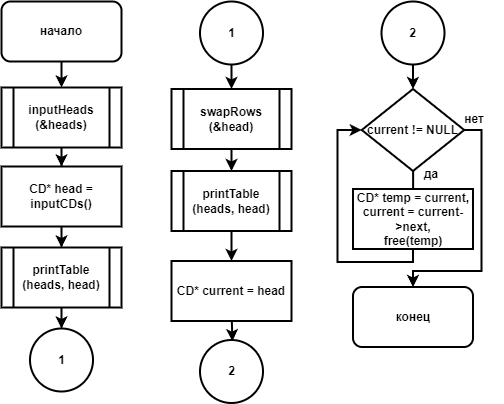
г) вывод на экран модифицированного списка в виде таблицы;

4) Выполнить программу на компьютере и оценить правильность ее работы.

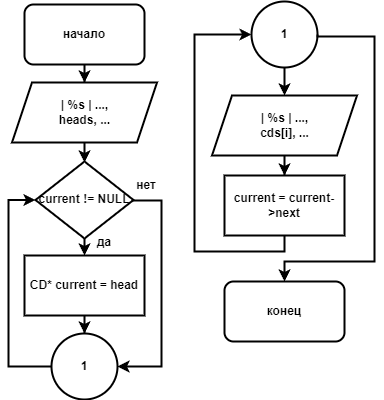
**Описание метода решение задачи**

Добавляются библиотеки и в функции начала программы просим пользователя ввести сначала кол-во треков, дальше шапку таблицы и данные на диске. Пользователь после ввода одного диска выбирает, нужно ли добавить ещё один диск или при вводе названия диска может написать символ, чтобы закончить добавление дисков. В результате выводим на экран приблизительную таблицу из задания.

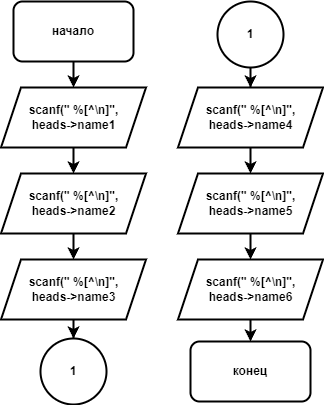
**Схема программы**



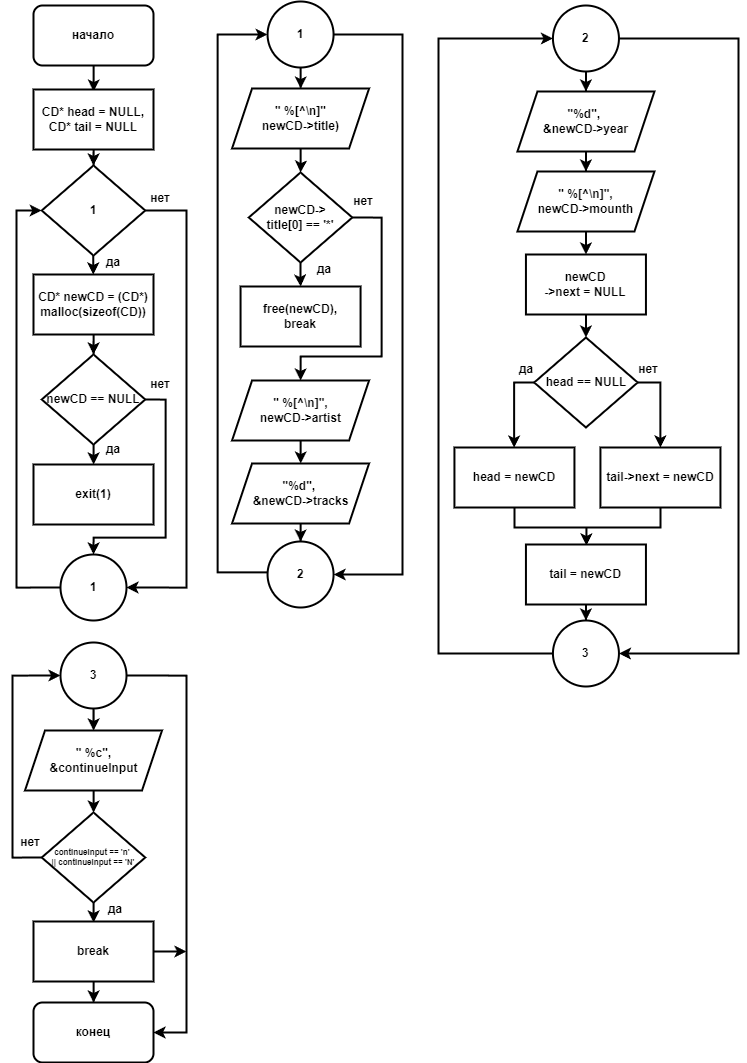
**Схема функции «printTable»**



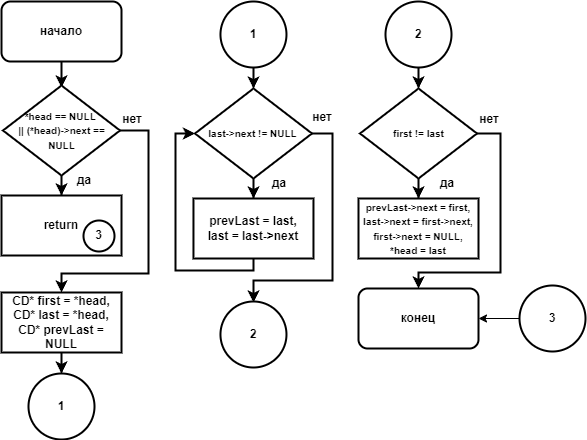
**Схема функции «inputHeads»**



**Схема функции «inputCDs»**



**Схема функции «swapRows»**



**Листинг**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <locale>

// Структура для хранения данных одного компакт-диска

typedef struct CD {

char title[50]; // Название диска

char artist[50]; // Исполнитель

int tracks; // Количество треков

int year; // Год выпуска

char mounth[50]; // Месяц выпуска

struct CD\* next; // Указатель на следующий элемент списка

} CD;

// Структура для хранения шапки таблицы

typedef struct {

char name1[50];

char name2[50];

char name3[50];

char name4[50];

char name5[50];

char name6[50];

} HEAD;

// Функция для вывода таблицы

void printTable(HEAD heads, CD\* head) {

// Заголовок таблицы

printf("\

| %s | %s | %s | %s |\n\

| %s | %s |\n",

heads.name1, heads.name2, heads.name3, heads.name4, heads.name5, heads.name6);

printf("-------------------------------------------------------------\n");

// Вывод данных из списка

CD\* current = head;

while (current != NULL) {

printf("| %s | %s | %d | %d | %s |\n", current->title, current->artist, current->tracks, current->year, current->mounth);

printf("-------------------------------------------------------------\n");

current = current->next;

}

}

// Функция для ввода шапки таблицы

void inputHeads(HEAD\* heads) {

printf("Введите шапку для таблицы:\n");

printf("Первая ячейка: ");

scanf(" %[^\n]", heads->name1);

printf("Вторая ячейка: ");

scanf(" %[^\n]", heads->name2);

printf("Третья ячейка: ");

scanf(" %[^\n]", heads->name3);

printf("Четвёртая ячейка: ");

scanf(" %[^\n]", heads->name4);

printf("Пятая ячейка: ");

scanf(" %[^\n]", heads->name5);

printf("Шестая ячейка: ");

scanf(" %[^\n]", heads->name6);

}

// Функция для ввода данных о компакт-дисках в динамический список

CD\* inputCDs() {

CD\* head = NULL; // Начало списка

CD\* tail = NULL; // Конец списка (для добавления элементов)

while (1) {

// Создаём новый элемент списка

CD\* newCD = (CD\*)malloc(sizeof(CD));

if (newCD == NULL) {

printf("Ошибка выделения памяти.\n");

exit(1);

}

printf("\nВведите данные для нового диска:\n");

printf("Название (или '\*' для завершения): ");

scanf(" %[^\n]", newCD->title);

// Завершаем ввод, если введен специальный символ '\*'

if (newCD->title[0] == '\*') {

free(newCD);

break;

}

printf("Исполнитель: ");

scanf(" %[^\n]", newCD->artist);

printf("Количество треков: ");

scanf("%d", &newCD->tracks);

printf("Год выпуска: ");

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

printf("Месяц выпуска: ");

scanf(" %[^\n]", newCD->mounth);

newCD->next = NULL; // Указатель на следующий элемент

// Добавляем новый элемент в список

if (head == NULL) {

head = newCD; // Если список пустой, новый элемент становится началом

}

else

tail->next = newCD; // Иначе, добавляем элемент в конец

tail = newCD; // Перемещаем указатель конца списка на новый элемент

// Запрашиваем продолжение ввода

char continueInput;

printf("Хотите ввести еще один диск? (y/n): ");

scanf(" %c", &continueInput);

if (continueInput == 'n' || continueInput == 'N') {

break;

}

}

return head;

}

// Функция для перестановки первого и последнего элемента списка

void swapRows(CD\*\* head) {

if (\*head == NULL || (\*head)->next == NULL) {

return; // Если список пустой или содержит только один элемент, перестановка не нужна

}

CD\* first = \*head;

CD\* last = \*head;

CD\* prevLast = NULL;

// Найдем последний элемент и его предыдущий элемент

while (last->next != NULL) {

prevLast = last;

last = last->next;

}

// Если первый и последний элементы не одинаковы, меняем их местами

if (first != last) {

// Изменяем указатели

prevLast->next = first;

last->next = first->next;

first->next = NULL;

// Меняем указатель на начало списка

\*head = last;

}

}

// Главная функция

int main() {

setlocale(LC\_ALL, ".UTF8");

HEAD heads; // Шапка таблицы

inputHeads(&heads); // Ввод шапки таблицы

// Ввод списка компакт-дисков

CD\* head = inputCDs();

// Вывод исходной таблицы

printf("\nИсходные данные:\n");

printTable(heads, head);

// Перестановка первого и последнего элемента списка

swapRows(&head);

// Вывод модифицированного списка

printf("\nДанные после перестановки:\n");

printTable(heads, head);

// Освобождаем память

CD\* current = head;

while (current != NULL) {

CD\* temp = current;

current = current->next;

free(temp);

}

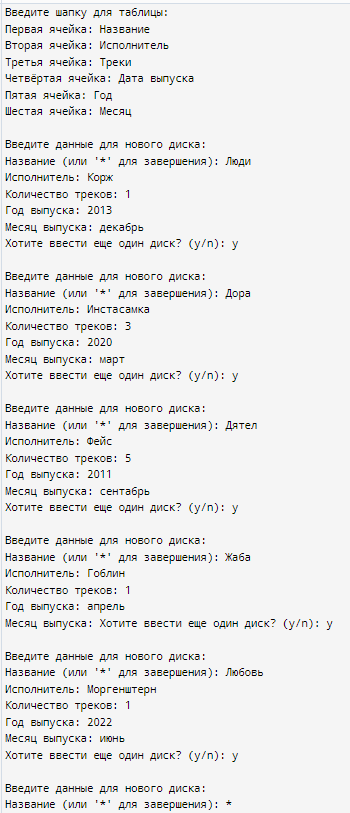
return 0;}

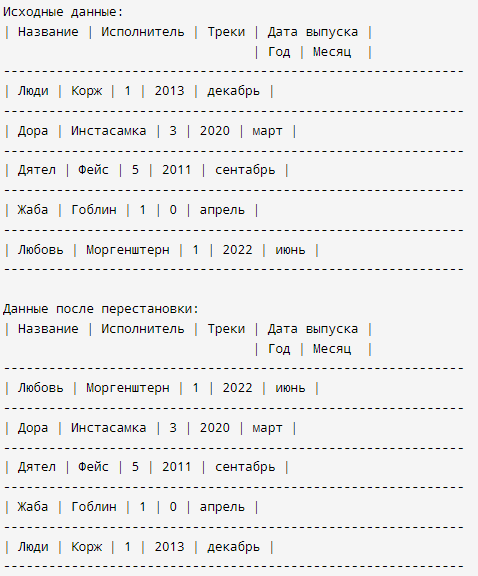
**Пояснительный текст к программе**

" %[^\n]" – читает введённую строку учитывая пробелы.

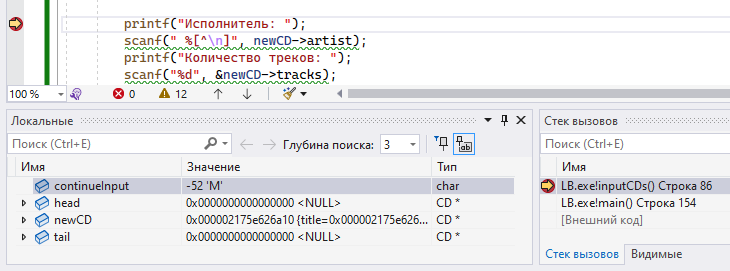
heads->name1 – записывает значение.

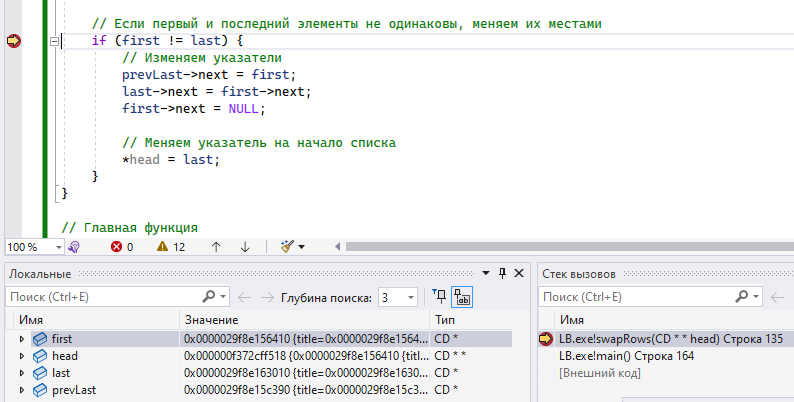
**Результат работы программы**





**Протокол трассировки программы**





**Вывод**

Были изучены правила описания, ввода-вывода и основных приемов работы со структурированными данными. Была разработана программа, в которой пользователь вводит все данные, которые будут отображены в виде таблицы. Пользователь в процессе добавления дисков может отменить ввод.