МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра Вычислительной техники

КУРСОВАЯ РАБОТА

по дисциплине «Программирование»

ТЕМА: «РАЗРАБОТКА ЭЛЕКТРОННОЙ КАРТОТЕКИ»

Студент гр. 3311 _____ Баймухамедов Р. Р.

Преподаватель _____ Хахаев И. А.

Санкт-Петербург

2024

Введение

Цель работы.

Полное решение содержательной задачи (содержательная и формальная постановка задачи, спецификация, включая описание диалога, выбор метода решения и структур данных, разработка алгоритма, программная реализация, тестирование и отладка, документирование).

Выбранная предметная область — фильмы.

Задание

Создать электронную картотеку, хранящуюся на диске, и программу, обеспечивающую взаимодействие с ней.

Программа должна выполнять следующие действия:

- Занесение данных в электронную картотеку
- Внесение изменений (исключение, корректировка, добавление)
- Поиск данных по различным признакам
- Сортировку данных по различным признакам
- Вывод результатов на экран и сохранение на диске

Выбор подлежащих выполнению команд реализован с помощью основного меню и вложенных меню. Выполнение программы является многократным по желанию пользователя. Данные первоначально считываются из файлов, в процессе работы данные вводятся с клавиатуры

Перечень пунктов меню:

0. Выход (Завершение работы программы)
1. Просмотр картотеки (Вывод списка на экран)
2. Добавление новых карточек (Внесение нового элемента)
3. Поиск карточки по параметру (Поиск по введённым данным в заданном пользователем поле)
4. Редактирование карточки (Изменение элемента, находящегося в списке)

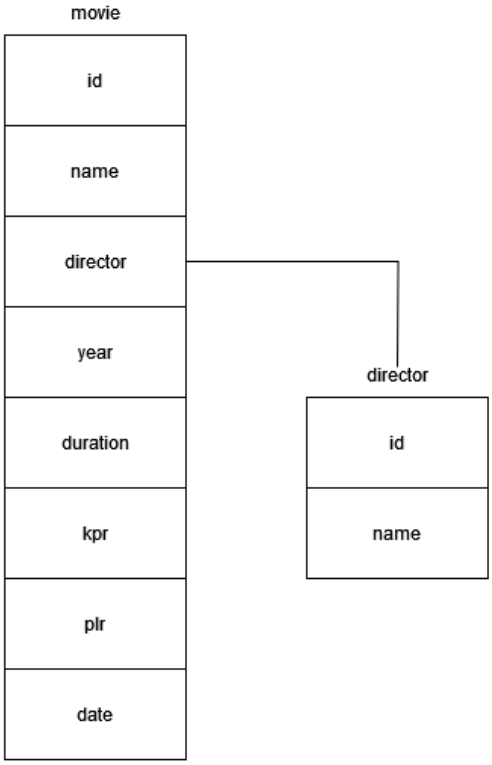
5. Удаление карточки (Удаление элемента)
6. Сортировка картотеки по параметру (Сортировка по заданному пользователем полю и направлению)
7. Справка

Поскольку выбранной предметной областью являются фильмы, то полями структуры будут идентификационный номер фильма, название фильма, режиссёр (эта характеристика объекта будет являться списком), год выхода, длительность, рейтинг КиноПоиска, личный рейтинг и дата просмотра. Структура режиссёра содержит поля идентификационного номера и имени режиссёра.

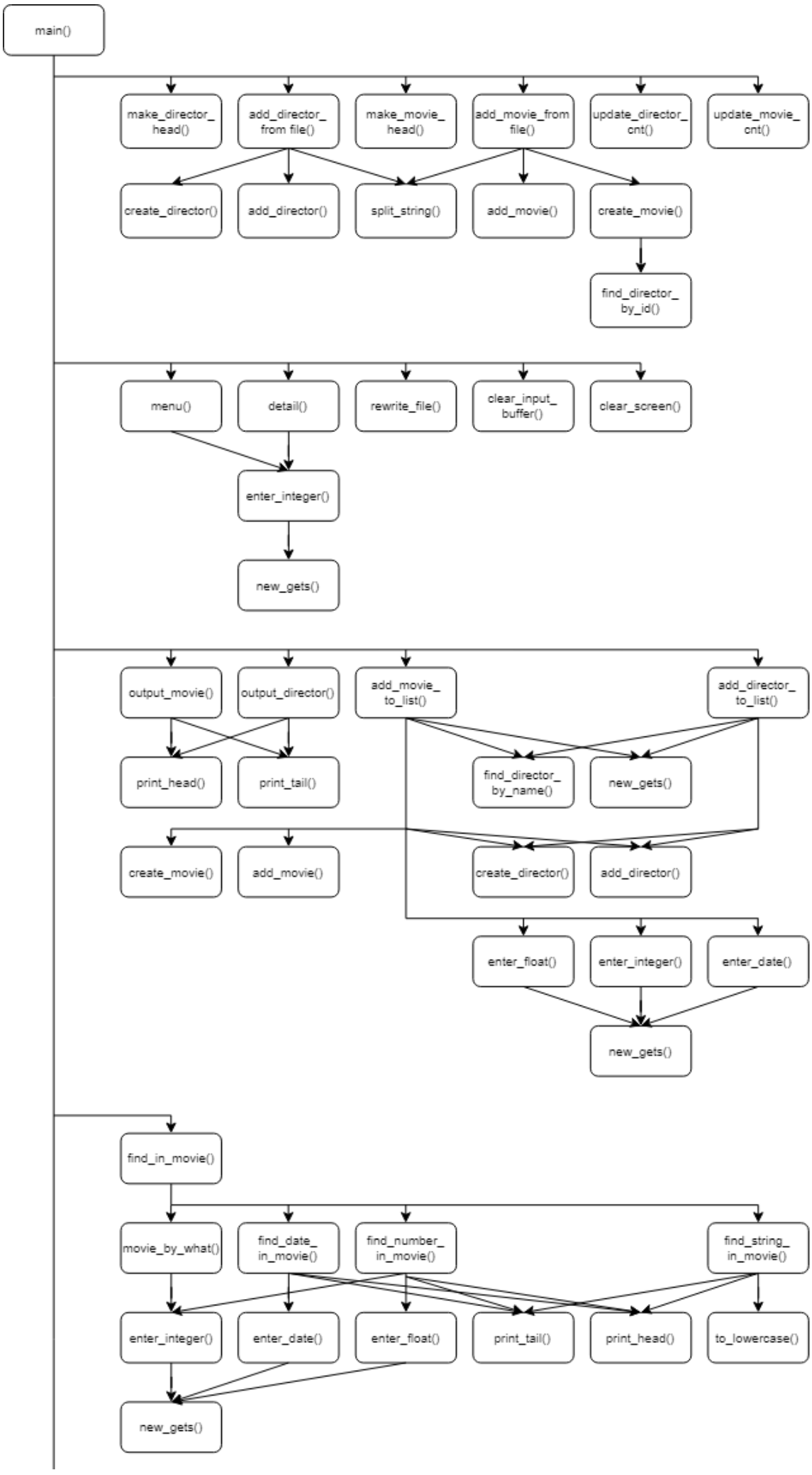
Описание общей архитектуры данных

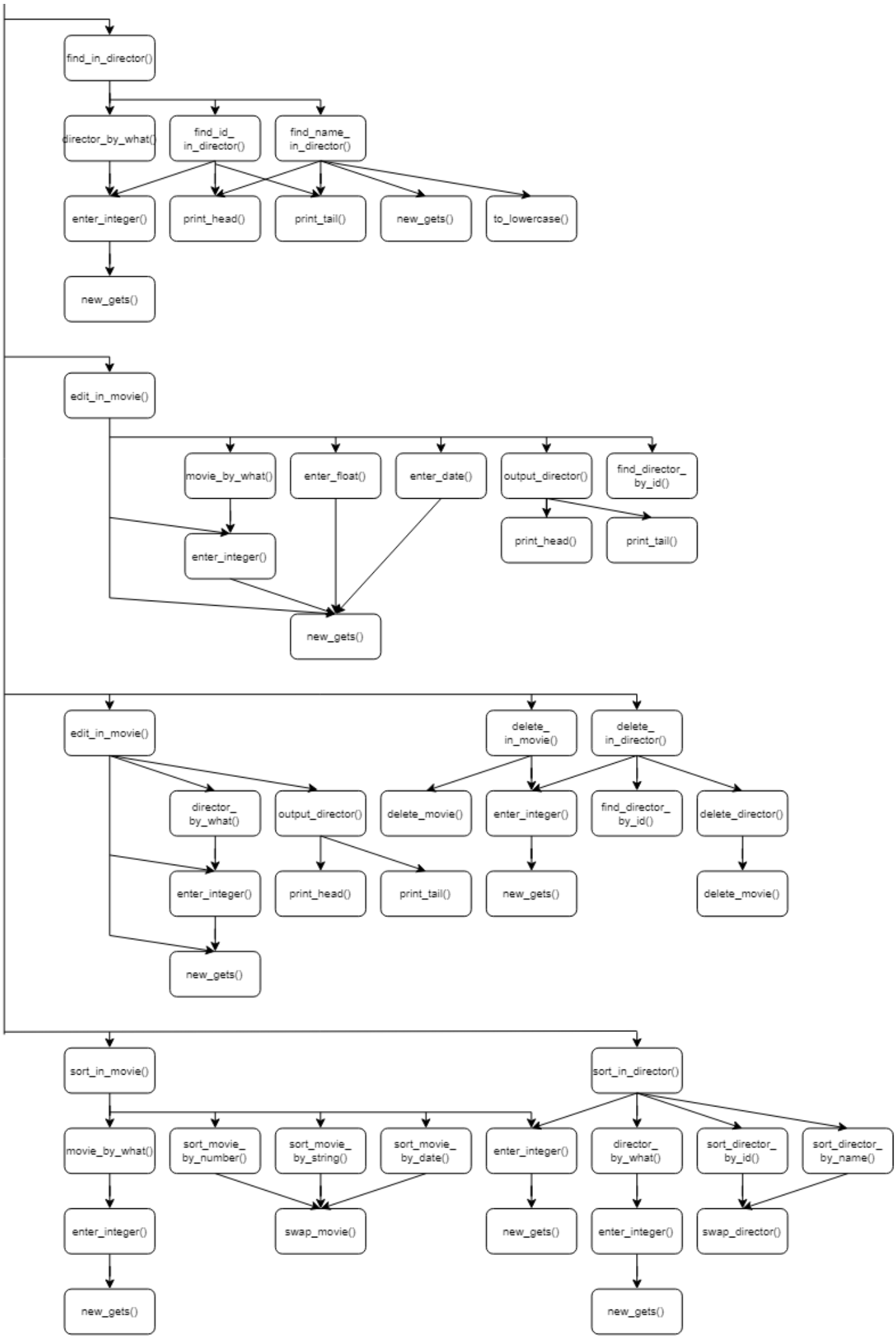
Работа с меню находится в основном теле программы (main). Все основные действия (пункты меню) вынесены в отдельные функции: просмотр, добавление, поиск, редактирование, удаление, сортировка карточек и справка. Сохранение картотеки происходит после выполнения любого основного действия. В качестве структуры данных — два двусвязных линейных списка, один из которых является характеристикой другого.

Сущности и их назначение:



Структура вызовов функций:


main()
make_director_head()
add_director_from file()
make_movie_head()
add_movie_from file()
update_director_cnt()
update_movie_cnt()
create_director()
add_director()
split_string()
add_movie()
create_movie()
find_director_by_id()
menu()
detail()
rewrite_file()
clear_input_buffer()
clear_screen()
enter_integer()
new_gets()
output_movie()
output_director()
add_movie_to_list()
add_director_to_list()
print_head()
print_tail()
find_director_by_name()
new_gets()
create_movie()
add_movie()
create_director()
add_director()
enter_float()
enter_integer()
enter_date()
new_gets()
find_in_movie()
movie_by_what()
find_date_in_movie()
find_number_in_movie()
find_string_in_movie()
enter_integer()
enter_date()
enter_float()
print_tail()
print_head()
to_lowercase()
new_gets()


find_in_director()
director_by_what()
find_id_ in_director()
find_name_ in_director()
enter_integer()
print_head()
print_tail()
new_gets()
to_lowercase()
new_gets()
edit_in_movie()
movie_by_what()
enter_float()
enter_date()
output_director()
find_director_ by_id()
print_head()
print_tail()
enter_integer()
new_gets()
edit_in_movie()
delete_ in_movie()
delete_ in_director()
director_ by_what()
output_director()
delete_movie()
enter_integer()
find_director_ by_id()
delete_director()
enter_integer()
print_head()
print_tail()
new_gets()
delete_movie()
new_gets()
sort_in_movie()
sort_in_director()
movie_by_what()
sort_movie_ by_number()
sort_movie_ by_string()
sort_movie_ by_date()
enter_integer()
director_ by_what()
sort_director_ by_id()
sort_director_ by_name()
enter_integer()
swap_movie()
new_gets()
enter_integer()
swap_director()
new_gets()
new_gets()

Описание структур

struct movie (MOV)

№	Название	Тип	Назначение
1	id	int	Идентификационный номер фильма
2	*name	char	Название фильма
3	*director	DIR (struct director)	Имя и фамилия режиссёра фильма
4	year	int	Год выхода фильма
5	duration	int	Длительность фильма
6	kpr	float	Оценка фильма на КиноПоиске
7	plr	float	Персональная оценка фильма
8	date[3]	int	Дата просмотра фильма
9	*prev	MOV (struct movie)	Указатель на предыдущий элемент списка
10	*next	MOV (struct movie)	Указатель на следующий элемент списка

struct movie_head (MHD)

№	Название	Тип	Назначение
1	cnt	int	Максимальное значение ID элементов списка
2	*first	MOV (struct movie)	Указатель на первый элемент списка фильмов
3	*last	MOV (struct movie)	Указатель на последний элемент списка фильмов

struct director (DIR)

№	Название	Тип	Назначение
1	id	int	Идентификационный номер режиссёра

2	*name	char	Имя и фамилия режиссёра
3	*prev	DIR (struct director)	Указатель на предыдущий элемент списка
4	*next	DIR (struct director)	Указатель на следующий элемент списка

struct movie_head (DHD)

№	Название	Тип	Назначение
1	cnt	int	Максимальное значение ID элементов списка
2	*first	DIR (struct director)	Указатель на первый элемент списка режиссёров
3	*last	DIR (struct director)	Указатель на последний элемент списка режиссеров

Описание функций

№	Функция	Тип	Назначение
1	main	int	Запускает программу, обрабатывает пользовательский ввод, вызывает другие функции в зависимости от выбора пользователя, и завершает программу при необходимости.
2	new_gets	void	Посимвольный ввод строки
3	enter_integer	int	Ввод целого числа и проверка его корректности
4	enter_float	float	Ввод вещественного числа и проверка его корректности
5	enter_date	void	Ввод даты и проверка её корректности
6	menu	int	Вывод меню программы и выбор опции
7	detail	int	Выбор списка фильмов или режиссёров
8	movie_by_what	int	Вывод названия полей структуры фильмов и выбор поля
9	director_by_what	int	Вывод названия полей структуры режиссеров и выбор поля
10	clear_screen	void	Очистка консоли

11	print_head	void	Вывод оглавления списка
12	print_tail	void	Вывод окончания списка
13	reference	void	Вывод справки
14	output_movie	void	Вывод списка фильмов
15	output_director	void	Вывод списка режиссёров
16	*make_movie_head	MHD (struct movie_head)	Инициализация головы списка фильмов
17	update_movie_cnt	void	Обновление счётчика головы фильмов
18	*make_director_head	DHD (struct director_head)	Инициализация головы списка режиссёров
19	update_director_cnt	void	Обновление счётчика головы режиссёров
20	*find_director_by_id	DIR (struct director)	Поиск элемента в списке режиссёров по ID
21	*find_director_by_name	DIR (struct director)	Поиск элемента в списке режиссёров по имени
22	*create_movie	MOV (struct movie)	Создание элемента списка фильмов
23	add_movie	void	Привязка нового элемента к списку фильмов
24	*create_director	DIR (struct director)	Создание элемента списка режиссёров
25	add_director	void	Привязка нового элемента к списку режиссёров
26	clear_input_buffer	void	Очистка входного буфера
27	to_lowercase	void	Приведение строки к нижнему регистру
28	rewrite_file	void	Сохранение изменений в списках фильмов и режиссёров в файл
29	split_string	void	Разделяет строку по символу-разделителю и записывает элементы строки в массив
30	add_movie_from_file	void	Чтение из файла и запись элементов строк в линейный двусвязный список фильмов
31	add_movie_to_list	void	Добавление нового элемента в список фильмов
32	add_director_from_file	void	Чтение из файла и запись элементов строк в линейный двусвязный список режиссёров
33	add_director_to_list	void	Добавление нового элемента в список режиссёров
34	find_number_in_movie	void	Поиск и вывод элементов списка фильмов по числовому полю структуры MOV
35	find_string_in_movie	void	Поиск и вывод элементов списка фильмов по символьному полю структуры MOV

36	find_date_in_movie	void	Поиск и вывод элементов списка фильмов по дате просмотра фильма
37	find_in_movie	void	Выбор поля структуры MOV для поиска
38	find_id_in_director	void	Поиск и вывод элементов списка режиссёров по ID
39	find_name_in_director	void	Поиск и вывод элементов списка режиссёров по имени
40	find_in_director	void	Выбор поля структуры DIR для поиска
41	edit_in_movie	void	Редактирование элемента структуры MOV
42	edit_in_director	void	Редактирование элемента структуры DIR
43	delete_movie	void	Удаление элемента из списка фильмов
44	delete_in_movie	void	Ввод ID удаляемого элемента списка фильмов
45	delete_director	void	Удаление элемента из списка режиссёров
46	delete_in_director	void	Ввод ID удаляемого элемента списка режиссёров
47	swap_movie	int	Меняет местами элементы двусвязного списка фильмов
48	sort_movie_by_number	void	Сортировка списка фильмов по числовому полю структуры MOV
49	sort_movie_by_string	void	Сортировка списка фильмов по символьному полю структуры MOV
50	sort_movie_by_date	void	Сортировка списка фильмов по дате просмотра
51	sort_in_movie	void	Выбор поля структуры MOV, по которой будет произведена сортировка списка фильмов
52	swap_director	int	Меняет местами элементы двусвязного списка режиссёров
53	sort_director_by_id	void	Сортировка списка режиссёров по ID
54	sort_director_by_name	void	Сортировка списка режиссёров по имени
55	sort_in_director	void	Выбор поля структуры DIR, по которой будет произведена сортировка списка режиссёров

Описание переменных

int main()

№	Переменная	Тип	Назначение
1	*mph	MHD (struct movie_head)	Голова списка фильмов
2	*dph	DHD (struct director_head)	Голова списка режиссёров

3	option	int	Опция, выбираемая пользователем
4	choise	int	Выбор списка фильмов или режиссёров

void new_gets(char s[], int lim)

№	Переменная	Тип	Назначение
1	c	char	Вводимый символ
2	i	i	Номер вводимого символа

int enter_integer(char *message, int a, int b)

№	Переменная	Тип	Назначение
1	input[MAXLEN]	char	Вводимая строка символов
2	number	int	Преобразованная в целое число строка
3	flag	int	Проверка корректности введённой строки символов

float enter_float(char *message, float a, float b)

№	Переменная	Тип	Назначение
1	input[MAXLEN]	char	Вводимая строка символов
2	number	int	Преобразованная в вещественное число строка
3	flag	int	Проверка корректности введённой строки символов

void enter_date(char *message, int *day, int *month, int *year)

№	Переменная	Тип	Назначение
1	input[MAXLEN]	char	Вводимая строка символов

2	flag	int	Проверка корректности введённой строки символов

int menu()

№	Переменная	Тип	Назначение
1	option	int	Опция, выбираемая пользователем

int detail()

№	Переменная	Тип	Назначение
1	option	int	Выбор списка режиссёров или фильмов

int movie_by_what(int i)

№	Переменная	Тип	Назначение
1	option	int	Выбор поля структуры MOV

int director_by_what(int i)

№	Переменная	Тип	Назначение
1	option	int	Выбор поля структуры DIR

void output_movie(MHD *mph)

№	Переменная	Тип	Назначение
1	*current	MOV (struct movie)	Текущий выводимый элемент списка фильмов

void output_director(DHD *dph)

№	Переменная	Тип	Назначение
1	*current	DIR (struct director)	Текущий выводимый элемент списка режиссёров

MHD *make_movie_head()

№	Переменная	Тип	Назначение
1	*mph	MHD (struct movie_head)	Голова линейного двусвязного списка фильмов

void update_movie_cnt(MHD *mph)

№	Переменная	Тип	Назначение
1	*current	MOV (struct movie)	Текущий элемент списка фильмов
2	max_id	int	Максимальное значение ID в списке фильмов

DHD *make_director_head()

№	Переменная	Тип	Назначение
1	*dph	DHD (struct director_head)	Голова линейного двусвязного списка режиссёров

void update_director_cnt(DHD *dph)

№	Переменная	Тип	Назначение
1	*current	DIR (struct director)	Текущий элемент списка режиссёров
2	max_id	int	Максимальное значение ID в списке режиссёров

DIR *find_director_by_id(DHD *dph, int id)

№	Переменная	Тип	Назначение
1	*current	DIR (struct director)	Текущий элемент списка режиссёров
2	flag	int	Результат на то, был ли найден элемент с заданным ID

DIR *find_director_by_name(DHD *dph, char *movie_director)

№	Переменная	Тип	Назначение

1	*current	DIR (struct director)	Текущий элемент списка режиссёров
2	flag	int	Результат на то, был ли найден элемент с заданным именем
3	lower_movie_director[MAXLEN]	char	Приведенное к нижнему регистру имя режиссёра для поиска
4	lower_current_name[MAXLEN]	char	Приведенное к нижнему регистру имя текущего режиссёра

MOV *create_movie(DHD *dph, int id_mov, char *movie_name, int id_dir, int movie_year, int movie_duration, float movie_kpr, float movie_plr, int watch_date[3])

№	Переменная	Тип	Назначение
1	*new_movie	MOV	Новый элемент списка фильмов
2	*movie_director	DIR	Элемент списка режиссёров
3	*name	char	Указатель на строку для хранения названия фильма

DIR *create_director(int id, char *movie_director);

№	Переменная	Тип	Назначение
1	*new_director	DIR	Новый элемент списка режиссёров
2	*director	char	Указатель на строку для хранения имени режиссёра

void clear_input_buffer()

№	Переменная	Тип	Назначение
1	c	int	Хранение символа

void to_lowercase(char *str)

№	Переменная	Тип	Назначение
1	i	int	Номер символа строки

void rewrite_file(MHD *mph, DHD *dph)

№	Переменная	Тип	Назначение
1	*mov_current	MOV	Текущий элемент списка фильмов
2	*dir_current	DIR	Текущий элемент списка режиссёров
3	*file1	FILE	Файл с данными о фильмах
4	*file2	FILE	Файл с данными о режиссёрах

void split_string(char *inputString, char **words, int *wordCount, char delimiter)

№	Переменная	Тип	Назначение
1	wordIndex	char	индекс текущего слова в массиве words
2	wordStart	char	индекс начала текущего слова в строке inputString
3	wordLength	char	длина текущего слова
4	inWord	int	флаг, указывающий на то, находится ли функция внутри слова или не внутри
5	i	int	переменная для итерации по символам в строке inputString.

void add_movie_from_file(MHD *mph, DHD *dph, char *filename, char sep)

№	Переменная	Тип	Назначение
1	*new_movie	MOV	Указатель на структуру MOV, представляющую новый фильм

2	line[MAXLEN]	char	Хранение информации строки из файла
3	*words[10]	char	Временный массив, куда записываются данные из строки файла
4	wordcount	int	Кол-во элементов, на которые строка была разделена
5	i	int	Переменная в цикле, отвечающая за номер элемента, на которые была разделена строка
6	date[3]	int	Массив для хранения даты просмотра фильма
7	*file	FILE	Исходный файл с данными

void add_movie_to_list(MHD *mph, DHD *dph)

№	Переменная	Тип	Назначение
1	*new_movie	MOV	Указатель на структуру MOV, представляющую новый фильм
2	*current	DIR	Указатель на структуру DIR, представляющую нового режиссёра
3	name[MAXLEN]	char	Название фильма
4	director[MAXLEN]	char	Имя режиссёра
5	kpr	float	Оценка фильма на КиноПоиске
6	plr	float	Персональная оценка фильма
7	id_dir	int	ID режиссёра фильма
8	year	int	Год выхода фильма
9	duration	int	Длительность фильма
10	date[3]	int	Дата просмотра фильма

void add_director_from_file(DHD *dph, char *filename, char sep)

№	Переменная	Тип	Назначение
1	*new_director	DIR	Указатель на структуру DIR, представляющую нового режиссёра
2	line[MAXLEN]	char	Хранение информации строки из файла
3	*words[2]	char	Временный массив, куда записываются данные из строки файла
4	wordcount	int	Кол-во элементов, на которые строка была разделена
5	i	int	Переменная в цикле, отвечающая за номер элемента, на которые была разделена строка
6	*file	FILE	Исходный файл с данными

void add_director_to_list(MHD *mph, DHD *dph)

№	Переменная	Тип	Назначение
1	*new_director	DIR	Указатель на структуру DIR, представляющую нового режиссёра
2	name[MAXLEN]	char	Имя режиссёра

void find_number_in_movie(MHD *mph, char *message, int a, int b, int i)

№	Переменная	Тип	Назначение
1	*current	MOV	Текущий элемент списка фильмов
2	record	float	Значение элемента в поле для поиска
3	number	float	Введённое для поиска числовое значение
4	count	int	Количество элементов списка,

			удовлетворяющих условию

void find_string_in_movie(MHD *mph, char *message, int i)

№	Переменная	Тип	Назначение
1	*current	MOV	Текущий элемент списка фильмов
2	input[MAXLEN]	char	Вводимая строка символов
3	*string	char	Значение элемента в поле для поиска
4	lstring[MAXLEN]	char	Значение элемента в поле для поиска приведённое к нижнему регистру
5	count	int	Количество элементов списка, удовлетворяющих условию
6	result	int	Результат сравнения строк

void find_date_in_movie(MHD *mph)

№	Переменная	Тип	Назначение
1	*current	MOV	Текущий элемент списка фильмов
2	date[3]	int	Вводимая для поиска дата просмотра фильма
3	count	int	Количество элементов списка, удовлетворяющих условию

void find_date_in_movie(MHD *mph)

№	Переменная	Тип	Назначение
1	option	int	Выбор поля структуры MOV

void find_id_in_director(DHD *dph)

№	Переменная	Тип	Назначение
1	*current	DIR	Текущий элемент списка режиссёров
2	id	int	Вводимый для поиска режиссёров ID
3	count	int	Количество элементов списка, удовлетворяющих условию

void find_name_in_director(DHD *dph)

№	Переменная	Тип	Назначение
1	*current	DIR	Текущий элемент списка режиссёров
2	input[MAXLEN]	char	Вводимая строка символов
3	lstring[MAXLEN]	char	Приведённое к нижнему регистру значение имени элемента списка режиссёров
4	count	int	Количество элементов списка, удовлетворяющих условию
5	result	int	Результат сравнения строк

void find_in_director(DHD *dph)

№	Переменная	Тип	Назначение
1	option	int	Выбор поля структуры DIR

void edit_in_movie(MHD *mph, DHD *dph)

№	Переменная	Тип	Назначение
1	*current	MOV	Редактируемый элемент списка фильма
2	*temp_dir	DIR	Новое значение имени режиссёра

3	input[MAXLEN]	char	Вводимая строка символов
4	kpr	float	Оценка фильма на КиноПоиске
5	plr	float	Персональная оценка фильма
6	option	int	Выбор поля структуры MOV
7	mov_id	int	ID редактируемого элемента списка фильмов
8	dir_id	int	ID режиссёра фильма
9	year	int	Год выхода фильма
10	duration	int	Длительность фильма
11	date[2]	int	Дата просмотра фильма
12	i	int	Номер элемента массива
13	flag	int	Результат нахождения элемента списка фильмов с введённым ID

void edit_in_director(DHD *dph)

№	Переменная	Тип	Назначение
1	*current	DIR	Редактируемый элемент списка режиссёров
2	*temp_dir	DIR	Новое значение имени режиссёра
3	input[MAXLEN]	char	Вводимая строка символов
4	option	int	Выбор поля структуры MOV
5	dir_id	int	ID редактируемого элемента списка режиссёров
6	flag	int	Результат нахождения элемента списка режиссёров с введённым ID

void delete_in_movie(MHD *mph)

№	Переменная	Тип	Назначение
1	*current	MOV	Текущий элемент списка фильмов
2	id_mov	int	ID удаляемого элемента списка фильмов
3	flag	int	Результат нахождения элемента списка фильмов с введённым ID

void delete_director(MHD *mph)

№	Переменная	Тип	Назначение
1	*current_movie	MOV	Текущий элемент списка фильмов
2	*next_movie	MOV	ID удаляемого элемента списка фильмов
3	flag	int	Результат нахождения элемента списка фильмов с введённым ID

void delete_in_director(MHD *mph)

№	Переменная	Тип	Назначение
1	*current	DIR	Текущий элемент списка режиссёра
2	id_dir	int	ID удаляемого элемента списка режиссёров

int swap_movie(MHD *mph, MOV *current)

№	Переменная	Тип	Назначение
1	*temp	MOV	Следующий после текущего (меняемого) элемента списка фильмов

void sort_movie_by_number(MHD *mph, int i, int direction)

№	Переменная	Тип	Назначение
1	*current	MOV	Текущий элемент списка фильмов
2	swapped	int	Факт, замены элементов списка
3	need_to_swap	int	Результат необходимости проведения замены

void sort_movie_by_number(MHD *mph, int i, int direction)

№	Переменная	Тип	Назначение
1	*current	MOV	Текущий элемент списка фильмов
2	swapped	int	Факт, замены элементов списка
3	need_to_swap	int	Результат необходимости проведения замены

void sort_movie_by_date(MHD *mph, int i, int direction)

№	Переменная	Тип	Назначение
1	*current	MOV	Текущий элемент списка фильмов
2	swapped	int	Факт, замены элементов списка
3	need_to_swap	int	Результат необходимости проведения замены

void sort_in_movie(MHD *mph)

№	Переменная	Тип	Назначение
1	option	int	Выбор поля структуры MOV, по которой будет произведена сортировка
2	direction	int	Направление сортировки

int swap_director(DHD *dph, DIR *current)

№	Переменная	Тип	Назначение
1	*temp	DIR	Следующий после текущего (меняемого) элемента списка режиссёров

void sort_director_by_id(DHD *dph, int direction)

№	Переменная	Тип	Назначение
1	*current	DIR	Текущий элемент списка режиссёров
2	swapped	int	Факт, замены элементов списка

void sort_director_by_name(DHD *dph, int direction)

№	Переменная	Тип	Назначение
1	*current	DIR	Текущий элемент списка режиссёров
2	swapped	int	Факт, замены элементов списка

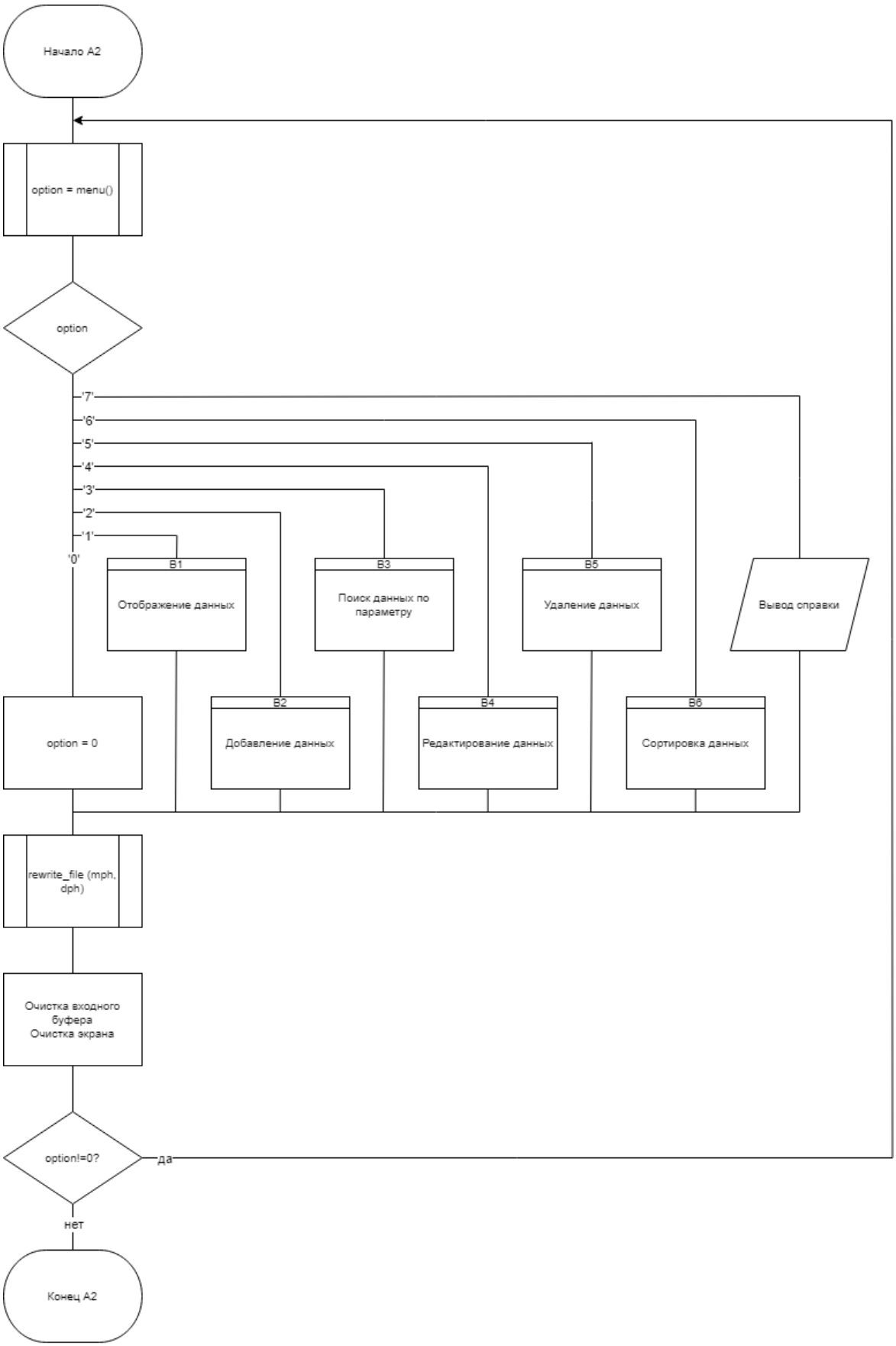
void sort_in_director(DHD *dph)

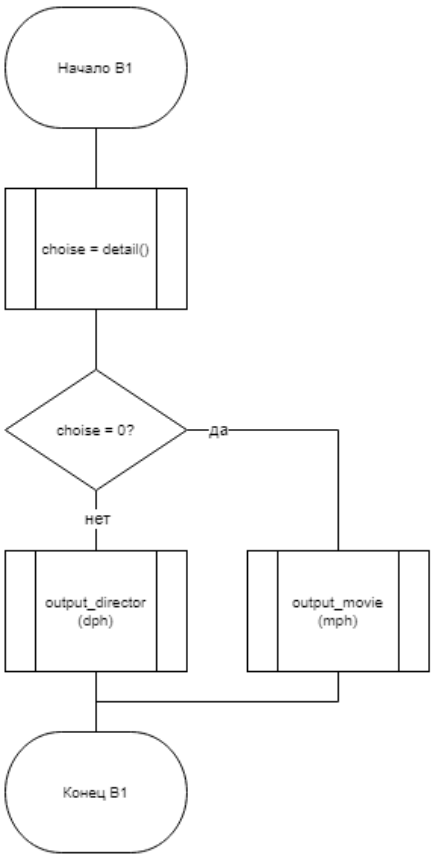
№	Переменная	Тип	Назначение
1	option	int	Выбор поля структуры DIR, по которой будет произведена сортировка
2	direction	int	Направление сортировки

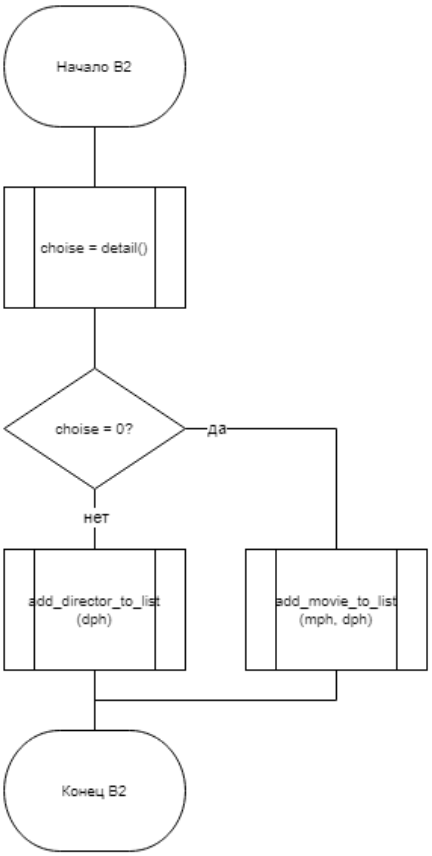
Схемы алгоритмов

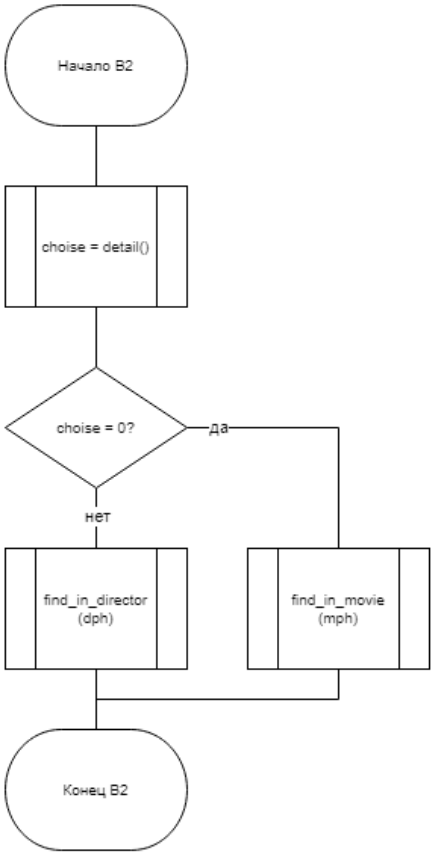

Начало
A1
Инициализация и подготовка данных
A2
Основной цикл обработки меню
Конец

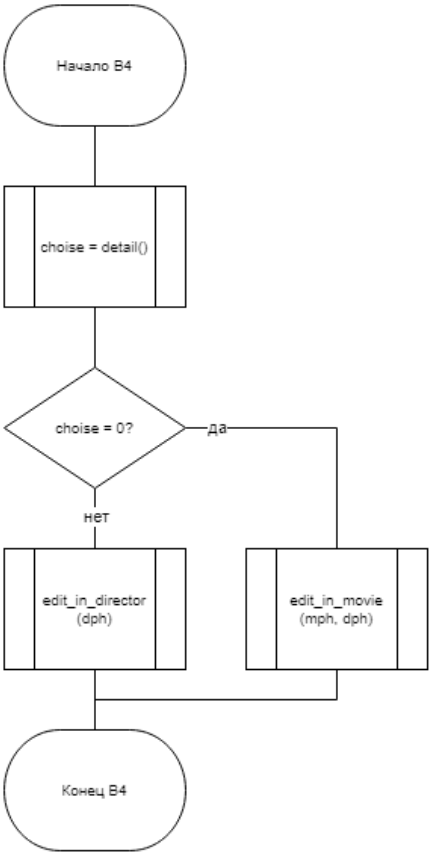

Начало A1
Создание голов списков фильмов и режиссёров
add_director_ from_file (dph, director_db, SEP)
add_movie_ from_file (mph, dph, movie_db, SEP)
update_director_cnt (dph)
update_director_cnt (dph)
Конец A1

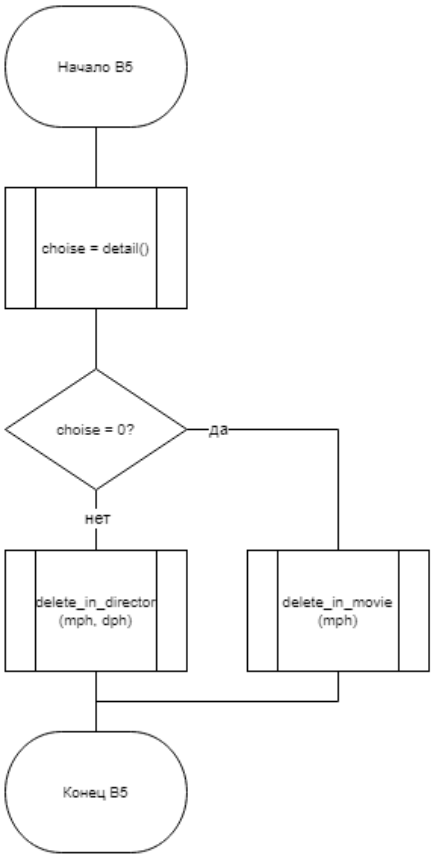

Начало A2
option = menu()
option
'7'
'6'
'5'
'4'
'3'
'2'
'1'
'0'
B1
Отображение данных
B3
Поиск данных по параметру
B5
Удаление данных
Вывод справки
option = 0
B2
Добавление данных
B4
Редактирование данных
B6
Сортировка данных
rewrite_file (mph, dph)
Очистка входного буфера
Очистка экрана
option!=0?
да
нет
Конец A2

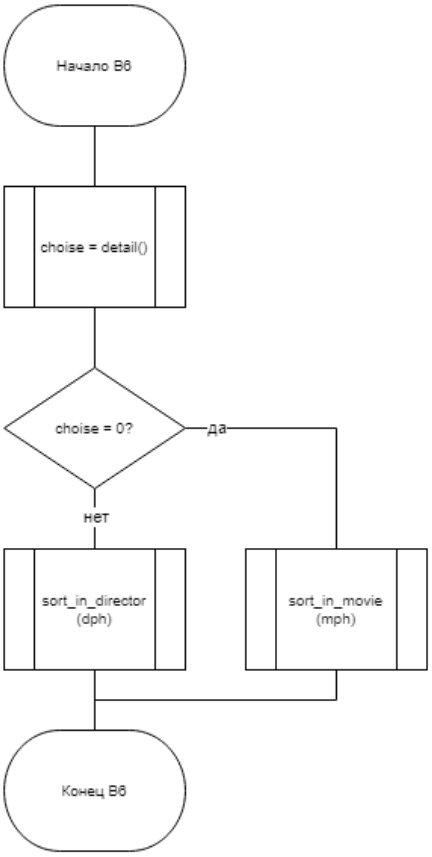

Начало B1
choise = detail()
choise = 0?
да
нет
output_director
(dph)
output_movie
(mph)
Конец B1

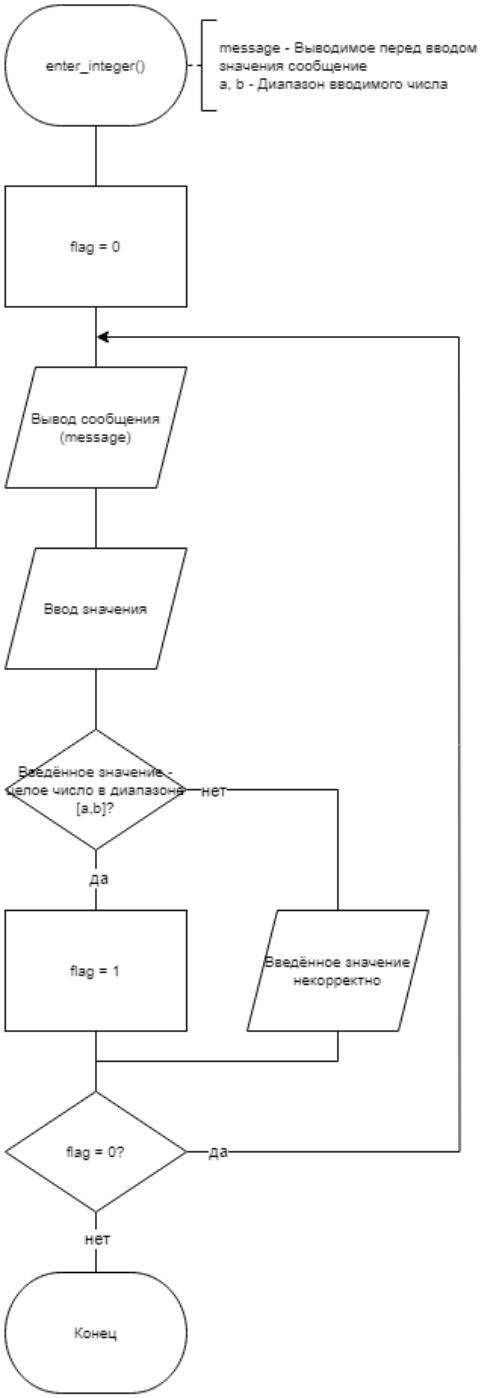

Начало B2
choise = detail()
choise = 0?
да
нет
add_director_to_list
(dph)
add_movie_to_list
(mph, dph)
Конец B2

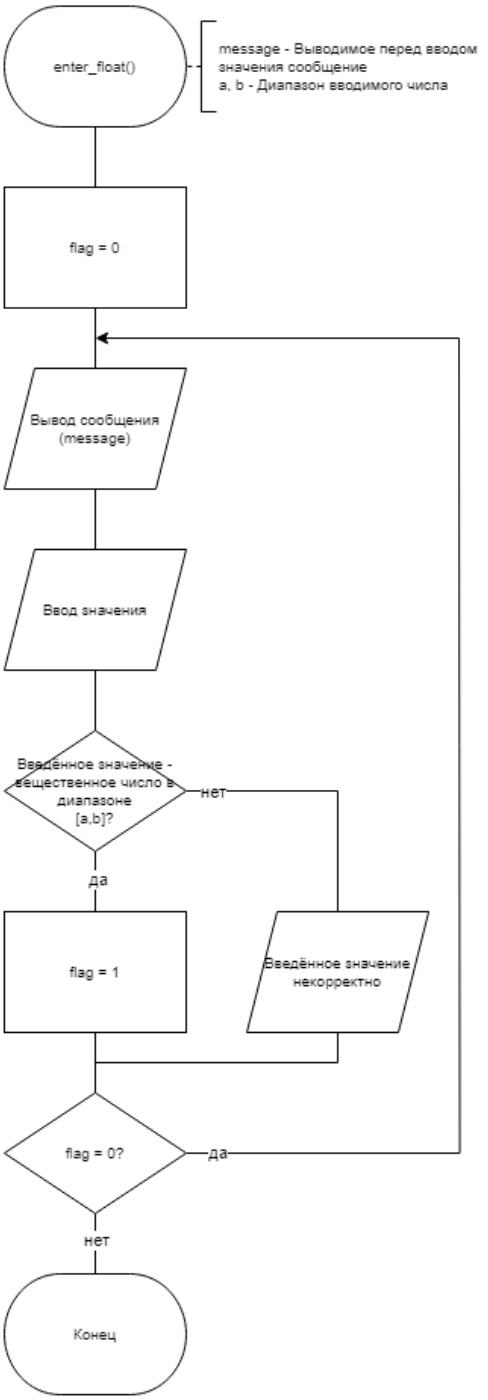

Начало B2
choise = detail()
choise = 0?
да
нет
find_in_director
(dph)
find_in_movie
(mph)
Конец B2

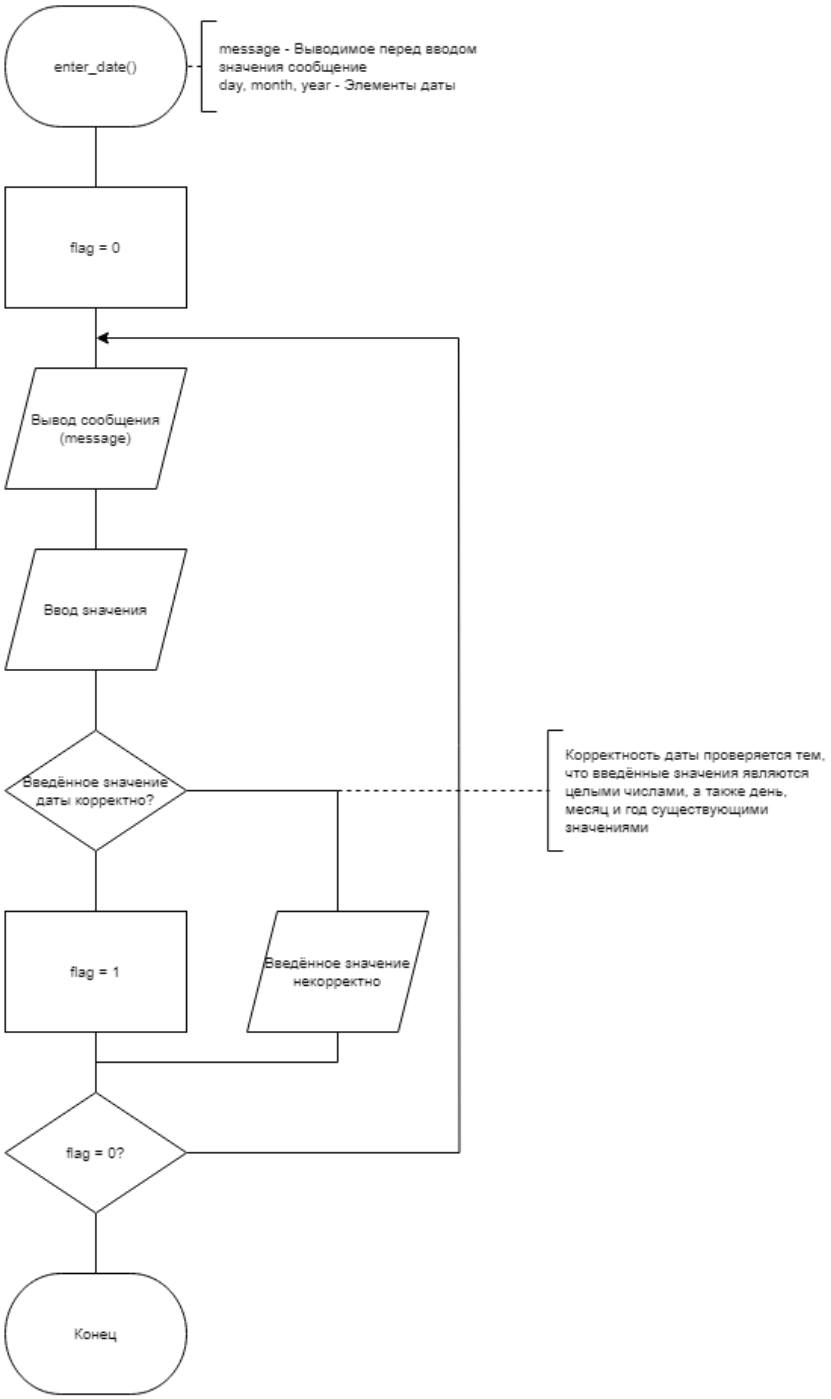

Начало B4
choise = detail()
choise = 0?
да
нет
edit_in_director
(dph)
edit_in_movie
(mph, dph)
Конец B4

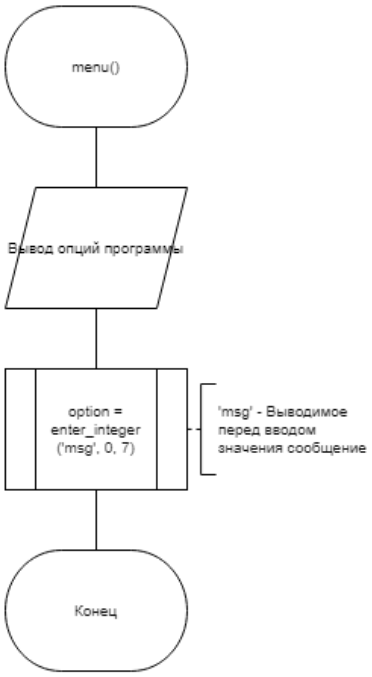

Начало B5
choise = detail()
choise = 0?
да
нет
delete_in_director
(mph, dph)
delete_in_movie
(mph)
Конец B5

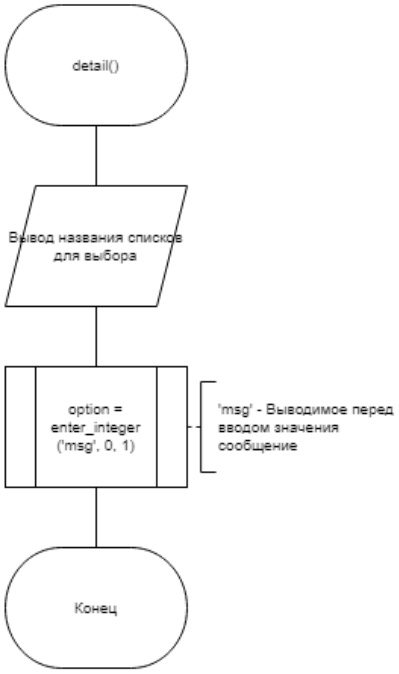

Начало B6
choise = detail()
choise = 0?
да
нет
sort_in_director
(dph)
sort_in_movie
(mph)
Конец B6

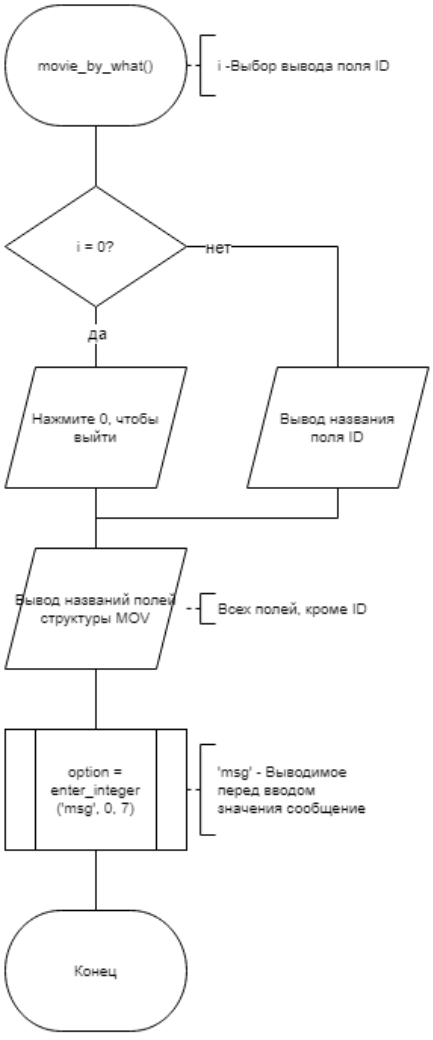

enter_integer()
message - Выводимое перед вводом значения сообщение
a, b - Диапазон вводимого числа
flag = 0
Вывод сообщения (message)
Ввод значения
Введённое значение - целое число в диапазоне [a,b]?
нет
да
flag = 1
Введённое значение некорректно
flag = 0?
да
нет
Конец

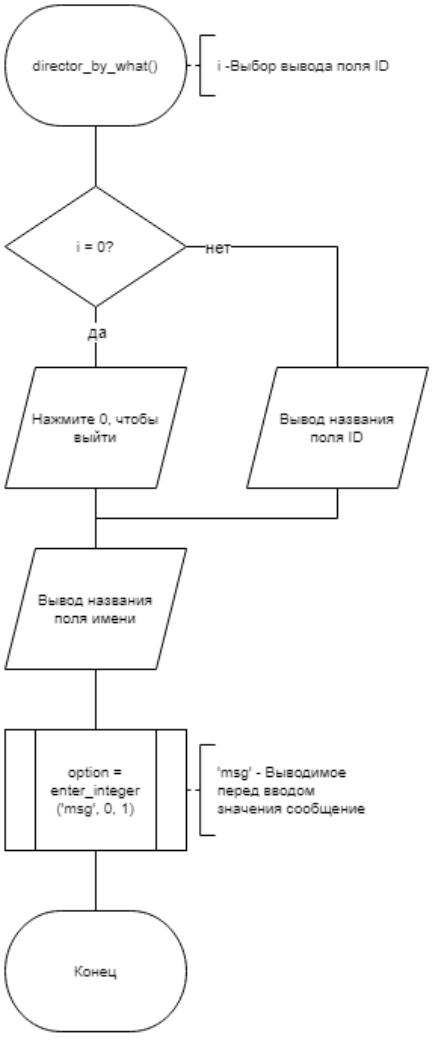

enter_float()
message - Выводимое перед вводом значения сообщение
a, b - Диапазон вводимого числа
flag = 0
Вывод сообщения (message)
Ввод значения
Введённое значение - вещественное число в диапазоне [a,b]?
нет
да
flag = 1
Введённое значение некорректно
flag = 0?
да
нет
Конец

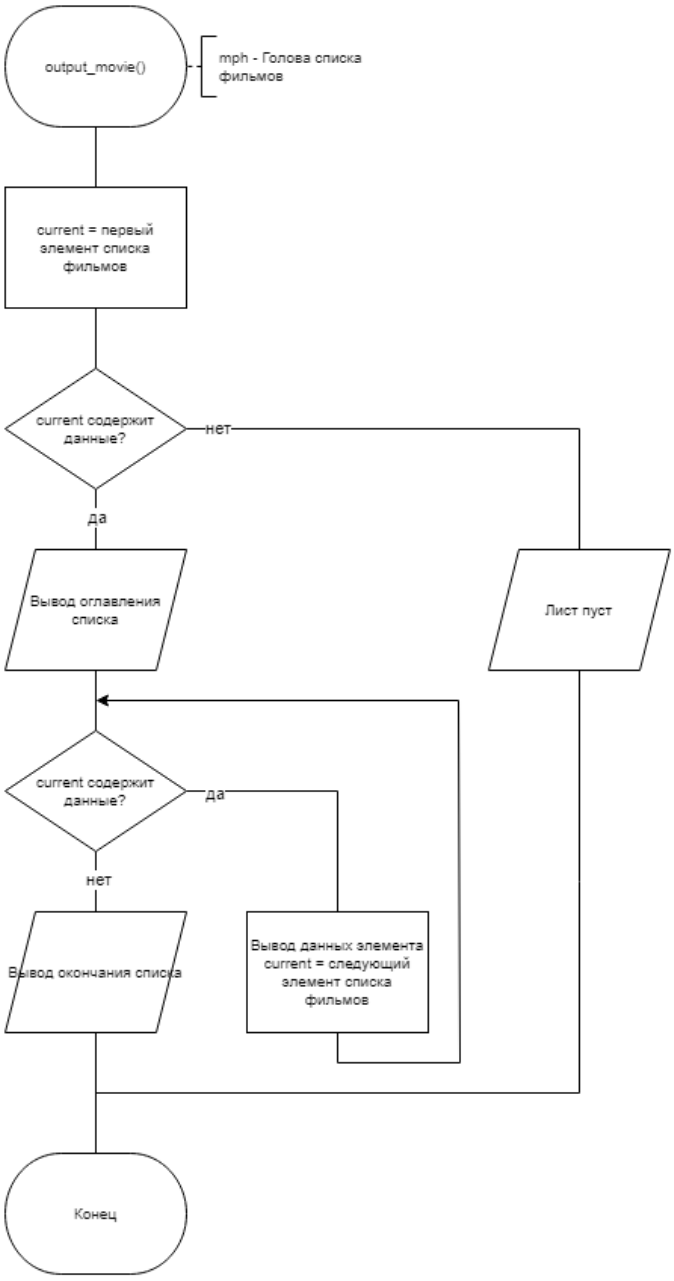

enter_date()
message - Выводимое перед вводом значения сообщение
day, month, year - Элементы даты
flag = 0
Вывод сообщения (message)
Ввод значения
Введённое значение даты корректно?
Корректность даты проверяется тем, что введённые значения являются целыми числами, а также день, месяц и год существующими значениями
flag = 1
Введённое значение некорректно
flag = 0?
Конец

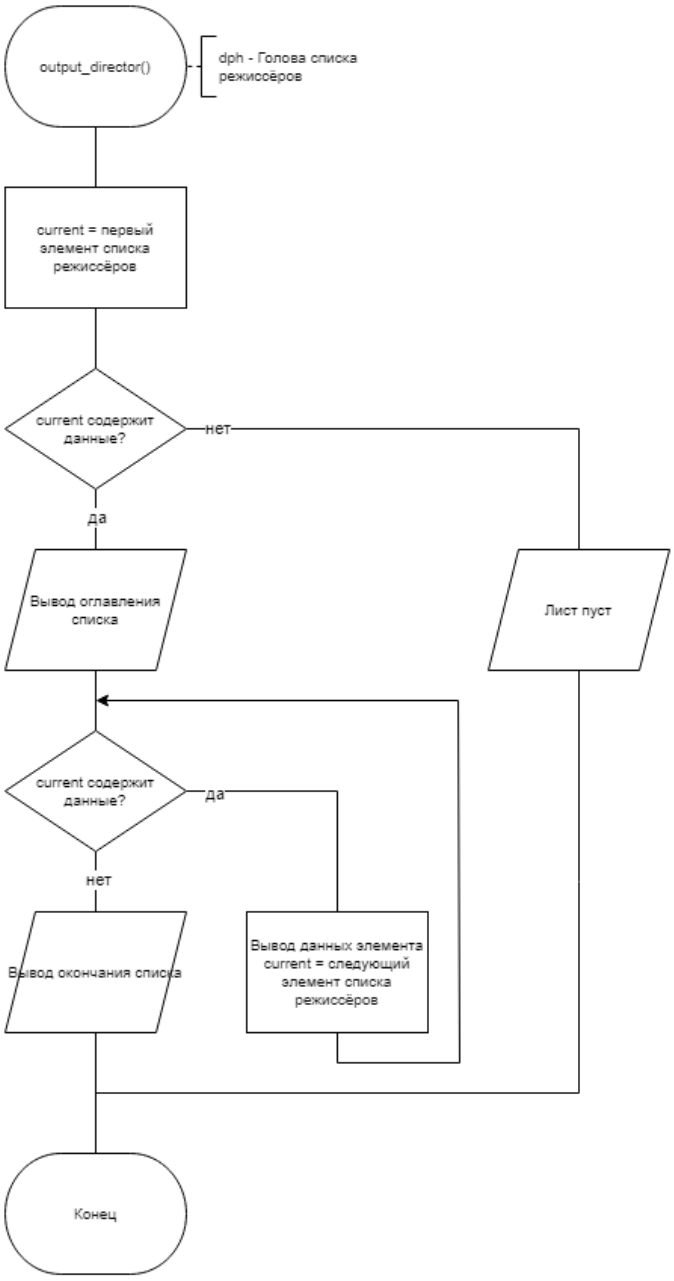

menu()
Вывод опций программы
option = enter_integer ('msg', 0, 7)
'msg' - Выводимое перед вводом значения сообщение
Конец

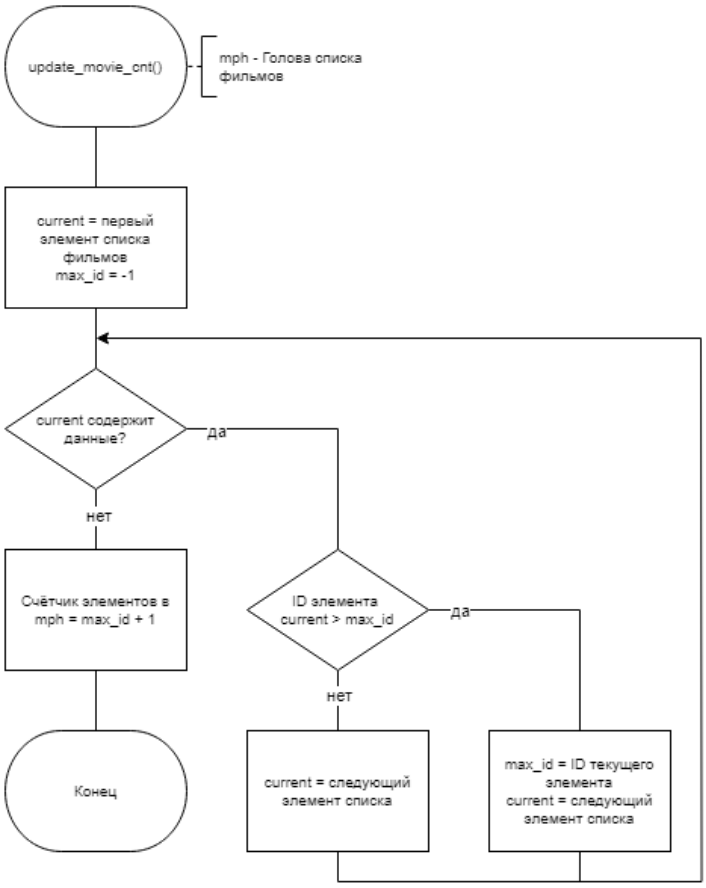

detail()
Вывод названия списков для выбора
option = enter_integer ('msg', 0, 1)
'msg' - Выводимое перед вводом значения сообщение
Конец

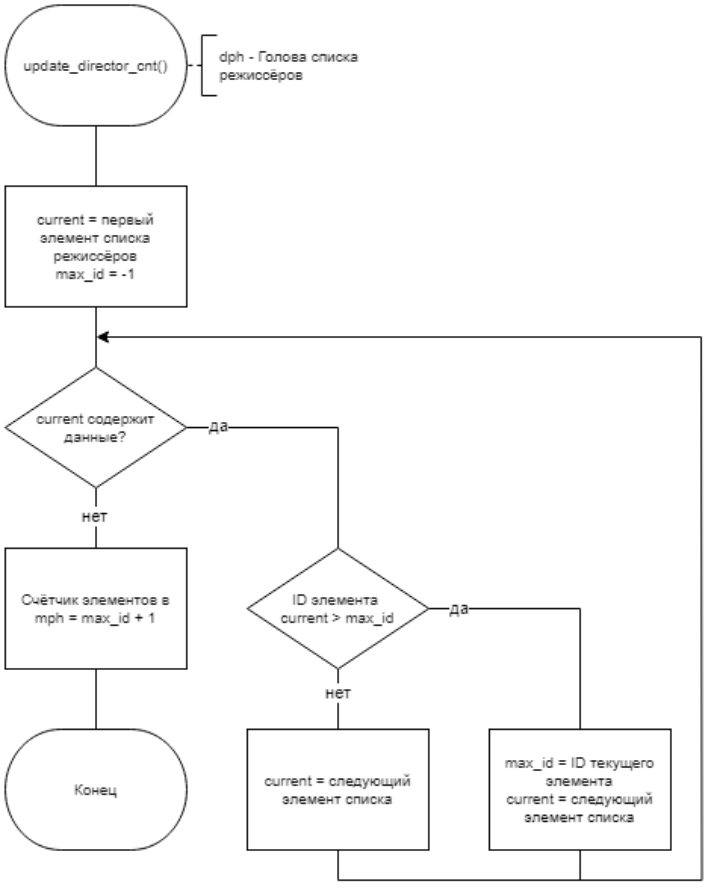

movie_by_what()
i -Выбор вывода поля ID
i = 0?
нет
да
Нажмите 0, чтобы выйти
Вывод названия поля ID
Вывод названий полей структуры MOV
Всех полей, кроме ID
option = enter_integer ('msg', 0, 7)
'msg' - Выводимое перед вводом значения сообщение
Конец

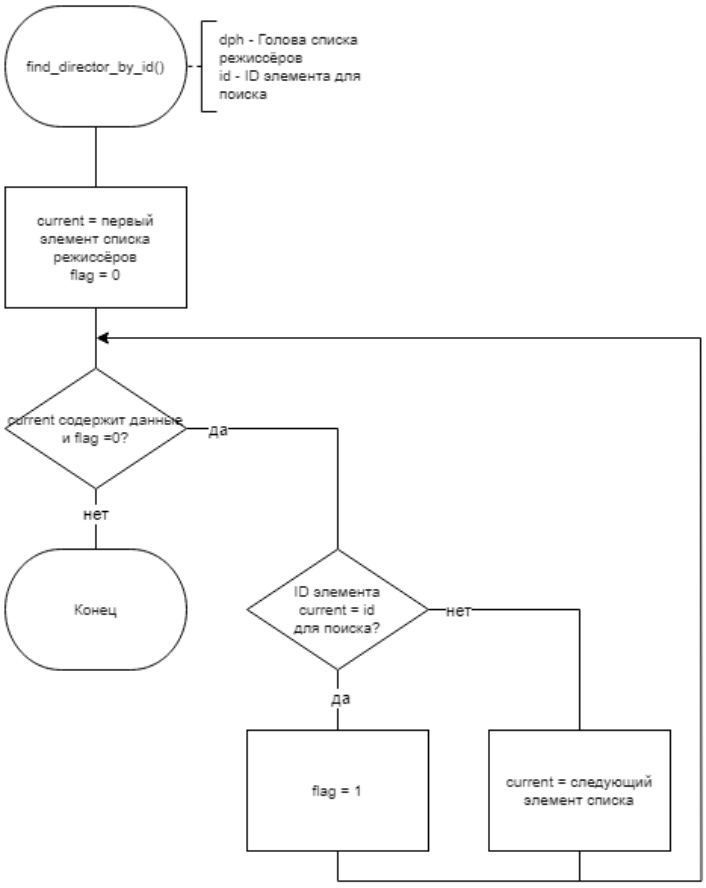

director_by_what()
i -Выбор вывода поля ID
i = 0?
нет
да
Нажмите 0, чтобы выйти
Вывод названия поля ID
Вывод названия поля имени
option = enter_integer ('msg', 0, 1)
'msg' - Выводимое перед вводом значения сообщение
Конец

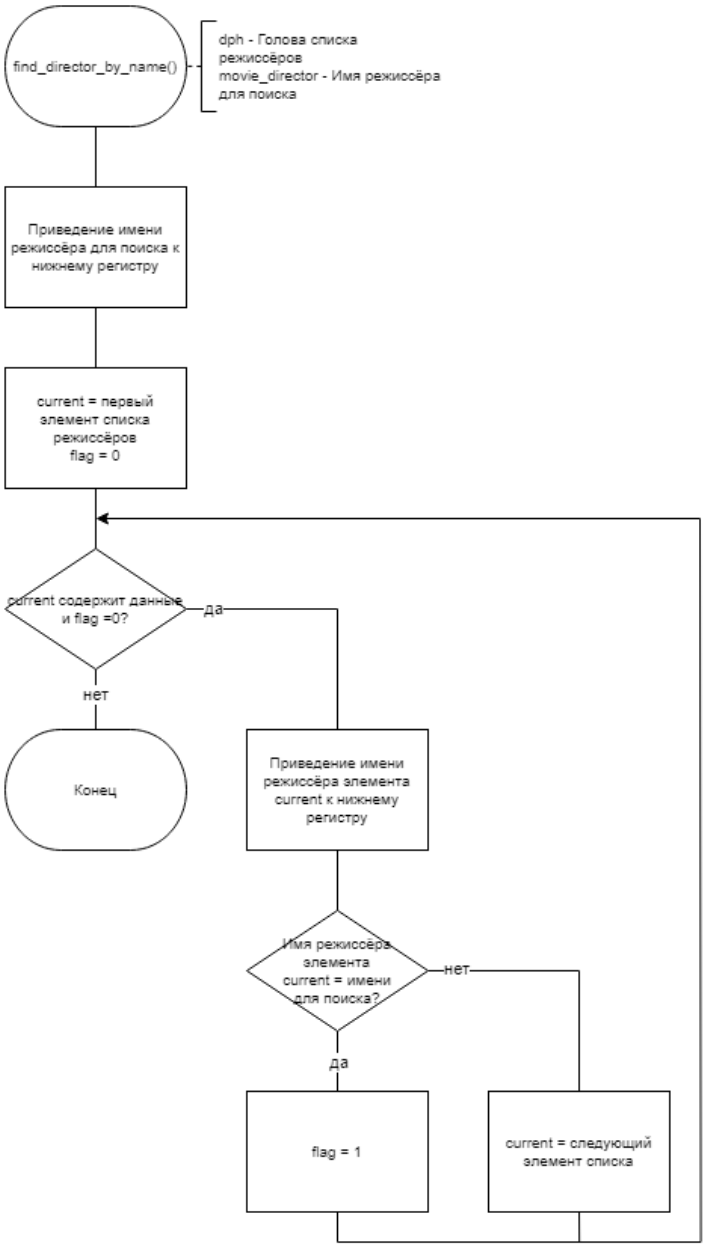

output_movie()
mph - Голова списка фильмов
current = первый элемент списка фильмов
current содержит данные?
нет
да
Вывод оглавления списка
Лист пуст
current содержит данные?
да
нет
Вывод окончания списка
Вывод данных элемента current = следующий элемент списка фильмов
Конец

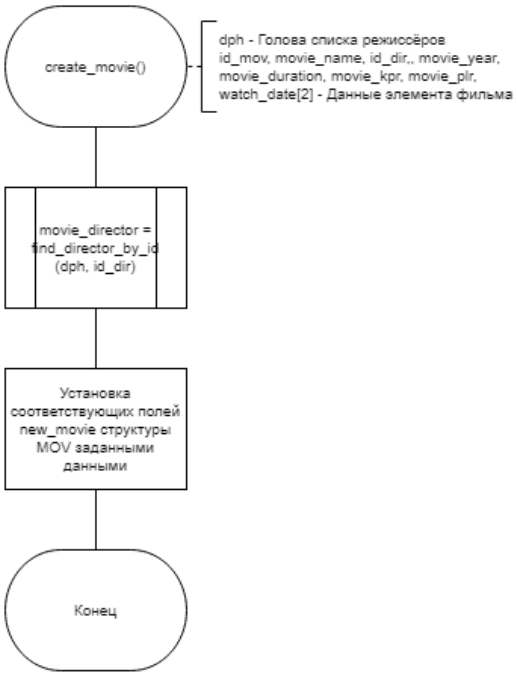

output_director()
dph - Голова списка режиссёров
current = первый элемент списка режиссёров
current содержит данные?
нет
да
Вывод оглавления списка
Лист пуст
current содержит данные?
да
нет
Вывод данных элемента current = следующий элемент списка режиссёров
Вывод окончания списка
Конец

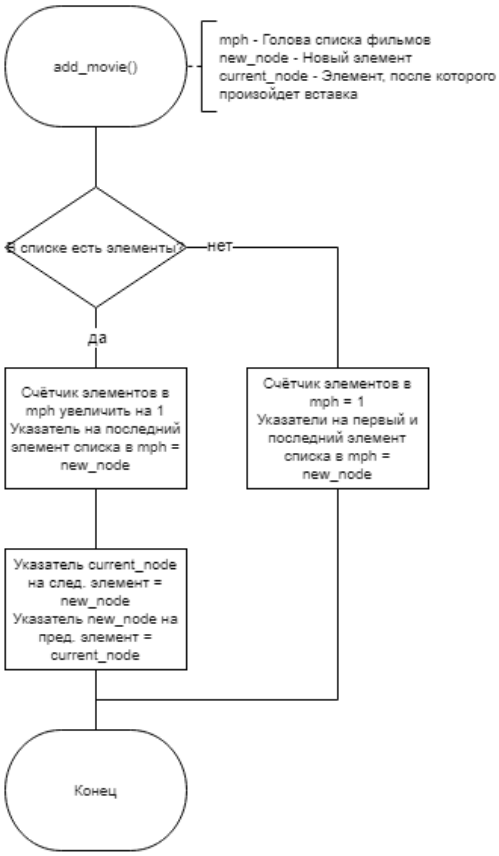

update_movie_cnt()
mph - Голова списка фильмов
current = первый элемент списка фильмов
max_id = -1
current содержит данные?
да
нет
Счётчик элементов в mph = max_id + 1
ID элемента current > max_id
да
нет
Конец
current = следующий элемент списка
max_id = ID текущего элемента
current = следующий элемент списка

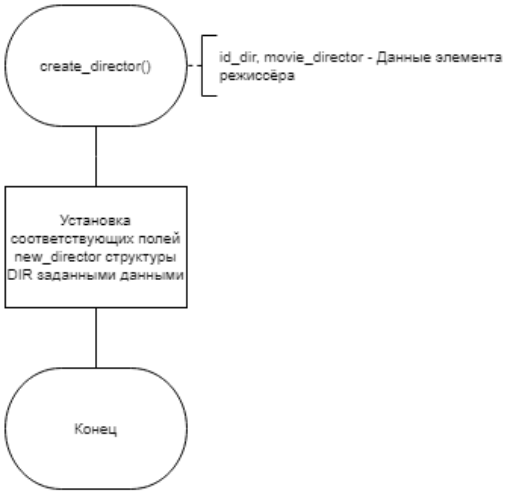

update_director_cnt()
dph - Голова списка режиссёров
current = первый элемент списка режиссёров
max_id = -1
current содержит данные?
да
нет
Счётчик элементов в mph = max_id + 1
ID элемента current > max_id
да
нет
Конец
current = следующий элемент списка
max_id = ID текущего элемента
current = следующий элемент списка


find_director_by_id()
dph - Голова списка режиссёров
id - ID элемента для поиска
current = первый элемент описка режиссёров
flag = 0
current содержит данные и flag =0?
да
нет
Конец
ID элемента current = id для поиска?
нет
да
flag = 1
current = следующий элемент списка

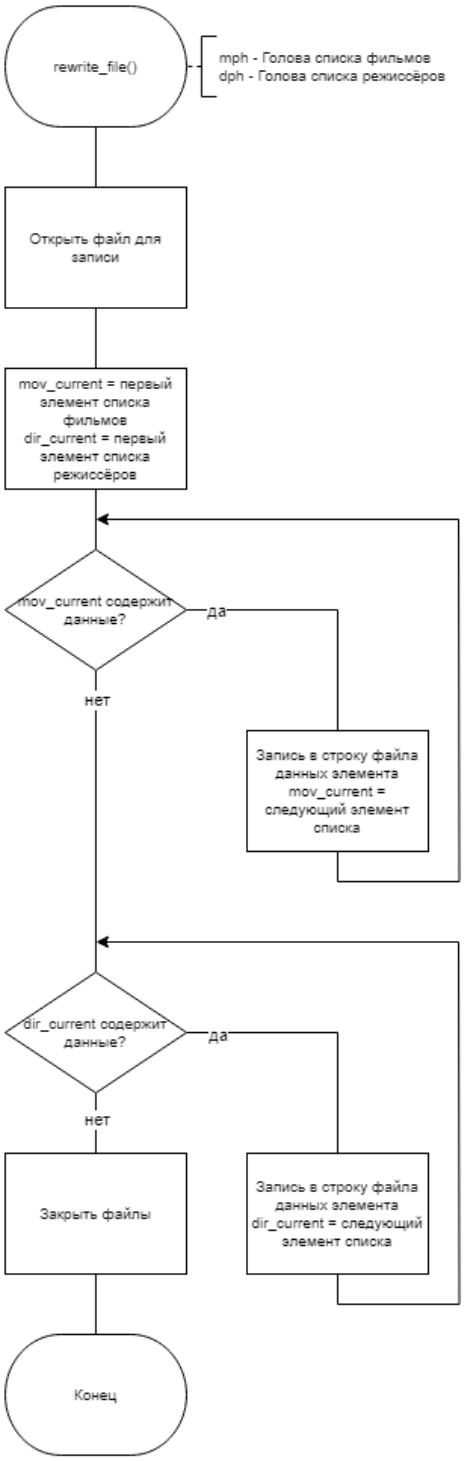

find_director_by_name()
dph - Голова списка режиссёров
movie_director - Имя режиссёра для поиска
Приведение имени режиссёра для поиска к нижнему регистру
current = первый элемент списка режиссёров
flag = 0
current содержит данные и flag =0?
да
нет
Конец
Приведение имени режиссёра элемента current к нижнему регистру
Имя режиссёра элемента current = имени для поиска?
нет
да
flag = 1
current = следующий элемент списка

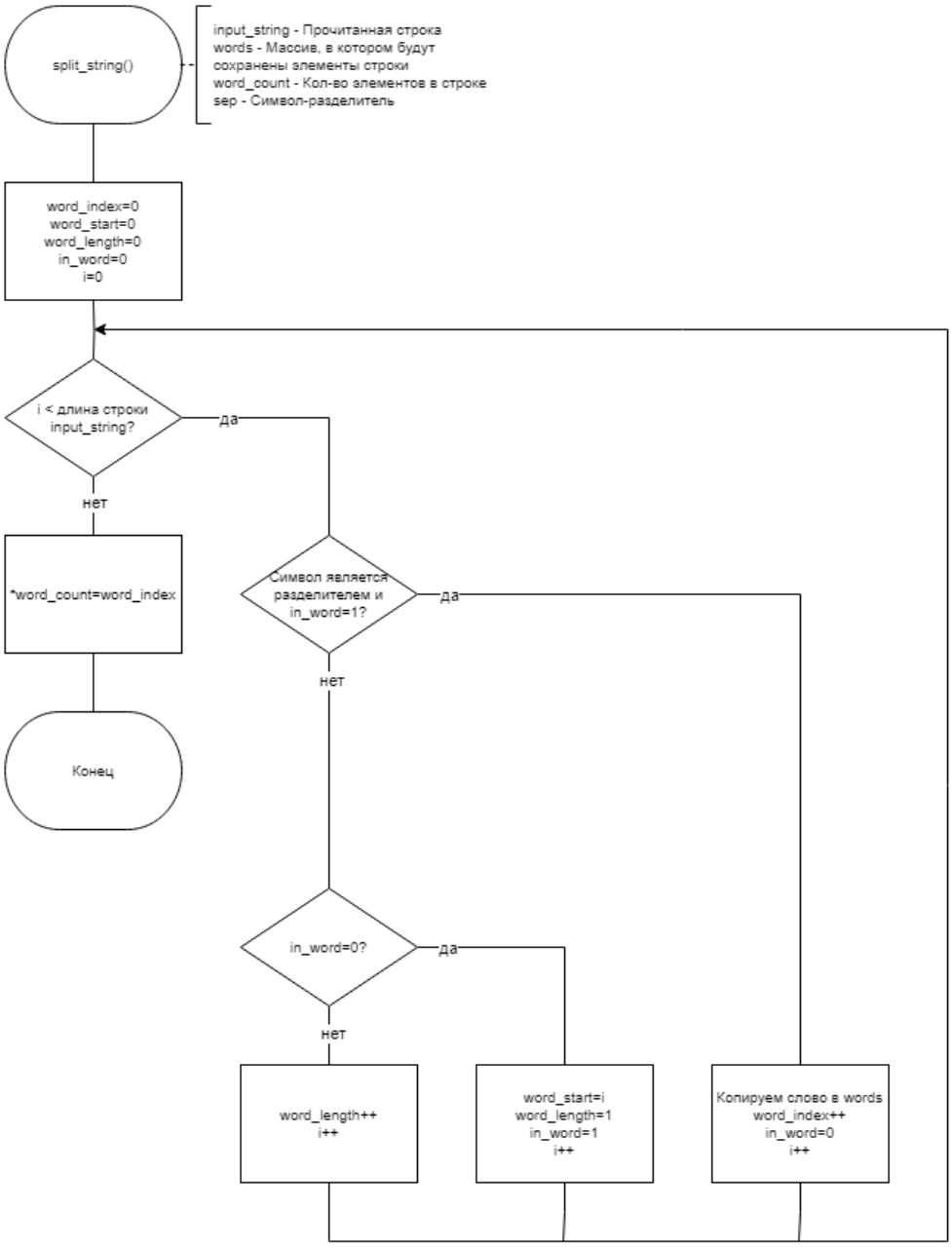

create_movie()
dph - Голова списка режисёров
id_mov, movie_name, id_dir,, movie_year,
movie_duration, movie_kpr, movie_plr,
watch_date[2] - Данные элемента фильма
movie_director =
find_director_by_id
(dph, id_dir)
Установка
соответствующих полей
new_movie структуры
MOV заданными
данными
Конец

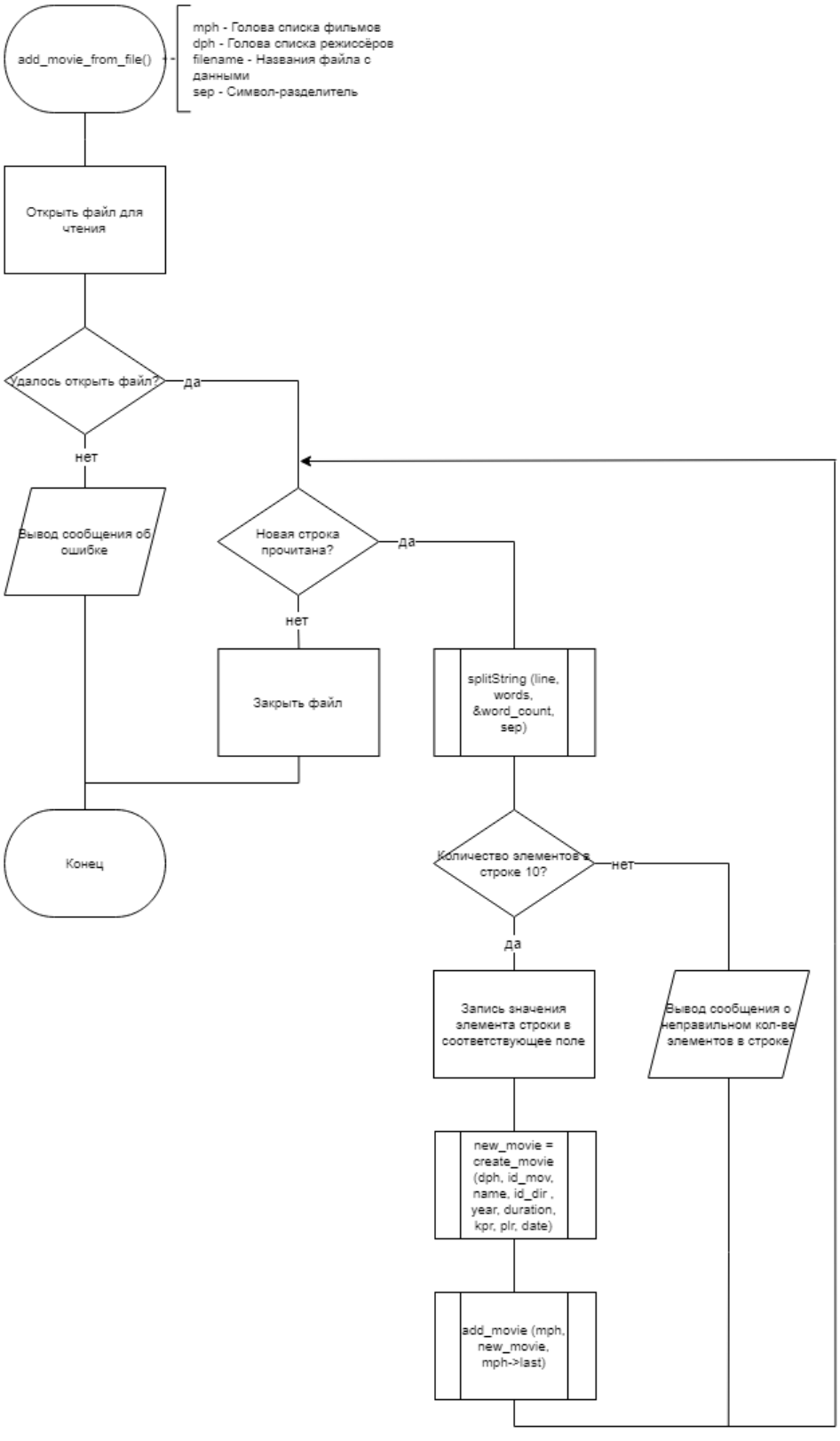

add_movie()
mph - Голова списка фильмов
new_node - Новый элемент
current_node - Элемент, после которого
произойдет вставка
В списке есть элементы?
нет
да
Счётчик элементов в
mph увеличить на 1
Указатель на последний
элемент списка в mph =
new_node
Счётчик элементов в
mph = 1
Указатели на первый и
последний элемент
списка в mph =
new_node
Указатель current_node
на след. элемент =
new_node
Указатель new_node на
пред. элемент =
current_node
Конец

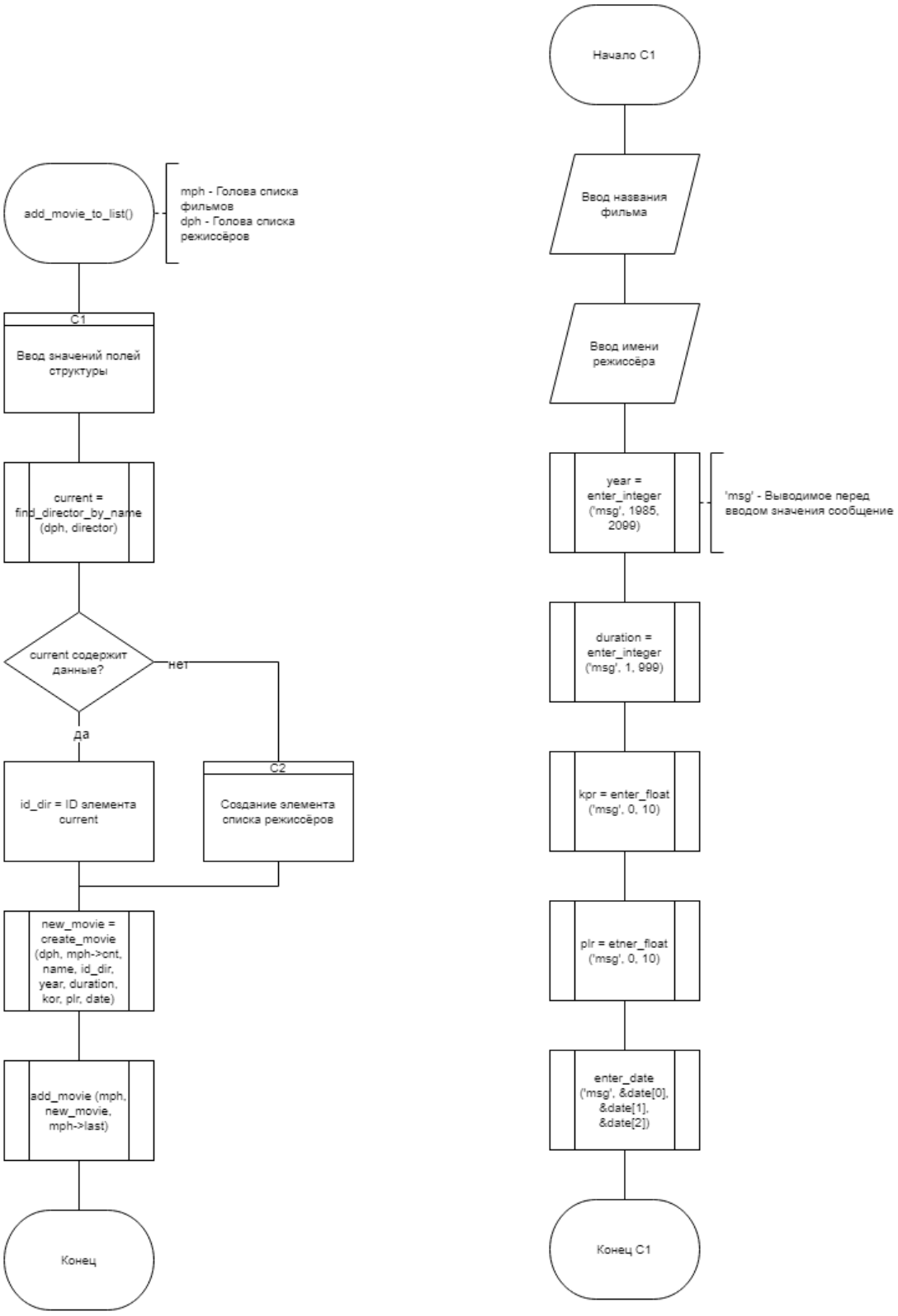

create_director()
id_dir, movie_director - Данные элемента
режисёра
Установка
соответствующих полей
new_director структуры
DIR заданными данными
Конец

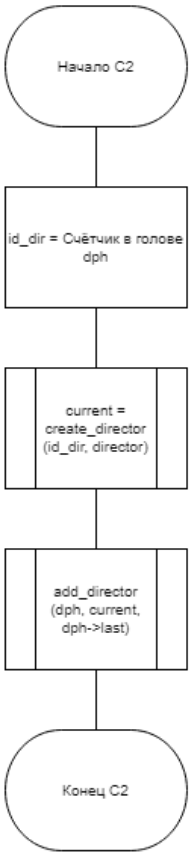

add_director()
dph - Голова списка режисёров
new_node - Новый элемент
current_node - Элемент, после которого
произойдет вставка
В списке есть элементы?
нет
да
Счётчик элементов в dph
увеличить на 1
Указатель на последний
элемент списка в dph =
new_node
Счётчик элементов в dph
= 1
Указатели на первый и
последний элемент
списка в dph = new_node
Указатель current_node
на след. элемент =
new_node
Указатель new_node на
пред. элемент =
current_node
Конец

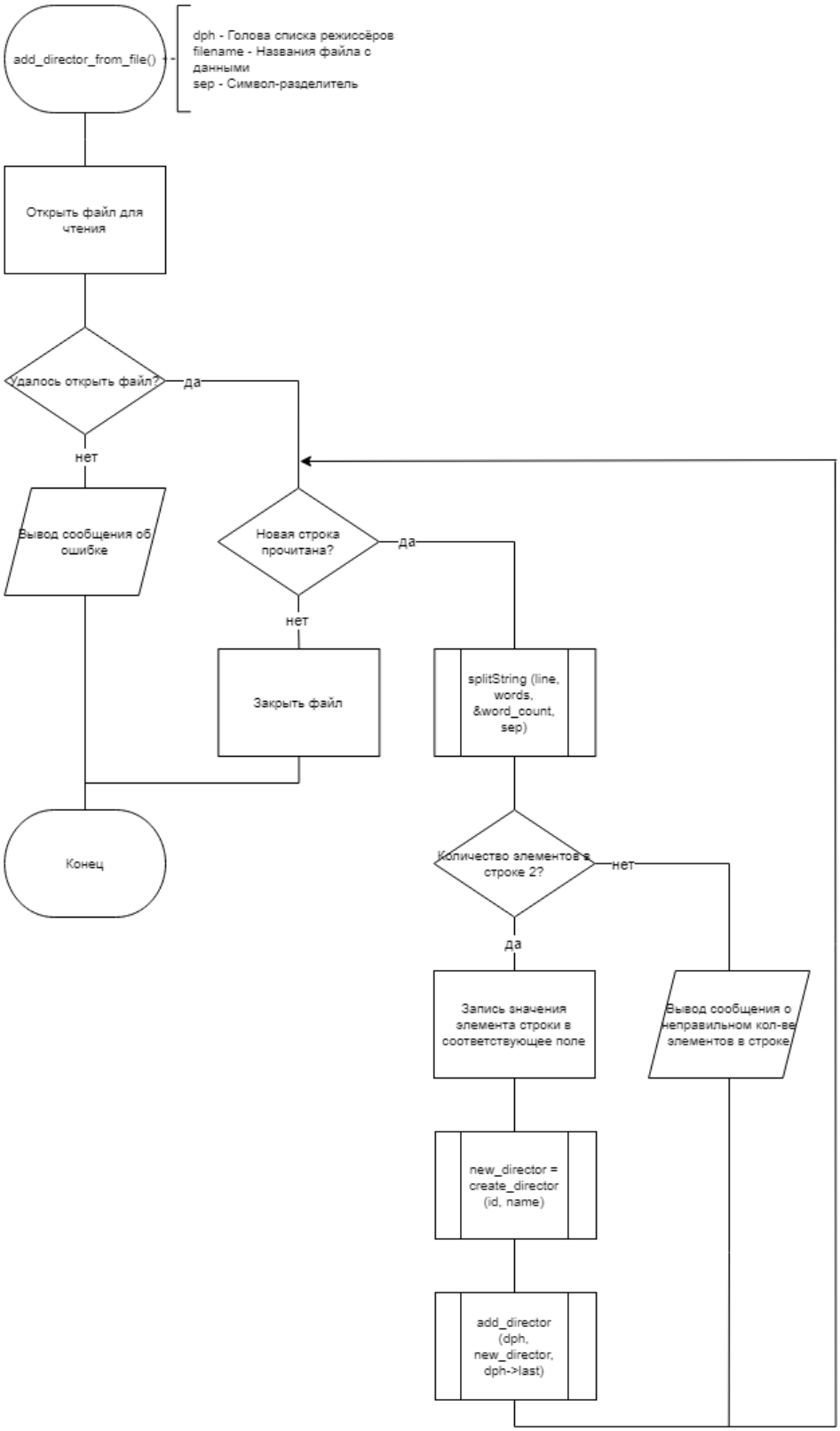

rewrite_file()
mph - Голова списка фильмов
dph - Голова списка режиссёров
Открыть файл для записи
mov_current = первый элемент списка фильмов
dir_current = первый элемент списка режиссёров
mov_current содержит данные?
да
нет
Запись в строку файла данных элемента mov_current = следующий элемент списка
dir_current содержит данные?
да
нет
Закрыть файлы
Запись в строку файла данных элемента dir_current = следующий элемент списка
Конец

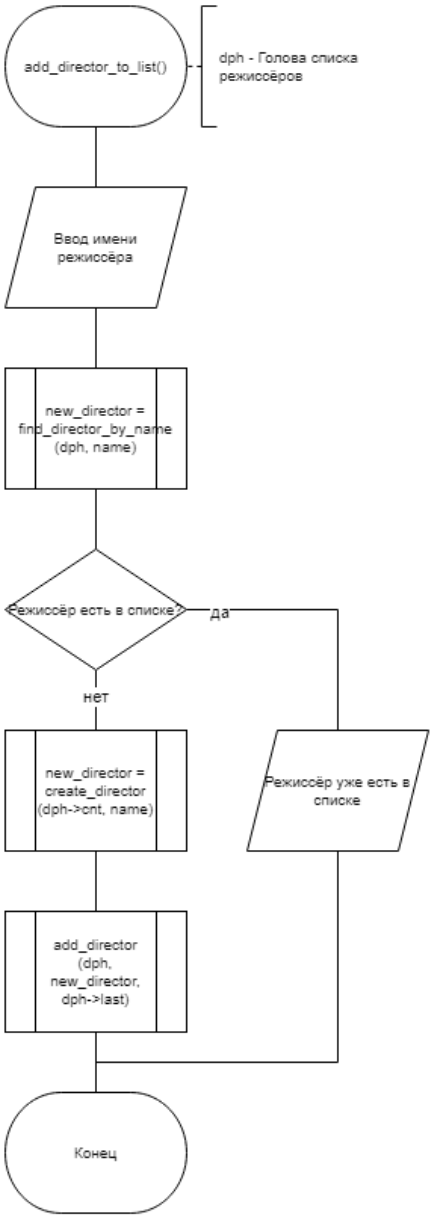

split_string()
input_string - Прочитанная строка
words - Массив, в котором будут
сохранены элементы строки
word_count - Кол-во элементов в строке
sep - Символ-разделитель
word_index=0
word_start=0
word_length=0
in_word=0
i=0
i < длина строки
input_string?
да
нет
*word_count=word_index
Конец
Символ является
разделителем и
in_word=1?
да
нет
in_word=0?
да
нет
word_length++
i++
word_start=i
word_length=1
in_word=1
i++
Копируем слово в words
word_index++
in_word=0
i++

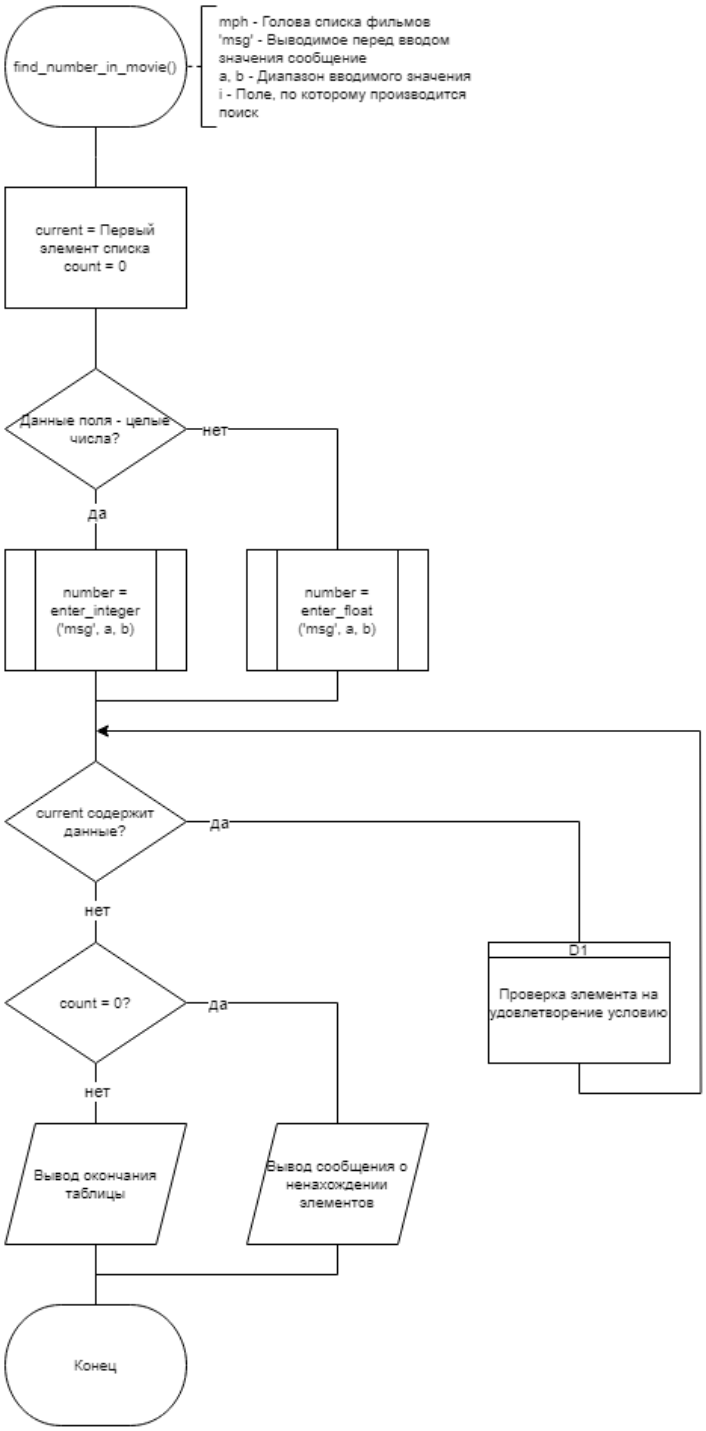

add_movie_from_file()
mph - Голова списка фильмов
dph - Голова списка режиссёров
filename - Названия файла с данными
sep - Символ-разделитель
Открыть файл для чтения
Удалось открыть файл?
да
нет
Вывод сообщения об ошибке
Новая строка прочитана?
да
нет
Закрыть файл
splitString (line, words, &word_count, sep)
Конец
Количество элементов в строке 10?
нет
да
Запись значения элемента строки в соответствующее поле
Вывод сообщения о неправильном кол-ве элементов в строке
new_movie = create_movie (dph, id_mov, name, id_dir , year, duration, kpr, plr, date)
add_movie (mph, new_movie, mph->last)

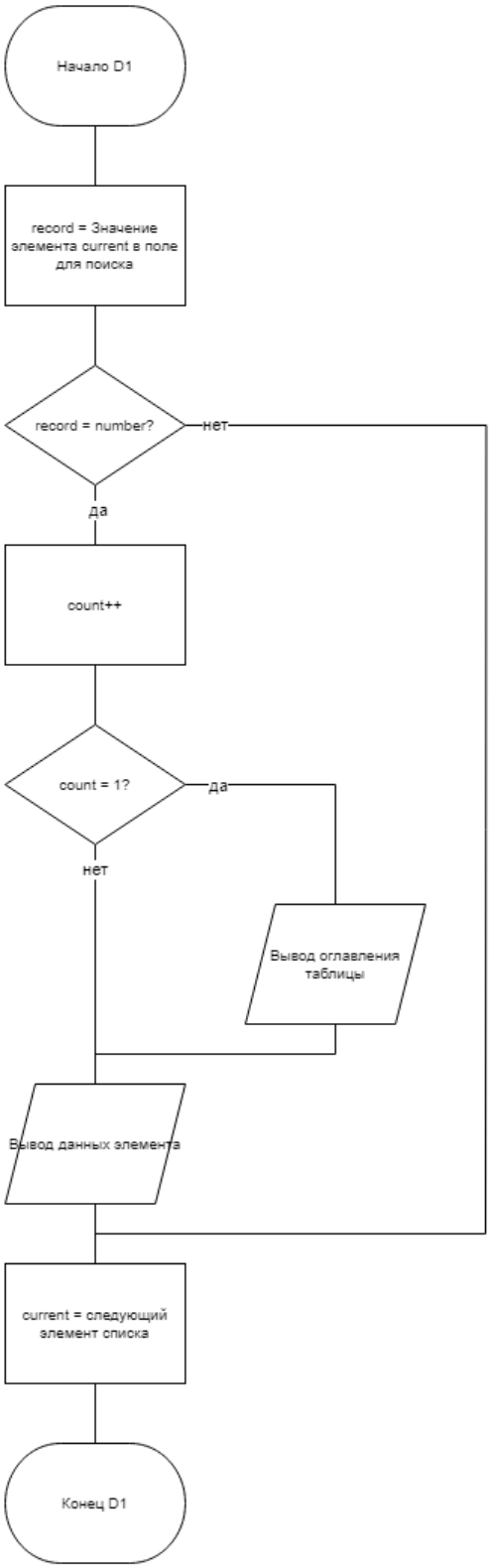

add_movie_to_list()
mph - Голова списка фильмов
dph - Голова списка режиссёров
C1
Ввод значений полей структуры
current = find_director_by_name (dph, director)
current содержит данные?
нет
да
id_dir = ID элемента current
C2
Создание элемента списка режиссёров
new_movie = create_movie (dph, mph->cnt, name, id_dir, year, duration, kor, plr, date)
add_movie (mph, new_movie, mph->last)
Конец
Начало C1
Ввод названия фильма
Ввод имени режиссёра
year = enter_integer ('msg', 1985, 2099)
'msg' - Выводимое перед вводом значения сообщение
duration = enter_integer ('msg', 1, 999)
kpr = enter_float ('msg', 0, 10)
plr = etner_float ('msg', 0, 10)
enter_date ('msg', &date[0], &date[1], &date[2])
Конец C1

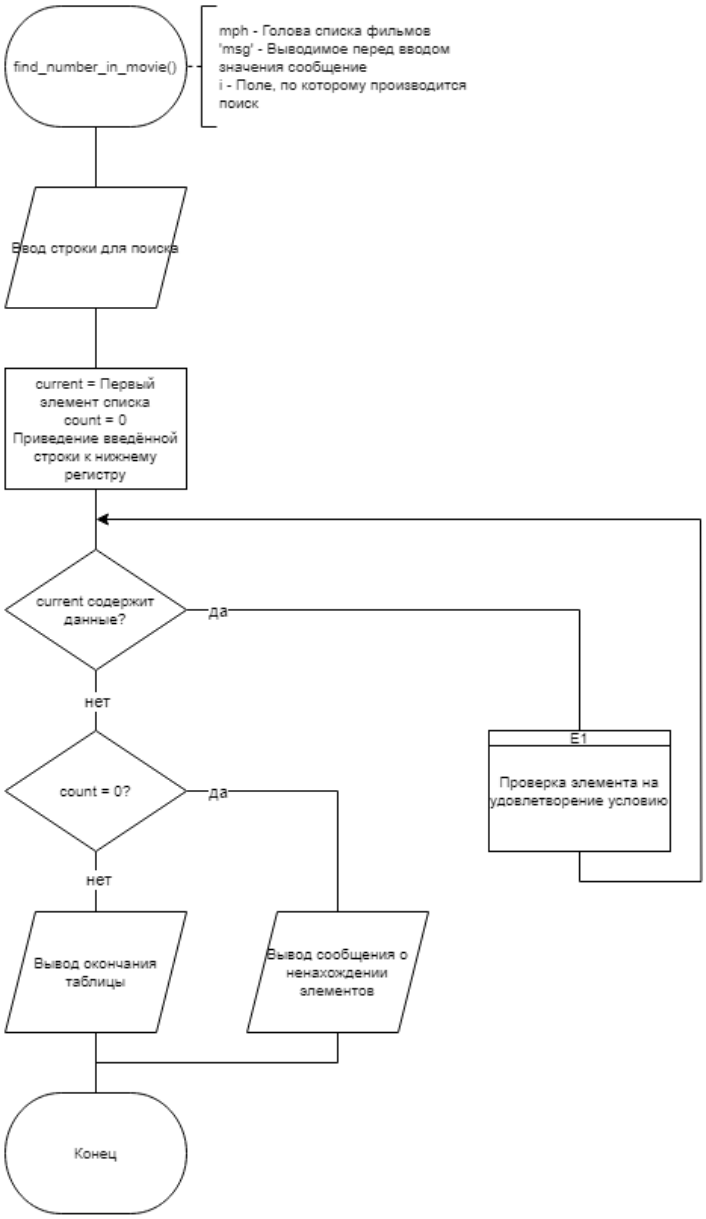

Начало C2
id_dir = Счётчик в голове dph
current = create_director (id_dir, director)
add_director (dph, current, dph->last)
Конец C2

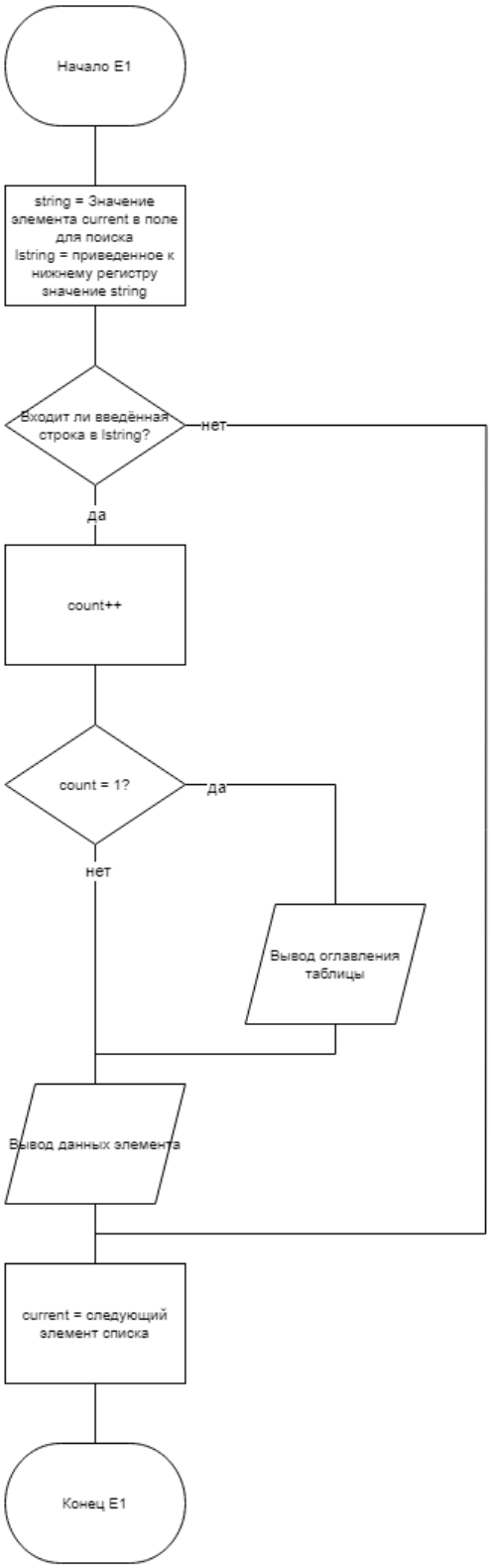

add_director_from_file()
dph - Голова списка режиссёров
filename - Названия файла с данными
sep - Символ-разделитель
Открыть файл для чтения
Удалось открыть файл?
да
нет
Вывод сообщения об ошибке
Новая строка прочитана?
да
нет
Закрыть файл
splitString (line, words, &word_count, sep)
Конец
Количество элементов в строке 2?
нет
да
Запись значения элемента строки в соответствующее поле
Вывод сообщения о неправильном кол-ве элементов в строке
new_director = create_director (id, name)
add_director (dph, new_director, dph->last)

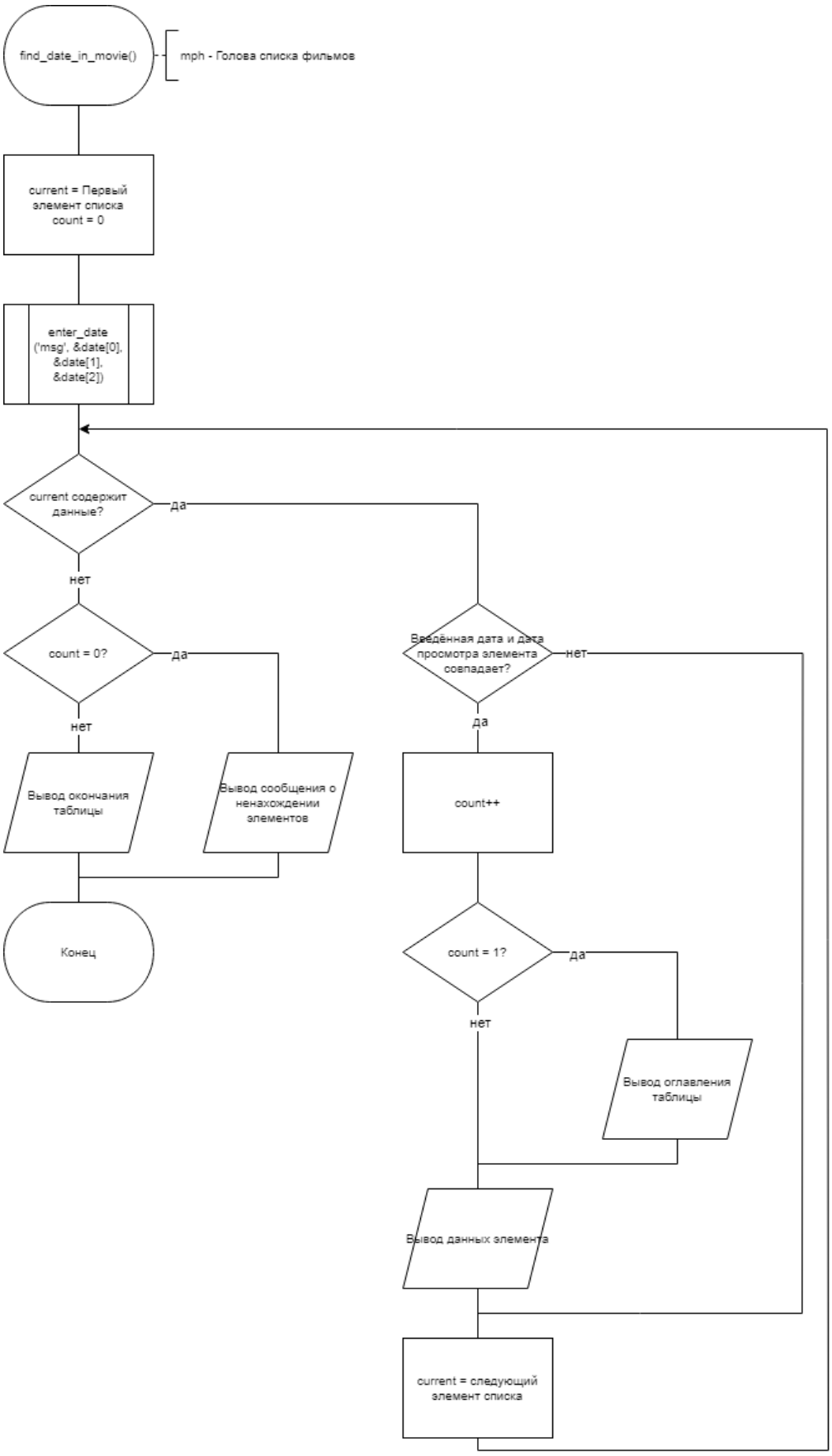

add_director_to_list()
dph - Голова списка режиссёров
Ввод имени режиссёра
new_director = find_director_by_name (dph, name)
Режиссёр есть в списке?
да
нет
new_director = create_director (dph->cnt, name)
Режиссёр уже есть в списке
add_director (dph, new_director, dph->last)
Конец

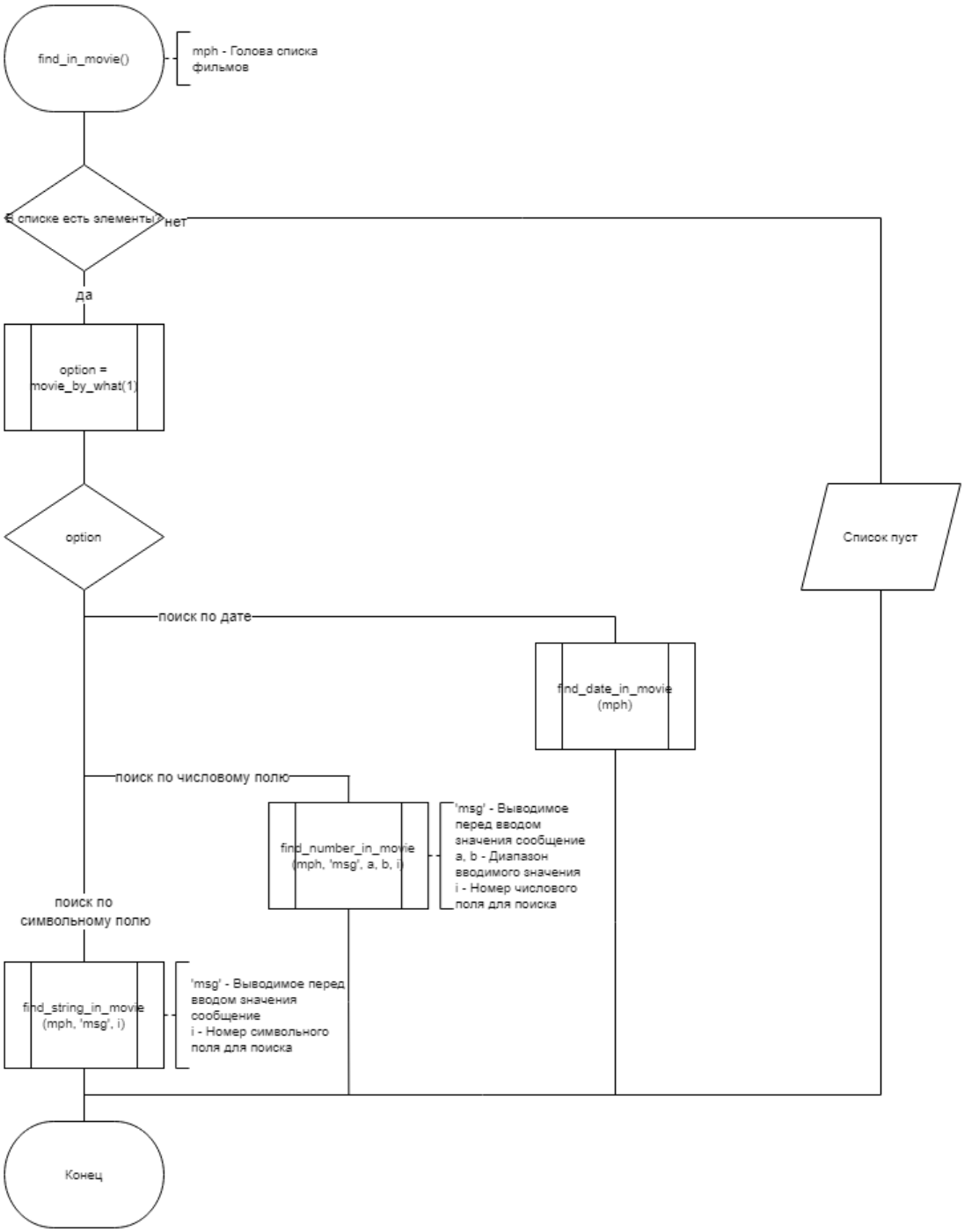

find_number_in_movie()
mph - Голова списка фильмов
'msg' - Выводимое перед вводом значения сообщение
a, b - Диапазон вводимого значения
i - Поле, по которому производится поиск
current = Первый элемент списка
count = 0
Данные поля - целые числа?
нет
да
number = enter_integer ('msg', a, b)
number = enter_float ('msg', a, b)
current содержит данные?
да
нет
D1
Проверка элемента на удовлетворение условию
count = 0?
да
нет
Вывод окончания таблицы
Вывод сообщения о ненахождении элементов
Конец

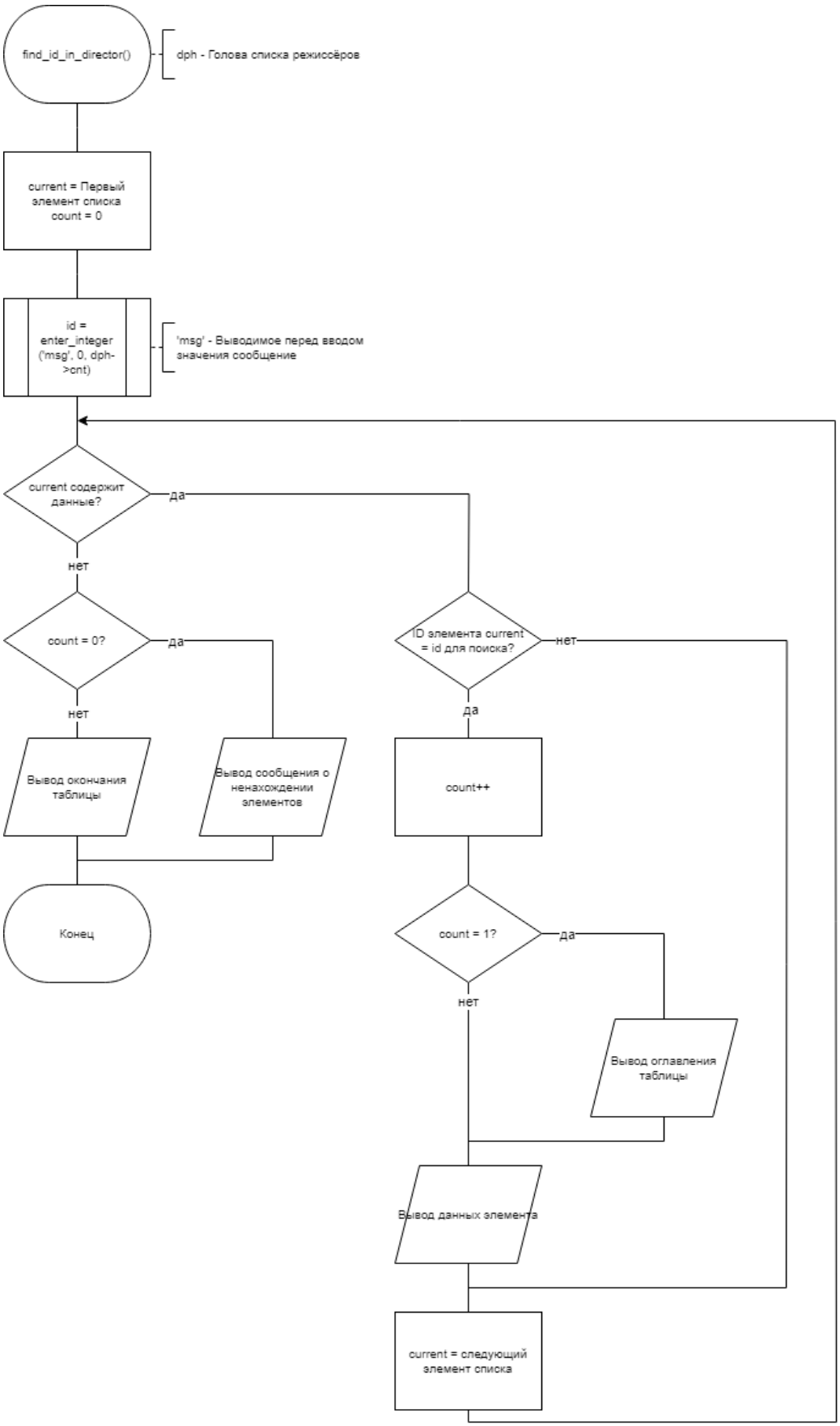

Начало D1
record = Значение элемента current в поле для поиска
record = number?
нет
да
count++
count = 1?
да
нет
Вывод оглавления таблицы
Вывод данных элемента
current = следующий элемент списка
Конец D1

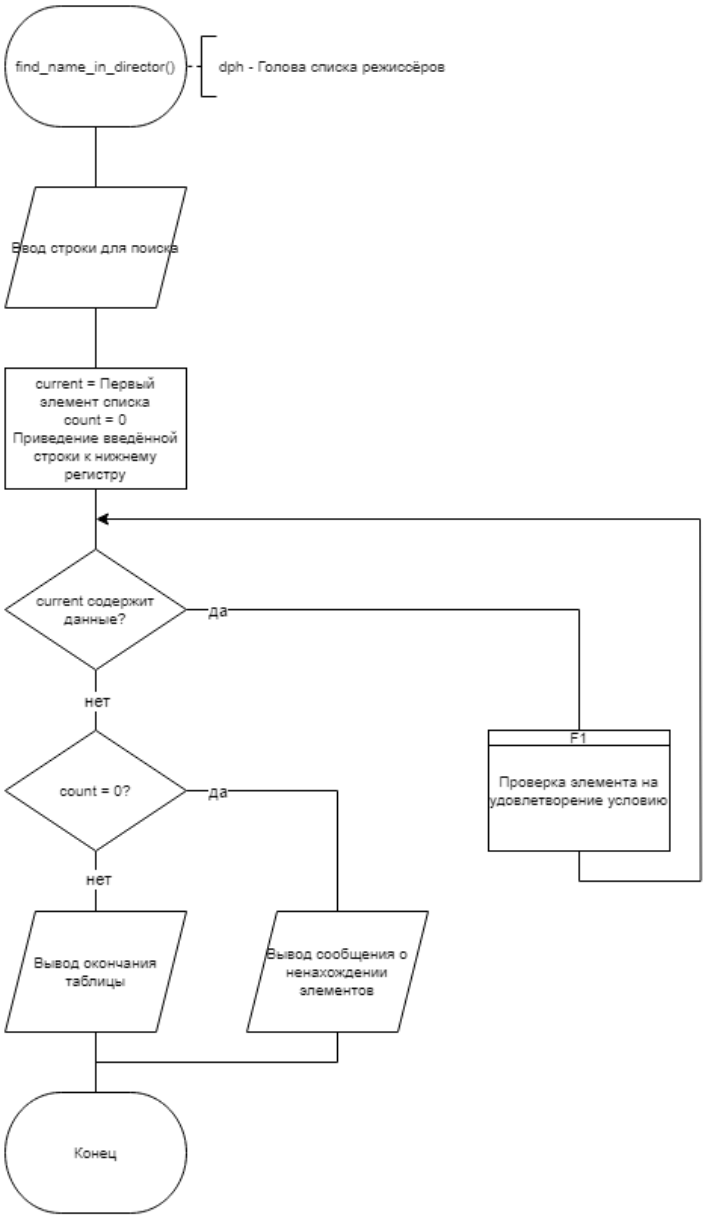

find_number_in_movie()
mph - Голова списка фильмов
'msg' - Выводимое перед вводом значения сообщение
i - Поле, по которому производится поиск
Ввод строки для поиска
current = Первый элемент списка
count = 0
Приведение введённой строки к нижнему регистру
current содержит данные?
да
нет
E1
Проверка элемента на удовлетворение условию
count = 0?
да
нет
Вывод окончания таблицы
Вывод сообщения о ненахождении элементов
Конец

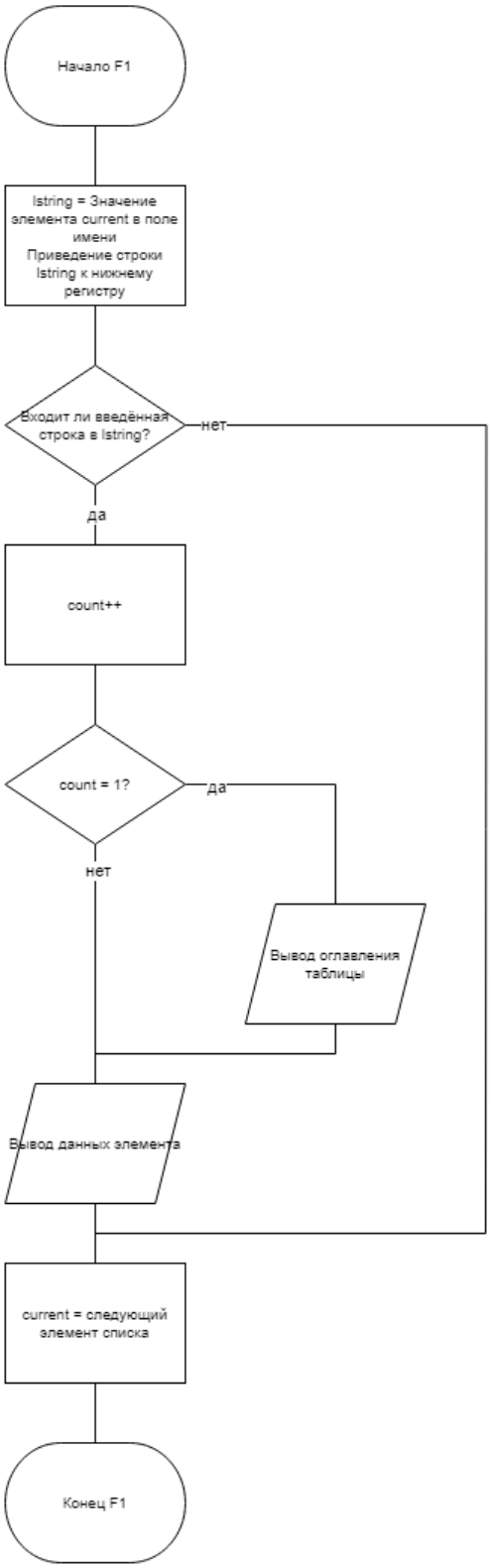

Начало E1
string = Значение элемента current в поле для поиска
lstring = приведенное к нижнему регистру значение string
Входит ли введённая строка в lstring?
нет
да
count++
count = 1?
да
нет
Вывод оглавления таблицы
Вывод данных элемента
current = следующий элемент списка
Конец E1

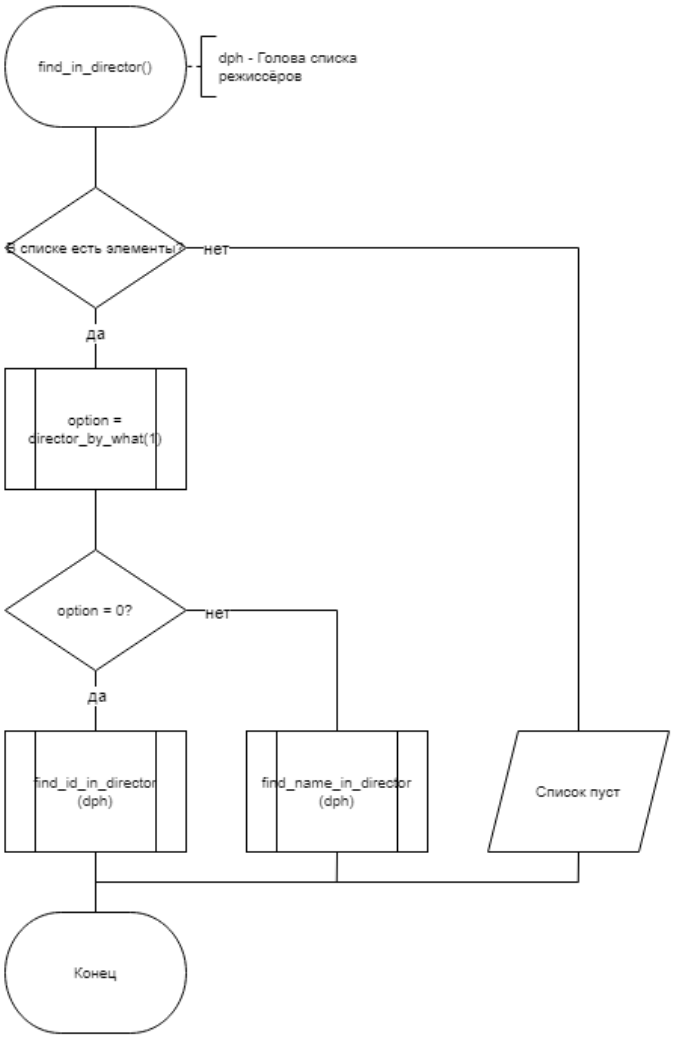

find_date_in_movie()
mph - Голова списка фильмов
current = Первый элемент списка
count = 0
enter_date ('msg', &date[0], &date[1], &date[2])
current содержит данные?
да
нет
count = 0?
да
нет
Вывод окончания таблицы
Вывод сообщения о ненахождении элементов
Конец
Введённая дата и дата просмотра элемента совпадает?
нет
да
count++
count = 1?
да
нет
Вывод оглавления таблицы
Вывод данных элемента
current = следующий элемент списка

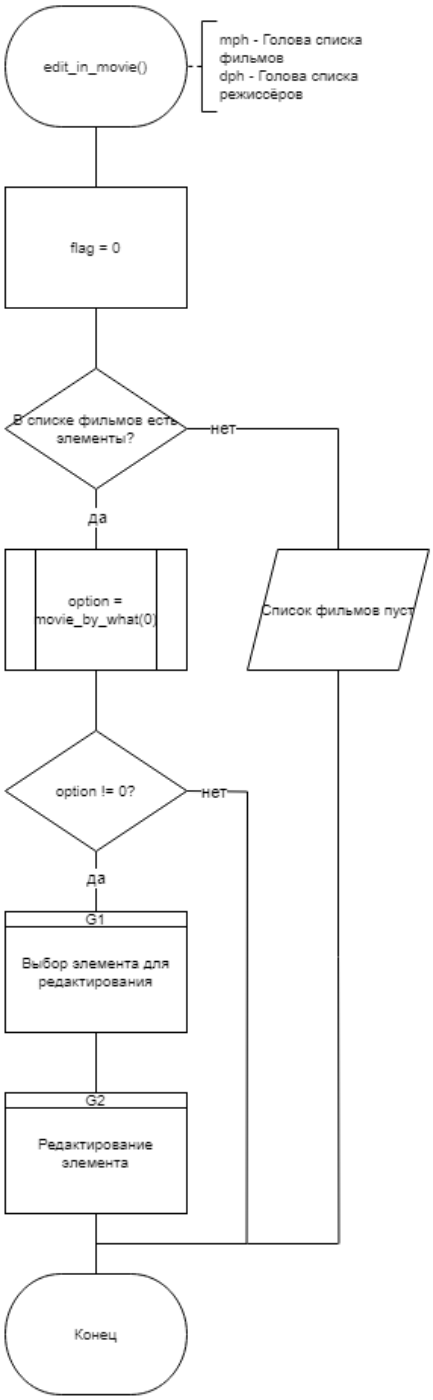

find_in_movie()
mph - Голова списка фильмов
В списке есть элементы?
нет
да
option = movie_by_what(1)
option
Список пуст
поиск по дате
find_date_in_movie (mph)
поиск по числовому полю
find_number_in_movie (mph, 'msg', a, b, i)
'msg' - Выводимое перед вводом значения сообщение
a, b - Диапазон вводимого значения
i - Номер числового поля для поиска
поиск по символьному полю
find_string_in_movie (mph, 'msg', i)
'msg' - Выводимое перед вводом значения сообщение
i - Номер символьного поля для поиска
Конец

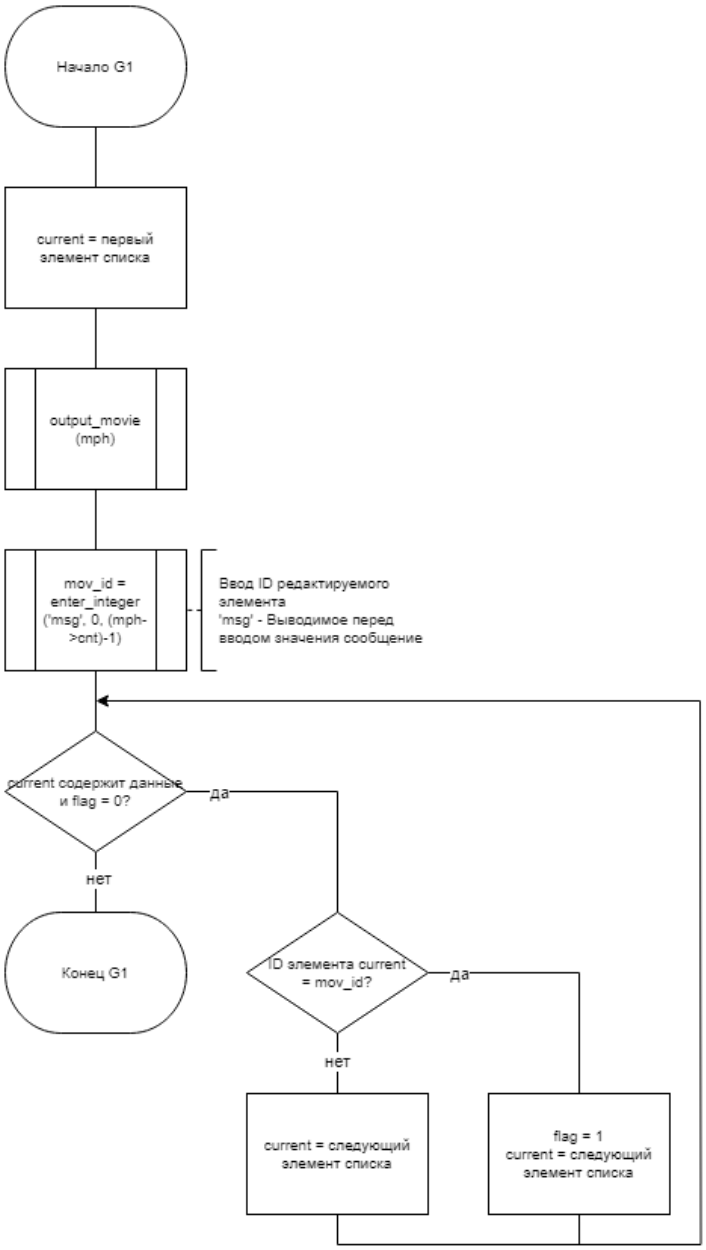

find_id_in_director()
dph - Голова списка режиссёров
current = Первый элемент списка count = 0
id = enter_integer ('msg', 0, dph->cnt)
'msg' - Выводимое перед вводом значения сообщение
current содержит данные?
да
нет
count = 0?
да
нет
Вывод окончания таблицы
Вывод сообщения о ненахождении элементов
Конец
ID элемента current = id для поиска?
нет
да
count++
count = 1?
да
нет
Вывод оглавления таблицы
Вывод данных элемента
current = следующий элемент списка

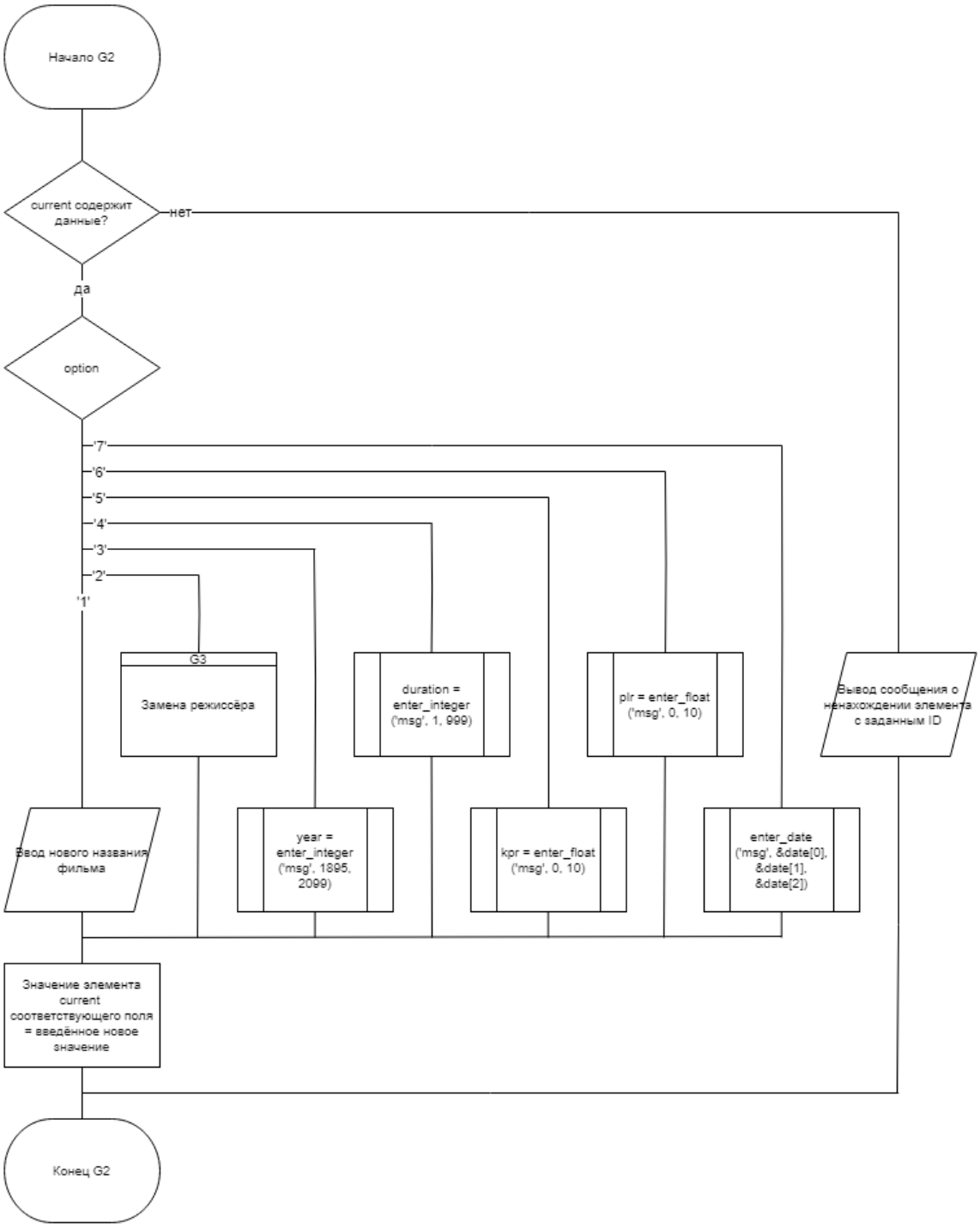

find_name_in_director()
dph - Голова списка режисёров
Ввод строки для поиска
current = Первый элемент списка
count = 0
Приведение введённой строки к нижнему регистру
current содержит данные?
да
нет
count = 0?
да
нет
F1
Проверка элемента на удовлетворение условию
Вывод окончания таблицы
Вывод сообщения о ненахождении элементов
Конец

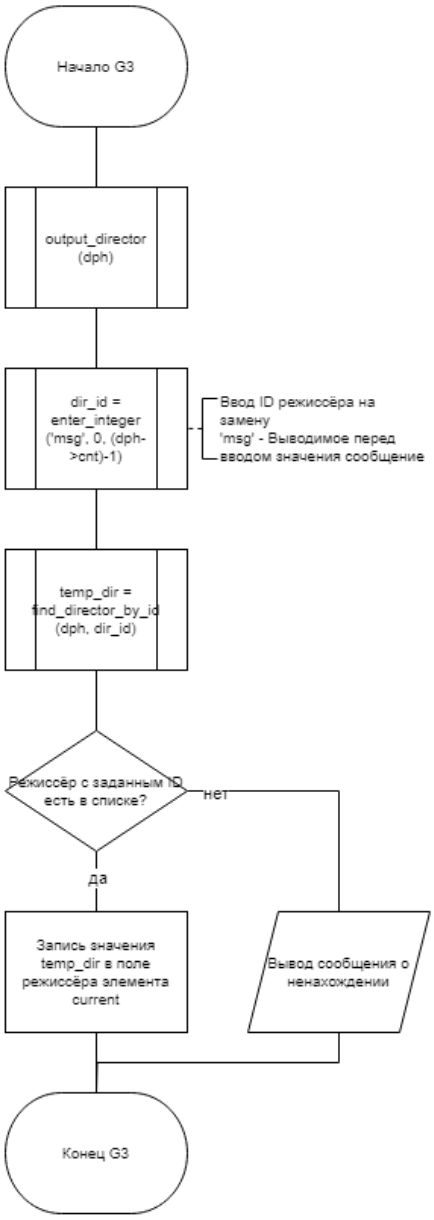

Начало F1
lstring = Значение элемента current в поле имени
Приведение строки lstring к нижнему регистру
Входит ли введённая строка в lstring?
нет
да
count++
count = 1?
да
нет
Вывод оглавления таблицы
Вывод данных элемента
current = следующий элемент списка
Конец F1

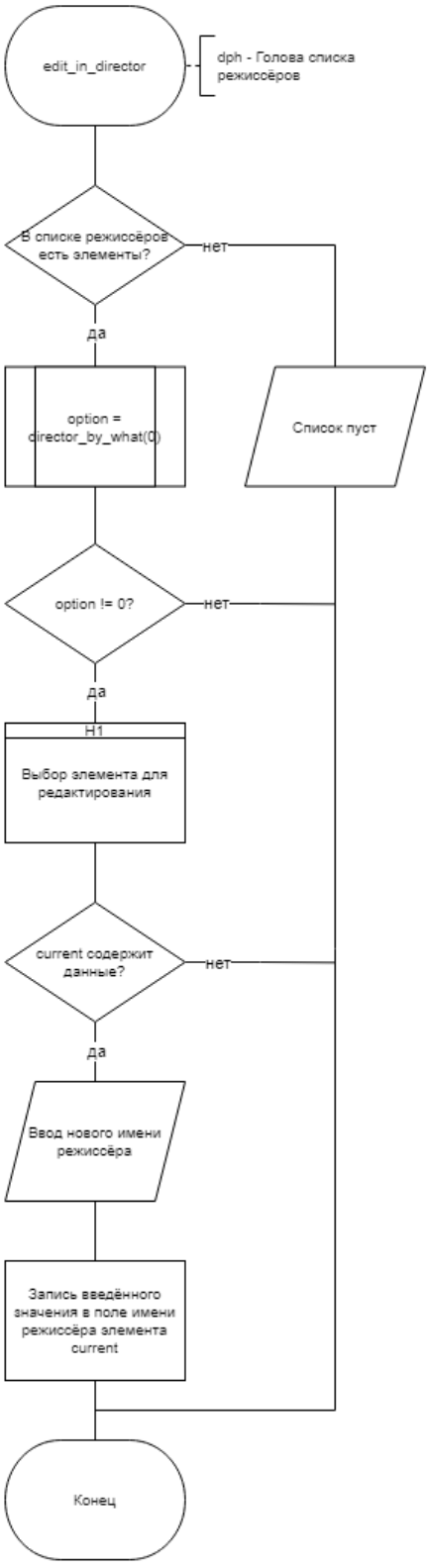

find_in_director()
dph - Голова списка режиссёров
В списке есть элементы?
нет
да
option = director_by_what(1)
option = 0?
нет
да
find_id_in_director (dph)
find_name_in_director (dph)
Список пуст
Конец

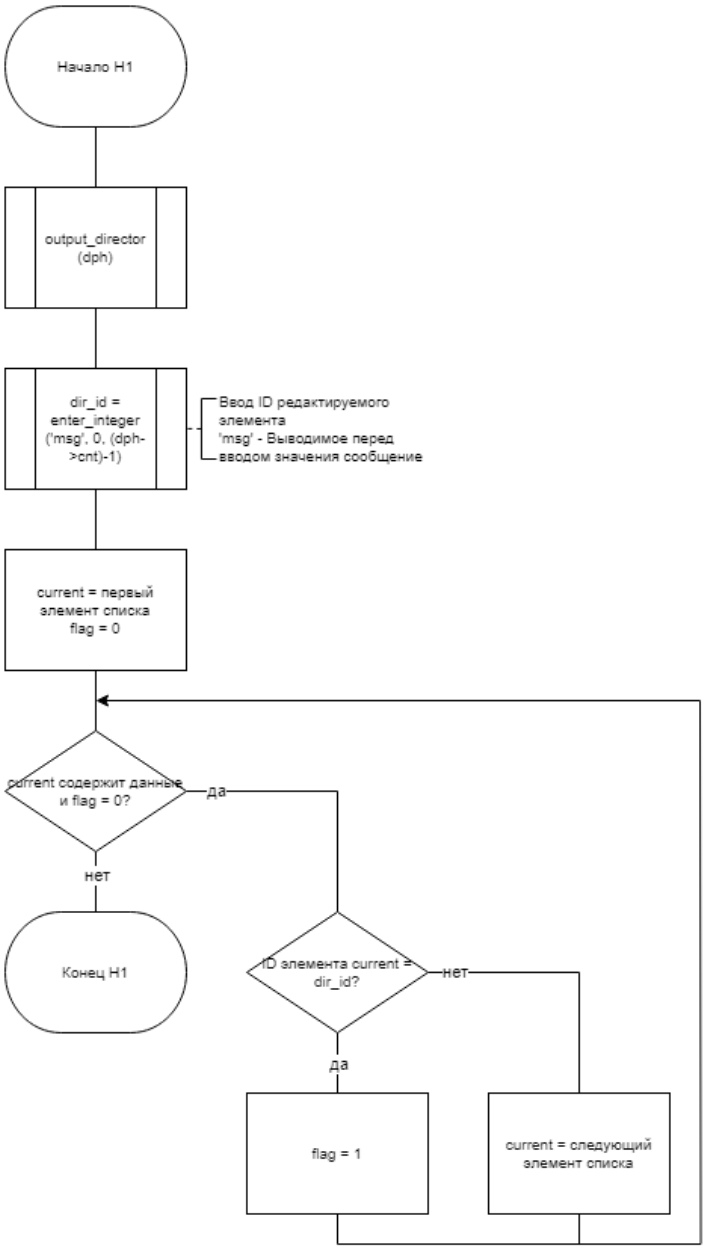

edit_in_movie()
mph - Голова списка фильмов
dph - Голова списка режиссёров
flag = 0
В списке фильмов есть элементы?
нет
да
option = movie_by_what(0)
Список фильмов пуст
option != 0?
нет
да
G1
Выбор элемента для редактирования
G2
Редактирование элемента
Конец

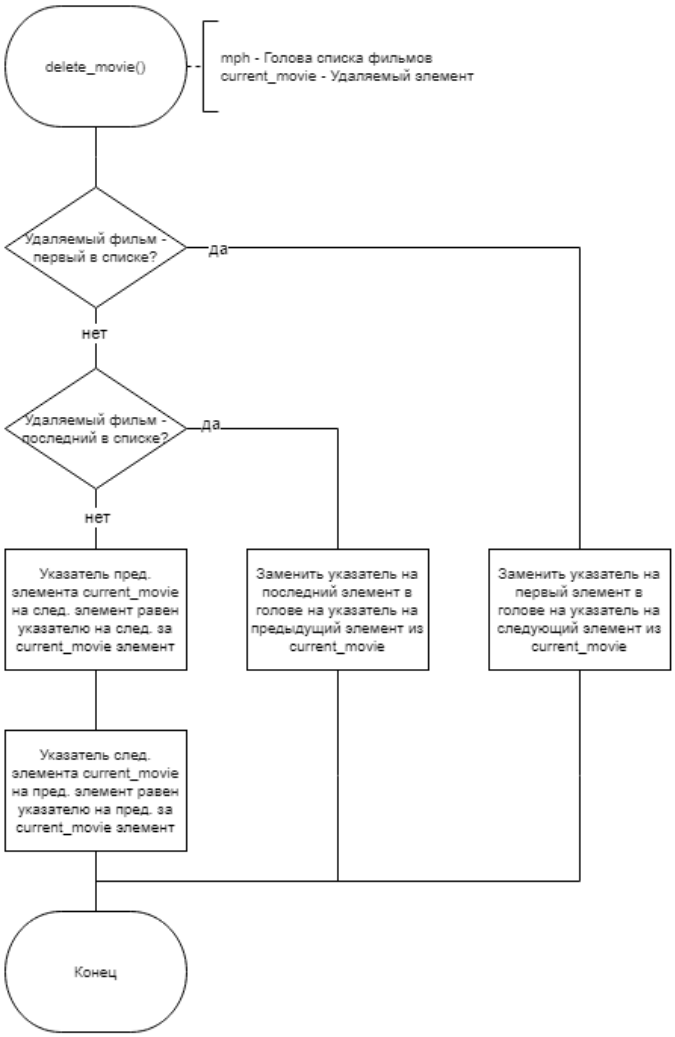

Начало G1
current = первый элемент списка
output_movie (mph)
mov_id = enter_integer ('msg', 0, (mph->cnt)-1)
Ввод ID редактируемого элемента
'msg' - Выводимое перед вводом значения сообщение
current содержит данные и flag = 0?
да
нет
Конец G1
ID элемента current = mov_id?
да
нет
current = следующий элемент списка
flag = 1
current = следующий элемент списка

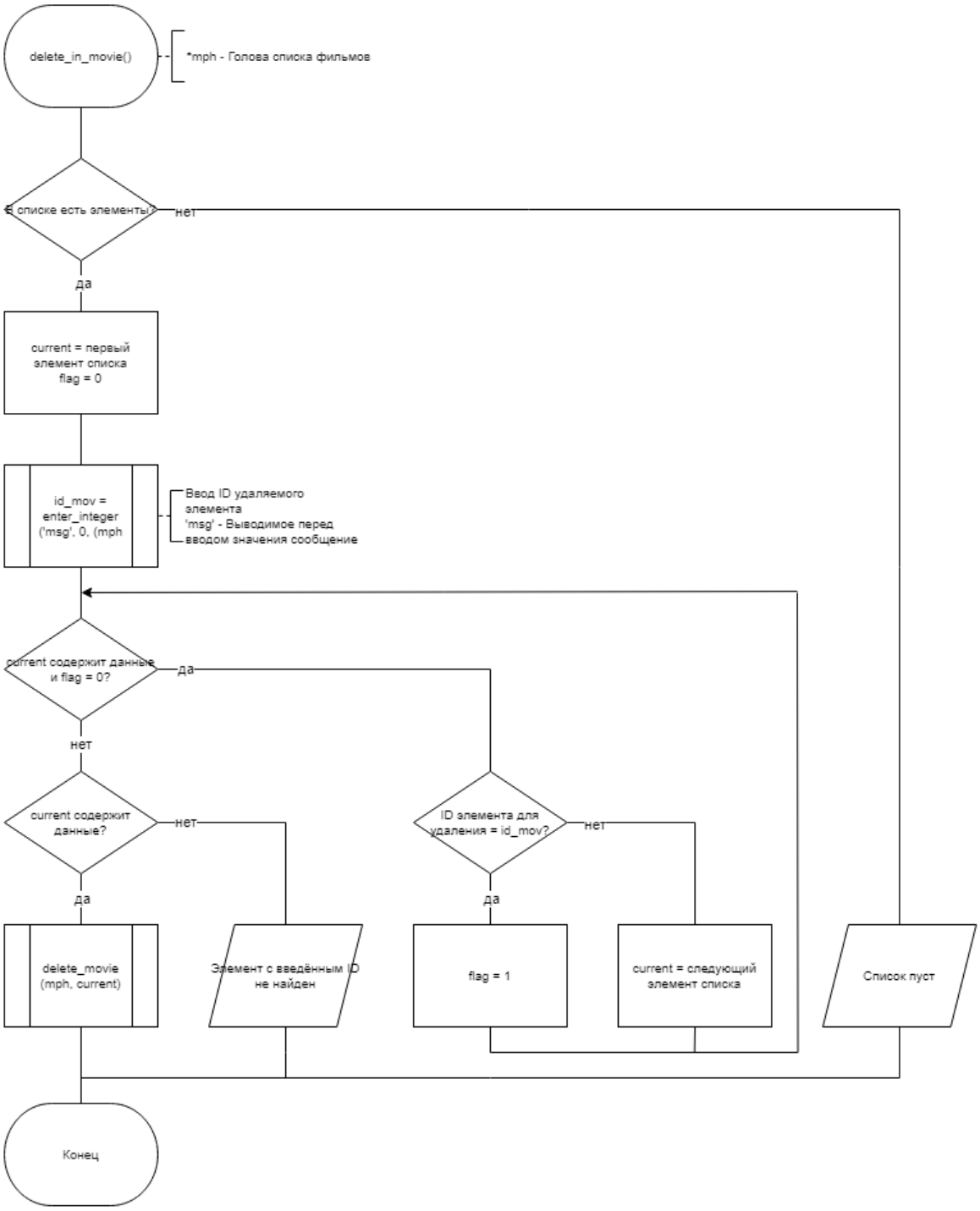

Начало G2
current содержит данные?
нет
да
option
'7'
'6'
'5'
'4'
'3'
'2'
'1'
G3
Замена режиссёра
duration = enter_integer ('msg', 1, 999)
plr = enter_float ('msg', 0, 10)
Вывод сообщения о ненахождении элемента с заданным ID
Ввод нового названия фильма
year = enter_integer ('msg', 1895, 2099)
kpr = enter_float ('msg', 0, 10)
enter_date ('msg', &date[0], &date[1], &date[2])
Значение элемента current соответствующего поля = введённое новое значение
Конец G2

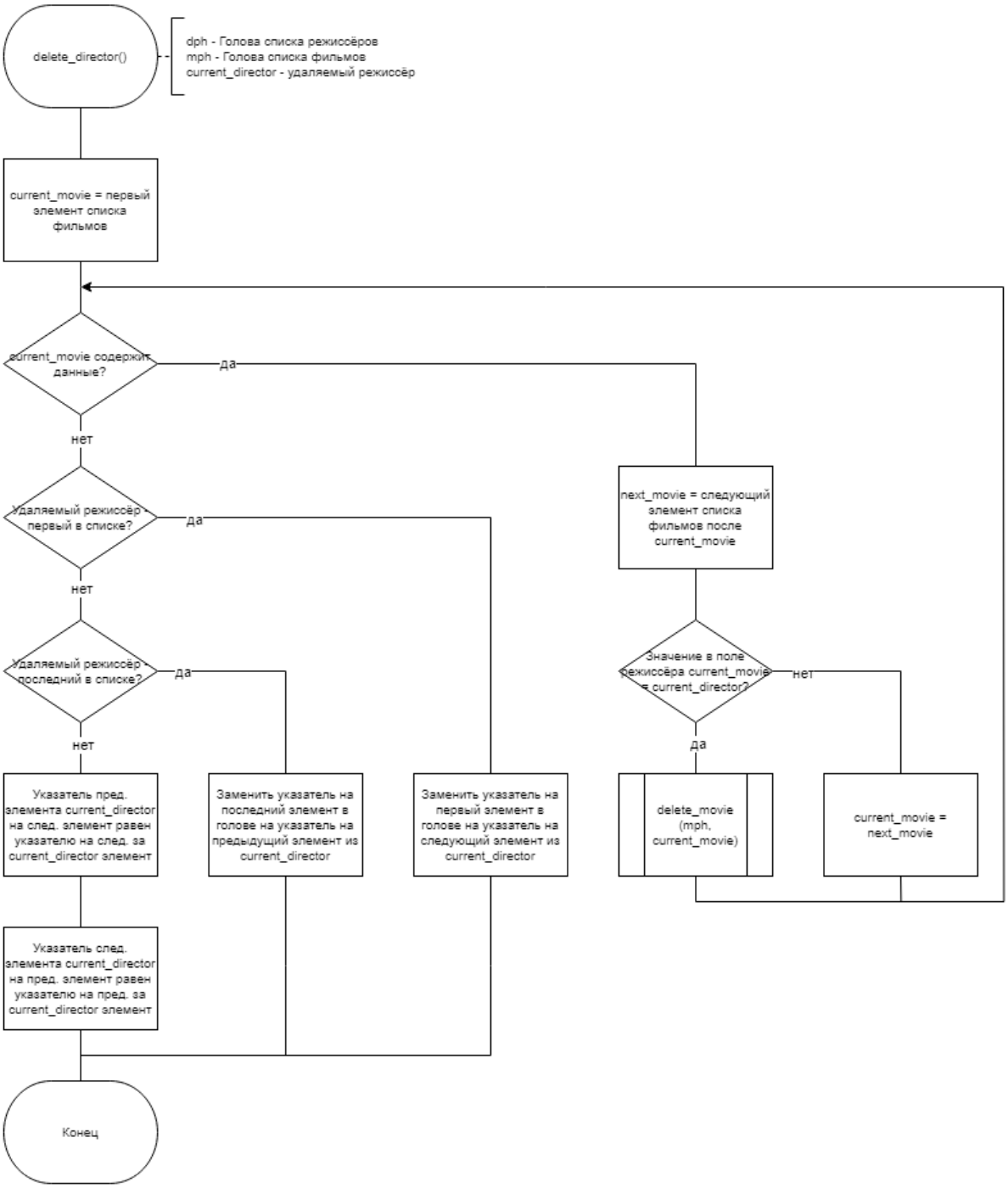

Начало G3
output_director (dph)
dir_id = enter_integer ('msg', 0, (dph->cnt)-1)
Ввод ID режисёра на замену
'msg' - Выводимое перед вводом значения сообщение
temp_dir = find_director_by_id (dph, dir_id)
Режиссёр с заданным ID есть в списке?
нет
да
Запись значения temp_dir в поле режиссёра элемента current
Вывод сообщения о ненахождении
Конец G3

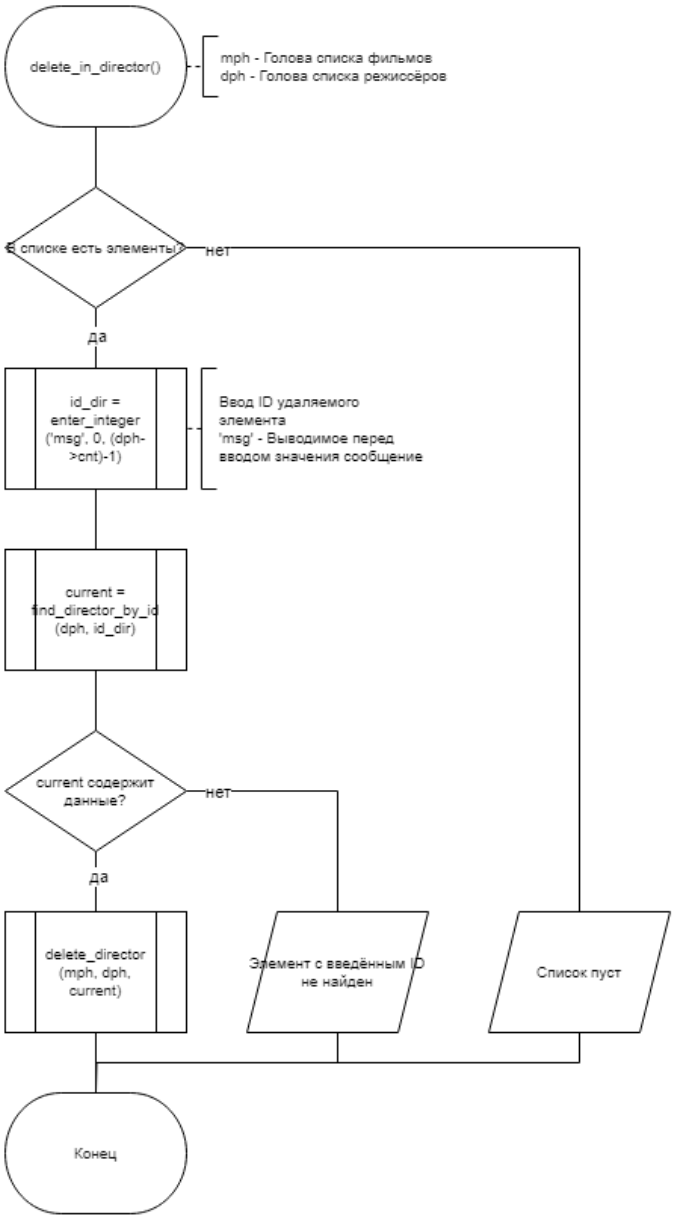

edit_in_director
dph - Голова списка режиссёров
В списке режиссёров есть элементы?
нет
да
option = director_by_what(0)
Список пуст
option != 0?
нет
да
H1
Выбор элемента для редактирования
current содержит данные?
нет
да
Ввод нового имени режиссёра
Запись введённого значения в поле имени режиссёра элемента current
Конец

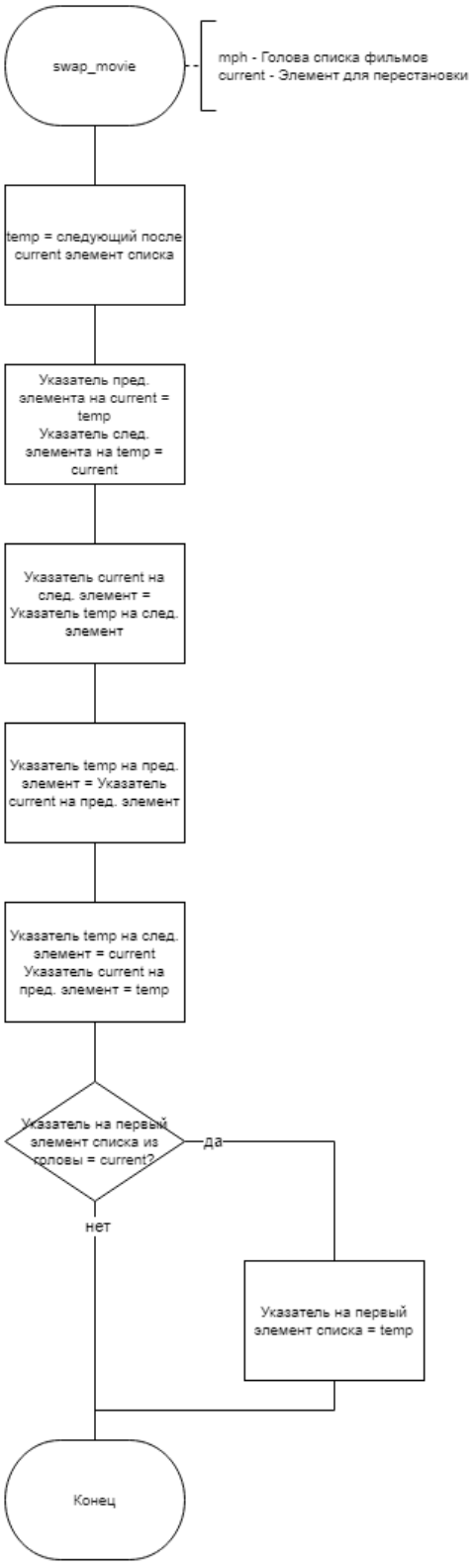

Начало H1
output_director (dph)
dir_id = enter_integer ('msg', 0, (dph->cnt)-1)
Ввод ID редактируемого элемента
'msg' - Выводимое перед вводом значения сообщение
current = первый элемент списка
flag = 0
current содержит данные и flag = 0?
да
нет
Конец H1
ID элемента current = dir_id?
нет
да
flag = 1
current = следующий элемент списка


delete_movie()
mph - Голова списка фильмов
current_movie - Удаляемый элемент
Удаляемый фильм - первый в списке?
да
нет
Удаляемый фильм - последний в списке?
да
нет
Указатель пред. элемента current_movie на след. элемент равен указателю на след. за current_movie элемент
Заменить указатель на последний элемент в голове на указатель на предыдущий элемент из current_movie
Заменить указатель на первый элемент в голове на указатель на следующий элемент из current_movie
Указатель след. элемента current_movie на пред. элемент равен указателю на пред. за current_movie элемент
Конец

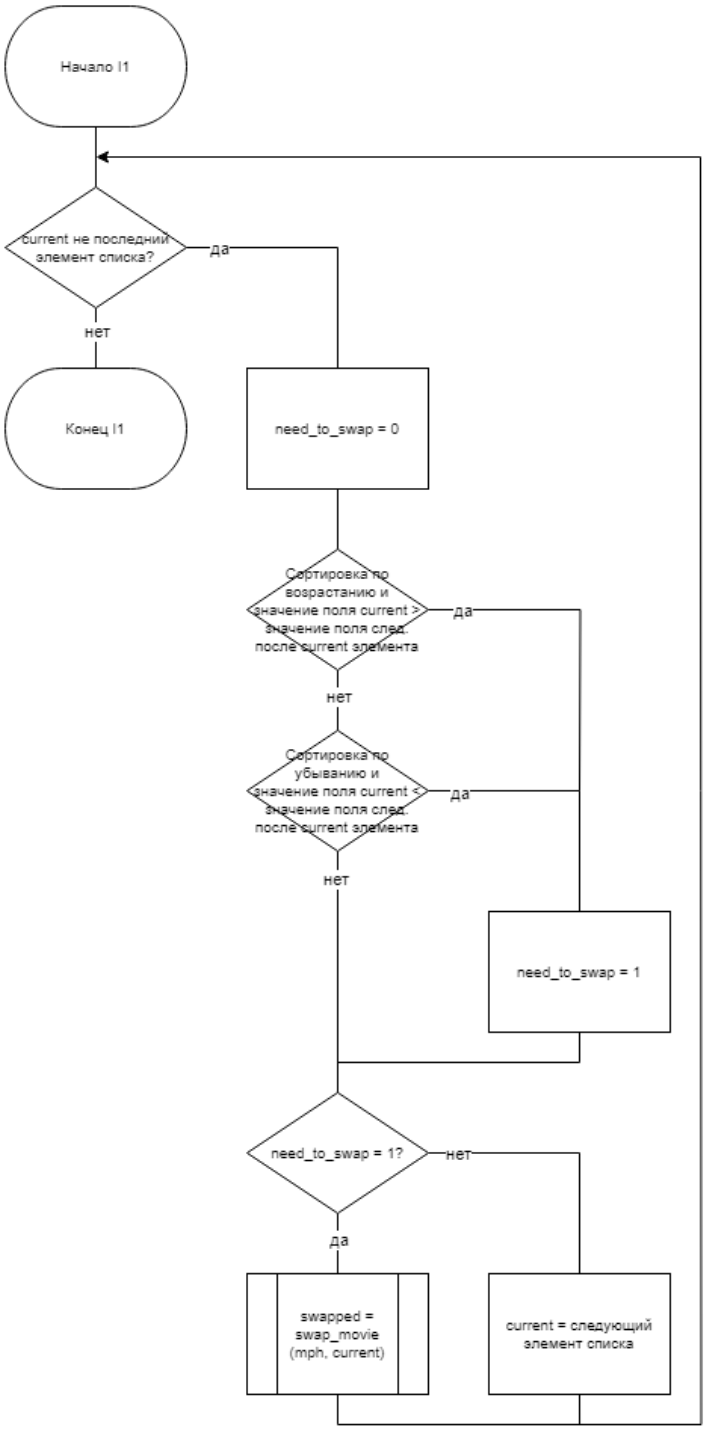

delete_in_movie()
*mph - Голова списка фильмов
В списке есть элементы?
нет
да
current = первый элемент списка
flag = 0
id_mov = enter_integer ('msg', 0, (mph
Ввод ID удаляемого элемента
'msg' - Выводимое перед вводом значения сообщение
current содержит данные и flag = 0?
да
нет
current содержит данные?
нет
да
ID элемента для удаления = id_mov?
нет
да
delete_movie (mph, current)
Элемент с введённым ID не найден
flag = 1
current = следующий элемент списка
Список пуст
Конец


delete_director()
dph - Голова списка режиссёров
mph - Голова списка фильмов
current_director - удаляемый режиссёр
current_movie = первый элемент списка фильмов
current_movie содержит данные?
да
нет
Удаляемый режиссёр первый в списке?
да
нет
Удаляемый режиссёр последний в списке?
да
нет
Указатель пред. элемента current_director на след. элемент равен указателю на след. за current_director элемент
Заменить указатель на последний элемент в голове на указатель на предыдущий элемент из current_director
Заменить указатель на первый элемент в голове на указатель на следующий элемент из current_director
Указатель след. элемента current_director на пред. элемент равен указателю на пред. за current_director элемент
Конец
next_movie = следующий элемент списка фильмов после current_movie
Значение в поле режиссёра current_movie = current_director?
нет
да
delete_movie (mph, current_movie)
current_movie = next_movie

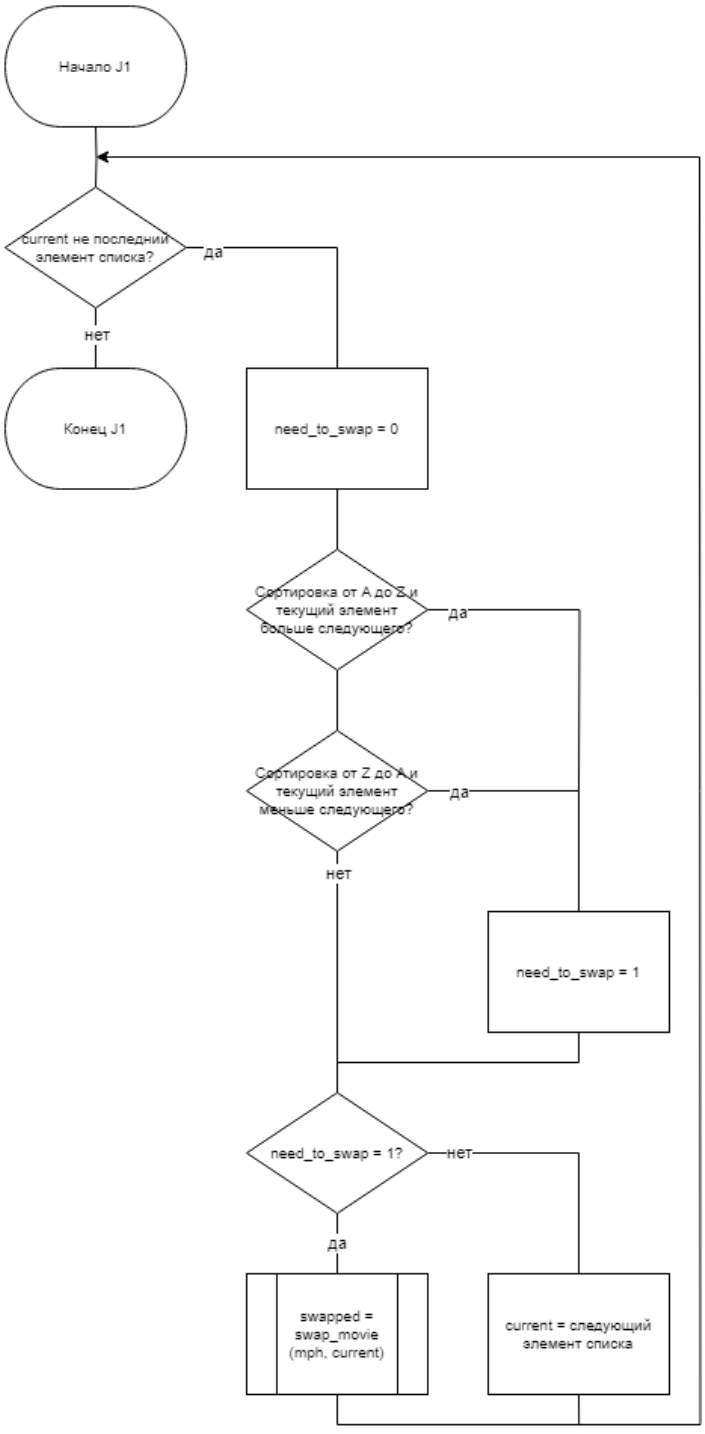

delete_in_director()
mph - Голова списка фильмов
dph - Голова списка режиссёров
В списке есть элементы?
нет
да
id_dir = enter_integer ('msg', 0, (dph->cnt)-1)
Ввод ID удаляемого элемента
'msg' - Выводимое перед вводом значения сообщение
current = find_director_by_id (dph, id_dir)
current содержит данные?
нет
да
delete_director (mph, dph, current)
Элемент с введённым ID не найден
Список пуст
Конец

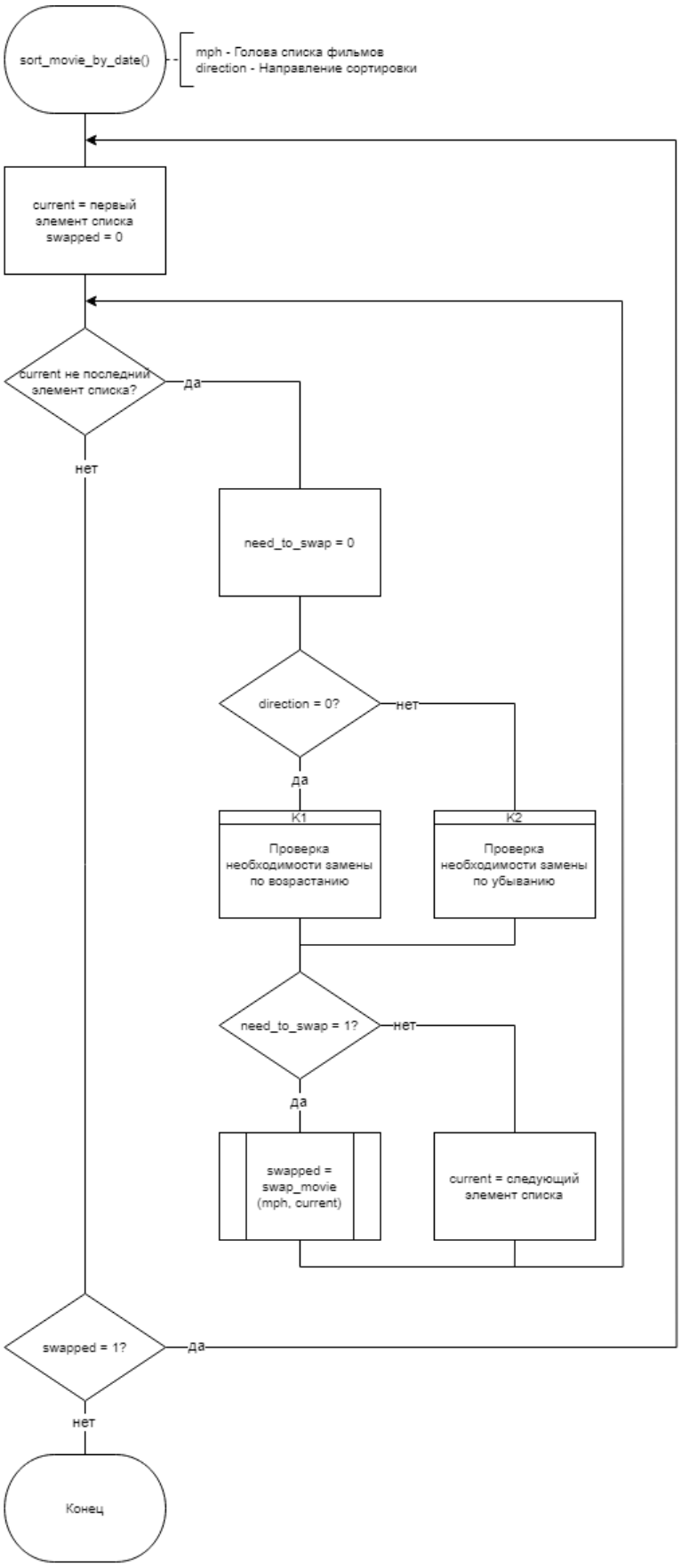

swap_movie
mph - Голова списка фильмов
current - Элемент для перестановки
temp = следующий после current элемент списка
Указатель пред. элемента на current = temp
Указатель след. элемента на temp = current
Указатель current на след. элемент = Указатель temp на след. элемент
Указатель temp на пред. элемент = Указатель current на пред. элемент
Указатель temp на след. элемент = current
Указатель current на пред. элемент = temp
Указатель на первый элемент списка из головы = current?
да
нет
Указатель на первый элемент списка = temp
Конец

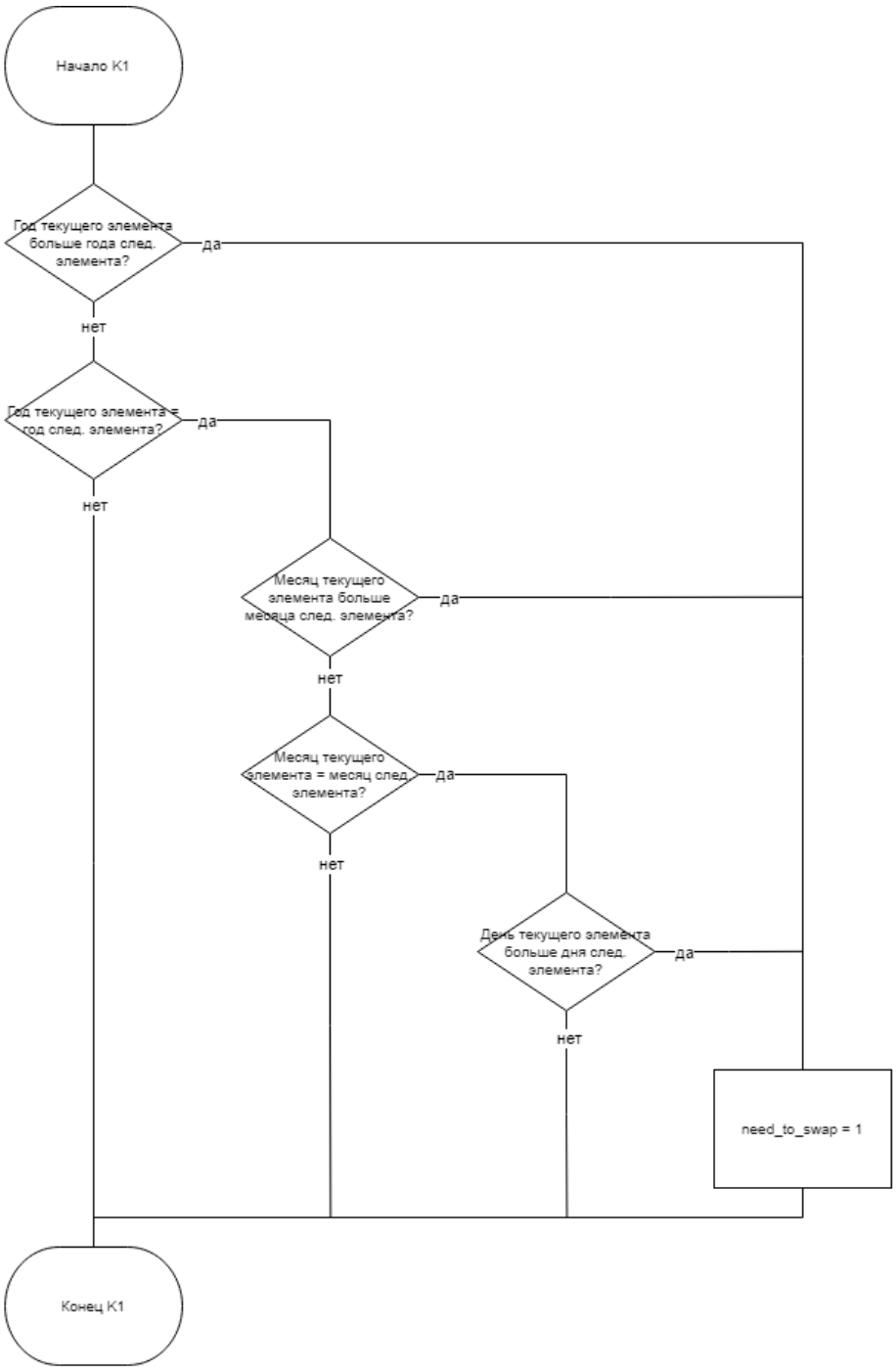

sort_movie_by_number()
mph - Голова списка фильмов
i - Номер поля для сортировки
direction - Направление сортировки
current = первый элемент списка
swapped = 0
i1
Перестановка элементов при удовлетворении условий
swapped = 1?
да
нет
Конец

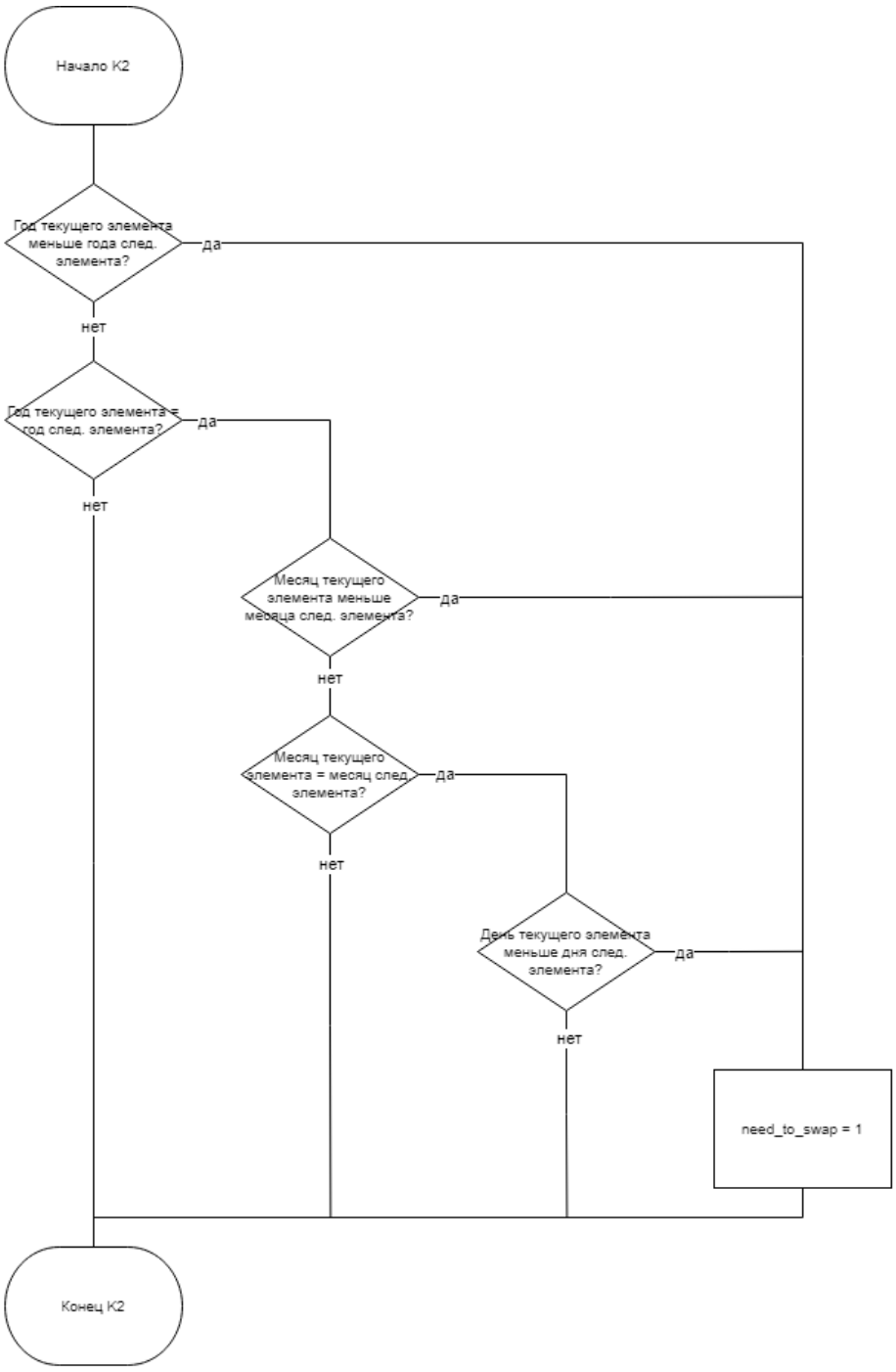

Начало l1
current не последний элемент списка?
да
нет
Конец l1
need_to_swap = 0
Сортировка по возрастанию и значение поля current > значение поля след. после current элемента
да
нет
Сортировка по убыванию и значение поля current < значение поля след. после current элемента
да
нет
need_to_swap = 1
need_to_swap = 1?
нет
да
swapped = swap_movie (mph, current)
current = следующий элемент списка

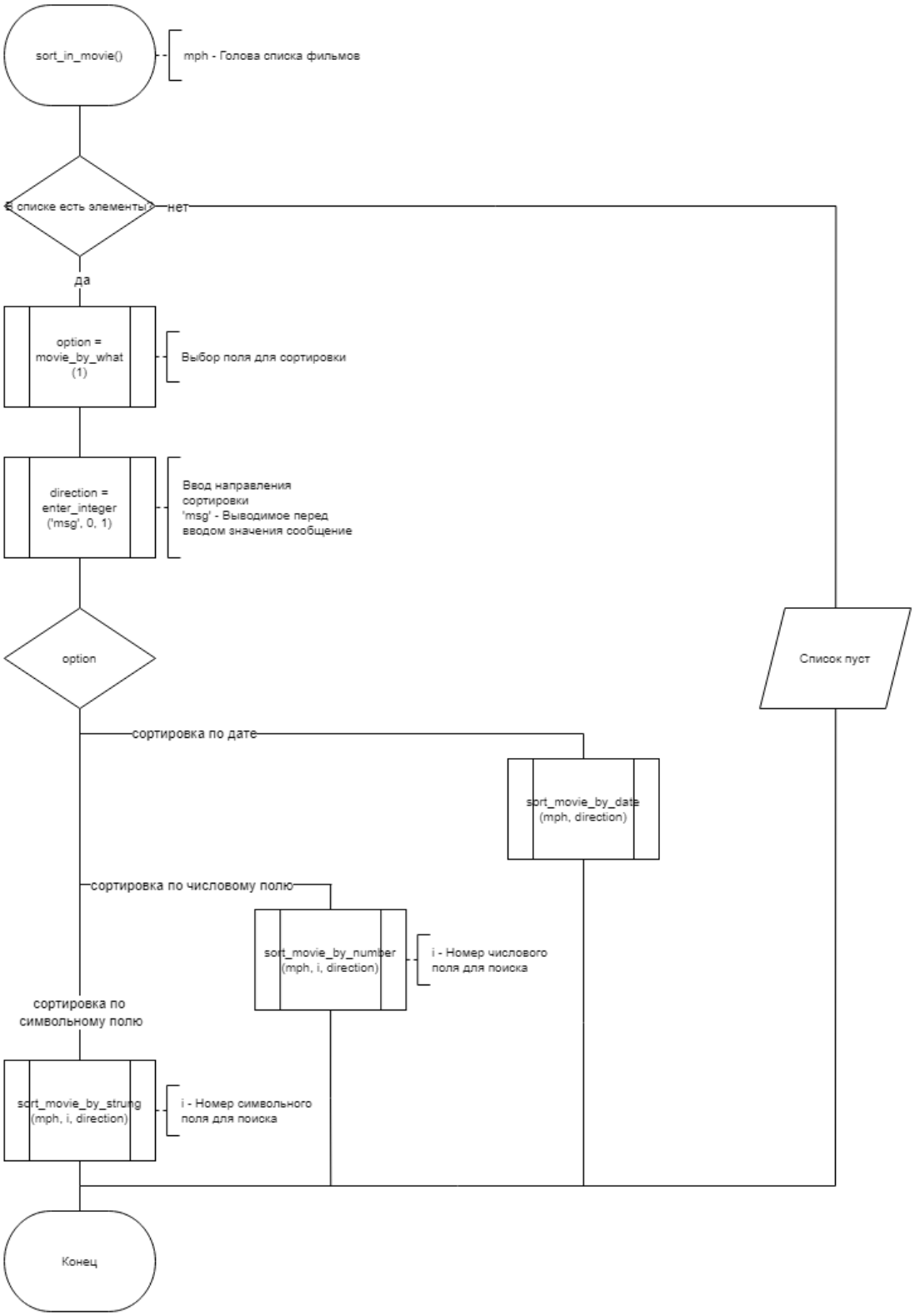

sort_movie_by_string()
mph - Голова списка фильмов
i - Номер поля для сортировки
direction - Направление сортировки
current = первый элемент списка
swapped = 0
J1
Перестановка элементов при удовлетворении условий
swapped = 1?
Да
нет
Конец

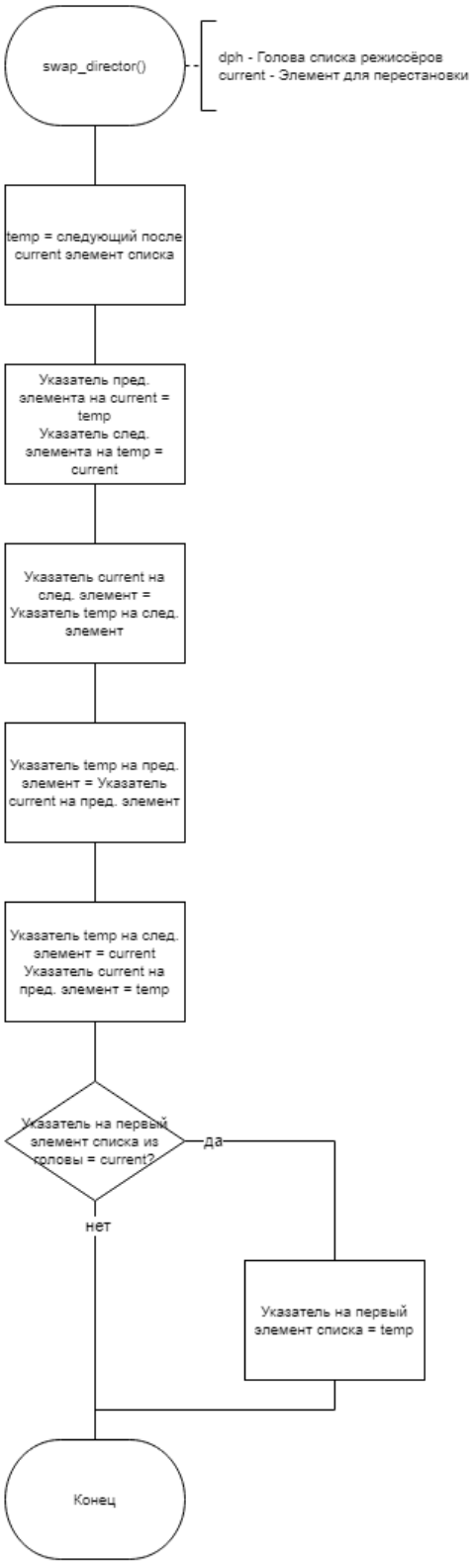

Начало J1
current не последний элемент списка?
да
нет
Конец J1
need_to_swap = 0
Сортировка от A до Z и текущий элемент больше следующего?
да
Сортировка от Z до A и текущий элемент меньше следующего?
да
нет
need_to_swap = 1
need_to_swap = 1?
нет
да
swapped = swap_movie (mph, current)
current = следующий элемент списка

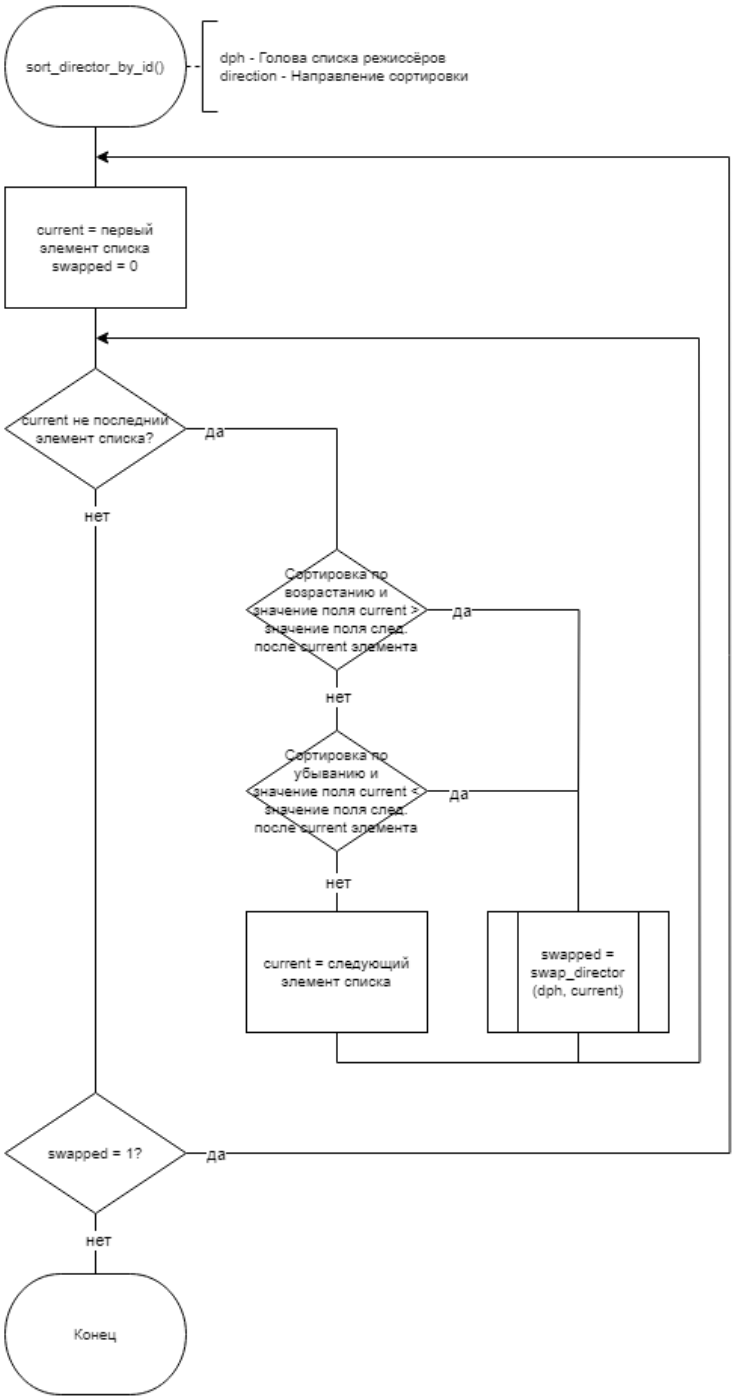

sort_movie_by_date()
mph - Голова списка фильмов
direction - Направление сортировки
current = первый элемент списка
swapped = 0
current не последний элемент списка?
да
нет
need_to_swap = 0
direction = 0?
нет
да
K1
Проверка необходимости замены по возрастанию
K2
Проверка необходимости замены по убыванию
need_to_swap = 1?
нет
да
swapped = swap_movie (mph, current)
current = следующий элемент списка
swapped = 1?
да
нет
Конец

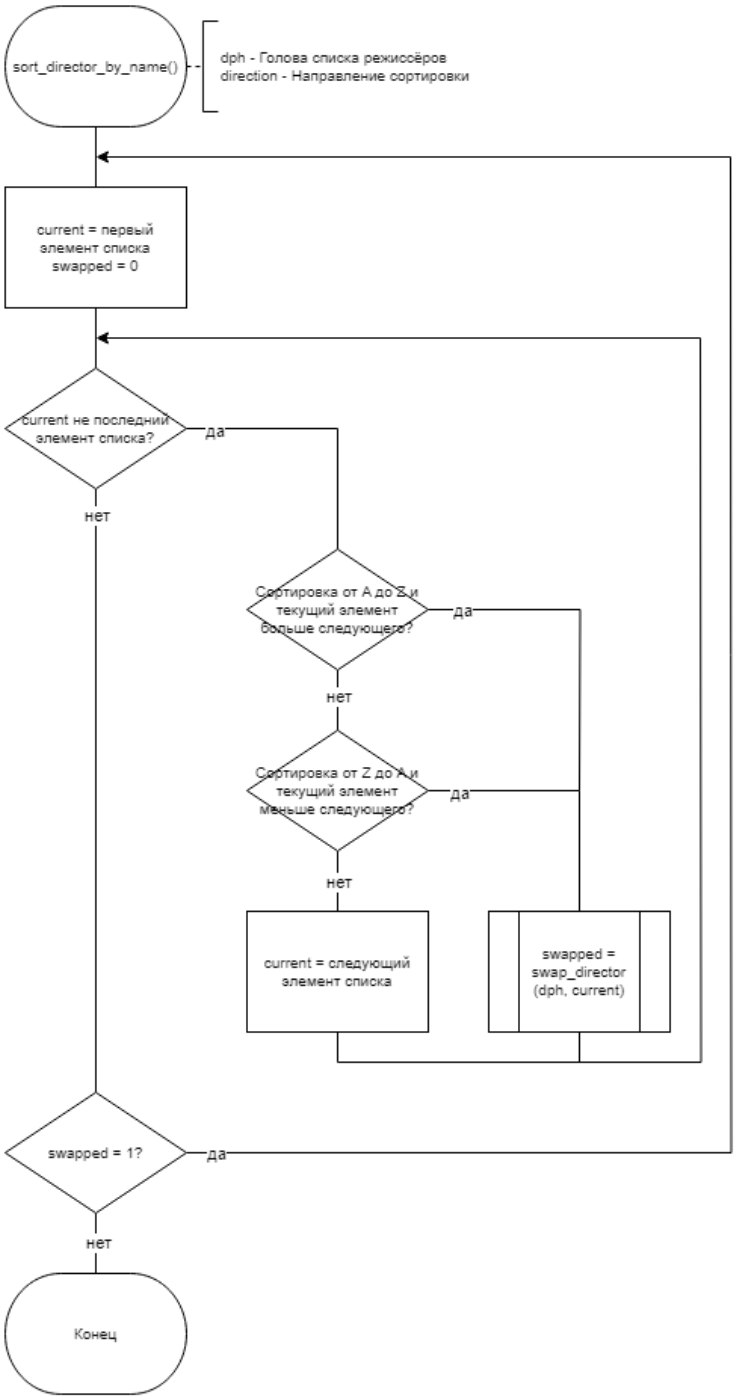

Начало K1
Год текущего элемента больше года след. элемента?
да
нет
Год текущего элемента = год след. элемента?
да
нет
Месяц текущего элемента больше месяца след. элемента?
да
нет
Месяц текущего элемента = месяц след. элемента?
да
нет
День текущего элемента больше дня след. элемента?
да
нет
need_to_swap = 1
Конец K1

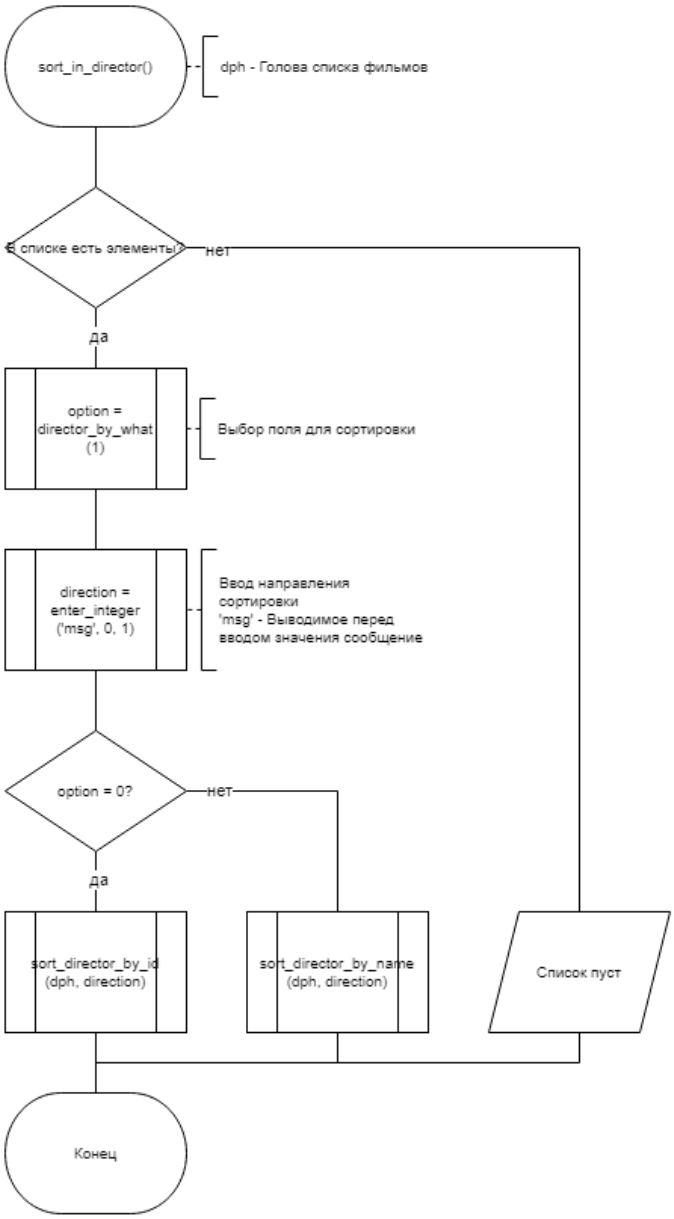

Начало K2
Год текущего элемента меньше года след. элемента?
да
нет
Год текущего элемента = год след. элемента?
да
нет
Месяц текущего элемента меньше месяца след. элемента?
да
нет
Месяц текущего элемента = месяц след. элемента?
да
нет
День текущего элемента меньше дня след. элемента?
да
нет
need_to_swap = 1
Конец K2


sort_in_movie()
mph - Голова списка фильмов
В списке есть элементы?
нет
да
option = movie_by_what (1)
Выбор поля для сортировки
direction = enter_integer ('msg', 0, 1)
Ввод направления сортировки
'msg' - Выводимое перед вводом значения сообщение
option
Список пуст
сортировка по дате
sort_movie_by_date (mph, direction)
сортировка по числовому полю
sort_movie_by_number (mph, i, direction)
i - Номер числового поля для поиска
сортировка по символьному полю
sort_movie_by_strung (mph, i, direction)
i - Номер символьного поля для поиска
Конец


swap_director()
dph - Голова списка режисёров
current - Элемент для перестановки
temp = следующий после current элемент списка
Указатель пред. элемента на current = temp
Указатель след. элемента на temp = current
Указатель current на след. элемент = Указатель temp на след. элемент
Указатель temp на пред. элемент = Указатель current на пред. элемент
Указатель temp на след. элемент = current
Указатель current на пред. элемент = temp
Указатель на первый элемент списка из головы = current?
да
нет
Указатель на первый элемент списка = temp
Конец


sort_director_by_id()
dph - Голова списка режиссёров
direction - Направление сортировки
current = первый элемент списка
swapped = 0
current не последний элемент списка?
да
нет
Сортировка по возрастанию и значение поля current > значение поля след. после current элемента
да
нет
Сортировка по убыванию и значение поля current < значение поля след. после current элемента
да
нет
current = следующий элемент списка
swapped = swap_director (dph, current)
swapped = 1?
да
нет
Конец


sort_director_by_name()
dph - Голова списка режиссёров
direction - Направление сортировки
current = первый элемент списка
swapped = 0
current не последний элемент списка?
да
нет
Сортировка от A до Z и текущий элемент больше следующего?
да
нет
Сортировка от Z до A и текущий элемент меньше следующего?
да
нет
current = следующий элемент списка
swapped = swap_director (dph, current)
swapped = 1?
да
нет
Конец



Контрольные примеры:

Пример 1 (Ознакомление с программой, добавление и вывод данных):

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 7

Your selection is REFERENCE

This program is card-file. Its functions are presented in the menu.
```

```
Subject area of the program - movie
Designations:
Name - Name of movie
Director - Director of the movie
Year - Year of movie release
Duration - Duration of the movie in minutes
KPR - Rating of the movie on KinoPoisk
PLR - Personal rating of the movie
Date - Date of viewing the movie

(...)

Enter the option: 2

Your selection is ADD THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

ID of new movie set automatically on 12
Enter the name of movie: Prisoners
Enter the director of movie: Deni Villeneuve
Enter the year of movie: qwe
Entered value is not correct. Please try again!
Enter the year of movie: 2013
Enter the duration of movie: -125
Entered value is not correct. Please try again!
Enter the duration of movie: 153
Enter the KPR of movie: 8.2
Enter the PLR of movie: 15
Entered value is not correct. Please try again!
Enter the PLR of movie: 9.5
Enter the date of viewing (DD MM YYYY): 08 12 1
Entered date is not correct. Please try again!
Enter the date of viewing (DD MM YYYY): 08 12 2023

Movie added successfully

(...)

Enter the option: 2

Your selection is ADD THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 1

ID of new director set automatically on 9
Enter the name of director: James Gunn
```

```
Director added successfully

(...)

Enter the option: 1

Your selection is SHOW THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

| ID |                      Name |                  Director | Year | Dur |   KPR |   PLR |  Watchdate |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+
|  0 |              Forrest Gump |           Robert Zemeckis | 1994 | 142 |   8.9 |   8.4 | 10.10.2015 |
|  1 |                 Inception |          Cristopher Nolan | 2010 | 148 |   8.7 |  10.0 | 10.05.2017 |
|  2 |                Fight Club |             David Fincher | 1999 | 139 |   8.7 |  10.0 | 12.11.2019 |
|  3 |                      1917 |                Sam Mendes | 2019 | 119 |   8.2 |   7.0 | 08.02.2020 |
|  4 |              Interstellar |          Cristopher Nolan | 2014 | 169 |   8.6 |   9.8 | 23.03.2024 |
|  5 |            Shutter Island |           Martin Scorsese | 2010 | 138 |   8.3 |   9.5 | 09.01.2024 |
|  6 |         The Boy and Heron |            Hayao Miyazaki | 2023 | 124 |   7.8 |   6.0 | 17.12.2023 |
|  7 |             Spirited Away |            Hayao Miyazaki | 2001 | 125 |   8.5 |   8.1 | 10.12.2021 |
|  9 |   The Wolf of Wall Street |           Martin Scorsese | 2013 | 180 |   8.1 |   8.7 | 30.04.2024 |
| 10 |  The Shawshank Redemption |            Frank Darabont | 1994 | 142 |   9.2 |   9.3 | 18.06.2023 |
| 11 |                  The Dune |           Deni Villeneuve | 2021 | 155 |   8.3 |   7.5 | 08.03.2024 |
| 12 |                  Prisoners |           Deni Villeneuve | 2013 | 153 |   8.2 |   9.5 | 08.12.2023 |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+

(...)

Enter the option: 1

Your selection is SHOW THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 1

| ID |                  Director |
+----+---------------------------+
|  0 |           Robert Zemeckis |
|  1 |          Cristopher Nolan |
|  2 |             David Fincher |
|  3 |                Sam Mendes |
|  4 |           Martin Scorsese |
|  5 |            Hayao Miyazaki |
|  7 |            Frank Darabont |
|  8 |           Deni Villeneuve |
|  9 |                James Gunn |
+----+---------------------------+
```

Пример 2 (Поиск данных и их редактирование):

```
(...)

Enter the option: 3

Your selection is FIND THE DATA BY PARAMETERS

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 1

Choose the parameter
0 - by ID
1 - by Name
Enter the option: 1

Enter the Name (or Substring): de

| ID |                  Director |
+----+---------------------------+
|  3 |                Sam Mendes |
|  8 |           Deni Villeneuve |
+----+---------------------------+

(...)

Enter the option: 4

Your selection is EDIT THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

Choose the parameter
0 - to EXIT
1 - by Name
2 - by Director
3 - by Year
4 - by Duration
5 - by KPR
6 - by PLR
7 - by Date
Enter the option: 5

| ID |                      Name |                  Director | Year | Dur |   KPR |   PLR |  Watchdate |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+
|  0 |              Forrest Gump |           Robert Zemeckis | 1994 | 142 |   8.9 |   8.4 | 10.10.2015 |
|  1 |                 Inception |          Cristopher Nolan | 2010 | 148 |   8.7 |  10.0 | 10.05.2017 |
|  2 |                Fight Club |             David Fincher | 1999 | 139 |   8.7 |  10.0 | 12.11.2019 |
```

```
|  3 |                      1917 |                Sam Mendes | 2019 | 119 |   8.2 |   7.0 | 08.02.2020 |
|  4 |              Interstellar |          Cristopher Nolan | 2014 | 169 |   8.6 |   9.8 | 23.03.2024 |
|  5 |            Shutter Island |           Martin Scorsese | 2010 | 138 |   8.3 |   9.5 | 09.01.2024 |
|  6 |         The Boy and Heron |            Hayao Miyazaki | 2023 | 124 |   7.8 |   6.0 | 17.12.2023 |
|  7 |             Spirited Away |            Hayao Miyazaki | 2001 | 125 |   8.5 |   8.1 | 10.12.2021 |
|  9 |   The Wolf of Wall Street |           Martin Scorsese | 2013 | 180 |   8.1 |   8.7 | 30.04.2024 |
| 10 |  The Shawshank Redemption |            Frank Darabont | 1994 | 142 |   9.2 |   9.3 | 18.06.2023 |
| 11 |                  The Dune |           Deni Villeneuve | 2021 | 155 |   8.3 |   7.5 | 08.03.2024 |
| 12 |                 Prisoners |           Deni Villeneuve | 2013 | 153 |   8.2 |   9.5 | 08.12.2023 |
+----+--------------------------+---------------------------+------+-----+-------+-------+------------+
Enter the ID of the record you want to edit: 0

Enter the new KPR of Movie: 15
Entered value is not correct. Please try again!
Enter the new KPR of Movie: 9.2

The data updated successfully

(...)

Enter the option: 3

Your selection is FIND THE DATA BY PARAMETERS

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

Choose the parameter
0 - by ID
1 - by Name
2 - by Director
3 - by Year
4 - by Duration
5 - by KPR
6 - by PLR
7 - by Date
Enter the option: 0

Enter the ID: 0

| ID |                      Name |                  Director | Year | Dur |   KPR |   PLR |  Watchdate |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+
|  0 |              Forrest Gump |           Robert Zemeckis | 1994 | 142 |   9.2 |   8.4 | 10.10.2015 |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+
```

Пример 3 (Сортировка и удаление данных):

```
(...)

Enter the option: 6
```

```
Your selection is SORT THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 1

Choose the parameter
0 - by ID
1 - by Name
Enter the option: 1

Enter the direction of sort (0 - Forward / 1 - Backward): 1

The director list was sorted

(...)

Enter the option: 1

Your selection is SHOW THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 1

| ID |                  Director |
+----+---------------------------+
|  3 |                Sam Mendes |
|  0 |           Robert Zemeckis |
|  4 |           Martin Scorsese |
|  9 |                James Gunn |
|  5 |            Hayao Miyazaki |
|  7 |            Frank Darabont |
|  8 |           Deni Villeneuve |
|  2 |             David Fincher |
|  1 |          Cristopher Nolan |
+----+---------------------------+

(...)

Enter the option: 6

Your selection is SORT THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

Choose the parameter
0 - by ID
```

```
1 - by Name
2 - by Director
3 - by Year
4 - by Duration
5 - by KPR
6 - by PLR
7 - by Date
Enter the option: 2

Enter the direction of sort (0 - Forward / 1 - Backward): 0

The movie list was sorted

(...)

Enter the option: 1

Your selection is SHOW THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0
```

ID	Name	Director	Year	Dur	KPR	PLR	Watchdate
1	Inception	Cristopher Nolan	2010	148	8.7	10.0	10.05.2017
4	Interstellar	Cristopher Nolan	2014	169	8.6	9.8	23.03.2024
2	Fight Club	David Fincher	1999	139	8.7	10.0	12.11.2019
11	The Dune	Deni Villeneuve	2021	155	8.3	7.5	08.03.2024
12	Prisoners	Deni Villeneuve	2013	153	8.2	9.5	08.12.2023
10	The Shawshank Redemption	Frank Darabont	1994	142	9.2	9.3	18.06.2023
6	The Boy and Heron	Hayao Miyazaki	2023	124	7.8	6.0	17.12.2023
7	Spirited Away	Hayao Miyazaki	2001	125	8.5	8.1	10.12.2021
5	Shutter Island	Martin Scorsese	2010	138	8.3	9.5	09.01.2024
9	The Wolf of Wall Street	Martin Scorsese	2013	180	8.1	8.7	30.04.2024
0	Forrest Gump	Robert Zemeckis	1994	142	9.2	8.4	10.10.2015
3	1917	Sam Mendes	2019	119	8.2	7.0	08.02.2020

```
(...)

Enter the option: 5

Your selection is DELETE THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

Enter the ID of the movie you want to delete: 15
Entered value is not correct. Please try again!
```

```
Enter the ID of the movie you want to delete: 12

Movie with entered ID was deleted

(...)

Enter the option: 1

Your selection is SHOW THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

| ID |                      Name |                  Director | Year | Dur |   KPR |   PLR |  Watchdate |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+
|  1 |                 Inception |          Cristopher Nolan | 2010 | 148 |   8.7 |  10.0 | 10.05.2017 |
|  4 |              Interstellar |          Cristopher Nolan | 2014 | 169 |   8.6 |   9.8 | 23.03.2024 |
|  2 |                Fight Club |             David Fincher | 1999 | 139 |   8.7 |  10.0 | 12.11.2019 |
| 11 |                  The Dune |           Deni Villeneuve | 2021 | 155 |   8.3 |   7.5 | 08.03.2024 |
| 10 |  The Shawshank Redemption |            Frank Darabont | 1994 | 142 |   9.2 |   9.3 | 18.06.2023 |
|  6 |         The Boy and Heron |            Hayao Miyazaki | 2023 | 124 |   7.8 |   6.0 | 17.12.2023 |
|  7 |             Spirited Away |            Hayao Miyazaki | 2001 | 125 |   8.5 |   8.1 | 10.12.2021 |
|  5 |            Shutter Island |           Martin Scorsese | 2010 | 138 |   8.3 |   9.5 | 09.01.2024 |
|  9 |   The Wolf of Wall Street |           Martin Scorsese | 2013 | 180 |   8.1 |   8.7 | 30.04.2024 |
|  0 |              Forrest Gump |           Robert Zemeckis | 1994 | 142 |   9.2 |   8.4 | 10.10.2015 |
|  3 |                      1917 |                Sam Mendes | 2019 | 119 |   8.2 |   7.0 | 08.02.2020 |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+
```

Текст программы

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#define MAXLEN 128
#define SEP ';'
#define movie_db "struct-data.txt"
#define director_db "struct-data2.txt"

/* Structure ------------------------------------------------------------------------------------------------------ */

struct movie;                                                   /* Define structure of Movie */

struct movie_head;                                              /* Define structure of Movie List Head*/

struct director;                                                /* Define structure of Director */

struct director_head;                                           /* Define structure of Director List Head */

typedef struct movie MOV;                                       /* Define Structured Type of Movie */

typedef struct movie_head MHD;                                  /* Define Structured Type of Movie List Head*/

typedef struct director DIR;                                    /* Define Structured Type of Director */

typedef struct director_head DHD;                               /* Define Structured Type of Director List Head*/

/* Input ---------------------------------------------------------------------------------------------------------- */

void new_gets(char s[], int lim);                               /*Read the string and write it in s[]*/
```

```
int enter_integer(char *message, int a, int b);                     /* Enter integer number in [a,b] and return it */

float enter_float(char *message, float a, float b);                 /* Enter float number in [a,b] and return it */

void enter_date(char *message, int *day, int *month, int *year);    /* Enter date */

int menu();                                                         /* Output main menu and enter option */

int detail();                                                       /* Action will be done in movie or director card-
               file */

int movie_by_what(int i);                                           /* Output name of movie fields (0 without id, 1
               with id) */

int director_by_what(int i);                                        /* Output name of director fields (0 without id, 1
               with id */

/* Output ------------------------------------------------------------------------------------------------------- */

void clear_screen();                                                /* Clear the console */

void print_head(int n);                                             /* Output head of sheet (0 - for MOVIE LIST, 1 -
               for DIRECTOR LIST) */

void print_tail(int n);                                             /* Output tail of sheet (0 - for MOVIE LIST, 1 -
               for DIRECTOR LIST) */

void reference();                                                   /* Information about program and card-file */

void output_movie(MHD *mph);                                        /* Output movie list */

void output_director(DHD *dph);                                     /* Output director list */

/* Heads of Lists ----------------------------------------------------------------------------------------------- */

MHD *make_movie_head();                                             /* Movie Head Initialization */

void update_movie_cnt(MHD *mph);                                    /* Update counter in movie head on max(id) to
               correct work */

DHD *make_director_head();                                          /* Director Head Initialization */

void update_director_cnt(DHD *dph);                                 /* Update counter in director head on max(id) to
               correct work */

/* Create Node of List ------------------------------------------------------------------------------------------ */

DIR *find_director_by_id(DHD *dph, int id);                         /* Get pointer to Node in Director List by ID */

DIR *find_director_by_name(DHD *dph, char *movie_director);         /* Get pointer to Node in Director List by name */
                                                                    /* Movie Node Initialization */
MOV *create_movie(DHD *dph, int id_mov, char *movie_name, int id_dir, int movie_year, int movie_duration, float
               movie_kpr, float movie_plr, int watch_date[3]);

void add_movie(MHD *mph, MOV *new_node, MOV *current_node);         /* Add new Movie Node to List */

DIR *create_director(int id, char *movie_director);                 /* Director Node Initialization */

void add_director(DHD *dph, DIR *new_node, DIR *current_node);      /* Add new Director Node to List */

/* Supporting --------------------------------------------------------------------------------------------------- */

void clear_input_buffer();                                          /* Clear input buffer */

void to_lowercase(char *str);                                       /* Converts string to lowercase */

void rewrite_file(MHD *mph, DHD *dph);                              /* Rewrite file using lists */

void split_string(char *inputString, char **words, int *wordCount, char sep);
                                                                    /* Split string by separator */
/* Fill the lists and Create their elements --------------------------------------------------------------------- */
                                                                    /* Adding the movie data of file to list */
void add_movie_from_file(MHD *mph, DHD *dph, char *filename, char sep);
```

```
void add_movie_to_list(MHD *mph, DHD *dph);                             /* Enter new data -> Create new element -> Add to
               list */

void add_director_from_file(DHD *dph, char *filename, char sep);    /* Adding the director data of file to list */

void add_director_to_list(DHD *dph);                                    /* Enter new data -> Create new element -> Add to
               list */

/* Find in movie ------------------------------------------------------------------------------------------------ */
                                                                        /* Enter number in [a;b] and output relevant
               records */
void find_number_in_movie(MHD *mph, char *message, int a, int b, int i);

void find_string_in_movie(MHD *mph, char *message, int i);          /* Enter string (or substring) and output relevant
               records */

void find_date_in_movie(MHD *mph);                                      /* Enter date and output relevant records */

void find_in_movie(MHD *mph);                                           /* Chose the field of list to find record */

/* Find in director --------------------------------------------------------------------------------------------- */

void find_id_in_director(DHD *dph);                                     /* Enter ID and output relevant records */

void find_name_in_director(DHD *dph);                                   /* Enter string (or substring) and output relevant
               records */

void find_in_director(DHD *dph);                                        /* Chose the field of list to find record */

/* Edit Element of the List ------------------------------------------------------------------------------------- */

void edit_in_movie(MHD *mph, DHD *dph);                                 /* Edit the data of record in Movie list */

void edit_in_director(DHD *dph);                                        /* Edit the data of record in Director list */

/* Delete Element of the List ----------------------------------------------------------------------------------- */

void delete_movie(MHD *mph, MOV *current_movie);                        /* Delete Movie Node */

void delete_in_movie(MHD *mph);                                         /* Enter ID and delete relevant record */

void delete_director(MHD *mph, DHD *dph, DIR *current_director);    /* Delete Director Node */

void delete_in_director(MHD *mph, DHD *dph);                            /* Enter ID and delete relevant record */

/* Sort the Movie List ------------------------------------------------------------------------------------------ */

int swap_movie(MHD *mph, MOV *current);                                 /* Swap movie nodes */

void sort_movie_by_number(MHD *mph, int i, int direction);          /* Sort movie list by number field (0 - FORWARD /
               1 - BACKWARD direction) */

void sort_movie_by_string(MHD *mph, int i, int direction);          /* Sort movie list by string field (0 - A-Z / 1 -
               Z-A direction) */

void sort_movie_by_date(MHD *mph, int direction);                   /* Sort movie list by date (0 - FORWARD / 1 -
               BACKWARD direction) */

void sort_in_movie(MHD *mph);                                           /* Chose the field to sort */

/* Sort the Director List --------------------------------------------------------------------------------------- */

int swap_director(DHD *dph, DIR *current);                              /* Swap director nodes */

void sort_director_by_id(DHD *dph, int direction);                  /* Sort director list by id (0 - FORWARD / 1 -
               BACKWARD direction) */

void sort_director_by_name(DHD *dph, int direction);                /* Sort director list by name (0 - A-Z / 1 - Z-A
               direction) */

void sort_in_director(DHD *dph);                                        /* Chose the field to sort */

/* Main program ------------------------------------------------------------------------------------------------- */
```

```
int main(){
    MHD *mph = NULL;
    DHD *dph = NULL;
    int option, choise;
    dph=make_director_head();
    mph=make_movie_head();
    add_director_from_file(dph, director_db, SEP);
    add_movie_from_file(mph, dph, movie_db, SEP);
    update_director_cnt(dph);
    update_movie_cnt(mph);

    do{
        option = menu();

        switch(option){
        case 0: puts("\nYour selection is EXIT"); break;

        case 1:{
            puts("\nYour selection is SHOW THE DATA");
            choise=detail();
            if(choise==0) output_movie(mph);
            else {puts(""); output_director(dph);}
            break;
        }

        case 2:{
            puts("\nYour selection is ADD THE DATA");
            choise=detail();
            if(choise==0) add_movie_to_list(mph, dph);
            else add_director_to_list(dph);
            break;
        }

        case 3:{
            puts("\nYour selection is FIND THE DATA BY PARAMETERS");
            choise=detail();
            if(choise==0) find_in_movie(mph);
            else find_in_director(dph);
            break;
        }

        case 4:{
            puts("\nYour selection is EDIT THE DATA");
            choise=detail();
            if(choise==0) edit_in_movie(mph, dph);
            else edit_in_director(dph);
            break;
        }

        case 5:{
            puts("\nYour selection is DELETE THE DATA");
            choise=detail();
            if(choise==0) delete_in_movie(mph);
            else delete_in_director(mph, dph);
            break;
        }

        case 6:{
            puts("\nYour selection is SORT THE DATA");
            choise=detail();
            if(choise==0) sort_in_movie(mph);
            else sort_in_director(dph);
            break;
        }

        case 7:{
            puts("\nYour selection is REFERENCE");
            reference();
            break;
        }
        }
    rewrite_file(mph, dph);
    puts("\nPress ENTER to continue");
    clear_input_buffer();
```

```
    clear_screen();
    } while (option!=0);
    return 0;
}

/* Functions ---------------------------------------------------------------------------------------------------- */

/* Structure ---------------------------------------------------------------------------------------------------- */

struct movie{
    int id;                     /* ID of movie */
    char *name;                 /* Name of movie */
    struct director *director;  /* Director of movie */
    int year;                   /* Year of movie release */
    int duration;               /* Duration of movie in minutes*/
    float kpr;                  /* Movie rating on KinoPoisk */
    float plr;                  /* Movie rating on my opinion */
    int date[3];                /* Day/Month/Year of watch the movie */
    struct movie *prev;         /* Link to previous node */
    struct movie *next;         /* Link to next node */
};

struct movie_head{
    int cnt;                    /* Number of all existing elements */
    struct movie *first;        /* Link to first element of list */
    struct movie *last;         /* Link to last element of list */
};

struct director{
    int id;                     /* ID of the director */
    char *name;                 /* Director name */
    struct director *next;      /* Link to previous node */
    struct director *prev;      /* Link to next node */
};

struct director_head{
    int cnt;                    /* Number of all existing elements */
    struct director *first;     /* Link to first element of list */
    struct director *last;      /* Link to last element of list */
};

/* Input -------------------------------------------------------------------------------------------------------- */

void new_gets(char s[], int lim){
    char c;
    int i;
    i=0;

    while (((c=getchar())!='\n') && (i < lim-1)){ /* Entering symbols to the end of the string or until the limit is
            reached */
        s[i]=c;
        i++;
    }
    s[i]='\0';
}

int enter_integer(char *message, int a, int b){
    char input[MAXLEN];
    int number, flag;
    flag = 0;

    do{
        printf(message);
        new_gets(input, MAXLEN);
        if (sscanf(input, "%d", &number) == 1 && number >= a && number <= b) flag = 1; /* If entered number is number
            and in [a, b] */
        else puts("Entered value is not correct. Please try again!");
    } while (flag == 0); /* While entered number is not correct */
    return number;
}

float enter_float(char *message, float a, float b){
    char input[MAXLEN];
    float number;
```

```
    int flag;
    flag = 0;

    do{
        printf(message);
        new_gets(input, MAXLEN);
        if (sscanf(input, "%f", &number) == 1 && number >= a && number <= b) flag = 1; /* If entered number is number
               and in [a, b] */
        else puts("Entered value is not correct. Please try again!");
    } while (flag == 0); /* While entered number is not correct */
    return number;
}

void enter_date(char *message, int *day, int *month, int *year){
    char input[MAXLEN];
    int flag;
    flag=0;

    do {
        printf(message);
        new_gets(input, MAXLEN);
        if (sscanf(input, "%d %d %d", day, month, year) == 3 && *day >= 1 && *day <= 31 && *month >= 1 && *month <= 12
               && *year >= 1895 && *year <= 2099) flag = 1;  /* If entered date is real and in [1895-...] year */
        else printf("Entered date is not correct. Please try again!\n");
               /* First movie out in 1895 | flag := 1 */
    } while (flag==0); /* While entered date is not correct */
}

int menu(){
    int option;

    puts("Choose the option");
    puts("0 - for EXIT PROGRAM");
    puts("1 - for SHOW THE DATA");
    puts("2 - for ADD THE DATA");
    puts("3 - for FIND THE DATA BY PARAMETERS");
    puts("4 - for EDIT THE DATA");
    puts("5 - for DELETE THE DATA");
    puts("6 - for SORT THE DATA");
    puts("7 - for REFERENCE");

    option = enter_integer("Enter the option: ", 0, 7);
    return option;
}

int detail(){
    int option;

    puts("\nChoose the card-file");
    puts("0 - for MOVIE card-file");
    puts("1 - for DIRECTOR card-file");

    option = enter_integer("Enter the option: ", 0, 1);
    return option;
}

int movie_by_what(int i){
    int option;

    puts("\nChoose the parameter");
    if(i==1) puts("0 - by ID");
    else puts("0 - to EXIT");
    puts("1 - by Name");
    puts("2 - by Director");
    puts("3 - by Year");
    puts("4 - by Duration");
    puts("5 - by KPR");
    puts("6 - by PLR");
    puts("7 - by Date");

    option = enter_integer("Enter the option: ", 0, 7);
    return option;
}
```

```
int director_by_what(int i){
    int option;

    puts("\nChoose the parameter");
    if(i==0) puts("0 - to EXIT");
    else puts("0 - by ID");
    puts("1 - by Name");

    option = enter_integer("Enter the option: ", 0, 1);
    return option;
}

/* Output ------------------------------------------------------------------------------------------------------------ */

void clear_screen(){
    #if defined(_WIN32) || defined(_WIN64)
        system("cls");
    #else
        system("clear");
    #endif
}

void print_head(int n){
    if(n==0){               /* 0 for movie / 1 for director */
        printf("| %2s | %25s | %25s | %4s | %3s | %5s | %5s | %10s
               |\n","ID","Name","Director","Year","Dur","KPR","PLR","Watchdate");
        printf("+----+---------------------------+---------------------------+------+-----+-------+-------+-----------
               -+\n");
    } else {
        printf("| %2s | %25s |\n","ID","Director");
        printf("+----+---------------------------+\n");
    }
}

void print_tail(int n){     /* 0 for movie / 1 for director */
    if(n==0) printf("+----+---------------------------+---------------------------+------+-----+-------+-------+------
               ------+\n");
    else printf("+----+---------------------------+\n");
}

void reference(){
    printf("\nThis program is card-file. Its functions are presented in the menu.\nSubject area of the program -
               movie\nDesignations:\nName - Name of movie\nDirector - Director of the movie\nYear - Year of movie
               release\nDuration - Duration of the movie in minutes\nKPR - Rating of the movie on KinoPoisk\nPLR -
               Personal rating of the movie\nDate - Date of viewing the movie\n");
}

void output_movie(MHD *mph){
    MOV *current;
    current = mph->first;

    puts("");
    if(current!=NULL){
        print_head(0);
        while (current!=NULL) {
                printf("| %2d | %25s | %25s | %d | %3d | %5.1f | %5.1f | %.2d.%.2d.%d |\n", current->id,
               current->name, current->director->name, current->year, current->duration, current->kpr, current->plr,
               current->date[0], current->date[1], current->date[2]);
                current = current->next;
        }
        print_tail(0);
    } else puts("The movie list is empty");
}

void output_director(DHD *dph){
    DIR *current;
    current = dph->first;

    if(current!=NULL){
    print_head(1);
    while (current!=NULL) {
        printf("| %2d | %25s |\n", current->id, current->name);
        current = current->next;
    }
```

```
    print_tail(1);
    } else puts("The director list is empty");
}

/* Heads of Lists ------------------------------------------------------------------------------------------------ */

MHD *make_movie_head(){
    MHD *mph=NULL; /* Define and init Head */
    mph=(MHD*)malloc(sizeof(MHD));
    mph->cnt = 0;
    mph->first=NULL;
    mph->last=NULL;
    return mph;
}

void update_movie_cnt(MHD *mph){
    MOV *current;
    int max_id;
    current = mph->first;
    max_id = -1;

    while (current != NULL) {
        if (current->id > max_id) max_id = current->id;
        current = current->next;
    }
    mph->cnt = max_id + 1;
}

DHD *make_director_head(){
    DHD *dph=NULL; /* Define and init Head */
    dph=(DHD*)malloc(sizeof(DHD));
    dph->cnt = 0;
    dph->first=NULL;
    dph->last=NULL;
    return dph;
}

void update_director_cnt(DHD *dph){
    DIR *current;
    int max_id;
    current = dph->first;
    max_id = -1;

    while (current != NULL) {
        if (current->id > max_id) max_id = current->id;
        current = current->next;
    }
    dph->cnt = max_id + 1;
}

/* Create Node of List ------------------------------------------------------------------------------------------- */

DIR *find_director_by_id(DHD *dph, int id){
    DIR *current = NULL;
    int flag;
    current = dph->first;
    flag = 0;

    while(current!=NULL && flag==0){
        if(current->id==id) flag = 1;
        else current=current->next;
    }
    return current;
}

DIR *find_director_by_name(DHD *dph, char *movie_director){
    DIR *current = NULL;
    int flag;
    current = dph->first;
    flag = 0;

    while (current!= NULL && flag == 0) {
        if (strcasecmp(current->name, movie_director) == 0) flag = 1;
        else current = current->next;
```

```
    }
    return current;
}

MOV *create_movie(DHD *dph, int id_mov, char *movie_name, int id_dir, int movie_year, int movie_duration, float
            movie_kpr, float movie_plr, int watch_date[3]){
    MOV *new_movie = NULL; /* Pointer to new node */
    DIR *movie_director = NULL; /* Pointer to director of movie */
    char *name = NULL;

    new_movie = (MOV*)malloc(sizeof(MOV));
    name = (char*)malloc((strlen(movie_name) + 1) * sizeof(char));
    movie_director = (DIR *)malloc(sizeof(DIR));

    movie_director=find_director_by_id(dph, id_dir);

    new_movie->id = id_mov;
    new_movie->name = name;
    new_movie->director = movie_director;
    new_movie->year = movie_year;
    new_movie->duration = movie_duration;
    new_movie->kpr = movie_kpr;
    new_movie->plr = movie_plr;
    memcpy(name, movie_name, strlen(movie_name) + 1);
    memcpy(new_movie->date, watch_date, sizeof(new_movie->date));
    new_movie->next = NULL;
    new_movie->prev = NULL;
    return new_movie; /* Return address of node */
}

void add_movie(MHD *mph, MOV *new_node, MOV *current_node){
    if (mph && new_node){
        if (current_node == NULL) {     /* Add first node of list */
            mph->cnt = 1;
            mph->first = new_node;
            mph->last = new_node;
            new_node->prev = NULL;
            new_node->next = NULL;
        } else {
            mph->cnt++;
            current_node->next = new_node;
            new_node->prev = current_node;
            new_node->next = NULL;
            mph->last = new_node;
        }
    } else printf("\nError! The head of movie list is missing\n");
}

DIR *create_director(int id, char *movie_director){
    DIR *new_director = NULL; /* Pointer to new node */
    char *director = NULL;

    new_director = (DIR*)malloc(sizeof(DIR));
    director = (char*)malloc((strlen(movie_director) + 1) * sizeof(char));

    new_director->id = id;
    new_director->name = director;
    memcpy(director, movie_director, strlen(movie_director) + 1);
    new_director->prev = NULL;
    new_director->next = NULL;
    return new_director; /* Return address of node */
}

void add_director(DHD *dph, DIR *new_node, DIR *current_node){
    if (dph && new_node){
        if (current_node == NULL) { /* Add first node of list */
            dph->cnt = 1;
            dph->first = new_node;
            dph->last = new_node;
            new_node->prev = NULL;
            new_node->next = NULL;
        } else {
            dph->cnt++;
            current_node->next = new_node;
```

```
            new_node->prev = current_node;
            new_node->next = NULL;
            dph->last = new_node;
        }
    } else printf("\nError! The head of director list is missing\n");
}

/* Supporting ------------------------------------------------------------------------------------------------------------ */

void clear_input_buffer(){
    int c;
    while ((c = getchar()) != '\n' && c != EOF);
}

void to_lowercase(char *str){
    int i;
    for (i = 0; str[i]; i++) {
        str[i] = tolower(str[i]);
    }
}

void rewrite_file(MHD *mph, DHD *dph){
    MOV *mov_current;
    DIR *dir_current;
    FILE *file1 = fopen(movie_db, "w");
    FILE *file2 = fopen(director_db, "w");
    mov_current = mph->first;
    dir_current = dph->first;

    while(mov_current!=NULL){
        fprintf(file1, "%d;%s;%d;%d;%d;%.1f;%.1f;%d;%d;%d\n", mov_current->id, mov_current->name,
               mov_current->director->id, mov_current->year, mov_current->duration, mov_current->kpr,
               mov_current->plr, mov_current->date[0], mov_current->date[1], mov_current->date[2]);
        mov_current=mov_current->next;
    }
    while(dir_current!=NULL){
        fprintf(file2, "%d;%s\n", dir_current->id, dir_current->name);
        dir_current=dir_current->next;
    }
    fclose(file1);
    fclose(file2);
}

void split_string(char *input_string, char **words, int *word_count, char sep) {
    int word_index, word_start, word_length, in_word, i, len;
    len = strlen(input_string);
    word_index = 0;
    word_start = 0;
    word_length = 0;
    in_word = 0;

    for (i = 0; i <= len; i++) {
        if ((input_string[i] == sep || input_string[i] == '\0')&&(in_word==1)) {
            words[word_index] = (char *)malloc(word_length + 1);
            strncpy(words[word_index], input_string + word_start, word_length);
            words[word_index][word_length] = '\0';
            word_index++;
            in_word = 0;
        } else {
            if (in_word==0) {
                word_start = i;
                word_length = 1;
                in_word = 1;
            } else word_length++;
        }
    }
    *word_count = word_index;
    /* For correct work on OS MAC and OS LINUX */
    for (i = 0; i < *word_count; i++) {
        len = strlen(words[i]);
        if (words[i][len - 1] == '\r') words[i][len - 1] = '\0';
    }
}
```

```
/* Fill the lists and Create their elements ------------------------------------------------------------------------ */

void add_movie_from_file(MHD *mph, DHD *dph, char *filename, char sep){
    MOV *new_movie;
    char line[MAXLEN], *words[10];  /* [10] - count of columns in filename and that card-file */
    int word_count, date[3], i;
    FILE *file = fopen(filename, "r");

    if (file == NULL) printf("Error opening file.\n");
    else {
        while (fgets(line, sizeof(line), file)){
            line[strcspn(line, "\n")] = 0;
            split_string(line, words, &word_count, sep);

            if (word_count == 10){
                int id_mov = atoi(words[0]);
                char *name = words[1];
                int id_dir = atoi(words[2]);
                int year = atoi(words[3]);
                int duration = atoi(words[4]);
                float kpr = atof(words[5]);
                float plr = atof(words[6]);
                date[0] = atoi(words[7]);
                date[1] = atoi(words[8]);
                date[2] = atoi(words[9]);

                new_movie = create_movie(dph, id_mov, name, id_dir, year, duration, kpr, plr, date);
                add_movie(mph, new_movie, mph->last);
            }
            else printf("Invalid number of attributes in line: %s\n", line);

            for (i = 0; i < word_count; i++){
                free(words[i]);
            }
        }
        fclose(file);
    }
}

void add_movie_to_list(MHD *mph, DHD *dph){
    MOV *new_movie = NULL;
    DIR *current = NULL;
    char name[MAXLEN],director[MAXLEN];
    float kpr, plr;
    int id_dir, year, duration, date[3];

    printf("\nID of new movie set automatically on %d\n", mph->cnt);
    printf("Enter the name of movie: ");
    new_gets(name,MAXLEN);
    printf("Enter the director of movie: ");
    new_gets(director,MAXLEN);
    year = enter_integer("Enter the year of movie: ", 1985, 2099);
    duration = enter_integer("Enter the duration of movie: ", 1, 999);
    kpr = enter_float("Enter the KPR of movie: ", 0, 10);
    plr = enter_float("Enter the PLR of movie: ", 0, 10);
    enter_date("Enter the date of viewing (DD MM YYYY): ", &date[0],&date[1],&date[2]);

    current = find_director_by_name(dph,director);
    if(current == NULL) {
        id_dir = dph->cnt;
        current = create_director(id_dir, director);
        add_director(dph, current, dph->last);
    }
    else id_dir = current->id;

    new_movie = create_movie(dph, mph->cnt, name, id_dir, year, duration, kpr, plr, date);
    add_movie(mph, new_movie, mph->last);
    printf("\nMovie added successfully\n");
}

void add_director_from_file(DHD *dph, char *filename, char sep){
    DIR *new_director;
    char line[MAXLEN], *words[2];   /* [2] - count of columns in filename and that card-file */
    int wordCount, i;
```

```
    FILE *file = fopen(filename, "r");

    if (file == NULL) printf("Error opening file.\n");
    else {
        while (fgets(line, sizeof(line), file)){
            line[strcspn(line, "\n")] = 0;
            split_string(line, words, &wordCount, sep);
            if (wordCount == 2){
                int id = atoi(words[0]);
                char *name = words[1];

                new_director = create_director(id, name);
                add_director(dph, new_director, dph->last);
            }
            else printf("Invalid number of attributes in line: %s\n", line);

            for (i = 0; i < wordCount; i++){
                free(words[i]);
            }
        }
        fclose(file);
    }
}

void add_director_to_list(DHD *dph){
    DIR *new_director = NULL;
    char name[MAXLEN];

    printf("\nID of new director set automatically on %d\n", dph->cnt);
    printf("Enter the name of director: ");
    new_gets(name, MAXLEN);
    new_director = find_director_by_name(dph, name);
    if(new_director==NULL){
        new_director = create_director(dph->cnt, name);
        add_director(dph, new_director, dph->last);
        printf("\nDirector added successfully\n");
    } else printf("\nThe director you entered is already on the list\n");
}

/* Find in movie ------------------------------------------------------------------------------------------------ */

void find_number_in_movie(MHD *mph, char *message, int a, int b, int i){
    MOV *current;
    float record, number;
    int count;
    current = mph->first;
    count = 0;

    if (i<3) number=enter_integer(message, a, b);
    else number=enter_float(message, a, b);
    puts("");

    while(current!=NULL){
        switch(i){
        case 0: record = current->id; break;
        case 1: record = current->year; break;
        case 2: record = current->duration; break;
        case 3: record = current->kpr; break;
        case 4: record = current->plr; break;
        }
        if(record == number){
            count++;
            if (count==1) print_head(0);
            printf("| %2d | %25s | %25s | %d | %3d | %5.1f | %5.1f | %.2d.%.2d.%d |\n", current->id, current->name,
               current->director->name, current->year, current->duration, current->kpr, current->plr,
               current->date[0], current->date[1], current->date[2]);
        }
        current = current->next;
    }
    if (count==0) printf("Movie with the entered value was not found\n");
    else print_tail(0);
}

void find_string_in_movie(MHD *mph, char *message, int i){
```

```
    MOV *current;
    char input[MAXLEN], *string, lstring[MAXLEN];
    int count, result;
    printf(message);
    new_gets(input, MAXLEN); /* Read the string */
    to_lowercase(input);
    current = mph->first;
    count = 0;

    puts("");

    while(current!=NULL){
        switch(i){
        case 0: string = current->name; break;
        case 1: string = current->director->name; break;
        }
        strcpy(lstring, string);
        to_lowercase(lstring);
        result = strncasecmp(input, lstring, strlen(input));
        if(result == 0 || strstr(lstring, input) != NULL) {
            count++;
            if (count==1) print_head(0);
            printf("| %2d | %25s | %25s | %d | %3d | %5.1f | %5.1f | %.2d.%.2d.%d |\n", current->id, current->name,
               current->director->name, current->year, current->duration, current->kpr, current->plr,
               current->date[0], current->date[1], current->date[2]);
        }
        current = current->next;
    }
    if (count==0) printf("Movie with the entered String (or Substring) was not found\n");
    else print_tail(0);
}

void find_date_in_movie(MHD *mph){
    MOV *current;
    int date[3], count;
    current = mph->first;
    count=0;

    enter_date("Enter the date of viewing (DD MM YYYY): ", &date[0], &date[1], &date[2]);
    puts("");

    while (current != NULL) {
        if (current->date[0] == date[0] && current->date[1] == date[1] && current->date[2] == date[2]) {
            count++;
            if (count == 1) print_head(0);
            printf("| %2d | %25s | %25s | %d | %3d | %5.1f | %5.1f | %.2d.%.2d.%d |\n",
                   current->id, current->name, current->director->name, current->year,
                   current->duration, current->kpr, current->plr, current->date[0],
                   current->date[1], current->date[2]);
        }
        current = current->next;
    }

    if (count == 0) printf("Movie with the entered Date was not found\n");
    else print_tail(0);
}

void find_in_movie(MHD *mph){
    int option;

    if(mph->first!=NULL){
        option=movie_by_what(1);
        puts("");
        switch (option) {
            case 0: find_number_in_movie(mph, "Enter the ID: ", 0, mph->cnt, 0); break;
            case 1: find_string_in_movie(mph, "Enter the Name of Movie (or Substring): ", 0); break;
            case 2: find_string_in_movie(mph, "Enter the Name of Director (or Substring): ", 1); break;
            case 3: find_number_in_movie(mph, "Enter the Year: ", 1895, 2099, 1); break;
            case 4: find_number_in_movie(mph, "Enter the Duration: ", 1, 999, 2); break;
            case 5: find_number_in_movie(mph, "Enter the KPR: ", 0, 10, 3); break;
            case 6: find_number_in_movie(mph, "Enter the PLR: ", 0, 10, 4); break;
            case 7: find_date_in_movie(mph); break;
        }
    } else printf("\nThe movie list is empty\n");
```

```
}

/* Find in director ------------------------------------------------------------------------------------------ */

void find_id_in_director(DHD *dph){
    DIR *current;
    int id, count;
    current = dph->first;
    count = 0;

    puts("");
    id = enter_integer("Enter the ID: ", 0, dph->cnt);
    puts("");
    while(current!=NULL){
        if (current->id == id){
            count++;
            if (count==1) print_head(1);
            printf("| %2d | %25s |\n", current->id, current->name);
        }
        current = current->next;
    }
    if (count==0) printf("Director with the entered ID was not found\n");
    else print_tail(1);
}

void find_name_in_director(DHD *dph){
    DIR *current;
    char input[MAXLEN], lstring[MAXLEN];
    int count, result;
    printf("\nEnter the Name (or Substring): ");
    new_gets(input, MAXLEN); /* Read the string */
    to_lowercase(input);
    current = dph->first;
    count = 0;

    puts("");

    while (current != NULL) {
        strcpy(lstring, current->name);
        to_lowercase(lstring);
        result = strncasecmp(input, lstring, strlen(input));
        if (result == 0 || strstr(lstring, input) != NULL) {
            count++;
            if (count == 1) print_head(1);
            printf("| %2d | %25s |\n", current->id, current->name);
        }
        current = current->next;
    }

    if (count == 0) printf("Director with the entered Name (or Substring) was not found\n");
    else print_tail(1);
}

void find_in_director(DHD *dph){
    int option;

    if(dph->first!=NULL){
        option=director_by_what(1);
        if(option==0) find_id_in_director(dph);
        else find_name_in_director(dph);
    } else printf("\nThe director list is empty\n");
}

/* Edit Element of the List ---------------------------------------------------------------------------------- */

void edit_in_movie(MHD *mph, DHD *dph){
    MOV *current;
    DIR *temp_dir;
    char input[MAXLEN];
    float kpr, plr;
    int option, mov_id, dir_id, year, duration, date[2], i, flag;
    flag=0;

    if(mph->first!=NULL){
```

```
        option=movie_by_what(0);
        if(option!=0){
            current=mph->first;
            output_movie(mph);

            mov_id = enter_integer("Enter the ID of the record you want to edit: ", 0, (mph->cnt)-1);
            while(current!=NULL && flag == 0){
                if(current->id == mov_id) flag = 1;
                else current = current->next;
            }
            if (current!=NULL){
                puts("");
                switch (option) {
                    case 1:
                        printf("Enter the new name of Movie: ");
                        new_gets(input,MAXLEN);
                        strcpy(current->name, input);
                        break;
                    case 2:
                        output_director(dph);
                        dir_id=enter_integer("Enter the new director ID of Movie: ", 0, (dph->cnt)-1);
                        temp_dir = find_director_by_id(dph, dir_id);
                        if(temp_dir!=NULL) current->director = temp_dir;
                        else printf("Director with entered ID was not found. Keeping the previous director\n");
                        break;
                    case 3:
                        year = enter_integer("Enter the new Year of Movie: ", 1895, 2099);
                        current->year = year;
                        break;
                    case 4:
                        duration = enter_integer("Enter the new Duration of Movie: ", 1, 999);
                        current->duration = duration;
                        break;
                    case 5:
                        kpr = enter_float("Enter the new KPR of Movie: ", 0, 10);
                        current->kpr = kpr;
                        break;
                    case 6:
                        plr = enter_float("Enter the new PLR of Movie: ", 0, 10);
                        current->plr = plr;
                        break;
                    case 7:
                        enter_date("Enter the new date of viewing (DD MM YYYY): ", &date[0],&date[1],&date[2]);
                        for(i=0; i<3; i++) current->date[i] = date[i];
                        break;
                }
                printf("\nThe data updated successfully\n");
            } else printf("\nMovie with entered ID was not found\n");
        }
    } else printf("\nThe movie list is empty\n");
}

void edit_in_director(DHD *dph){
    DIR *current = NULL;
    char input[MAXLEN];
    int option, dir_id, flag;

    if(dph->first!=NULL){
        option = director_by_what(0);
        if (option!=0){
            puts("");
            output_director(dph);
            dir_id = enter_integer("Enter the ID of the record you want to edit: ", 0, (dph->cnt)-1);
            current = dph->first;
            flag=0;
            while(current!=NULL && flag == 0){
                if(current->id == dir_id) flag = 1;
                else current = current->next;
            }
            if(current!=NULL){
                printf("Enter the new name Name of Director: ");
                new_gets(input, MAXLEN);
                strcpy(current->name, input);
            } else printf("\nDirector with entered ID was not found\n");
```

```
        }
    } else printf("\nThe director list is empty\n");
}

/* Delete Element of the List ---------------------------------------------------------------------------------- */

void delete_movie(MHD *mph, MOV *current_movie) {
    if (current_movie == mph->first) { /* If deleted node is the first in the list */
        mph->first = current_movie->next;
        if (mph->first) {
            mph->first->prev = NULL;
        } else { /* If deleted node is alone in the list */
            mph->last = NULL;
        }
    } else if (current_movie == mph->last) { /* If deleted node is the last in the list */
        mph->last = current_movie->prev;
        if (mph->last) {
            mph->last->next = NULL;
        } else { /* If deleted node is alone in the list */
            mph->first = NULL;
        }
    } else { /* If deleted node not first or last of the list */
        current_movie->prev->next = current_movie->next;
        current_movie->next->prev = current_movie->prev;
    }
    current_movie->next = NULL;
    current_movie->prev = NULL;
    free(current_movie);
}

void delete_in_movie(MHD *mph){
    MOV *current = NULL;
    int id_mov, flag;

    if(mph->first!=NULL){
        current=mph->first;
        flag = 0;

        puts("");
        id_mov = enter_integer("Enter the ID of the movie you want to delete: ", 0, (mph->cnt)-1);
        while(current!=NULL && flag==0){
            if(current->id == id_mov) flag=1;
            else current = current->next;
        }

        if(current==NULL) printf("\nMovie with entered ID was not found\n");
        else {delete_movie(mph, current); printf("\nMovie with entered ID was deleted\n");}
    } else printf("\nThe movie list is empty\n");
}

void delete_director(MHD *mph, DHD *dph, DIR *current_director){
    MOV *next_movie = NULL;
    MOV *current_movie = mph->first;

    while (current_movie != NULL) {
        next_movie = current_movie->next;
        if (current_movie->director == current_director) {
            delete_movie(mph, current_movie);
        }
        current_movie = next_movie;
    }

    if (current_director == dph->first) {
        dph->first = current_director->next;
        if (dph->first) {
            dph->first->prev = NULL;
        } else dph->last = NULL;
    } else if (current_director== dph->last) {
        dph->last = current_director->prev;
        if (dph->last) {
            dph->last->next = NULL;
        } else dph->first = NULL;
    } else {
        current_director->prev->next = current_director->next;
```

```
        current_director->next->prev = current_director->prev;
    }

    current_director->next = NULL;
    current_director->prev = NULL;

    free(current_director);
}

void delete_in_director(MHD *mph, DHD *dph){
    DIR *current = NULL;
    int id_dir;

    if(dph->first!=NULL){
        puts("");
        id_dir = enter_integer("Enter the ID of the director you want to delete: ", 0, (dph->cnt)-1);
        current=find_director_by_id(dph, id_dir);
        if(current != NULL) {delete_director(mph, dph, current); printf("\nDirector with entered ID was deleted\n");}
        else printf("\nDirector with entered ID was not found\n");
    } else printf("\nThe director list is empty\n");
}

/* Sort the Movie List ------------------------------------------------------------------------------------------ */

int swap_movie(MHD *mph, MOV *current){
    MOV *temp = current->next;

    if (current->prev != NULL) current->prev->next = temp;
    else mph->first = temp;

    if (temp->next != NULL) temp->next->prev = current;
    else mph->last = current;

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

    return 1;
}

void sort_movie_by_number(MHD *mph, int i, int direction) {
    MOV *current = NULL;
    int swapped, need_to_swap;

    do {
        current = mph->first;
        swapped = 0;

        while (current->next != NULL) {
            need_to_swap = 0;
            switch (i) {
                case 0:
                    if ((direction == 0 && current->id > current->next->id) || (direction == 1 && current->id <
               current->next->id)) need_to_swap = 1;
                    break;
                case 1:
                    if ((direction == 0 && current->year > current->next->year) || (direction == 1 && current->year <
               current->next->year)) need_to_swap = 1;
                    break;
                case 2:
                    if ((direction == 0 && current->duration > current->next->duration) || (direction == 1 &&
               current->duration < current->next->duration)) need_to_swap = 1;
                    break;
                case 3:
                    if ((direction == 0 && current->kpr > current->next->kpr) || (direction == 1 && current->kpr <
               current->next->kpr)) need_to_swap = 1;
                    break;
                case 4:
                    if ((direction == 0 && current->plr > current->next->plr) || (direction == 1 && current->plr <
               current->next->plr)) need_to_swap = 1;
                    break;
            }
```

```
            if (need_to_swap == 1) swapped = swap_movie(mph, current);
            else current = current->next;
        }
    } while (swapped);
}

void sort_movie_by_string(MHD *mph, int i, int direction) {
    MOV *current = NULL;
    int swapped, need_to_swap;

    do {
        swapped = 0;
        current = mph->first;

        while (current->next != NULL) {
            need_to_swap = 0;

            switch (i) {
                case 0:
                    if ((direction == 0 && strcmp(current->name, current->next->name) > 0)  || (direction == 1 &&
               strcmp(current->name, current->next->name) < 0)) need_to_swap = 1;
                    break;
                case 1:
                    if ((direction == 0 && strcmp(current->director->name, current->next->director->name) > 0)  ||
               (direction == 1 && strcmp(current->director->name, current->next->director->name) < 0)) need_to_swap =
               1;
                    break;
            }

            if (need_to_swap == 1) swapped = swap_movie(mph, current);
            else current = current->next;
        }
    } while (swapped);
}

void sort_movie_by_date(MHD *mph, int direction) {
    MOV *current = NULL;
    int swapped, need_to_swap;

    do {
        swapped = 0;
        current = mph->first;

        while (current->next != NULL) {
            need_to_swap = 0;

            if ((direction == 0 && current->date[2] > current->next->date[2]) || (direction == 1 && current->date[2] <
               current->next->date[2])) need_to_swap = 1;
            else if (current->date[2] == current->next->date[2]) {
                if ((direction == 0 && current->date[1] > current->next->date[1]) || (direction == 1 &&
               current->date[1] < current->next->date[1])) need_to_swap = 1;
                else if (current->date[1] == current->next->date[1]) {
                    if ((direction == 0 && current->date[0] > current->next->date[0]) || (direction == 1 &&
               current->date[0] < current->next->date[0])) need_to_swap = 1;
                }
            }

            if (need_to_swap == 1) swapped = swap_movie(mph, current);
            else current = current->next;
        }
    } while (swapped);
}

void sort_in_movie(MHD *mph){
    int option, direction;

    if(mph->first!=NULL){
        option = movie_by_what(1);
        puts("");
        direction = enter_integer("Enter the direction of sort (0 - Forward / 1 - Backward): ", 0, 1);
        switch(option){
        case 0: sort_movie_by_number(mph, 0, direction); break;
        case 1: sort_movie_by_string(mph, 0, direction); break;
        case 2: sort_movie_by_string(mph, 1, direction); break;
```

```
        case 3: sort_movie_by_number(mph, 1, direction); break;
        case 4: sort_movie_by_number(mph, 2, direction); break;
        case 5: sort_movie_by_number(mph, 3, direction); break;
        case 6: sort_movie_by_number(mph, 4, direction); break;
        case 7: sort_movie_by_date(mph, direction); break;
        }
        printf("\nThe movie list was sorted\n");
    } else printf("\nThe movie list is empty\n");
}

/* Sort the Director List ------------------------------------------------------------------------------------------ */

int swap_director(DHD *dph, DIR *current) {
    DIR *temp = current->next;

    if (current->prev != NULL) current->prev->next = temp;
    else dph->first = temp;

    if (temp->next != NULL) temp->next->prev = current;
    else dph->last = current;

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

    return 1;
}

void sort_director_by_id(DHD *dph, int direction){
    DIR *current = NULL;
    int swapped;

    do {
        swapped = 0;
        current = dph->first;

        while (current->next != NULL) {
            if ((direction == 0 && current->id > current->next->id) ||  (direction == 1 && current->id <
               current->next->id)) swapped = swap_director(dph, current);
            else current = current->next;
        }
    } while (swapped);
}

void sort_director_by_name(DHD *dph, int direction){
    DIR *current = NULL;
    int swapped;

    do {
        swapped = 0;
        current = dph->first;

        while (current->next != NULL) {
            if ((direction == 0 && strcmp(current->name, current->next->name) > 0) ||  (direction == 1 &&
               strcmp(current->name, current->next->name) < 0)) swapped = swap_director(dph, current);
            else current = current->next;
        }
    } while (swapped);
}

void sort_in_director(DHD *dph){
    int option, direction;

    if(dph->first!=NULL){
        option = director_by_what(1);
        puts("");
        direction = enter_integer("Enter the direction of sort (0 - Forward / 1 - Backward): ", 0, 1);
        if(option==0) sort_director_by_id(dph, direction);
        else sort_director_by_name(dph, direction);
        printf("\nThe director list was sorted\n");
    } else printf("\nThe director list is empty\n");
}
```

Примеры работы программы

Исходные данные:

Текстовые файлы:

```
0;Forrest Gump;0;1994;142;8.9;8.4;10;10;2015
1;Inception;1;2010;148;8.7;10.0;10;5;2017
2;Fight Club;2;1999;139;8.7;10.0;12;11;2019
3;1917;3;2019;119;8.2;7.0;8;2;2020
4;Interstellar;1;2014;169;8.6;9.8;23;3;2024
5;Shutter Island;4;2010;138;8.3;9.5;9;1;2024
6;The Boy and Heron;5;2023;124;7.8;6.0;17;12;2023
7;Spirited Away;5;2001;125;8.5;8.1;10;12;2021
9;The Wolf of Wall Street;4;2013;180;8.1;8.7;30;4;2024
10;The Shawshank Redemption;7;1994;142;9.2;9.3;18;6;2023
11;The Dune;8;2021;155;8.3;7.5;8;3;2024
```

```
0;Robert Zemeckis
1;Cristopher Nolan
2;David Fincher
3;Sam Mendes
4;Martin Scorsese
5;Hayao Miyazaki
7;Frank Darabont
8;Deni Villeneuve
```

Списки:

ID	Name	Director	Year	Dur	KPR	PLR	Watchdate
0	Forrest Gump	Robert Zemeckis	1994	142	8.9	8.4	10.10.2015
1	Inception	Cristopher Nolan	2010	148	8.7	10.0	10.05.2017
2	Fight Club	David Fincher	1999	139	8.7	10.0	12.11.2019
3	1917	Sam Mendes	2019	119	8.2	7.0	08.02.2020
4	Interstellar	Cristopher Nolan	2014	169	8.6	9.8	23.03.2024
5	Shutter Island	Martin Scorsese	2010	138	8.3	9.5	09.01.2024
6	The Boy and Heron	Hayao Miyazaki	2023	124	7.8	6.0	17.12.2023
7	Spirited Away	Hayao Miyazaki	2001	125	8.5	8.1	10.12.2021
9	The Wolf of Wall Street	Martin Scorsese	2013	180	8.1	8.7	30.04.2024
10	The Shawshank Redemption	Frank Darabont	1994	142	9.2	9.3	18.06.2023
11	The Dune	Deni Villeneuve	2021	155	8.3	7.5	08.03.2024

ID	Director
0	Robert Zemeckis
1	Cristopher Nolan
2	David Fincher
3	Sam Mendes
4	Martin Scorsese
5	Hayao Miyazaki
7	Frank Darabont
8	Deni Villeneuve

Пример 1 (Ознакомление с программой, добавление и вывод данных):

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 7

Your selection is REFERENCE

This program is card-file. Its functions are presented in the menu.
Subject area of the program - movie
Designations:
Name - Name of movie
Director - Director of the movie
Year - Year of movie release
Duration - Duration of the movie in minutes
KPR - Rating of the movie on KinoPoisk
PLR - Personal rating of the movie
Date - Date of viewing the movie

Press ENTER to continue
```

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 2

Your selection is ADD THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

ID of new movie set automatically on 12
Enter the name of movie: Prisoners
Enter the director of movie: Deni Villeneuve
Enter the year of movie: qwe
Entered value is not correct. Please try again!
Enter the year of movie: 2013
Enter the duration of movie: -125
Entered value is not correct. Please try again!
Enter the duration of movie: 153
Enter the KPR of movie: 8.2
Enter the PLR of movie: 15
Entered value is not correct. Please try again!
Enter the PLR of movie: 9.5
Enter the date of viewing (DD MM YYYY): 08 12 1
Entered date is not correct. Please try again!
Enter the date of viewing (DD MM YYYY): 08 12 2023

Movie added successfully

Press ENTER to continue
```

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 2

Your selection is ADD THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 1

ID of new director set automatically on 9
Enter the name of director: James Gunn

Director added successfully

Press ENTER to continue
```

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 1

Your selection is SHOW THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

| ID |                      Name |                  Director | Year | Dur |   KPR |   PLR |  Watchdate |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+
|  0 |              Forrest Gump |           Robert Zemeckis | 1994 | 142 |   8.9 |   8.4 | 10.10.2015 |
|  1 |                 Inception |          Cristopher Nolan | 2010 | 148 |   8.7 |  10.0 | 10.05.2017 |
|  2 |                Fight Club |             David Fincher | 1999 | 139 |   8.7 |  10.0 | 12.11.2019 |
|  3 |                      1917 |                Sam Mendes | 2019 | 119 |   8.2 |   7.0 | 08.02.2020 |
|  4 |              Interstellar |          Cristopher Nolan | 2014 | 169 |   8.6 |   9.8 | 23.03.2024 |
|  5 |           Shutter Island |           Martin Scorsese | 2010 | 138 |   8.3 |   9.5 | 09.01.2024 |
|  6 |         The Boy and Heron |            Hayao Miyazaki | 2023 | 124 |   7.8 |   6.0 | 17.12.2023 |
|  7 |             Spirited Away |            Hayao Miyazaki | 2001 | 125 |   8.5 |   8.1 | 10.12.2021 |
|  9 |   The Wolf of Wall Street |           Martin Scorsese | 2013 | 180 |   8.1 |   8.7 | 30.04.2024 |
| 10 |  The Shawshank Redemption |            Frank Darabont | 1994 | 142 |   9.2 |   9.3 | 18.06.2023 |
| 11 |                  The Dune |          Deni Villeneuve | 2021 | 155 |   8.3 |   7.5 | 08.03.2024 |
| 12 |                 Prisoners |          Deni Villeneuve | 2013 | 153 |   8.2 |   9.5 | 08.12.2023 |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+

Press ENTER to continue
```

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 1

Your selection is SHOW THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 1

| ID |                  Director |
+----+---------------------------+
|  0 |           Robert Zemeckis |
|  1 |          Cristopher Nolan |
|  2 |             David Fincher |
|  3 |                Sam Mendes |
|  4 |           Martin Scorsese |
|  5 |            Hayao Miyazaki |
|  7 |            Frank Darabont |
|  8 |          Deni Villeneuve |
|  9 |                James Gunn |
+----+---------------------------+

Press ENTER to continue
```

Пример 2 (Поиск данных и их редактирование):

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 3

Your selection is FIND THE DATA BY PARAMETERS

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 1

Choose the parameter
0 - by ID
1 - by Name
Enter the option: 1

Enter the Name (or Substring): de

| ID |                  Director |
+----+---------------------------+
|  3 |                Sam Mendes |
|  8 |           Deni Villeneuve |
+----+---------------------------+

Press ENTER to continue
```

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 4

Your selection is EDIT THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

Choose the parameter
0 - to EXIT
1 - by Name
2 - by Director
3 - by Year
4 - by Duration
5 - by KPR
6 - by PLR
7 - by Date
Enter the option: 5

| ID |                      Name |                  Director | Year | Dur |   KPR |   PLR |  Watchdate |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+
|  0 |               Forrest Gump |            Robert Zemeckis | 1994 | 142 |   8.9 |   8.4 | 10.10.2015 |
|  1 |                  Inception |           Cristopher Nolan | 2010 | 148 |   8.7 |  10.0 | 10.05.2017 |
|  2 |                 Fight Club |              David Fincher | 1999 | 139 |   8.7 |  10.0 | 12.11.2019 |
|  3 |                       1917 |                  Sam Mendes | 2019 | 119 |   8.2 |   7.0 | 08.02.2020 |
|  4 |               Interstellar |           Cristopher Nolan | 2014 | 169 |   8.6 |   9.8 | 23.03.2024 |
|  5 |             Shutter Island |            Martin Scorsese | 2010 | 138 |   8.3 |   9.5 | 09.01.2024 |
|  6 |          The Boy and Heron |             Hayao Miyazaki | 2023 | 124 |   7.8 |   6.0 | 17.12.2023 |
|  7 |             Spirited Away |             Hayao Miyazaki | 2001 | 125 |   8.5 |   8.1 | 10.12.2021 |
|  9 |   The Wolf of Wall Street |            Martin Scorsese | 2013 | 180 |   8.1 |   8.7 | 30.04.2024 |
| 10 |  The Shawshank Redemption |             Frank Darabont | 1994 | 142 |   9.2 |   9.3 | 18.06.2023 |
| 11 |                   The Dune |           Deni Villeneuve | 2021 | 155 |   8.3 |   7.5 | 08.03.2024 |
| 12 |                  Prisoners |           Deni Villeneuve | 2013 | 153 |   8.2 |   9.5 | 08.12.2023 |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+
Enter the ID of the record you want to edit: 0

Enter the new KPR of Movie: 15
Entered value is not correct. Please try again!
Enter the new KPR of Movie: 9.2

The data updated successfully

Press ENTER to continue
```

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 3

Your selection is FIND THE DATA BY PARAMETERS

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

Choose the parameter
0 - by ID
1 - by Name
2 - by Director
3 - by Year
4 - by Duration
5 - by KPR
6 - by PLR
7 - by Date
Enter the option: 0

Enter the ID: 0

| ID |                      Name |                  Director | Year | Dur |   KPR |   PLR |  Watchdate |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+
|  0 |              Forrest Gump |           Robert Zemeckis | 1994 | 142 |   9.2 |   8.4 | 10.10.2015 |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+

Press ENTER to continue
```

Пример 3 (Сортировка и удаление данных):

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 6

Your selection is SORT THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 1

Choose the parameter
0 - by ID
1 - by Name
Enter the option: 1

Enter the direction of sort (0 - Forward / 1 - Backward): 1

The director list was sorted

Press ENTER to continue
```

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 1

Your selection is SHOW THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 1

| ID |                  Director |
+----+---------------------------+
|  3 |                Sam Mendes |
|  0 |           Robert Zemeckis |
|  4 |           Martin Scorsese |
|  9 |                James Gunn |
|  5 |             Hayao Miyazaki |
|  7 |             Frank Darabont |
|  8 |            Deni Villeneuve |
|  2 |              David Fincher |
|  1 |           Cristopher Nolan |
+----+---------------------------+

Press ENTER to continue
```

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 6

Your selection is SORT THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

Choose the parameter
0 - by ID
1 - by Name
2 - by Director
3 - by Year
4 - by Duration
5 - by KPR
6 - by PLR
7 - by Date
Enter the option: 2

Enter the direction of sort (0 - Forward / 1 - Backward): 0

The movie list was sorted

Press ENTER to continue
```

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 1

Your selection is SHOW THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

| ID |                      Name |                  Director | Year | Dur |   KPR |   PLR |  Watchdate |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+
|  1 |                 Inception |          Cristopher Nolan | 2010 | 148 |   8.7 |  10.0 | 10.05.2017 |
|  4 |              Interstellar |          Cristopher Nolan | 2014 | 169 |   8.6 |   9.8 | 23.03.2024 |
|  2 |                Fight Club |             David Fincher | 1999 | 139 |   8.7 |  10.0 | 12.11.2019 |
| 11 |                  The Dune |           Deni Villeneuve | 2021 | 155 |   8.3 |   7.5 | 08.03.2024 |
| 12 |                 Prisoners |           Deni Villeneuve | 2013 | 153 |   8.2 |   9.5 | 08.12.2023 |
| 10 |  The Shawshank Redemption |            Frank Darabont | 1994 | 142 |   9.2 |   9.3 | 18.06.2023 |
|  6 |         The Boy and Heron |            Hayao Miyazaki | 2023 | 124 |   7.8 |   6.0 | 17.12.2023 |
|  7 |             Spirited Away |            Hayao Miyazaki | 2001 | 125 |   8.5 |   8.1 | 10.12.2021 |
|  5 |            Shutter Island |           Martin Scorsese | 2010 | 138 |   8.3 |   9.5 | 09.01.2024 |
|  9 |   The Wolf of Wall Street |           Martin Scorsese | 2013 | 180 |   8.1 |   8.7 | 30.04.2024 |
|  0 |               Forrest Gump |           Robert Zemeckis | 1994 | 142 |   9.2 |   8.4 | 10.10.2015 |
|  3 |                      1917 |                Sam Mendes | 2019 | 119 |   8.2 |   7.0 | 08.02.2020 |
+----+---------------------------+---------------------------+------+-----+-------+-------+------------+

Press ENTER to continue
```

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 5

Your selection is DELETE THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

Enter the ID of the movie you want to delete: 15
Entered value is not correct. Please try again!
Enter the ID of the movie you want to delete: 12

Movie with entered ID was deleted

Press ENTER to continue
```

```
Choose the option
0 - for EXIT PROGRAM
1 - for SHOW THE DATA
2 - for ADD THE DATA
3 - for FIND THE DATA BY PARAMETERS
4 - for EDIT THE DATA
5 - for DELETE THE DATA
6 - for SORT THE DATA
7 - for REFERENCE
Enter the option: 1

Your selection is SHOW THE DATA

Choose the card-file
0 - for MOVIE card-file
1 - for DIRECTOR card-file
Enter the option: 0

| ID |                     Name |                  Director | Year | Dur |   KPR |   PLR |  Watchdate |
+----+--------------------------+---------------------------+------+-----+-------+-------+------------+
|  1 |                Inception |          Cristopher Nolan | 2010 | 148 |   8.7 |  10.0 | 10.05.2017 |
|  4 |              Interstellar |          Cristopher Nolan | 2014 | 169 |   8.6 |   9.8 | 23.03.2024 |
|  2 |                Fight Club |             David Fincher | 1999 | 139 |   8.7 |  10.0 | 12.11.2019 |
| 11 |                  The Dune |           Deni Villeneuve | 2021 | 155 |   8.3 |   7.5 | 08.03.2024 |
| 10 |  The Shawshank Redemption |            Frank Darabont | 1994 | 142 |   9.2 |   9.3 | 18.06.2023 |
|  6 |         The Boy and Heron |            Hayao Miyazaki | 2023 | 124 |   7.8 |   6.0 | 17.12.2023 |
|  7 |             Spirited Away |            Hayao Miyazaki | 2001 | 125 |   8.5 |   8.1 | 10.12.2021 |
|  5 |            Shutter Island |           Martin Scorsese | 2010 | 138 |   8.3 |   9.5 | 09.01.2024 |
|  9 |   The Wolf of Wall Street |           Martin Scorsese | 2013 | 180 |   8.1 |   8.7 | 30.04.2024 |
|  0 |              Forrest Gump |           Robert Zemeckis | 1994 | 142 |   9.2 |   8.4 | 10.10.2015 |
|  3 |                      1917 |                Sam Mendes | 2019 | 119 |   8.2 |   7.0 | 08.02.2020 |
+----+--------------------------+---------------------------+------+-----+-------+-------+------------+

Press ENTER to continue
```

Заключение

Заголовочный файлы стандартной библиотеки

<stdio.h>:

- fopen – Открытие файла
- fclose – Закрытие файла
- rewind – Перемещение файла на начало

- fgets – Ввод строки
- fprintf – Печати строки в файл
- getchar – Очистка буфера
- sscanf – Преобразование строки в переменные необходимого формата
- puts, printf – Вывод и интерфейс

<stdlib.h>:

- system - Для функции clear_screen, обращение к системе для очистки терминала
- malloc – Динамическое выделение памяти
- free – Освобождение выделенной памяти
- atoi -Преобразование строки в целое число
- atof – Преобразование строки в вещественное число

<string.h>:

- strcmp – Сравнение строк с учётом регистра
- strlen – Получение длины строки
- memcpy – Копирование блока памяти
- strncpy – Копирование строки с ограничением длины
- strcpy – Копирование строки
- strcspn - Определение длины участка строки не содержащего указанные символы
- strstr – Поиск подстроки в строке

<ctype.h>:

- tolower – Преобразование символа в нижний регистр

Вывод

В результате выполнения работы была изучена работа со структурами в языке С и получены практические навыки в создании электронных картотек