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

КУРСОВАЯ РАБОТА

по дисциплине «Программирование»

Тема: Программная реализация электронной картотеки

Студент гр. 9305 _____ Китаев И.А.

Преподаватель _____ Перязева Ю.В.

Санкт-Петербург
2020

ЗАДАНИЕ

НА КУРСОВУЮ РАБОТУ

Студент Китаев И.А.

Группа 9305

Тема работы: программная реализация электронной картотеки

Исходные данные:

Csv-файл, содержащий электронную картотеку и пустой csv-файл

Содержание пояснительной записки:

«Введение», «Электронная картотека», «Программная реализация», «Приложения», «Заключение», «Список использованных источников»

Предполагаемый объем пояснительной записки:

38 страниц.

Дата выдачи задания: 01.04.2020

Дата сдачи реферата: 08.06.2020

Дата защиты реферата: 08.06.2020

Студент _____ Китаев И.А.

Преподаватель _____ Перязева Ю.В.

АННОТАЦИЯ

В данной курсовой работе рассматривается электронная картотека с функционалом. В ней реализованы такие функции, как добавление, удаление, поиск, сортировка и редактирование элементов списка. Помимо этого, должны быть реализованы функции считывания списка из файла и занесение его в файл, программа должна быть устойчива к случайным нажатиям пользователя на клавиши. В содержание работы входят: описание картотеки, описание функций, примеры работы программы и блок-схемы, а также ссылка на репозиторий, где хранится сам код.

SUMMARY

In this course work is considered an electronic file cabinet with functionality. It implements functions such as adding, deleting, searching, sorting and editing list items. In addition, the functions of reading the list from the file and entering it into the file must be implemented, the program must be resistant to accidental user keystrokes. The contents of the work include: a description of the file cabinet, a description of the functions, examples of the program and the flowchart, as well as a link to the repository where the code itself is stored.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель работы:

Написать программу, позволяющую работай с картотекой на определенную тематику.

Задачи

- Изучить поставленную задачу.
- Придумать пути её решения.
- Отобразить алгоритм работы картотеки в виде блок-схемы.
- Написать программную реализацию картотеки.
- Проанализировать полученные результаты.

1. ЭЛЕКТРОННАЯ КАРТОТЕКА

1.1. Тематика и структуры

Электронная картотека основана на теме государств и международных организаций.

Поля таблицы картотеки:

1) Название страны

2) Международная организация

3) Территория страны

4) Население страны в миллионах

5) Население столицы в миллионах

6) Год вступления в организацию

7) ВВП страны

Структура данных:

```
typedef struct states states;
struct states
{
   char *name;        //Название государства
   char *inter_org;   //Название международной организации
   int terr;          //Территория страны
   float pop;         //Население страны в миллионах
   float pop_cap;     // Население столицы в миллионах
   int year_of_entry; //Год вступления в организацию
   int GDP[1];        //ВВП страны
};
```

1.2. Функционал картотеки

Возможности картотеки:

1) Справка

2) Добавление карточки

3) Удаление

4) Поиск

5) Редактирование

6) Сортировка

8) Выход

2. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

2.1. Описание решения

В электронной картотеке в качестве хранения информации были использованы списки.

Добавление осуществляется по введенному id.

Удаление возможно по номеру карточки.

Вывод картотеки осуществлен с начала.

Сложный поиск работает по принципу выбора пользователем конкретного поля и ввода значения поля, которое надо найти.

Редактирование работает сразу для всей карточки, чей id ввел пользователь.

Сортировка возможна по любому из семи полей картотеки.

2.2. Описание структур

Описание головы списка

Имя поля	Тип	Назначение
count	int	количество узлов
*first	Node	адрес первого элемента списка
*last	Node	адрес последнего элемента списка

Описание узла

Имя поля	Тип	Назначение
id	int	порядковый номер
data	states	элемент, содержащий структуру данных
*next	Node	адрес следующего узла
*prev	Node	адрес предыдущего узла

Описание структуры данных

Имя поля	Тип	Назначение
name	char	название страны
inter_org	char	название международной организации
terr	int	территория государства
pop	float	население страны
pop_cap	float	население столицы
year_of_entry	int	год вступления в организацию
GDP	int	ВВП страны

2.3 Описание функций

Функция main

Описание: Является точкой входа в программу.

Прототип: int main()

Пример вызова: main()

Описание переменных: Нет

Возвращаемое значение: 0

Функции для работы с файлом
Функция **split

Описание: Функция для разбиения строки из файла на подстроки по заданному символу. Сами подстроки записываются в строки двумерного массива.

Прототип: char **split(char **mes, char *string)

Пример вызова: mes = split(mes, message)

Описание переменных:

Имя переменной	Тип	Назначение
*str	char	Подстрока
*sep	char	Символ разделителя
i	int	Счетчик символов в строке
flag	int	Флаг
cnt_clear	int	Счетчик строк в двумерном массиве, подлежащим дальнейшей очистке
cnt	int	Счетчик разделителей

Возвращаемое значение: массив строк.

Функция fill_list

Описание: Функция для заполнения структур.

Прототип: void fill_list(Head *q)

Пример вызова: fill_list(ph);

Описание переменных:

Имя переменной	Тип	Назначение
message[MAXLEN]	char	Строка из файла
**mes	char	Массив строк после разбиения строки из файла
id	int	Id узла
*p	Node	Указатель на новый узел в списке
*p1	Node	Указатель на последний узел в списке
*fp	FILE	Указатель на файл
flag	int	Флаг, отслеживающий правильность заполнения узла

Возвращаемое значение:

Функция adding_to_node

Описание: Функция, преобразующая массив строк в узел.
Прототип: Node *adding_to_node(char **arr, int id)
Пример вызова: p = adding_to_node(mes, id)
Описание переменных:

Имя переменной	Тип	Назначение
*p	Node	Указатель на узел

Возвращаемое значение: указатель на узел

Функция clear_array

Описание: Функция для очистки массива строк.

Прототип: void clear_array(char **arr, int cnt)

Пример вызова: clear_array(mes, cnt_clear);

Описание переменных:

Имя переменной	Тип	Назначение
i	int	Индекс строки

Возвращаемое значение:

Функция Writing_list_to_new_doc

Описание: Функция для записи измененного списка во второй файл

Прототип: void Writing_list_to_new_doc(Head* head)

Пример вызова: Writing_list_to_new_doc(head);

Описание переменных:

Имя переменной	Тип	Назначение
*df	FILE	Индекс строки
*node	Node	Указатель на узел
*str	char	Строка со структурой

Возвращаемое значение:

Функция writing_struct_to_string

Описание: Функция, в которой непосредственно происходит запись структуры в строку при помощи sprintf.

Прототип: char *writing_struct_to_string(states* st)

Пример вызова: str = writing_struct_to_string(node->data);

Описание переменных:

Имя переменной	Тип	Назначение
*str	char	Указатель на строку со структурой
sep	char	разделитель

Возвращаемое значение: строка со структурой

Функция structure_filling

Описание: Функция, при помощи которой заполняются поля изменяемой структуры.

Прототип: states *structure_filling()

Пример вызова: st = structure_filling();

Описание переменных:

Имя переменной	Тип	Назначение
*sr	states	Элемент, содержащий адрес структуры

Возвращаемое значение: адрес измененной структуры

Функция reading_float

Описание: Функция считывания вещественного числа с клавиатуры.

Прототип: float reading_float()

Пример вызова: st->pop = reading_float();

Описание переменных:

Имя переменной	Тип	Назначение
read	float	Введенное значение

Возвращаемое значение: введенное число

Функция reading_int

Описание: Функция считывания целого числа с клавиатуры.

Прототип: int reading_int()

Пример вызова: st->terr = reading_int();

Описание переменных:

Имя переменной	Тип	Назначение
read	int	Введенное значение

Возвращаемое значение: введенное число

Функции меню

Функция menu_delete_element

Описание: Функция, удаляющая элемент списка по id.

Прототип: void menu_delete_element(Head* head, int flag)

Пример вызова: menu_delete_element(ph, flag);

Описание переменных:

Имя переменной	Тип	Назначение
*tmp	Node	Указатель на узел
idd	int	Id структуры, которая удаляется

Возвращаемое значение:

Функция menu

Описание: Функция, позволяющая пользователю выбрать интересующий его пункт меню работы с картотекой.

Прототип: void menu(int flag)

Пример вызова: menu(0);

Описание переменных:

Имя переменной	Тип	Назначение
ph	Head	Адрес головы списка
opt	int	Выбранный пользователем пункт меню

Возвращаемое значение:

Функция menu_output_list

Описание: Функция вывода картотеки.

Прототип: void menu_output_list(Head* head, int flag)

Пример вызова: menu_output_list(ph, flag);

Описание переменных:

Возвращаемое значение:

Функция menu_add_element

Описание: Функция добавления узла в список.

Прототип: void menu_add_element(Head* head, int flag)

Пример вызова: menu_add_element(ph, flag);

Описание переменных:

Имя переменной	Тип	Назначение
*tmp	Node	Указатель на узел
*node	Node	Указатель на добавленный узел
uid	int	Id структуры, которая удаляется

Возвращаемое значение:

Функция menu_sort_element

Описание: Функция, позволяющая пользователю выбрать, по какому полю будет осуществлена сортировка. Вызов функции сортировки.

Прототип: void menu_sort_element(Head* head, int flag)

Пример вызова: menu_sort_element(ph, flag);

Описание переменных:

Имя переменной	Тип	Назначение
c	int	Введенный параметр для сортировки

Возвращаемое значение:

Функция menu_change_element

Описание: Функция, изменяющая поля структуры в зависимости от выбора.

Прототип: void menu_change_element(Head* head, int flag)

Пример вызова: menu_change_element(ph, flag);

Описание переменных:

Имя переменной	Тип	Назначение
c	int	Номер структуры для изменения
i	int	Счетчик

Возвращаемое значение:

Функция menu_find_element

Описание: Функция, осуществляющая выбор структур из списка по различным параметрам.

Прототип: void menu_find_element(Head* head, int flag)

Пример вызова: menu_ find_element(ph, flag);

Описание переменных:

Имя переменной	Тип	Назначение

c	int	Введенный параметр для поиска
*node	Node	Адрес первого узла
*head0	Head	Указатель на голову

Возвращаемое значение:

Функции списка

Функция create_head

Описание: Функция, формирующая голову списка.

Прототип: Head *create_head()

Пример вызова: ph = create_head();

Описание переменных:

Имя переменной	Тип	Назначение
*p	Head	Указатель на голову

Возвращаемое значение: указатель на первый элемент списка

Функция create_node

Описание: Функция создает новый узел двусвязного списка. Отводится память под новую запись, устанавливаются значения полей, переданные в аргументах, ссылка на следующий элемент устанавливается в NULL.

Прототип: Node *create_node(Head *q)

Пример вызова: temp = create_node(q)

Описание переменных:

Имя переменной	Тип	Назначение
*temp	Node	Узел списка

Возвращаемое значение: указатель на узел списка

Функция malloc_node

Описание: Функция выделения памяти под узел.

Прототип: void malloc_node(Node *temp)

Пример вызова: malloc_node(temp);

Возвращаемое значение:

Функция add_first_node

Описание: Добавление первого узла в список.

Прототип: void add_first(Head *head , Node *node)

Пример вызова: add_first(q, p);

Возвращаемое значение:

Функция filling

Описание: Добавление узлов в список.

Прототип: void filling(Head *head, Node *p1, Node *p)

Пример вызова: filling(q, p1, p);

Возвращаемое значение:

Функция free_list

Описание: Функция для очистки памяти, выделенной под список

Прототип: void free_list(Head *head)

Пример вызова: free_list(ph);

Описание переменных:

Имя переменной	Тип	Назначение
*p	Node	Указатель на первый узел
*tmp	Node	Указатель на узел

Возвращаемое значение:

Функция free_del

Описание: Удаление узла из памяти

Прототип: void free_del(Node *tmp)

Пример вызова: free_del(tmp);

Возвращаемое значение:

Функция decrease_id

Описание: Уменьшение id у элементов, идущих после удаленного

Прототип: void decrease_id(Head *head, Node *node)

Пример вызова: decrease_id(head, tmp -> next -> next);

Возвращаемое значение:

Функция increase_id

Описание: Увеличение id элементов идущих после добавленного в список элемента.

Прототип: void increase_id(Head *head, Node *node)

Пример вызова: increase_id(head, node -> next);

Возвращаемое значение:

Функция get_node

Описание: Функция выделяющая память под новый узел и осуществляющая заполнение узла с клавиатуры.

Прототип: Node *get_node()

Пример вызова: node = get_node();

Описание переменных:

Имя переменной	Тип	Назначение
* new_node	Node	Указатель на новый узел

Возвращаемое значение: указатель на новый, заполненный с клавиатуры, узел.

Функция scan

Описание: Заполнение структуры в узле с клавиатуры

Прототип: void scan(Node *temp)

Пример вызова: scan(new_node);

Возвращаемое значение:

Функция sort_by_name

Описание: Осуществляет лексикографическую проверку двух строк

Прототип: int sort_by_name(Node *node)

Пример вызова: sort[1] = sort_by_name;

Возвращаемое значение: положительное число, если код первого отличающегося символа в первой строке больше кода символа на той же позиции во второй строке

Функция sort_by_inter_org

Описание: Осуществляет лексикографическую проверку двух строк

Прототип: int sort_by_inter_org (Node *node)

Пример вызова: sort[2] = sort_by_inter_org;

Возвращаемое значение: положительное число, если код первого отличающегося символа в первой строке больше кода символа на той же позиции во второй строке

Функция sort_by_territory

Описание: Сравнение территорий стран

Прототип: int sort_by_territory (Node *node)

Пример вызова: sort[3] = sort_by_territory;

Возвращаемое значение: положительное число, если условие верно, иначе 0

Функция sort_by_population

Описание: Сравнение населений стран

Прототип: int sort_by_population (Node *node)

Пример вызова: sort[4] = sort_by_population;

Возвращаемое значение: положительное число, если условие верно, иначе 0

Функция sort_by_cap_popul

Описание: Сравнение населений столиц

Прототип: int sort_by_cap_popul (Node *node)

Пример вызова: sort[5] = sort_by_cap_popul;

Возвращаемое значение: положительное число, если условие верно, иначе 0

Функция sort_by_year

Описание: Сравнение по году

Прототип: int sort_by_year (Node *node)

Пример вызова: sort[6] = sort_by_year;

Возвращаемое значение: положительное число, если условие верно, иначе 0

Функция sort_by_GDP

Описание: Сравнение по ВВП

Прототип: int sort_by_GDP (Node *node)

Пример вызова: sort[7] = sort_by_GDP;

Возвращаемое значение: положительное число, если условие верно, иначе 0

Функция sort_cards

Описание: Сортировка пузырьком по возрастанию.

Прототип: void sort_cards(Head *head, int c)

Пример вызова: sort_cards(head, c);

Описание переменных:

Имя переменной	Тип	Назначение
* node	Node	Указатель на первый узел
*tab	states	Указатель на структуру данных

Возвращаемое значение:

Функция delete_node

Описание: Удаление узла

Прототип: void delete_node(Head *head, Node *node)

Пример вызова: delete_node(head, node);

Возвращаемое значение:

Функция find_by_terr

Описание: Нахождение структур по параметру

Прототип: Head* find_by_terr(Head *head)

Пример вызова: find[3] = find_by_terr;

Описание переменных:

Имя переменной	Тип	Назначение
* node	Node	Указатель на первый узел

Возвращаемое значение: указатель на голову

Функция find_by_popul

Описание: Нахождение структур по параметру

Прототип: Head* find_by_popul (Head *head)

Пример вызова: find[4] = find_by_popul;

Описание переменных:

Имя переменной	Тип	Назначение
* node	Node	Указатель на первый узел

Возвращаемое значение: указатель на голову

Функция find_by_cap_popul

Описание: Нахождение структур по параметру

Прототип: Head* find_by_cap_popul (Head *head)

Пример вызова: find[5] = find_by_cap_popul;

Описание переменных:

Имя переменной	Тип	Назначение
* node	Node	Указатель на первый узел

Возвращаемое значение: указатель на голову

Функция find_by_year

Описание: Нахождение структур по диапазону

Прототип: Head* find_by_year (Head *head)

Пример вызова: find[6] = find_by_year;

Описание переменных:

Имя переменной	Тип	Назначение
* node	Node	Указатель на первый узел
min	int	Минимальное значение
max	int	Максимальное значение

Возвращаемое значение: указатель на голову

Функция find_by_GDP

Описание: Нахождение структур по диапазону

Прототип: Head* find_by_GDP (Head *head)

Пример вызова: find[7] = find_by_GDP;

Описание переменных:

Имя переменной	Тип	Назначение
* node	Node	Указатель на первый узел
min	int	Минимальное значение
max	int	Максимальное значение

Возвращаемое значение: указатель на голову

Функция find_by_name

Описание: Нахождение структур по названию страны

Прототип: Head* find_by_name (Head *head)

Пример вызова: find[1] = find_by_name;

Описание переменных:

Имя переменной	Тип	Назначение
* node	Node	Указатель на первый узел
*value	char	Считанная строка

Возвращаемое значение: указатель на голову

Функция find_by_inter_org

Описание: Нахождение структур по названию международной организации

Прототип: Head* find_by_inter_org (Head *head)

Пример вызова: find[2] = find_by_inter_org;

Описание переменных:

Имя переменной	Тип	Назначение
* node	Node	Указатель на первый узел
*value	char	Считанная строка

Возвращаемое значение: указатель на голову

Функции вывода

Функция print_header

Описание: Вывод заголовка таблицы.

Прототип: void print_header()

Пример вызова: print_header();

Описание переменных:

Возвращаемое значение:

Функция output_list

Описание: Функция для вывода списка.

Прототип: void output_list(Head *q)

Пример вызова: output_list(q);

Возвращаемое значение: нет

Функция output_menu

Описание: Вывод главного меню.

Прототип: void output_menu ()

Пример вызова: output_menu ();

Описание переменных:

Возвращаемое значение:

Функция information

Описание: Вывод справки.

Прототип: void information(Head *head)

Пример вызова: information(ph);

Описание переменных:

Возвращаемое значение:

Функция parameters

Описание: Вывод справки.

Прототип: void parameters()

Пример вызова: parameters();

Описание переменных:

Возвращаемое значение:

Функция node_out

Описание: Функция для вывода узла в виде таблицы

Прототип: void node_out(Node *temp)

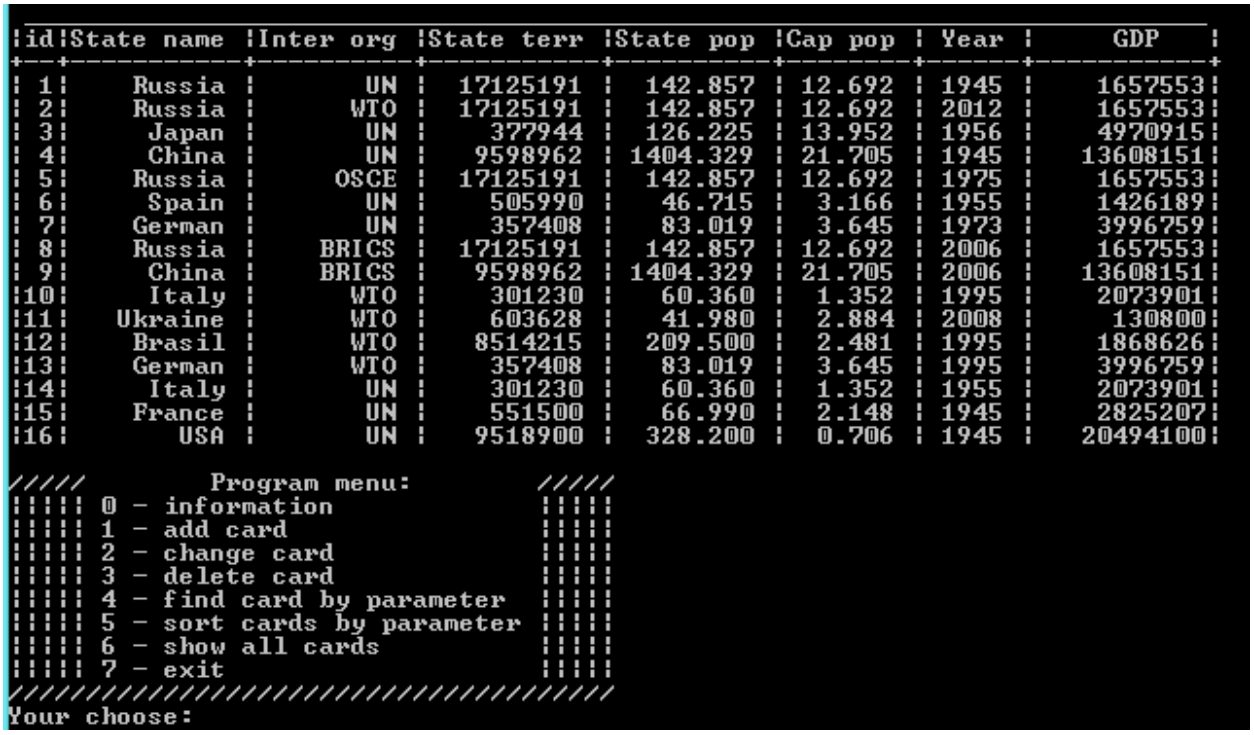
Пример вызова: node_out(temp);

Возвращаемое значение:

2.4 Пример работы программы

Основное меню

Вывод списка, считанного из файла



Добавление узла по номеру

```
//////////////////////////////////////
Your choose:
1

Fill new struct

Enter the state name:
1
Enter the international organization:
1
Enter the territory of the country:
1
Enter the population of the country:
1
Enter the population of the capital:
1
Enter the year this country joined the inter. org:
1
Enter state establishment GDP: 1

Enter element`s id
1
```

id	State name	Inter org	State terr	State pop	Cap pop	Year	GDP
1	1	1	1	1.000	1.000	1	1
2	Russia	UN	17125191	142.900	12.700	1945	1657553
3	Russia	WTO	17125191	142.900	12.700	2012	1657553
4	Japan	UN	377944	126.200	14.000	1956	4970915
5	China	UN	9598962	1404.300	21.700	1945	13608151
6	Russia	OSCE	17125191	142.900	12.700	1975	1657553
7	Spain	UN	505990	46.700	3.200	1955	1426189
8	German	UN	357408	83.000	3.600	1973	3996759
9	Russia	BRICS	17125191	142.900	12.700	2006	1657553
10	China	BRICS	9598962	1404.300	21.700	2006	13608151
11	Italy	WTO	301230	60.400	1.400	1995	2073901
12	Ukraine	WTO	603628	42.000	2.900	2008	130800
13	Brasil	WTO	8514215	209.500	2.500	1995	1868626
14	German	WTO	357408	83.000	3.600	1995	3996759
15	Italy	UN	301230	60.400	1.400	1955	2073901
16	France	UN	551500	67.000	2.100	1945	2825207
17	USA	UN	9518900	328.200	0.700	1945	20494100

```
Structure successfully added!
```

Удаление узла по номеру

```
//////////////////////////////////////
Your choose:
3
Enter the node id
1
```

id	State name	Inter org	State terr	State pop	Cap pop	Year	GDP
1	Russia	UN	17125191	142.900	12.700	1945	1657553
2	Russia	WTO	17125191	142.900	12.700	2012	1657553
3	Japan	UN	377944	126.200	14.000	1956	4970915
4	China	UN	9598962	1404.300	21.700	1945	13608151
5	Russia	OSCE	17125191	142.900	12.700	1975	1657553
6	Spain	UN	505990	46.700	3.200	1955	1426189
7	German	UN	357408	83.000	3.600	1973	3996759
8	Russia	BRICS	17125191	142.900	12.700	2006	1657553
9	China	BRICS	9598962	1404.300	21.700	2006	13608151
10	Italy	WTO	301230	60.400	1.400	1995	2073901
11	Ukraine	WTO	603628	42.000	2.900	2008	130800
12	Brasil	WTO	8514215	209.500	2.500	1995	1868626
13	German	WTO	357408	83.000	3.600	1995	3996759
14	Italy	UN	301230	60.400	1.400	1955	2073901
15	France	UN	551500	67.000	2.100	1945	2825207
16	USA	UN	9518900	328.200	0.700	1945	20494100

Нахождение структур по параметру и сохранение полученного списка для дальнейшей работы

```
//////////////////////////////////////
Your choose:
4
Please choose parameter:
1 - Country name(char)
2 - International organization(char)
3 - Territory(int)
4 - Country population(float)
5 - Capital population(float)
6 - Year of entry(int)
7 - GDP of country(int)
Your choose:
1
Enter country name:
Russia
_____________________________________________________________________________
|id|State name |Inter org |State terr |State pop |Cap pop | Year |     GDP    |
+--+-----------+----------+-----------+----------+--------+------+------------+
| 1|    Russia |       UN |  17125191 |  142.900 | 12.700 | 1945 |    1657553|
| 2|    Russia |      WTO |  17125191 |  142.900 | 12.700 | 2012 |    1657553|
| 5|    Russia |     OSCE |  17125191 |  142.900 | 12.700 | 1975 |    1657553|
| 8|    Russia |    BRICS |  17125191 |  142.900 | 12.700 | 2006 |    1657553|
Do you want to add another condition to search?
||   1 - YES    ||
||   0 - NO     ||
0
Do you want to save result?
||   1 - YES    ||
||   0 - NO     ||
1
```

Сортировка по параметру в порядке возростания

```
//////////////////////////////////////
Your choose:
5
Please choose parameter:
1 - Country name(char)
2 - International organization(char)
3 - Territory(int)
4 - Country population(float)
5 - Capital population(float)
6 - Year of entry(int)
7 - GDP of country(int)
Your choose:
6
```

```
_____________________________________________________________________________
|id|State name |Inter org |State terr |State pop |Cap pop | Year |     GDP    |
+--+-----------+----------+-----------+----------+--------+------+------------+
| 1|    Russia |       UN |  17125191 |  142.900 | 12.700 | 1945 |    1657553|
| 2|    Russia |     OSCE |  17125191 |  142.900 | 12.700 | 1975 |    1657553|
| 3|    Russia |    BRICS |  17125191 |  142.900 | 12.700 | 2006 |    1657553|
| 4|    Russia |      WTO |  17125191 |  142.900 | 12.700 | 2012 |    1657553|
```

Заключение

В ходе проделанной работы была реализована электронная картотека на языке программирования Си для хранения различных данных о странах. Картотека позволяет сортировать карточки (по возрастанию), удалять элементы, добавлять с клавиатуры, редактировать и проводить поиск карточек по выбранному значению и полю. При её реализации были изучены: принципы работы с двусвязными линейными списками, программная разработка, реализация и отладка конечной программы. Были получены ценные практические знания о синтаксисе и правилах написания кода на языке Си.

Список используемых источников

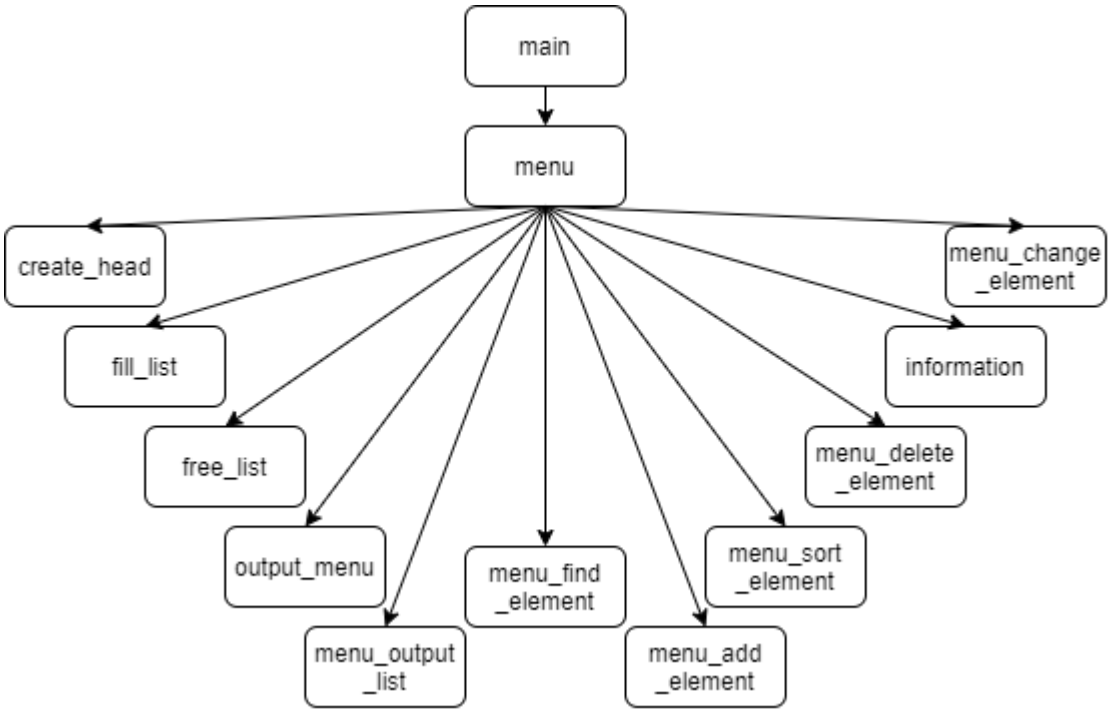
https://prog-cpp.ru/data-dls/

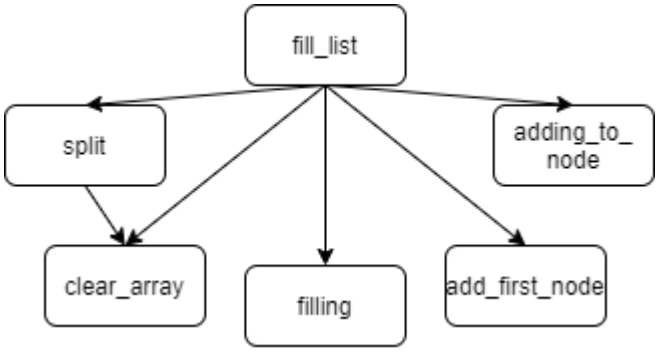
https://learnc.info/adt/double_linked_list.html

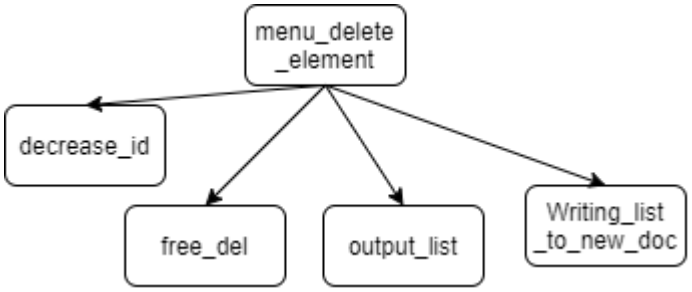
https://prog-cpp.ru/c-files/

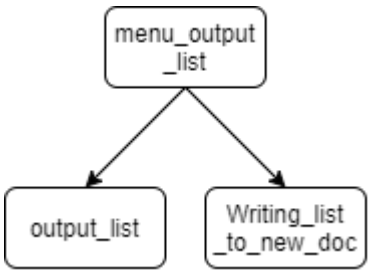
3. Приложения

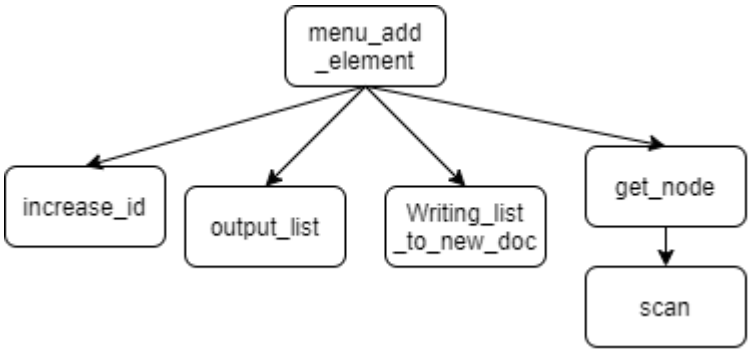
A. Схема вызова функций

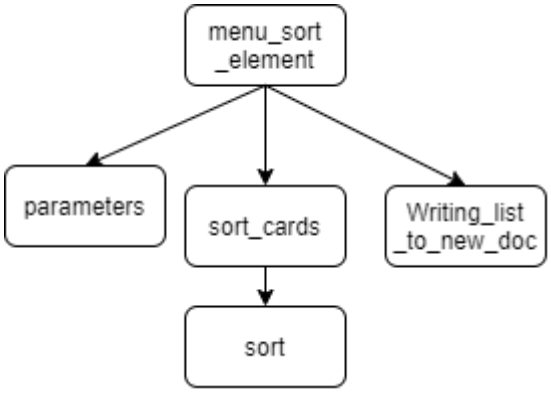


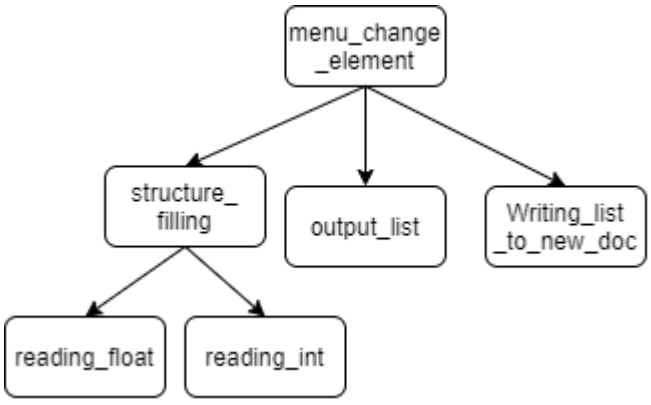


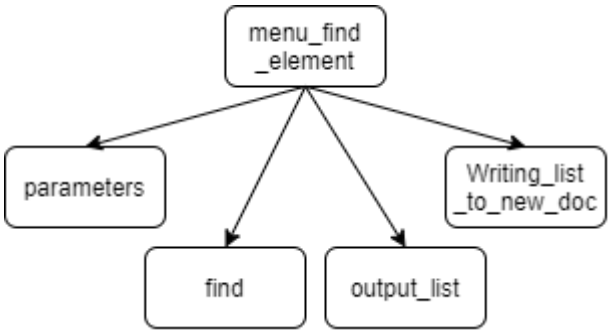


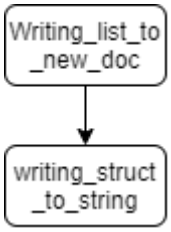

menu_output
_list
output_list
Writing_list
_to_new_doc

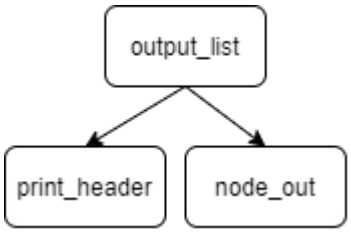

menu_add
_element
increase_id
output_list
Writing_list
_to_new_doc
get_node
scan


menu_sort
_element
parameters
sort_cards
Writing_list
_to_new_doc
sort


menu_change
_element
structure_
filling
output_list
Writing_list
_to_new_doc
reading_float
reading_int

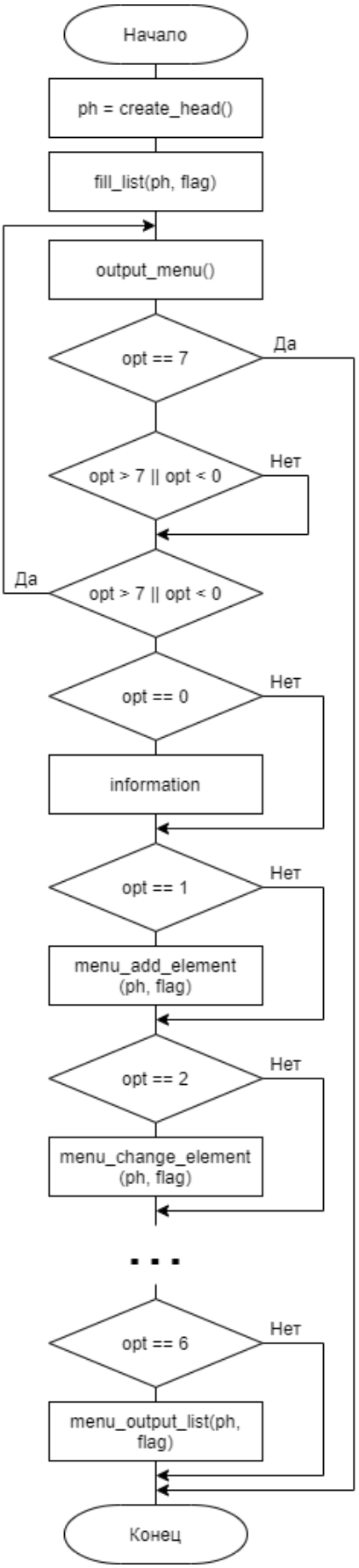

menu_find
_element
parameters
find
output_list
Writing_list
_to_new_doc


Writing_list_to
_new_doc
writing_struct
_to_string

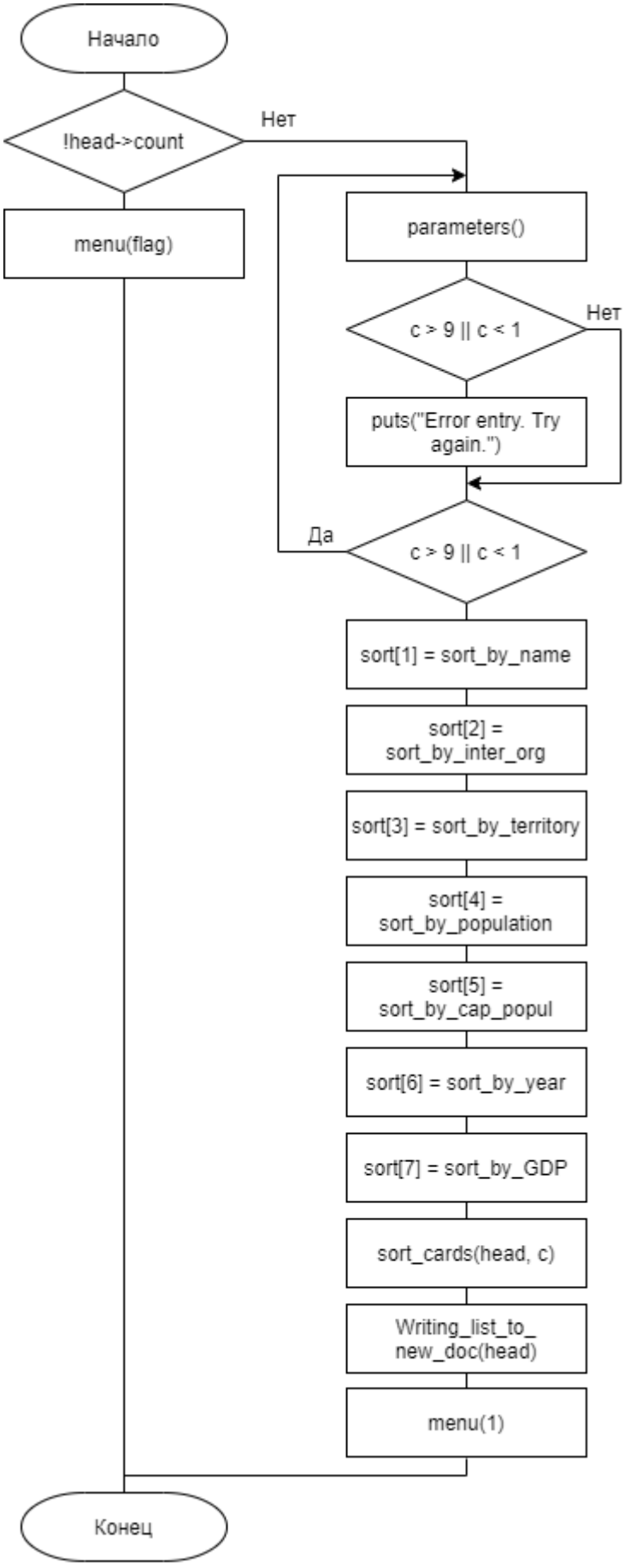

output_list
print_header
node_out

В. Схема функций

Функция menu.



Функция menu_sort_element



C. Текст программы

Ссылка на github: https://github.com/onekitaev/Laboratory-works.-Kitaev