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

Кафедра вычислительной техники

Отчет по лабораторной работе №11-12
по дисциплине «Программирование»
Тема: Двусвязные и кольцевые списки

Студент гр. 9305 _____ Китаев И.А.

Преподаватель _____ Перязева Ю.В.

Санкт-Петербург
2020

Содержание:

Введение

Данная лабораторная работа выполнена с целью получения практических навыков в разработке алгоритма и написании программы на языке Си для знакомства с синтаксисом, а также особенностями работы с двусвязными и кольцевыми списками.

Задание

Для выбранной предметной области создать динамический массив структур, содержащих характеристики объектов предметной области.

Обязательный набор полей:

1) динамический массив символов, включая пробелы (name)
2) произвольный динамический массив символов
3) числовые поля типов int и float (не менее двух полей каждого типа)
4) поле с числовым массивом.

Написать программу, обеспечивающую начальное формирование массива структур при чтении из файла (текст с разделителями — CSV) с последующим возможным дополнением элементов массива при вводе с клавиатуры. Следует использовать указатели на структуры и указатели на функции обработки массива в соответствии с вариантом задания.

Во всех случаях, когда при поиске записей результат отсутствует, следует вывести сообщение

Задание варианта

Разработать подалгоритм добавления элемента в односвязный кольцевой список после заданного по номеру элемента и в «конец» списка, если такого элемента нет.

Разработать подалгоритм ввода с клавиатуры значений информационных полей и добавления в двусвязный список нового элемента перед последним элементом и в начало списка, если список пуст.

Постановка задачи и описание решения

Исходные данные: Значение символьного поля, таблица полей предметной области.

Выходные данные: Сначала на экран будет выведен начальный список, затем пользователю будет предложено выбрать действия, которые бы он хотел проделать со списком. Набор действий зависит от номера лабораторной работы, но общий набор действий таков: добавление структуры в начало списка, в конец, удаление структуры по выбранному id, добавление структуры перед последним элементом списка, добавление структуры после элемента, чей id был введен, вывод списка, выход из программы.

Для начала объявим структуру states, которая выглядит следующим образом:

```
typedef struct states states;
//Структура из лабораторной 9
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 data_of_app[3]; //Дата основания государства
};
```

Создадим структуры элемента списка и головы списка:

```
typedef struct Head Head;
typedef struct Node Node;
struct Node
{
  int id; //Id элемента
```

```
    states *data; //данные
    Node *next; // Указатель на следующий элемент списка
    Node *prev; // Указатель на предыдущий элемент списка
};
// Структура головы списка
struct Head
{
    int count; // Количество элементов в списке
    Node *first; // Указатель на первый элемент в списке
    Node *last; // Указатель на последний элемент списка
};
```

Разделим код программы на модули.

Теперь рассмотрим работу программы пошагово. В функции main создается указатель с типом Head, вызывается функция create_head, где формируется голова списка. После этого в main вызывается функция fill_list, внутри которой мы открываем файл с данными, построчно считываем, разбиваем каждую строку по разделителю и записываем в двумерный массив при помощи функции split, переносим данные в узел с помощью функции adding_to_node и связываем его с точкой начала (головой), используя такие функции, как add_first и filling. Очищаем двумерный массив функцией clear_array и проделываем те же действия, пока не считаем все строки из файла. В конце выводим получившийся список в виде таблицы.

Далее внутри main вызывается функция command_selecting. Внутри нее выводится список команд, а затем считывается выбор пользователя, в зависимости от которого, производится дальнейший вызов функций. Важно заметить, что помимо, так называемых, командных функций, есть еще две очень важные. Это - charToInt и str_len, благодаря им при ошибке ввода со стороны пользователя, программа не сломается, так как вводится строка, а затем при помощи первой преобразуется в int. Вторая же следит, чтобы строка

имела длину 1. В конце вызывается функция free_list, которая очищает память, выделенную под список.

Описание структур

Описание структуры states

Имя поля	Тип	Назначение
name	char	Название государства
inter_org	char	Название международной организации
terr	int	Территория страны
pop	float	Население страны в миллионах
pop_cap	float	Население столицы в миллионах
year_of_entry	int	Год вступления в организацию
data_of_app	int	Дата основания государства

Описание структуры Head

Имя поля	Тип	Назначение
count	int	Количество элементов в списке
*first	Node	Указатель на первый элемент в списке
*last	Node	Указатель на последний элемент списка

Описание структуры Node

Имя поля	Тип	Назначение
id	int	Id элемента
*data	states	Данные
*next	Node	Указатель на следующий элемент списка
*prev	Node	Указатель на предыдущий элемент смиска

Структура вызовов функций

Функция main

Описание: Является точкой входа в программу.

Прототип: int main()

Пример вызова: main()

Описание переменных:

Имя переменной	Тип	Назначение
*ph	Head	Указатель на голову списка

Возвращаемое значение:

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 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 print_header

Описание: Вывод заголовка таблицы.

Прототип: void print_header()

Пример вызова: print_header();

Описание переменных: нет

Возвращаемое значение:

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 output_list

Для двусвязного списка:

Описание: Функция для вывода списка.

Прототип: void output_list(Head *q)

Пример вызова: output_list(q);

Возвращаемое значение: нет

Для циклического списка

Описание: Функция для вывода списка.

Прототип: void output_list(Head *q)

Пример вызова: output_list(q);

Описание переменных:

Имя переменной	Тип	Назначение
i	int	Счетчик узла
*temp	Node	Указатель на узел

Функция malloc_node

Описание: Функция выделения памяти под узел.

Прототип: void malloc_node(Node *temp)

Пример вызова: malloc_node(temp);

Возвращаемое значение: нет

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: нет

Функция create_node

Описание: Функция создает новый узел связного списка. Отводится память под новую запись, устанавливаются значения полей, переданные в аргументах, ссылка на следующий елемент 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 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 add_first

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 node_out

Описание: Функция для вывода узла в виде таблицы

Прототип: void node_out(Node *temp)

Пример вызова: node_out(temp);

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:

Функции для пользовательского меню

Функция command_selecting

Описание: Функция выбора команды. Пользователь вводит номер действия, которое бы он хотел выполнить со списком. В зависимости от выбора, данная функция обращается к функции, отвечающей за то или иное действие.

Прототип: void command_selecting(Head *q)

Пример вызова: command_selecting(ph);

Описание переменных:

Имя переменной	Тип	Назначение
s[MAXLEN]	char	Строка, введенная пользователем при выборе команды
f	int	Строка, записанная в тип int
*temp	Node	Указатель на узел

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 add_first1

Описание: Функция выделяющая память под новый узел и записывающая его в начало списка

Прототип: Node * add_first1 (Head *q)

Пример вызова: add_first1 (q);

Описание переменных:

Имя переменной	Тип	Назначение
*temp	Node	Указатель на новый узел, добавленный в начало списка

Возвращаемое значение: указатель на первый узел в списке

Функция add_last

Описание: Функция выделяющая память под новый узел и записывающая его в конец списка

Прототип: Node *add_last(Head *q)

Пример вызова: add_last(q);

Описание переменных:

Имя переменной	Тип	Назначение
*temp	Node	Указатель на новый узел, добавленный в конец списка

Возвращаемое значение: указатель на последний узел в списке

Функция scan

Описание: Заполнение структуры в узле с клавиатуры

Прототип: void scan(Node *temp)

Пример вызова: scan(new_node);

Возвращаемое значение:

Функция str_len

Описание: Нахождение длины введенной строки

Прототип: int str_len(char *s)

Пример вызова: str_len(s)

Описание переменных:

Имя переменной	Тип	Назначение
r	int	Индекс символа строки

Возвращаемое значение: длина строки

Функция charToInt

Описание: Перевод строки в число при помощи таблицы ASCII

Прототип: int charToInt(char numeric)

Пример вызова: f = charToInt(s[0])

Возвращаемое значение: полученное число

Функция delete_elem

Описание: Функция удаления узла по индексу. Находится элемент с введенным индексом, затем указатель next у предыдущего элемента принимает адрес элемента, идущего через один, а выбранный узел перестает ссылаться на следующий элемент и стирается из памяти.

Прототип: void delete_elem(Head *head)

Пример вызова: delete_elem(q);

Описание переменных:

Имя переменной	Тип	Назначение
idd	int	Id, по которому удаляется узел
*tmp	Node	Указатель на узел
*tmp1	Node	Указатель на элемент, идущий после удаляемого

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 free_del

Описание: Удаление узла из памяти

Прототип: void free_del(Node *tmp)

Пример вызова: free_del(tmp);

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:

Функции для двусвязного списка

Функция insert_before_last

Описание: Функция добавления элемента перед последним узлом в списке. Если список пуст, то элемент добавляется в начало списка.

Прототип: void insert_before_last(Head *head)

Пример вызова: insert_before_last(q);

Описание переменных:

Имя переменной	Тип	Назначение
*tmp	Node	Указатель на узел
*node	Node	Указатель на элемент, введенный с клавиатуры

Возвращаемое значение:

Функции для циклического списка

Функция insert_after

Описание: Функция добавления элемента после элемента по id.

Прототип: void insert_before_last(Head *head)

Пример вызова: insert_before_last(q);

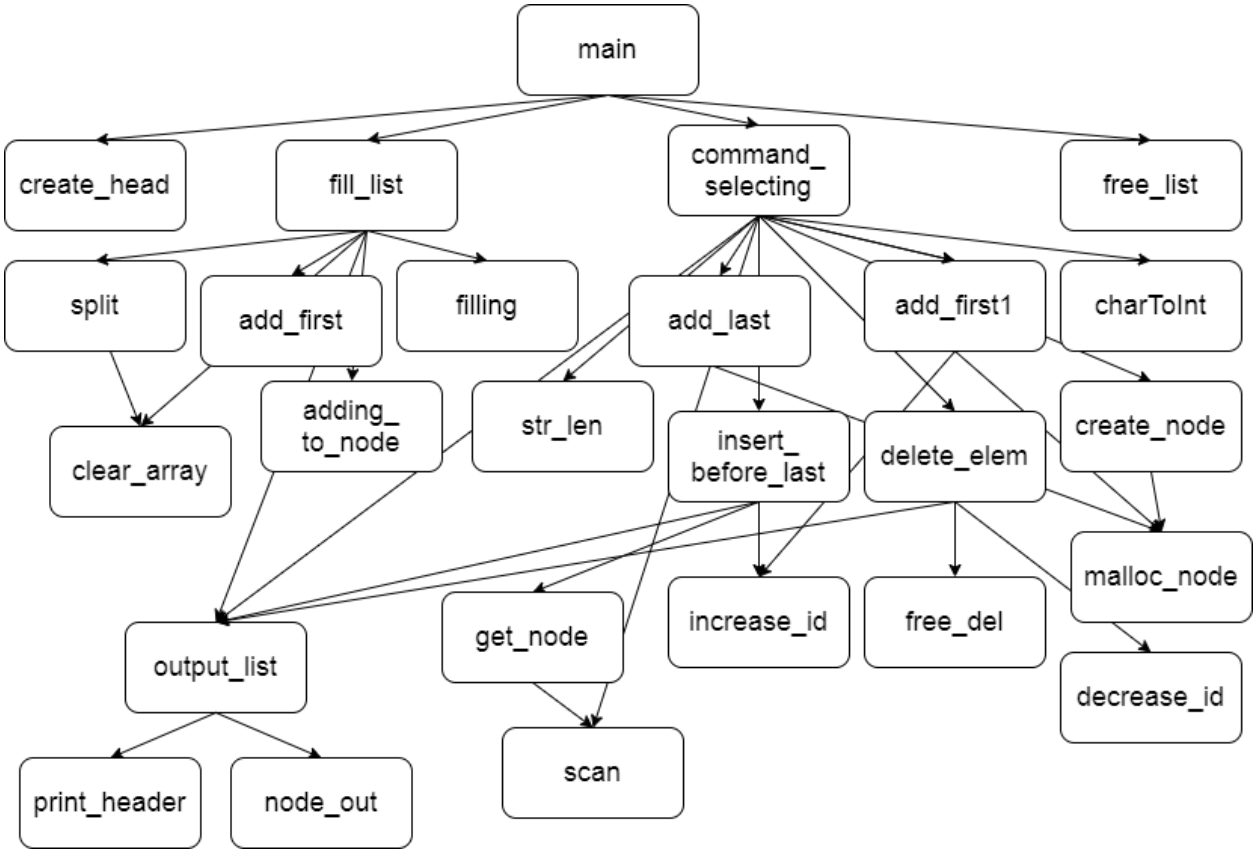
Описание переменных:

Имя переменной	Тип	Назначение
id	int	Номер элемента, введенный пользователем
*tmp	Node	Указатель на первый элемент всписке

Возвращаемое значение:

Схема вызовов функций

Для двусвязного списка



Для циклического схема выглядит аналогично, только на месте функции insert_before_last должна стоять insert_after.

Текст программы

Текст программы представлен на ресурсе GitHub:

https://github.com/onekitaev/Laboratory-works.-Kitaev/tree/master/lab11

https://github.com/onekitaev/Laboratory-works.-Kitaev/tree/master/lab12

Примеры работы программы для двусвязного списка

1. **Файловые данные:**

Russia;UN;17125191;142.857;12.692;1945;881;12;25

Russia;WTO;17125191;142.857;12.692;1995;881;12;25

Japan;UN;377944;126.225;13.952;1956;660;2;11

China;UN;9598962;1404.329;21.705;1945;1911;10;1

Russia;OSCE;17125191;142.857;12.692;1975;881;12;25

Spain;UN;505990;46.715;3.166;1955;1479;11;7

German;UN;357408;83.019;3.645;1973;843;8;10

Russia;BRICS;17125191;142.857;12.692;2006;881;12;25

China;BRICS;9598962;1404.329;21.705;2006;1911;10;1

Ввод:

В этом примере рассматривается работа трех функций: Добавление элемента в начало списка, добавление в конец и удаление элемента по id.

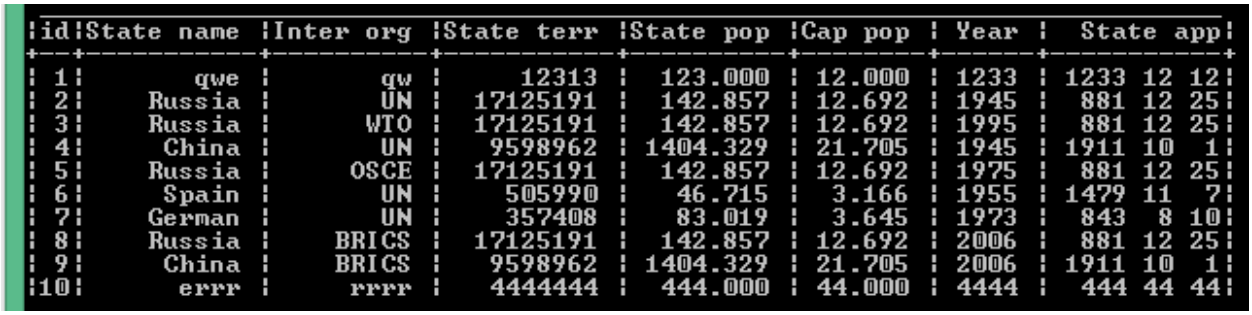
```
|id|State name |Inter org |State terr |State pop |Cap pop | Year |  State app|
+--+-----------+----------+-----------+----------+--------+------+-----------+
| 1|     Russia |       UN |  17125191 |  142.857 | 12.692 | 1945 |  881 12 25|
| 2|     Russia |      WTO |  17125191 |  142.857 | 12.692 | 1995 |  881 12 25|
| 3|      Japan |       UN |    377944 |  126.225 | 13.952 | 1956 |  660  2 11|
| 4|      China |       UN |   9598962 | 1404.329 | 21.705 | 1945 | 1911 10  1|
| 5|     Russia |     OSCE |  17125191 |  142.857 | 12.692 | 1975 |  881 12 25|
| 6|      Spain |       UN |    505990 |   46.715 |  3.166 | 1955 | 1479 11  7|
| 7|     German |       UN |    357408 |   83.019 |  3.645 | 1973 |  843  8 10|
| 8|     Russia |    BRICS |  17125191 |  142.857 | 12.692 | 2006 |  881 12 25|
| 9|      China |    BRICS |   9598962 | 1404.329 | 21.705 | 2006 | 1911 10  1|

1 - Add first
2 - Add last
3 - Delete element
4 - Insert before the last
5 - List output
0 - Exit
1
Enter the state name:
qwe
Enter the international organization:
qw
Enter the territory of the country:
12313
Enter the population of the country:
123
Enter the population of the capital:
12
Enter the year this country joined the inter. org:
1233
Enter state establishment data (XXXX XX XX): 1233 12 12
```

```
2
Enter the state name:
errr
Enter the international organization:
rrrr
Enter the territory of the country:
4444444
Enter the population of the country:
444
Enter the population of the capital:
44
Enter the year this country joined the inter. org:
4444
Enter state establishment data (XXXX XX XX): 444 44 44
```

```
3
Enter the node id
4
```

Вывод программы:



id	State name	Inter org	State terr	State pop	Cap pop	Year	State app
1	qwe	qw	12313	123.000	12.000	1233	1233 12 12
2	Russia	UN	17125191	142.857	12.692	1945	881 12 25
3	Russia	WTO	17125191	142.857	12.692	1995	881 12 25
4	China	UN	9598962	1404.329	21.705	1945	1911 10 1
5	Russia	OSCE	17125191	142.857	12.692	1975	881 12 25
6	Spain	UN	505990	46.715	3.166	1955	1479 11 7
7	German	UN	357408	83.019	3.645	1973	843 8 10
8	Russia	BRICS	17125191	142.857	12.692	2006	881 12 25
9	China	BRICS	9598962	1404.329	21.705	2006	1911 10 1
10	errr	rrrr	4444444	444.000	44.000	4444	444 44 44

2. **Файловые данные:**

Russia;UN;17125191;142.857;12.692;1945;881;12;25

Russia;WTO;17125191;142.857;12.692;1995;881;12;25

Japan;UN;377944;126.225;13.952;1956;660;2;11

China;UN;9598962;1404.329;21.705;1945;1911;10;1

Russia;OSCE;17125191;142.857;12.692;1975;881;12;25

Spain;UN;505990;46.715;3.166;1955;1479;11;7

German;UN;357408;83.019;3.645;1973;843;8;10

Russia;BRICS;17125191;142.857;12.692;2006;881;12;25

China;BRICS;9598962;1404.329;21.705;2006;1911;10;1

Ввод:

В данном примере показана работа функции по добавлению нового элемента перед последним элементом списка.

```
|id|State name |Inter org |State terr |State pop |Cap pop | Year |  State app|
+--+----------+----------+-----------+----------+--------+------+-----------+
| 1|    Russia |       UN |  17125191 |  142.857 | 12.692 | 1945 |  881 12 25|
| 2|    Russia |      WTO |  17125191 |  142.857 | 12.692 | 1995 |  881 12 25|
| 3|     Japan |       UN |    377944 |  126.225 | 13.952 | 1956 |  660  2 11|
| 4|     China |       UN |   9598962 | 1404.329 | 21.705 | 1945 | 1911 10  1|
| 5|    Russia |     OSCE |  17125191 |  142.857 | 12.692 | 1975 |  881 12 25|
| 6|     Spain |       UN |    505990 |   46.715 |  3.166 | 1955 | 1479 11  7|
| 7|    German |       UN |    357408 |   83.019 |  3.645 | 1973 |  843  8 10|
| 8|    Russia |    BRICS |  17125191 |  142.857 | 12.692 | 2006 |  881 12 25|
| 9|     China |    BRICS |   9598962 | 1404.329 | 21.705 | 2006 | 1911 10  1|

1 - Add first
2 - Add last
3 - Delete element
4 - Insert before the last
5 - List output
0 - Exit
4

Fill new struct:

Enter the state name:
qwerty
Enter the international organization:
weewee
Enter the territory of the country:
23333
Enter the population of the country:
233.233
Enter the population of the capital:
23.333
Enter the year this country joined the inter. org:
2323
Enter state establishment data (XXXX XX XX): 2323 23 23
```

Вывод программы:

```
|id|State name |Inter org |State terr |State pop |Cap pop | Year |  State app|
+--+----------+----------+-----------+----------+--------+------+-----------+
| 1|    Russia |       UN |  17125191 |  142.857 | 12.692 | 1945 |  881 12 25|
| 2|    Russia |      WTO |  17125191 |  142.857 | 12.692 | 1995 |  881 12 25|
| 3|     Japan |       UN |    377944 |  126.225 | 13.952 | 1956 |  660  2 11|
| 4|     China |       UN |   9598962 | 1404.329 | 21.705 | 1945 | 1911 10  1|
| 5|    Russia |     OSCE |  17125191 |  142.857 | 12.692 | 1975 |  881 12 25|
| 6|     Spain |       UN |    505990 |   46.715 |  3.166 | 1955 | 1479 11  7|
| 7|    German |       UN |    357408 |   83.019 |  3.645 | 1973 |  843  8 10|
| 8|    Russia |    BRICS |  17125191 |  142.857 | 12.692 | 2006 |  881 12 25|
| 9|    qwerty |   weewee |     23333 |  233.233 | 23.333 | 2323 | 2323 23 23|
|10|     China |    BRICS |   9598962 | 1404.329 | 21.705 | 2006 | 1911 10  1|
```

3. **Файловые данные:**

Russia;UN;17125191;142.857;12.692;1945;881;12;25

Russia;WTO;17125191;142.857;12.692;1995;881;12;25

Japan;UN;377944;126.225;13.952;1956;660;2;11

China;UN;9598962;1404.329;21.705;1945;1911;10;1

Russia;OSCE;17125191;142.857;12.692;1975;881;12;25

Spain;UN;505990;46.715;3.166;1955;1479;11;7

German;UN;357408;83.019;3.645;1973;843;8;10

Russia;BRICS;17125191;142.857;12.692;2006;881;12;25

China;BRICS;9598962;1404.329;21.705;2006;1911;10;1

Ввод:

В этом примере продемонстрирована работа функции “Insert before the last” в условиях, когда все элементы списка были удалены. В условии сказано, если список пуст, то элемент добавляется в начало списка. Предварительно удалим все данные из списка при помощи функции “Delete element”.

```
 The list is empty!

1 - Add first
2 - Add last
3 - Delete element
4 - Insert before the last
5 - List output
0 - Exit
4

Fill new struct:

Enter the state name:
evhjfe
Enter the international organization:
weff
Enter the territory of the country:
32332
Enter the population of the country:
233
Enter the population of the capital:
23
Enter the year this country joined the inter. org:
2323
Enter state establishment data (XXXX XX XX): 2323 23 23
```

Вывод программы:

```
¦id¦State name ¦Inter org ¦State terr ¦State pop ¦Cap pop ¦ Year ¦  State app¦
+--+-----------+----------+-----------+----------+--------+------+-----------+
¦ 1¦    evhjfe ¦     weff ¦     32332 ¦  233.000 ¦ 23.000 ¦ 2323 ¦ 2323 23 23¦
```

Примеры работы программы для циклического списка

1. **Файловые данные:**

Russia;UN;17125191;142.857;12.692;1945;881;12;25

Russia;WTO;17125191;142.857;12.692;1995;881;12;25

Japan;UN;377944;126.225;13.952;1956;660;2;11

China;UN;9598962;1404.329;21.705;1945;1911;10;1

Russia;OSCE;17125191;142.857;12.692;1975;881;12;25

Spain;UN;505990;46.715;3.166;1955;1479;11;7

German;UN;357408;83.019;3.645;1973;843;8;10

Russia;BRICS;17125191;142.857;12.692;2006;881;12;25

China;BRICS;9598962;1404.329;21.705;2006;1911;10;1

Ввод:

Показана работа функций удаления элементов списка и добавлению элементов в начало списка, в конец и после элемента по id.

```
1 - Add first
2 - Add last
3 - Delete element
4 - Insert after
5 - List output
0 - Exit
1
Enter the state name:
qwqwqw
Enter the international organization:
qw
Enter the territory of the country:
122222
Enter the population of the country:
12
Enter the population of the capital:
2
Enter the year this country joined the inter. org:
1222
Enter state establishment data (XXXX XX XX): 1212 12 12
```

```
1 - Add first
2 - Add last
3 - Delete element
4 - Insert after
5 - List output
0 - Exit
2
Enter the state name:
err
Enter the international organization:
er
Enter the territory of the country:
555
Enter the population of the country:
555
Enter the population of the capital:
555
Enter the year this country joined the inter. org:
55
Enter state establishment data (XXXX XX XX): 555 55 55
```

```
Enter the element's id:
5
```

id	State name	Inter org	State terr	State pop	Cap pop	Year	State app
1	qwqwqw	qw	122222	12.000	2.000	1222	1212 12 12
2	Russia	UN	17125191	142.857	12.692	1945	881 12 25
3	Russia	WTO	17125191	142.857	12.692	1995	881 12 25
4	Japan	UN	377944	126.225	13.952	1956	660 2 11
5	Russia	OSCE	17125191	142.857	12.692	1975	881 12 25
6	Spain	UN	505990	46.715	3.166	1955	1479 11 7
7	German	UN	357408	83.019	3.645	1973	843 8 10
8	Russia	BRICS	17125191	142.857	12.692	2006	881 12 25
9	China	BRICS	9598962	1404.329	21.705	2006	1911 10 1
10	ewfew	we	0	0.000	0.000	0	0 0 0
11	err	er	555	555.000	555.000	55	555 55 55

```
4
Enter the state name:
8888
Enter the international organization:
8888
Enter the territory of the country:
888
Enter the population of the country:
88
Enter the population of the capital:
88
Enter the year this country joined the inter. org:
8888
Enter state establishment data (XXXX XX XX): 888 88 88

Enter the element's id:
7
```

id	State name	Inter org	State terr	State pop	Cap pop	Year	State app
1	qwqwqw	qw	122222	12.000	2.000	1222	1212 12 12
2	Russia	UN	17125191	142.857	12.692	1945	881 12 25
3	Russia	WTO	17125191	142.857	12.692	1995	881 12 25
4	Japan	UN	377944	126.225	13.952	1956	660 2 11
5	Russia	OSCE	17125191	142.857	12.692	1975	881 12 25
6	Spain	UN	505990	46.715	3.166	1955	1479 11 7
7	German	UN	357408	83.019	3.645	1973	843 8 10
8	8888	8888	888	88.000	88.000	8888	888 88 88
9	Russia	BRICS	17125191	142.857	12.692	2006	881 12 25
10	China	BRICS	9598962	1404.329	21.705	2006	1911 10 1
11	ewfew	we	0	0.000	0.000	0	0 0 0
12	err	er	555	555.000	555.000	55	555 55 55

Заключение

При выполнении лабораторной работы были получены практические навыки в разработке алгоритма и написании программы на языке Си для знакомства с синтаксисом, а также особенностями работы с двусвязными и кольцевыми списками.