Техническое задание
на разработку программного кода к рубежному контролю №2

Реализовать динамическую структуру данных, состоящих из элементов одного типа – **кольцевой односвязный список целых чисел** на языке С (С99).

Код программы должен быть реализован в предоставленном файле main.c.

В исходный файл main.c должен быть добавлен код следующих компонентов.

1. Тип элементов списка -- `l_elem`.

```
typedef struct list_elem
{
// Here's your code
}l_elem;
```

2. Тип переменной, обеспечивающей доступ к списку -- `list1`

```
typedef struct
{
    // Here's your code
}list1;
```

3. Код реализации функций основных операций со списком, заданных протототипами.

Во всех функциях формальный параметр `list_ptr` – указатель на переменную типа `list1`, обеспечивающая доступ к списку.

Во всех функциях, где есть формальный параметр `int index`, значение `index` указывает порядковый номер (индекс) элемента списка относительно первого (головного) элемента списка. Индекс первого (головного) элемента списка равен нулю.

Значение `index` должно быть по модулю строго меньше количества элементов в списке. Иначе, функция выводит в консоль сообщение, что индекс находится за границами списка.

```
void init_list(list1 *list_ptr);
```

Инициализация переменной типа `list1` – задает начальные значения переменной для пустого списка.

```
void append(list1 *list_ptr, int value);
```

Добавляет в конец списка значение переменной `value`

```
void print_list(list1 *list_ptr);
```

С новой строки выводит в консоль значения элементов списка, разделенных пробелом в одну строку. Строка начинается с надписи `print_list():`.

Например,

```
print_list(): 0 1 2 3 4 5 6 7 8 9
```

```
void clear_list(list1 *list_ptr);
```

Удаляет все элементы из списка. С новой строки выводит в консоль значения удаляемых элементов списка, разделенных пробелом в одну строку. Строка начинается с надписи `clear_elements:`

Например,

```
clear_elements: 10 -10 2 3 10 5 -10 7 8 -10
```

```
int get_by_index(list1 *list_ptr, int index);
```

Возвращает значение элемента списка с индексом `index.`

Если индекс находится за границами списка, функция выводит в консоль сообщение и возвращает значение равное нулю. Сообщение начинается с надписи `get_by_index(x):`, где `x` – значение формального параметра `index`.

Например,

```
get_by_index(10): index 10 out of range
```

```
void delete_by_index(list1 *list_ptr, int index);
```

Функция удаляет элемент из списка с индексом, указанным в формальном параметре `index`.

Если индекс находится за границами списка, функция выводит в консоль сообщение. Сообщение начинается с надписи `delete_by_index(x):`, где `x` – значение формального параметра `index`.

Например,

```
delete_by_index(10): index 10 out of range
```

```
void delete_all_odd(list1 *list_ptr);
```

Функция удаляет все элементы из списка, у которых значение нечетно.

```
void delete_all_even(list1 *list_ptr);
```

Функция удаляет все элементы из списка, у которых значение четно.

Функция `int main()` содержит код, тестирующий работу вышеперечисленных функций. Этот код менять ***нельзя****!!!*

В файле main.c используются два заголовочных файла.

```
#include <stdio.h>
#include <stdlib.h>
```

Другие заголовочные файлы использовать ***нельзя!!!***

Разработчиком могут быть добавлены собственные функции необходимые для реализации вышеуказанных функций.

Результаты разработки кода программы должны быть представлены в виде архивного zip-файла.

Имя файла должно соответствовать шаблону:

iu42N*RK2.zip (rt221*RK2.zip для студентов факультета РТ)

где N – номер группы студента;

* -- последовательность латинских букв – транскрипция фамилии студента.

Например, iu423VetrovRK2.zip -- файл студента Ветрова из группы ИУ4-23.

Zip-файл должен содержать только файл main.c. Файл не должен быть вложен в другой каталог. Иначе задание считается невыполненным.

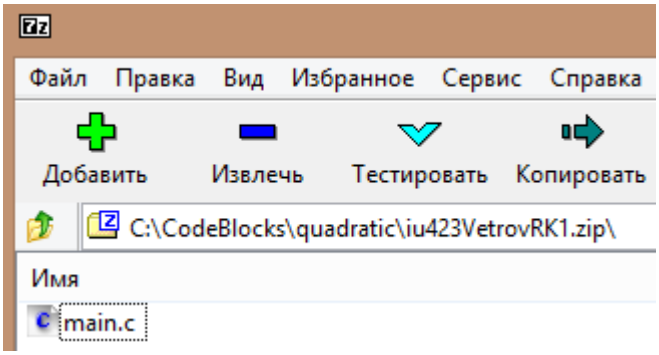

Рис.1 Содержимое zip-файла