**Лабораторная работа №6**

«Имеется однонаправленное кольцо. Элементом кольца является указатель на стек (учебная группа) и число учащихся в группе. Элементом стека – указатель на текстовую информацию (фамилия). В кольце найти группу с минимальным количеством учащихся и добавить ее стек к стеку следующей группы. Выбранный элемент кольца удалить».

struct stack //стек  
{

char lastname[30]; //фамилия студента  
stack\* next; //указатель на следующий элемент

};

struct ring //однонаправленное кольцо  
{

int count; //кол-во студентов в группе  
stack\* group; //группа   
ring\* next; //указатель на следующий элемент

};

**Блок-схема main()**





**Функция check()**

**1. НАЧАЛО**

2. Входные значения: *head* — элемент-точка входа в кольцо.

Промежуточные значения: *node* — удаляемый элемент кольца, *k* — кол-во студентов очередной группы (элемента кольца), *str* — строка, переносимая из стека удаляемого элемента в стек соседнего.

Выходные значения: *head* — элемент-точка входа в кольцо.

3. Если в кольце один элемент, то перейти к **пункту 11**.

4. Анализ содержимого кольца, на которое указывает *head*. Поиск элемента с наименьшим *k.* Найденный элемент записать в *node.*

5. Если *head* и *node* указывают на один и тот же элемент, то переместить *head* на следующий элемент кольца.

6. Удалить элемент стека из *node* с помощью функции *pop()*, поместив хранящуюся строку в *str*.

7. В стек элемента кольца, следующего за *node*, с помощью функции *push()* добавить элемент, содержащий строку *str*.

8. Если в стеке элемента *node* остались элементы, то переходим к **пункту 6**, иначе — к **пункту 9**.

9. Исключение *node* из кольца путём присвоения элементу, находящемуся перед *node*, указатель на элемент, находящийся после *node*.

10. Очистка памяти *node*.

**11. КОНЕЦ**