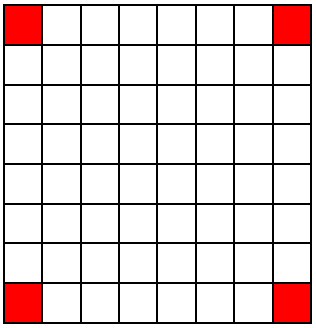
Coursework 2: 2D Led Array

In this task, we assume that an 8×8 LED array is controlled by 2 words specified in led_array. Each bit in this memory space controls the on/off state of one LED with 1 turns on an LED and 0 turns off an LED. Each byte of the led_array maps to one row of LEDs and the least significant bit of this byte controls the right-most LED of this row.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0th byte
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	1st byte
.	.	.	.	.	.	.	Bit 16	2nd byte
.	.	.	.	.	.	.	Bit 24	3rd byte
.	.	.	.	.	.	.	Bit 32	4th byte
.	.	.	.	.	.	.	Bit 40	5th byte
.	.	.	.	.	.	.	Bit 48	6th byte
Bit 63	Bit 62	Bit 61	Bit 60	Bit 59	Bit 58	Bit 57	Bit 56	7th byte

One example is shown below. 0x81 is the first byte of led_array, the first bit is 1 and the last bit is also 1, this will turn on the top left and top right LEDs.

```
led_array DCD 0x00000081, 0x81000000
```



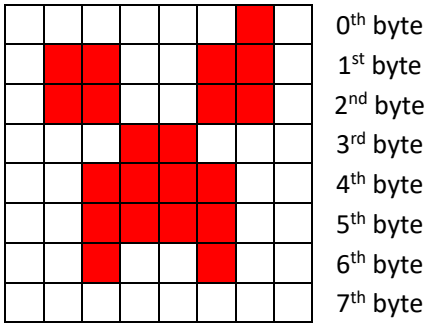
If there were issues found in this coursework specification after its initial release, it will be updated and announced on LMO.

Task 1: (60%)

In this task, we assume that only one LED on the edge of this led array is turned on. Implement the function “spin_single”, which rotates the position of the light-emitting LED clock-wise once.

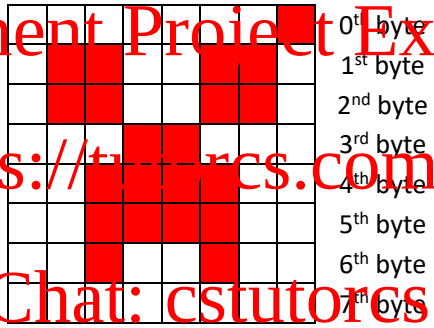
For example, if the original state of led_array is:

```
led_array DCD 0x18666602, 0x00243C3C
```



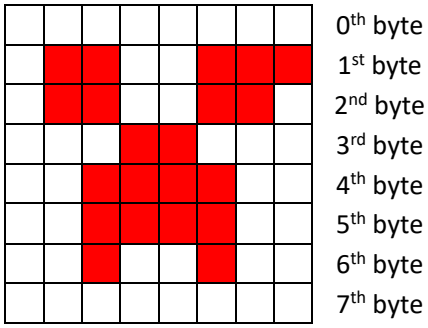
After calling function “spin_single” once, led_array should become:

```
led_array DCD 0x18666601, 0x00243C3C
```



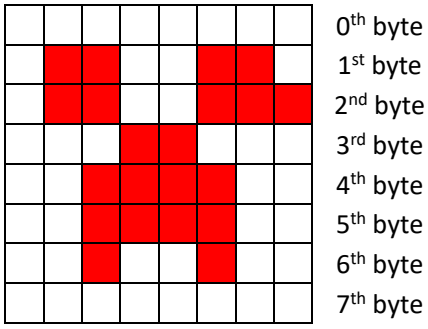
Calling it once more, led_array should become:

```
led_array DCD 0x18666700, 0x00243C3C
```



Calling it once more, led_array should become:

```
led_array DCD 0x18676600, 0x00243C3C
```

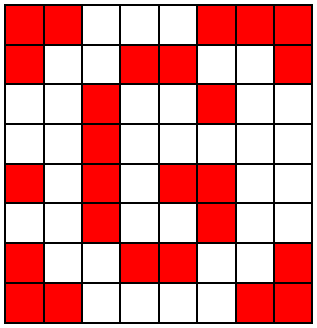


There are 4 important requirements/restrictions on your program:

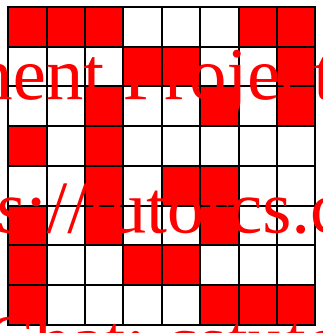
1. Your function should not modify the state of any LEDs in the middle. Only the LEDs on the edge can be changed.
2. You are not allowed to use DCD (or similar assembler directives like DCW or DCB) to declare new memory blocks.
3. Your function must return to its caller when finished.
4. You should not assume a fixed initial position of the light-emitting LED on the edge. It may appear at any position when spin_single is called.

Task 2: (40%)

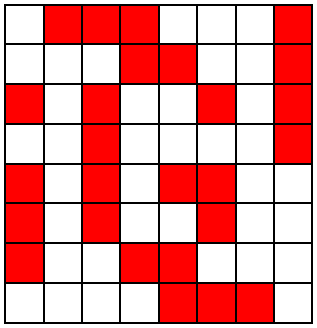
Implement the function called “spin_multi”, which rotates light-emitting LEDs on the edge clock-wise once. There’s no fixed pattern for light-emitting LEDs on the edge. Given the example state of led_array below:



After calling spin_multi once, it should become:



Calling spin_multi again, it should become:



Just like task 1, There are 4 important requirements/restrictions on your program:

1. Your function should not modify the state of any LEDs in the middle. Only the LEDs on the edge can be changed.
2. You are not allowed to use DCD (or similar assembler directives like DCW or DCB) to declare new memory blocks.
3. Your function must return to its caller when finished.
4. You should not assume a fixed initial position of the light-emitting LED sequences on the edge.