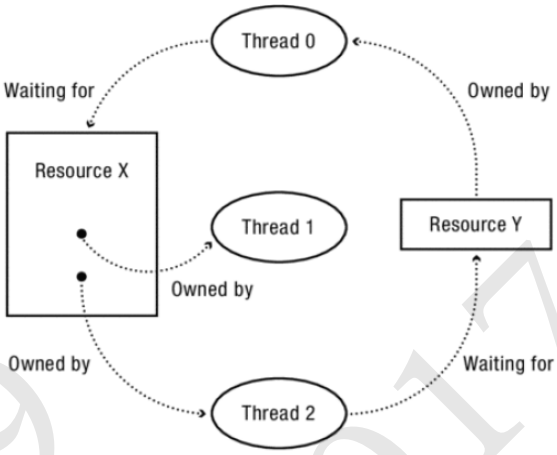
SJSU CS 149 HW5 Fall 2017

REMINDER: Each homework (including programming question) is individual. "Every single byte must come from you." Cut&paste from others is not allowed.
[This assignment does not have programming question, except the optional extra credit question.]

1. (15 pts) Given resource type X with two instances, resource type Y with a single instance, and three threads. As illustrated in the resource allocation graph,
   - Thread 0 owns resource type Y's instance and is waiting for resource type X.
   - Thread 1 owns one instance of resource type X.
   - Thread 2 owns the other instance of resource type X and is waiting for resource type Y.

a. Are Coffman conditions true in this state?
b. Is there any deadlock? Why or why not?



2. (15 pts) Both segment table and page table are used to translate from logical address to physical address. But the structures of these tables are different; Each entry in a segment table is {base, length} while each entry in a page table is {frame#}. Why the differences? Can we record base in a page table? Why there is no length in a page table?

3. (40 pts) Consider the following page reference string: 7, 2, 3, 1, 2, 5, 3, 4, 6, 7, 7, 1, 0, 5, 4
Assuming demand paging with 3 frames, fill in the table to indicate pages in the frames, page fault if any, and total number of page faults, for the following page replacement algorithms.

a. LRU

Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reference	7	2	3	1	2	5	3	4	6	7	7	1	0	5	4
Frame 0	7	7	7												
Frame 1	X	2	2												
Frame 2	X	X	3												
Page fault? (Y/N)															

Total page faults =_____

b. FIFO

Time	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Reference	7	2	3	1	2	5	3	4	6	7	7	1	0	5	4
Frame 0	7	7	7												
Frame 1	X	2	2												
Frame 2	X	X	3												
Page fault? (Y/N)															

Total page faults =_____

4. (30 pts) On a machine with 16 bytes page size, given the following page table for a process, and four of these 8 entries are mapped to page frames. Frame 0 starts at physical address 0. (All numbers given are in decimal.)

Page number	Frame number

0	3
1	0
2	Not in main memory
3	Not in main memory
4	2
5	Not in main memory
6	1
7	Not in main memory

a. Make a list of all virtual address ranges (in decimals, byte-level) that would cause page faults.
b. What are the corresponding physical addresses (in decimals, byte-level) of the following virtual addresses (in decimals, byte-level)
    - 0
    - 17
    - 31
    - 32
    - 100

Submit the following file:

- `CS149_HW5_YourName_L3SID` (.pdf, .doc, or .docx), which includes answers to all questions.

The ISA and/or instructor leave feedback to your homework as comments and/or annotated comment. To access annotated comment, click "view feedback" button. For details, see the following URL::

http://guides.instructure.com/m/4212/l/106690-how-do-i-use-the-submission-details-page-for-an-assignment

Optional - Extra credit (up to additional 15 points on top of 100 points)

5. Assume that a system has a 32-bit virtual address with N-KB page size (where N >= 1, and 1KB = 1024 bytes). Write a C program that accepts two command line parameters, the first one being the value of N (in decimal notation) and the second one being a virtual address in decimal notation, and have it output the page size, the page number and offset for the given virtual address. For example,

```
./vaddr 4 19985
```

The program should output

```
Virtual address translation by <YourName> <L3SID>
Page size = 4096, virtual address 19985 => page number = 4, offset = 3601
```

Replace YourName and L3SID with your own name, and last 3 digits of your SID. Test your program with the following four runs

```
./vaddr 1 19981
./vaddr 2 19982
./vaddr 4 19984
./vaddr 8 19988
```

and capture screenshots of your program execution.

Submission:

- At the end of the regular report, include screenshots of those four runs. Note each screenshot must include "Virtual address translation by …".
- Your source code, named as `vaddr_<YourName>_<L3SID>.c`