Spring 2018 Operating Systems Qualifying Exam

CODE__________________________________

This is a closed book test.

Correct, clear and precise answers receive full marks

Please start a new page for each question.

There are five (5) questions, **20 points each**

1. We wish to increase the size of a RAID array by adding one new disk. Is it easier to add the new disk if the array's RAID level is 4 or 5, and why? You may assume the new disk contains all zeroes.

   Answer: RAID 4. Since the parity disk is the same, adding another data disk, will not affect the resulting value of the parity sector. Since the added sector is 0, then there is no need to calculate the parity value. The sectors are then added to the free list and any updates to the sectors would update the parity sector. In RAID 5, the position of the parity sector is distributed which means that shuffling of the parity sector would have to be made. Thus, effectively, at least the size of the new disk of data would have to be shuffled.

2. In a paged operating system, when a page is to be brought into main memory from disc, it must be placed in a page frame. If there is no empty page frame available, a page must be evicted from main memory to make room for the new page. The choice of which frame to select is made by a page replacement algorithm. Describe, excluding the working set model, two page replacement algorithms. Please include the performance challenge of your two examples.

   Answer: There are several answers to include, Least Recently Used, FIFO (very bad), LIFO (usually better than LIFO), Working set, CLOCK,

3. Write a Producer/Consumer solution using a non-counting semaphore and threads. You should have a shared global buffer where data is placed for production and taken when consumer. You are to ensure that all race conditions are addressed while not blocking processes that don't need to access the shared resources.

```
semaphore notfull = 1;
semaphore notempty = 0;
semaphore buffer_mutex=1;
size=0;

procedure producer()

{    while (true)

    {    item = produceItem();
         down(notfull);
         down(buffer_mutex);

             putItemIntoBuffer(item);
             size ++;
              if (size!= BUF_SIZE) up(notfull);
             up(notempty);

       up(buffer_mutex);

    }
}

procedure consumer()
{

    while (true)
 {
   down(notempty);
   down(buffer_mutex);
       item = removeItemFromBuffer();
       size--;
       if (size > 0)  up(notempty);
      up(notfull);
  up(bufer_mutex);

   consumeItem(item);

    }

}
```

a) What sort of possible speedup will you get when utilizing User-Level Threads?

No real possible speed up because User-Level threads runs on one processor. Since there is no I/O, non-blocking I/O is immaterial

b) What sort of possible speedup will you get when utilizing Kernel-Level Threads?

With Kernel threads, it is possible to get speed up. This now depends on the Operating system and its process scheduling algorithm.

4. Consider a system with 3 physical frames of memory that is given the following page memory reference sequence:

   1, 3, 6, 7, 1, 3, 6, 7, 1, 3, 6, 7

   What is the number of page faults that would occur for each of the following page replacement algorithms?

   a. An optimal page replacement algorithm

   b. LRU

   c. 2nd chance clock replacement

Answer:

a)   FRAME A: 1
FRAME B: 3
FRAME C: 6, 7, 6, 7
6 faults

b) FRAME A:1, 7, 6, 3
FRAME B: 3, 1, 7, 6
FRAME C: 6, 3, 1, 7
12 faults

c) FRAME A: 1, 7, 6, 3
FRAME B: 3, 1, 7, 6
FRAME C: 6, 3, 1, 7
12 faults – because when we do second chance, we clear the access bits and nothing resets them

5. A system is composed of four processes, p1 through p4, and three types of consumable resources, R1 through R3. There is one unit each of R1 and R3 available.

   p1 requests a unit of R1 and a unit of R3.
   p2 produces a unit of R1 and a unit of R3 and requests one unit of R2.
   p3 requests a unit of R1 and a unit of R3.
   p4 produces a unit of R2 and requests one unit of R3.

   Show the consumable resource graph to represent the system state. Which, if any, of the processes are deadlocked in this state?

   In a consumable resource graph edges from process to resource indicate a request, edges from resource to process are producer edges.

   The system will stall. There are 3 requests for R3, yet there is only 2 total units of R3. Ultimately p4 will stall waiting on R3.