CS342 Operating Systems - Fall 2020
Homework #3

Assigned: Dec 26, 2020, Saturday.
Due date: Jan 3, 2020, Sunday, 23:59 Document version: 1.0

1) Design a monitor that will allow either a P process or a Q process to access a file. Processes may pop-up at any time. Solve the same question with mutex and condition variables. This is the problem asked in Midterm 1.

2) A resource R has 20 instances. There are 2 processes P1 and P2. P1's max demand is 12 and P2's max demand is 14 (instances). From deadlock avoidance perspective, how many allocations are safe? For example, allocation (0,0) is safe. Allocation (1, 2), which indicates that P1 is allocated 1 instance and P2 is allocated 2 instances at the moment, is also safe state. But (12, 14) may not be safe. Check it.

3) Consider the following request and allocation matrices. Is there a deadlock at the moment? Assume initially (without allocations yet) there are 5 A, 4 B, and 4 C in the system (i.e., it is the existing vector).

```
      Alloc   Request
      A B C   A B C
P1    0 1 0   2 0 1
P2    1 1 1   3 0 1
P3    1 0 1   0 1 0
P4    2 0 1   0 3 3
P5    1 1 0   1 0 2
```

4) Assume in a computer supporting paging, virtual addresses are 36 bits long. Page size is 16 KB, no matter which paging structure is used. Assume a page table entry is 8 bytes long, no matter which paging structure is used. The computer has 4 GB of RAM (physical memory).

We have three processes (A, B, C) in the system at the moment. The virtual memory layouts of the processes are like the following:
--- A is using 64 MB from the start of its VM and 32 MB from the end. The rest of its VM is unused.
--- B is using 128 MB from the start and 64 MB from the end. The rest of its VM is unused.
--- C is using 32 MB from the start and 32 MB from the end. The rest of its VM is unused.
Find out the physical memory required to hold the page table information for these three processes in total, for the following paging structures (i.e., methods).

a) One level page table.
b) Two level page table with address division scheme (11, 11, 14).
c) Inverted page table.

6) Assume there are 4 frames that can be used by a process. The page reference string of the process is shown below.

3 7 4 2 5 2 1 4 5 8 1 8 3 5 1 8 5 2 3 5 4

Show the memory state (what pages are in 4 frames and the R bit value of each page– if applicable) after each page reference for the following page replacement algorithms. Also mark if a reference is a page fault or not. At the end, count the page faults.

a) LRU
b) Optimal
c) FIFO
d) R bit. Assume reference bits are cleared after every 6th reference.
e) Second chance. Assume reference bits are cleared after every 6th reference.

7) Assume we have a disk that is of size 64 GB. Block size is 16 KB. FAT entry size is 8 bytes. How many entries do we need for that FAT (assume we need an entry for every disk block). Assume disk pointer size is 8 bytes. How many disk blocks are occupied by the FAT? What is the maximum file size? If instead of FAT, two level index allocation would be used to keep track of the data blocks of a file, what would be the maximum size of the file (assuming disk pointer size is 8 bytes).

8) Assume combined indexed allocation scheme is used in a file system (like in Unix/Linux). For each file, we have an inode. In the inode, we can keep 10 direct block addresses at most. Block size is 1 KB. Disk address (pointer) size if 4 bytes long. What can be the maximum file size? How many index blocks (of all kind, like first level, second level, third level index blocks), we need for files A, B, C, of sizes 3 KB, 200 KB, 32 MB and 1 GB.