Solver: Chen Yinya

1) a) i) False. In many to one mapping, when one thread blocks, other user threads mapped to the same kernel thread are also blocked.

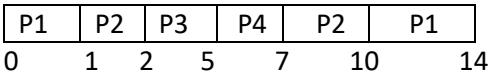
ii) True. There is a cycle in the graph and only one instance per resource. R2 requires R1 to execute while R1 is given to P1.

iii) False. Kernel mode is a hardware operation mode while admin/root privileges still belong to the user mode.

iv) False, it denotes how many more processes can hold this semaphore before the next process will get blocked.

v) True. PCB keeps information associated with each process, which includes base and limit register values.

b) i) SRTF:



ii) Turnaround time is the amount of time to execute a process, from time of submission to time of completion.
P1 = 14 – 0 = 14
P2 = 10 – 1 = 9
P3 = 5 – 2 = 3
P4 = 7 – 3 = 4

Waiting time = turnaround time – burst
P1 = 14 – 5 – 2 = 7
P2 = 9 – 4 – 2 = 3
P3 = 3 – 3 = 0
P4 = 4 – 2 = 2

c) Initial:
lock = true;
Entry:
while(!TestAndSelect(&lock));
critical section
Exit:
Lock = true;
Remainder section

d) Yes. Because the system calls are atomic, they ensure mutual exclusion.

2) a) i) True. A thread is called a lightweight process to emphasise the fact that a thread is like a process but is more efficient and uses fewer resources.

ii) True. This is the definition of a trap.

iii) False. Multiprogramming means there are one or more programs loaded in main memory which are ready to execute. Only one program at a time can get the CPU for executing its instructions. The main idea of multiprogramming is to maximise the use of CPU time. Multithreading is an execution model that allows a single process to have multiple threads running concurrently within the "context" of that process.

iv) True, only if there are multiple cores, each core can run one process.

v) False. Context is stored in registers (PCB).

b) i) Total number = available + allocation
A: 4 + 1 + 1 = 6
B: 3 + 1 + 1 = 5

ii) Yes. There exists a safety sequence for all process to complete: P1-P4-P3-P2-P5

iii) The minimum matrix in Max is (3,3). Assuming (3,3) is enough, there does not exist a safety sequence. Next, we try (3,4), there exists a safety sequence and so does (4,3). Thus, the minimum values required is (3,4) or (4,3).

iv) Assume the request is granted. Available matrix becomes (1,0). Max matrix for P2 becomes (1,2). However, the Available matrix is smaller than any one of the matrices in Max column. Thus, there does not exist a safety sequence. The request should not be granted.

3) a) i) False. When TLB hit ratio is very low, EAT is increased by TLBs lookup.

ii) False. The page transfer out is not necessary if there has been no modifications made. (these pages are identified using a modify or dirty bit)

iii) False. Only fixed allocation implies local replacement. Variable allocation can imply either local replacement or global replacement.

b) Similarities: Blocks of varying size used for both methods, no internal fragmentation. Both methods suffer from external fragmentation.

Differences: Dynamic partitioning uses contiguous allocation while segmentation is non-contiguous allocation and allows the blocks to be placed anywhere in memory.

c) i) Page size is 1024 bytes = 2^{10} bytes (10 bits of offset)

ii) There are 2^{20} frames in physical memory

$$size = 2^{20}(number\ of\ frames) * 2^{10}(size\ of\ a\ frame) = 2^{30}$$

iii) Let the size of page table entry be y.
Number of entries for 2nd level page table = 2^{12} y
Number of entries for outer level page table = $4y^2$
$4y^2 = 2^8$
y = 8
Hence, 3 bits are used for each entry
The size of outer page is 8 + 3 = 11 bits > 10 bits
Thus, cannot fit in a frame

iv) 32(logical addresses have 32 bits) – 10(size of a page) = 22
2^{22} entries

v) Offset remains 10 bits (because the size of a frame does not change), middle part remains 7 bits, outer = 8 + 7 = 15
The format is (15,7,10).

d) i) Belady's anomaly states that increasing the number of page frames results in an increase in the number of page faults for a given memory access pattern.

ii) No. Because LRU is a stacking algorithm, using k frames will always be a subset of k+n frames for LRU. Thus, any page-faults that may occur for k+n frames will also occur for k frames, which in turn means that LRU does not suffer from Belady's anomaly as it fulfils inclusion property.

e) When the working set window is too small, the number of pages in the memory for each process is small, page fault frequency is high, number of processes is high. When the working set window is too large, the number of pages in the memory for each process is large, page fault frequency is low, number of processes is low.

4) a) i) True. A file can have different names in an acyclic-graph directory.

ii) False. Large blocks may result in low disk utilisation rate because space would be wasted when the file does not occupy the entire last block.

iii) False. A process will keep running if non-blocking IO is used.

iv) False. There is only one copy of data in buffering.

b) 5200 / 512 = 10…80 (byte 5200 is within 11th block)
(5200+600) / 512 = 11…168 (byte 5800 is within 12th block)

Contiguous: 2 disk I/O required to read 11th and 12th blocks
Linked: Need to read from first block to traverse to 12th block. 12 disk I/O required.
Indexed: Need to read index block first then read 2 data blocks, total 3 disk I/O required.

c) i) SSTF: 110, 80, 150, 160, 190, 5

This is because if the system is heavily loaded with requests clustered at one end of the disk, requests at the other end of disk may never be serviced. We can use aging to solve this problem by increasing priority of processes over time.

--End of Answers--