CSE 2431
AU 25 **NAME: [*REPLACE WITH YOUR NAME*]**

HOMEWORK 2

Due: Friday, September 19th, 2025 by 11:30 pm on Carmen (as long as you submit before midnight, the assignment will be counted as on time, but at midnight or after (even 1 second!), it will be counted as late)

NOTE: ***Please keep a copy of your homework to study for the exam.***

1. CPU Scheduling

For A) and B) below, analyze scheduling algorithms for the following four processes given the process CPU burst time, and arrival time.

Processes	CPU Burst (ms)	Arrival Time
P1	6	0
P2	3	2
P3	2	4
P4	5	6

For each of the following three scheduling algorithms (i) Draw a Gantt diagram (you can use a one row table with an appropriate number of columns, as shown in the example below); (ii) Calculate clearly (show how you do the calculation; see the class slides on Carmen for formulas) the turnaround time for each process, and (iii) Calculate the average turnaround time. **NOTE:** If the scheduling algorithm cannot uniquely determine which of 2 or more processes should be scheduled, the process among the alternatives which has the earliest arrival time should be scheduled.

A) (Non-preemptive) Shortest Job First

(i) Gantt chart (EXAMPLE given below, but ***it is not correct for this problem***; you can add columns if needed, and you edit the table by adding or deleting columns, and change the process in each box; you can also move the lines dividing the columns; make other Gantt charts below this way)

P1	P3	P4	P2
0	6	8	13	16

(ii) Turnaround time per process (See class slides for formula)

P1:

P2:
P3:
P4:

(iii) Average turnaround time per process

B) (Preemptive) STCF

(i) Gantt chart (you can add or delete columns if needed, and write all transition times below the dividing line in the table when the transition takes place)

0

(ii) Turnaround time per process

P1:
P2:
P3:
P4:

(iii) Average turnaround time per process

C) **For Round Robin Scheduling,** analyze the scheduling algorithm for the following four processes given the process CPU burst time, and arrival time. The order of the four processes in the ready queue is P1 first, then P2, then P3, and last, P4. Assume that the time quantum (time slice) is 2 ms, and make the Gantt chart below. NOTE: If aprocess finishes executing before the end of the 2 ms time slice, it will make an exit system call and the dispatcher will get the CPU back to schedule another process waiting to run in the ready queue, if any are still waiting.

Processes	**CPU Burst (ms)**	**Arrival Time**
P1	6	0
P2	2	0
P3	3	0
P4	4	0

(i) Gantt chart (you can add or delete columns if needed, and write all transition times below the dividing line in the table when the transition takes place)

0

2. Addressing for Paged Memory

Complete the table below by filling in the empty boxes with the correct value for number of virtual address bits; number of virtual page bits; number of page offset bits; number of virtual pages; and number of bytes in each page.

NOTE: For page size, express the value in KB (see note below). For number of virtual pages, write the value as a power of 2 (See the table for an example; if you prefer, you can write the power given as 2^22).

NOTE: Assume 1 KB = 1024 bytes (2^{10} bytes)

Virtual Address Bits	Virtual Page Bits	Page Offset Bits	No. Virtual Pages	No. Bytes per Page
42		12		
32			2^{22}	
30				2 KB
26	16			

3. Virtual memory

Fill in the boxes in the table below for the approaches to virtual memory discussed in class, with T, if the characteristic applies to the approach, or F if it does not. **BE SURE TO WRITE T OR F; DO NOT WRITE "X" OR ANY OTHER SYMBOL EXCEPT T OR F, and do not leave any boxes empty, or you will receive no credit for that box.**

Characteristics

A. No MMU hardware is needed to implement the approach (no MMU or registers in an MMU).
B. No general mechanism for protecting processes from each other (i.e., preventing any process from accessing another process's address space) is provided.
C. The whole address space for the process must be loaded into a contiguous sequence of bytes in memory in order for the process to run.
D. The whole process, or parts of it, can be dynamically relocated by the OS while the process is in execution (that is, between the time it is created and the time it terminates execution).
E. Each segment of the process must be loaded into a contiguous sequence of bytes in memory. (NOTE that even when segmentation is not used, processes still have a code segment with instructions first in memory, then a heap above the code, an empty space (hole) for the heap and stack to grow, and finally, a stack at the top of their memory).
F. The whole address space of the process must be loaded into memory (but not necessarily in a contiguous sequence of bytes) in order for the process to run.

Approach	**Characteristic**	A	B	C	D	E	F

Static relocation	X						
Dynamic Relocation with Base	X						
Dynamic Relocation with Base + bounds	X						
Segmentation	X						
Paging	X						

4. Virtual Memory – Segmentation

A. Suppose the use of segmentation and an 18 bit virtual address space (Note that this is 5 hex digits, but that the most significant hex digit will always correspond to binary 00xy, where the xy bits specify the segment) for a program running as a process with (a maximum of) 4 segments.

B. Here is the segment table

Segment	Base	Bounds	RW (Protection bits)
0	0xc000	0xffff	10
1	0x8000	0x0fff	11
2	0x3000	0x0dff	11
3	0x6000	0x07ff	11

C. Translate the following virtual addresses in the program to the corresponding physical addresses.

i. ***Show clearly how you calculated the physical address (Do this even if you say for Part ii. that the reference results in a segmentation fault).***

ii. Say whether the memory reference results in a segmentation fault (invalid memory access), and ***if so, say why it does clearly***.

Virtual Addresses

a. Read 0x100e8
Physical address:
Segmentation fault? [ ] Yes [ ] No; If so, why?

b. Write 0x30a10
Physical address:
Segmentation fault? [ ] Yes [ ] No; If so, why?

c. Write 0x20e00
Physical address:
Segmentation fault? [ ] Yes [ ] No; If so, why?

d. Write 0x01bc0
Physical address:
Segmentation fault? [ ] Yes [ ] No; If so, why?

The homework must be typed or clearly handwritten, including diagrams (Gantt charts for problem 1), and submitted on Carmen as a docx or pdf file.