ISLAMIC UNIVERSITY OF TECHNOLOGY (IUT)

ORGANISATION OF ISLAMIC COOPERATION (OIC)

Department of Computer Science and Engineering (CSE)

SEMESTER FINAL EXAMINATION **WINTER SEMESTER, 2020-2021**

DURATION: 1 Hour 30 Minutes **FULL MARKS: 75**

CSE 4501: Operating Systems

Programmable calculators are not allowed. Do not write anything on the question paper.

There are **3 (three)** questions. Answer all of them.

Figures in the right margin indicate marks.

File naming format: "**Student ID_C4501F.pdf**"

---

1. a) 10 + 10 *(CO1, CO3, CO4) (PO1, PO2, PO4, PO10)*

Table 1: Process Scheduling Parameters

Process	Arrival Time	Priority*	CPU Burst (t_n)
P_1	0	3	6
P_2	2	1	5
P_3	6	4	5
P_4	7	5	10
P_5	10	2	4

*Priority of a process increases inversely with the value of "*Priority*".

With short term scheduling, there is no way to know the length of the next CPU burst. One way to solve this problem is to predict its value. We might expect that the next CPU burst will be similar in length to the previous ones. Generally, CPU bursts are predicted using the following formula –

$$\tau_n = \alpha \times t_n + (1-\alpha) \times \tau_{n-1}$$

Where,

τ_n = Predicted value for the next CPU burst.

t_n = Length of the $\boldsymbol{n^{th}}$ CPU burst.

τ_{n-1} = The value of $(\boldsymbol{n-1})^{\boldsymbol{th}}$ predicted CPU burst.

α = Controls the relative weight of the recent and past history. Consider $\alpha = 0.5$.

Generate the list of predicted CPU bursts for the 5 processes using the formula mentioned above. For the first process, P_1, compute the value of τ_0 using the following formula –

$$\tau_0 = \left\lceil \frac{last\ 3\ digits\ of\ Student_{ID}}{10} \right\rceil$$

Using these predicted burst times (τ_i) of the processes along with the information in q, calculate the average waiting times for the following scheduling algorithms and show their appropriate Gantt charts –

i. Shortest Remaining Time First (SRTF)

ii. Preemptive Priority Scheduling

b) What impact does the choice of quantum value in *Round Robin* have on the corresponding *average turnaround time*? Explain with an example. 5 *(CO1, CO2, CO4) (PO1, PO4, PO10)*

2. Consider a system with 5 processes, P_1 through P_5, and 3 resources of type A, B, and C with 12, 8, and 10 instances, respectively. Suppose at time t_0, the snapshot of the system is as shown in *Table 2*. 10 + 5 + 10 *(CO1, CO2, CO3, CO4) (PO1, PO2, PO4, PO10)*

Table 2: System Snapshot at time t_0

Process	**Available**			**Allocation**			**Max**		
	A	**B**	**C**	**A**	**B**	**C**	**A**	**B**	**C**
P_1				0	1	0	8	6	3
P_2				2	0	1	5	1	2
P_3	3	3	3	3	2	2	7	5	4
P_4				2	1	4	2	3	5
P_5				2	1	0	6	4	1

Table 3: Resource Request by processes at time t_1

Process	**Request**		
	A	**B**	**C**
P_1	0	0	2
P_2	2	0	0
P_3	1	1	1
P_4	0	0	1
P_5	2	1	1

Based on the above information, answer the following questions.

i. Find whether the system is in a safe state. [Show step-by-step execution.]

ii. Based on the snapshot at t_0, can the requests $(0, 1, 0)$ and $(1, 1, 1)$ generated by $\boldsymbol{P_4}$ and $\boldsymbol{P_2}$, respectively, be granted? Why? Why not?

iii. If the requests in **(ii)** are granted, generate a snapshot of the system at time t_1, consisting of the updated values of the corresponding data structures. Based on this snapshot, detect whether the requests, as shown in *Table 3*, will put the system in a deadlock or not.

3. a) Consider six memory partitions, 300 KB, 600 KB, 350 KB, 200 KB, 750 KB and 125 KB (in order), how would the first-fit, best-fit, and worst-fit algorithms place processes of size 115 KB, 500 KB, 358 KB, 200 KB, and 375 KB (in order)? Rank the algorithms from the worst to the best in terms of efficient use of memory. 10 *(CO1, CO2, CO3, CO4) (PO1, PO2, PO4, PO10)*

b) Considering the processes P_1 and P_2, give an illustration of a *Complete Solution* using synchronization H/W that satisfies the properties of a "*Solution to Critical Section Problem*". 10 *(CO1, CO2, CO3, CO4) (PO1, PO2, PO10)*

c) What are the conditions that must hold simultaneously for occurrence of deadlock in any operating system? 5 *(CO1, CO3) (PO1)*