Department of Computer Science and Engineering
Midterm Examination Fall 2022
CSE 321: Operating Systems
[Set A]

Duration: 1 Hour 10 Minutes | **Total Marks:** 25

Answer the following questions.
Figures in the right margin indicate marks.

---

1. a) **State** time-sharing system with an example. [3]
CO1 b) **Mention** the roles of the system call interface. [2]

2. a) **Distinguish** between CPU scheduler and job scheduler. [3]
CO2 b) **Find** the output of the following code snippet. [4]

```
int main(){
      int id;
      static int x = 10;
      int y = 5;
      id = fork();
      if (id < 0){
            printf("fork failed\n");
      }
      else if(id == 0){
            printf("child started\n");
            printf("child finished\n");
      }
      else{
            wait(NULL);
            printf("parent started\n");
            x=x-2;
            y=y+5;
            printf("values of x: %d & y: %d\n",x,y);
            printf("parent finished\n");
      }
      x=x+5;
      y=y-5;
      printf("values of x: %d & y: %d\n",x,y);
      printf("terminating\n");

      return 0;
```

```
    }
```

3. **CO3**

a) **When** is CPU scheduling required? [2]

b) **Draw** a Gantt chart and illustrate the execution of the process using the **Round Robin** scheduling algorithm **(time quantum = 11 units)**. **Calculate** the **average waiting time** and **number of context switching**. [3+2+1]

Processes	**Arrival Time**	**Burst Time**
P1	3	37
P2	12	17
P3	58	28

P4	59	21
P5	68	19

c) Consider the following set of processes with the length of the CPU-burst time given in milliseconds. **Draw** the Gantt Charts illustrating the execution of these processes using **preemptive priority** (the lowest number implies a higher priority). **Calculate** the **average turnaround time** for the below data set. [3+2]

Processes	Priority	Arrival Time	Burst Time
P1	12	0	4
P2	8	1	2
P3	6	2	3
P4	2	3	5
P5	4	4	1
P6	1	5	4
P7	3	6	6

Department of Computer Science and Engineering
Midterm Examination Fall 2022
CSE 321: Operating Systems
[Set B]

Duration: 1 Hour | **Total Marks:** 25

Answer the following questions.
Figures in the right margin indicate marks.

1.	a) **Explain** dual-mode operation.	[3]
CO1	b) **Briefly explain** any two services of the OS.	[2]

2. a) **Explain** each process state with an example. [3]

CO2 b) **Find** the output of the following code snippet. [4]

```
int main(){
      int id;
      static int x = 10;
      int y = 5;
      id = fork();
      if (id < 0){
            printf("fork failed\n");
      }
      else if(id == 0){
            printf("child started\n");
            x=x+5;
            y=y-3;
            printf("values of x: %d & y: %d\n",x,y);
            printf("child finished\n");
      }
      else{
            wait(NULL);
            printf("parent started\n");
            printf("parent finished\n");
      }
      x=x+5;
      y=y-5;
      printf("values of x: %d & y: %d\n",x,y);
      printf("terminating\n");

      return 0;
}
```

3. a) **"**Multilevel-queue can prevent starvation problem"-**Justify** your answer. [2]

CO3

Draw a Gantt chart and illustrate the execution of the process using the **Round Robin** scheduling algorithm **(time quantum = 12 units)**. **Calculate** the **average waiting time** and **number of context switching**. [3+2+1]

Processes	Arrival Time	Burst Time
P1	3	37
P2	12	17
P3	62	28

P4	63	21
P5	72	19

b. Consider the following set of processes with the length of the CPU-burst time given in milliseconds. **Draw** the Gantt Charts illustrating the execution of these processes using preemptive priority (the highest number implies a higher priority). **Calculate** the **average turnaround** time for the below data set. [3+2 points] [3+2]

Processes	Priority	Arrival Time	Burst Time
P1	2	0	4
P2	4	1	2
P3	6	2	3
P4	10	3	5
P5	8	4	1
P6	12	5	4
P7	9	6	6

A

Department of Computer Science and Engineering

Final Examination Fall 2022

CSE 321: Operating Systems

Duration: 2 Hours **Total Marks:** 40

Answer the following questions.
Figures in the right margin indicate marks.

---

1. CO4	**a)** A system has processes to execute of which **50%** is parallel. If the number of cores is increased from **1** to **4**, what will be the **increase** in performance? **b) Distinguish** between **many-to-many** and **many-to-one** multithreading models.	[2] [2]

2. CO5	**a)** Suppose a faculty member can take a maximum of 5 groups of students for doing a thesis under him in a semester. In a particular semester, a total of 9 groups applied for doing a thesis under his supervision. Among them, he selected 5 groups and kept the rest of the other groups on a waiting list for the next semester where groups will be selected according to a first come first serve manner from the waiting list if any of his slots gets free. **Logically explain** which synchronization method has been used here.
	
	b) For Peterson's problem below conditions will be applied.
	• There are two processes: P1 and P2.
	• Each Statement takes 3ms to execute.
	• Context Switch will occur after 9ms.
	• Both the Critical & Remainder section contains 3 statements.
	• For P1: i=0 and j=1
	• For P2: i=1 and j=0
	• turn=0
	• flag[0] = FALSE, flag[1] = TRUE
	
	The structure of process P_i in Peterson's solution:
	`do{`
	`    flag[i] = true;`
	`    turn = j;`
	`    while(flag[j] == true && turn == 1){`
	`        //busy wait`
	`    }`
	`    //critical section`
	`    flag[i] = false;`
	`    //remainder section`
	`}while(true);`
	
	Complete the table given below for processes P1 and P2 using **Peterson's**
		[2]

solution. [4]

Process 1: i=0, j=1	Process 2: i=1, j=0

3.
CO5

a) Consider the following snapshot of a system:

		Allocation					**MAX**			
		A	**B**	**C**	**D**		**A**	**B**	**C**	**D**
P0		5	1	1	4		10	5	10	5
P1		2	2	6	2		5	9	10	3
P2		2	8	6	4		9	12	9	10
P3		4	6	8	2		4	6	8	3

Available			
1	1	0	3

[4]

[2+3]

i. Is the system in a safe state? **Apply Banker's safety algorithm** to find out the safe sequence. You need to **calculate** the need matrix.

ii. **P_1** requests for **(1 0 0 1)**. **Check** the validity of the request. If the request is valid, does the system enter a **deadlock**?

b) Suppose, in an office, we have a set of resource types, R = {R1, R2, R3} and a set of processes, P = {P1, P2, P3, P4, P5}. **R1, R2, and R3** have **3, 2, and 1**

instance respectively.

- P1 requests for 1 instance of R3
- P1 is holding 2 instances of R2
- P2 requests for 1 instance of R2
- P2 is holding 1 instance of R1
- P3 is holding 1 instance of R3
- P3 requests 1 instance of R1
- P4 is holding 1 instance of R1
- P5 is holding 1 instance of R1
- P5 requests 1 instance of R2

Construct a resource allocation graph for the above scenario. **Mention** the number of cycles found and **identify** whether there is a deadlock or not. [3]

4. CO6

a) At a particular time, the snapshot of the Main memory is given below for dynamic partitioning where gray portions of the memory represent occupied spaces. **Apply worst fit** and **first fit** algorithms to place processes with the space requirement of P1=300k, P2=200k, P3=149k, P4=146k, P5=100k, P6= 50k, P7=22k and P8=29k (in order). **Explain** which algorithm makes the most effective use of memory. [2+2+1]

400K	300K	160K	50K	200K	260K

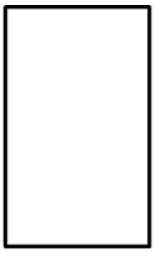
b) Assume that the page size is 3 bytes and the Physical Memory size is 36 bytes. **Show** the users' view of memory which is mapped into physical memory. [2]

P0	Red
P1	Blue
P2	Green
P3	Yellow
P4	Pink
P5	Orange

Logical space of processes

0	6
1	2
2	9
3	4
4	10
5	7

PMT



Main Memory

c) If the page size is **2 KB**, **how many** frames will be needed in Main memory for a process size of **38,767 Bytes**? Is there any **internal fragmentation**? - If yes, **calculate** the value. [1 KB = 1024 Bytes] [2]

d) What are the **differences** between **static** and **dynamic** techniques for partitioning main memory? [3]

5. CO6

Consider a computer with a main memory that has 3 frames and page reference string of 0-7 pages: **[7 0 6 4 2 6 1 2 0 5 1]**. The page reference string represents the order in which the pages are accessed by a program. **Apply FIFO** & **LRU** algorithm to **simulate** the page replacement that occurs when the main memory can hold at most 3 pages at a time. **Record** the number of **page faults** and [4+1+1]

	compare the result. **Mention** which algorithm performs better in this scenario.	

Department of Computer Science and Engineering

Final Examination Fall 2022

CSE 321: Operating Systems

B

Duration: 2 Hours **Total Marks:** 40

Answer the following questions.
Figures in the right margin indicate marks.

---

1. CO4

a) A system has processes to execute of which **30%** is serial. If the number of cores is increased from **1** to **3**, what will be the **increase** in performance? [2]

b) Distinguish between **many-to-one** and **one-to-one** multithreading models**.** [2]

2. CO5

a) Suppose a medical center is providing Covid vaccination. In that center maximum of 6 people can take vaccines at a time in separate booths. But approximately 50 people went there to take vaccines on a particular day. Therefore, the authorities have decided that they will provide vaccines to 6 people at a time and keep others waiting in a queue. If any of the vaccine booths get free a person from the queue will be taken to that booth according to the first come first serve manner for vaccination. **Logically explain** which synchronization method has been used here [2]

b) For Peterson's problem below conditions will be applied.

- There are two processes: P1 and P2.
- Each Statement takes 4ms to execute.
- Context Switch will occur after 12ms.
- Both the Critical & Remainder section contains 3 statements.
- For P1: i=0 and j=1
- For P2: i=1 and j=0
- turn=0
- flag[0] = FALSE, flag[1] = FALSE

The structure of process P_i in Peterson's solution:

```
do{
       flag[i] = true;
       turn = j;
       while(flag[j] == true && turn == 1){
              //busy wait
       }
       //critical section
       flag[i] = false;
       //remainder section
}while(true);
```

Complete the table given below for processes P1 and P2 using **Peterson's**

solution. [4]

Process 1: i=0, j=1	Process 2: i=1, j=0
⬇	

3. CO5 **a)** Consider the following snapshot of a system:

		Allocation					MAX			
		A	**B**	**C**	**D**		**A**	**B**	**C**	**D**
P_0		5	1	1	4		10	5	10	11
P_1		2	8	6	4		9	12	9	10
P_2		2	2	6	2		5	9	10	3
P_3		4	6	8	2		4	6	8	3

Available			
4	4	6	4

i. Is the system in a safe state? **Apply Banker's safety algorithm** to find out the safe sequence. You need to calculate the need matrix. [4]

ii. **P_0** requests for **(3 3 6 1**), **check** the validity of the request. If the request is valid, does the system enter a **deadlock**? [2+3]

b) Suppose, in an office, we have a set of resource types, R = {R1, R2, R3} and a set of processes, P = {P1, P2, P3, P4, P5}. **R1, R2, and R3** have **3, 2, and 1** instance respectively.

- P1 is holding 1 instance of R3
- P1 requests 1 instance of R1
- P3 requests for 1 instance of R3
- P3 is holding 2 instances of R2
- P2 requests for 1 instance of R2
- P2 is holding 1 instance of R1
- P4 is holding 1 instance of R1
- P4 requests 1 instance of R2
- P5 is holding 1 instance of R1

Construct a resource allocation graph for the above scenario. **Mention** the [3] number of cycles found and **identify** whether there is a deadlock or not.

4. **CO6** **a)** At a particular time, the snapshot of the Main memory is given below for dynamic partitioning where gray portions of the memory represent occupied spaces. **Apply worst fit** and **first fit** algorithms to place processes with the space requirement of P1=300k, P2=200k, P3=149k, P4=146k, P5=100k, P6= 50k, P7=22k and P8=29k (in order). **Explain** which algorithm makes the most effective use of memory. [2+2+1]

400K	300K	160K	50K	200K	260K

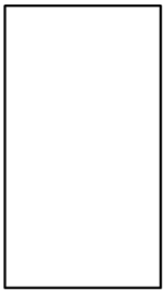
b) Assume that in a paged memory management system the page size for processes is 4 bytes and the Physical Memory size is 36 bytes. **Show** the users' view of memory which is mapped into physical memory. [2]

P0	Free
P1	Finite
P2	Fruit
P3	From
P4	Flower
P5	Freedom

Logical space of a process

0	2
1	7
2	4
3	5
4	3
5	1

PMT



Main Memory

c) If the page size is **6 KB**, **how many** frames will be needed in Main memory for a process size of **102,506 Bytes**? Is there any **internal fragmentation**? - If yes, **calculate** the value. [1 KB = 1024 Bytes] [2]

d) Discuss the purpose of **MMU**. [3]

5. **CO6** Consider a computer with a main memory that has 3 frames and page reference string of 0-7 pages: **[3 0 6 4 2 6 7 2 0 1 7]**. The page reference string represents the order in which the pages are accessed by a program. **Apply FIFO** & **LRU** algorithm to simulate the page replacement that occurs when the main memory can hold at most 3 pages at a time. **Record** the number of **page faults** and compare the result. **Mention** which algorithm performs better in this scenario. [4+1+1]