



Department of Computer Science and Engineering
Midterm Examination Spring 2025
CSE 321: Operating Systems

Duration: 1 Hour 30 Minutes

Total Marks: 30

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

Name: XXXXXXXXXX	ID: XXXXXX	S: XXXXXX
--	--	---

1. a) Write the answers in the question paper.

CO1

[1*2
=2]

- i) Let's consider a scenario where a total of 4 processes are available for CPU allocation. Now, they have burst times like 4, 5, 6, 8. The OS is configured to use Round-Robin scheduling with a time quantum of 10. At this point, figure out the problem with this amount of time quantum.

Answer: It will behave like a FIFO queue and follow FCFS algorithm because of max(burst time)

- ii) Observe the following code. Calculate how many processes and how many threads will be created.

```
pid_t pid;
pthread_create( . . . );
pid = fork();
if (pid == 0) { /* child process */
    fork();
}
fork();
```

Answer: Thread = 1
Process = 6

b) Find outputs of the given code:

[10*
0.5=
5]

(P)

```

int main(){
    pid_t a,b;
    int c[]={5,7,3};
    ① a=fork();
    if(a<0){
        printf("error\n");
    }
    else if(a==0){
        c[0]=c[1]-c[2];
        c[1]=c[0]-c[2];
        c[2]=c[0]-c[1];
        ② b=fork();
        if(b<0){

        }
        else if(b>0){
            wait(); — C1
            c[0]=c[1]*c[2];
            c[1]=c[0]*c[2];
            c[2]=c[0]*c[1];
        }
        else{
            c[0]=c[1]+c[2];
            c[1]=c[0]+c[2];
            c[2]=c[0]+c[1];
        }
    }
    else{
        printf("hello\n");
        wait(); — P
        c[0]=c[1]+c[2];
        c[1]=c[0]+c[2];
        c[2]=c[0]+c[1];
    }
    for(int i=0;i<3;i++){
        printf("c[%d]= %d\n",i,c[i]);
    }
    return 0;
}

```

Write the outputs in the question paper. Output sequence should be exactly matched.

Outputs

1	hello
2	c[0] = 4
3	c[1] = 7
4	c[2] = 11
5	c[0] = 3
6	c[1] = 9
7	c[2] = 27
8	c[0] = 10
9	c[1] = 13
10	c[2] = 23

✓ c) Write the answers in the answer script.

Dipu is implementing his own OS. For his OS he has decided to develop **Multilevel Feedback Queue** as a scheduler. To design the scheduler he decided to divide the ready queue into **three** levels. According to his preferences, in every level round-robin algorithm will be used and some constraints should be maintained in each level which are given below.

- Q1: time quantum = 3, priority = 1
- Q2: time quantum = 5, priority = 5
- Q3: time quantum = 10, priority = 7

Here, the lower priority value indicates the higher priority and in case of an I/O request, the scheduler will promote a process by one level upon I/O operation completion. At a certain moment, Dipu has collected information on burst times, arrival times, and I/O times of four processes

Process	Burst Time	Arrival Time	I/O Time
P1	19	0	6 [After 15s of total CPU allocation]
P2	8	2	N/A
P3	9	12	4 [After 7s of total CPU allocation]
P4	17	7	N/A

✓ i. Draw a Gantt chart using multilevel feedback queue scheduling algorithm showing the states of the ready queues of different levels. [12]

✓ ii. Calculate the average waiting time and average turnaround time from the Gantt chart. [1.5+1.5=3]

2. a) Write the answers in the question paper. [1*3=3]

CO2

✓ i) You are given an array of **100** integers. You are required to calculate the sum of these integers using a **10-core** CPU. **Identify** what kind of parallelism you need to implement to ensure that the operation is performed most efficiently.



Answer: Data Parallelism. (Divide 100 by 10 and sum 10 integers in 10 CPU cores)

- 20 PCs are dedicated to AI/ML-related thesis groups.
- 18 PCs are dedicated to NLP-related thesis groups.
- The remaining PCs are assigned to Image Processing-related thesis groups. 17

Answer: Counting Semaphore (For NLP, $S = 18$, Group = 23)
↓
instances

```
lock = false
```

```

        while(test_and_set(&lock));
        if (available_seats > 0) {    // Check if seats are
available
            available_seats = available_seats - 1 // Deduct
seat

```

Figure out the error from the above code.

Answer: `while (test_and_set(&lock) != True);`

[10*
0.5=
5]

b) Find outputs of the given code:

```
int t_id[]={1,2,3,4};
void *func(int *id);
int a=123;
int b=50;
sem_t s1,s2,s3,s4;

int main(){
    pthread_t t[4];
    sem_init(&s1,0,0);
    sem_init(&s2,0,1);
    sem_init(&s3,0,0);
    sem_init(&s4,0,0);
    for(int i=0;i<4;i++){
        pthread_create(&t[i],NULL,(void *)func,&t_id[i]);
    }
    for(int i=0;i<4;i++){
        pthread_join(t[i],NULL);
    }
    sem_destroy(&s1);
    sem_destroy(&s2);
    sem_destroy(&s3);
    sem_destroy(&s4);
    printf("Final a: %d\nFinal b: %d\n",a,b);
    return 0;
}


void *func(int *id){
    if(*id==1){
        sem_wait(&s3);
        printf("a: %d\nb: %d\n",a,b);
        a=a+b;
        b=b-a;
        sem_post(&s3);
    }
    else if(*id==2){
        sem_wait(&s1);
        printf("a: %d\nb: %d\n",a,b);
        a=a*b;
        b=b*5;
        sem_post(&s1);
        sem_post(&s3);
    }
    else if(*id==3){
        sem_wait(&s2);
        printf("a: %d\nb: %d\n",a,b);
        a=a-b;
        b=b-a;
        sem_post(&s2);
        sem_post(&s4);
    }
    else{
        sem_wait(&s4);
        printf("a: %d\nb: %d\n",a,b);
        a=a-12;
        b=b-16;
        sem_post(&s4);
        sem_post(&s1);
    }
}
```

Handwritten annotations in the code:

- Next to `printf("a: %d\nb: %d\n",a,b);` for `*id==1`: A circle containing "12" and a crossed-out circle containing "12".
- Next to `printf("a: %d\nb: %d\n",a,b);` for `*id==2`: A circle containing "111".
- Next to `printf("a: %d\nb: %d\n",a,b);` for `*id==3`: A circle containing "1".
- Next to `printf("a: %d\nb: %d\n",a,b);` for the `else` block: A circle containing "11".

Write the outputs in the question paper. Output sequence should be exactly matched.

Outputs



1	a : 123
2	b : 50
3	a : 73
4	b : -23
5	a : 61
6	b : -39
7	a : -2379
8	b : -195
9	Final a: -2574
10	Final b: 2379

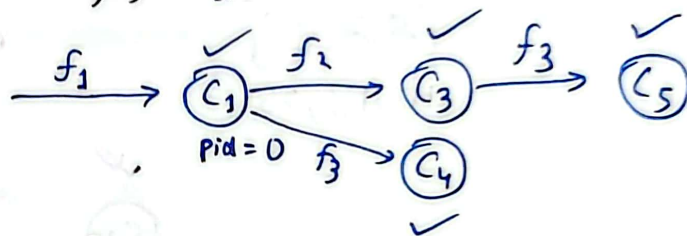
Ans to the Q.N-1

pid > 0
a) ii) (P)

pid = & pid ;

pthread_create (...) ; t1

① pid = fork () ;



if (pid == 0) {

② fork ()

}

③ fork ()



b)

(P)

$a > 0$

b

$c = \{8, 7, 3\}$

10, 13, 23

(C₁)

$c = \begin{bmatrix} 5 & 7 & 3 \\ 4 & 1 & 3 \\ 3 & 9 & 27 \end{bmatrix}$

$b > 0$

$a = 0$

(C₂)

$c = \begin{bmatrix} 4 & 8 & 3 \\ 4 & 7 & 11 \end{bmatrix}$

$b = 0$

Output

hello

$c[0] = 4$

$c[1] = 7$

$c[2] = 11$

$c[0] = 3$

$c[1] = 9$

$c[2] = 27$

$c[0] = 10$

$c[1] = 13$

$c[2] = 23$

C) i) Given,

Process	Arrival Time	Burst Time	I/O
P ₁	0	19 16 14 0	6s after 15 CPU
P ₂	2	8 5 0	—
P ₃	12	9 6 2 0	4s after 7 CPU
P ₄	7	17 14 9 0	—

Q₁ (n = 3)

P₁ P₂ P₄ P₃ P₃

Q₂ (n = 5)

P₁ P₂ P₄ P₃ P₁

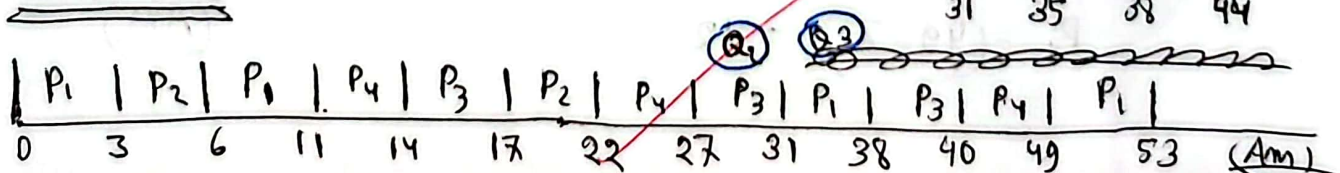
Q₃ (n = 10)

P₁ P₄

I/O queue

P₃ P₁

Gantt chart



i) Waiting Time:

$$P_1 \rightarrow (0-0) + (6-3) + (31-11) + (49-44) = 28$$

$$P_2 \rightarrow (3-2) + (17-6) = 12$$

$$P_3 \rightarrow (14-12) + (27-17) + (38-35) = 15$$

$$P_4 \Rightarrow (11-7) + (22-14) + (40-27) = 25$$

$$\therefore \text{Average waiting time} = 20 \text{ s}$$

(Ans)

Turnaround time:

$$P_1 = (53-0) = 53$$

$$P_2 = (22-2) = 20$$

$$P_3 = (40-12) = 28$$

$$P_4 = (49-7) = 42$$

$$\therefore \text{Average Turnaround time}$$

$$= 35.75 \text{ s}$$

(Ans)

i=3

Ans to the Q.N-2

b)

id = [1 , 2 , 3 , 4]

a = ~~123~~ ~~73~~ ~~61~~ -2379 - 2574

b = ~~50~~ -23 ~~-39~~ -195 2379

S2

Output:

- a: 123

- b: 50

- a: 73

- b: -23

- a: 61

- b: -39

a: -2379

~~b: -195~~

Final a: -2574

~~Final b: 2379~~

Final b: 2379