Operating Systems (CS2006)

Sessional-II Exam

Date: November 5th 2024

Course Instructor(s)

Mr. Razi Uddin
Mr. Mubashar Hussain
Ms. Rubab Anam
Ms. Namra Absar

Total Time (Hrs): 1
Total Marks: 35
Total Questions: 2

[redacted]

Roll No [redacted] Section

[redacted]
Student Signature

Do not write below this line

Attempt all the questions.

Follow the order of the questions as given. You must show all relevant work, reasoning, and steps as specified in the question statements to receive full credit.

CLO 3: Evaluate the commonly used mechanisms for scheduling of tasks and implement synchronization mechanisms like Semaphores, TSL, etc.

Q1: *[Marks: 20]*

A CPU scheduling system needs to handle six processes, where some processes are scheduled based on **preemptive Priority Scheduling** (where 1 is the highest priority) and others using **Shortest Remaining Time First (SRTF)**. The system always gives preference to **Priority Scheduling** for high-priority tasks, but if two processes have the same priority, the **SRTF** rule applies.

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

TAT = $\frac{66}{6}$ = 11

W = $\frac{38}{6}$ = 6.3333

Requirements:

1. **Draw a Gantt chart** showing the order of execution of these processes according to the scheduling rules.
2. Calculate the following:

- Average **Turnaround Time** for all processes.
- Average **Waiting Time** for all processes.

CLO 3: Evaluate the commonly used mechanisms for scheduling of tasks and implement synchronization mechanisms like Semaphores, TSL, etc.

Q2: [Marks: 10+5=15]

A)

In a busy restaurant, serving an order requires both a chef and a waiter. An order can only be completed if both roles are present.
There are two processes: one for chefs and one for waiters. The kitchen has only one serving station where orders are picked up. An order can be fulfilled if both a chef and a waiter are available. If a waiter arrives at the station and a chef has already prepared the order, the waiter picks it up and serves it, and vice versa. However, if a waiter arrives and no chef is available, they'll wait at the serving station until a chef arrives with the prepared order, and the same goes if a chef arrives first.
Additionally, no two chefs or two waiters can be at the serving station simultaneously; each order requires exactly one chef and one waiter working together. Implement these rules using **semaphores**.

Note: To receive full credit, explicitly declare and initialize all semaphores and variables.

Process 1 (Chef)	Process 2 (waiters)
Serving station (Critical Section)	Serving station (Critical Section)

B)

```
#define BUFFER_SIZE 10
typedef struct
{
  ...
} item;
item buffer[BUFFER_SIZE];
int in=0;
int out=0;
```

The code for the producer process is:

```
while(1)
{
   /*Produce an item in nextProduced*/
   while(counter == BUFFER_SIZE); /*do nothing*/
   buffer[in]=nextProduced;
   in=(in+1)%BUFFER_SIZE;
   counter++;
}
```

The code for the consumer process is:

```
while(1)
{
    while(counter==0); //do nothing
    nextConsumed=buffer[out];
    out=(out+1)%BUFFER_SIZE;
    counter--;
    /*Consume the item in nextConsumed*/
}
```

The code above addresses the famous Producer-Consumer problem and is intended to synchronize the producer and consumer processes:

- Identify a potential problem with this code. Explain the cause of this issue and the specific conditions under which it might occur.