NATIONAL TEXTILE

UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE

SUBMITTED BY:

Eman Faisal 23-NTU-CS-1149

SECTION SE: 5th (A)

Operating System-Hometask lab

SUBMITTED TO:

Sir Naisr mahmood

SUBMISSION DATE:

18 dec 1025

Operating Systems – COC 3071

SE 5th A – Fall 2025

After-mid Homework -1

Part 1: Semaphore theory

1. **A counting semaphore is initialized to 7. If 10 wait() and 4 signal() operations are performed, find the final value of the semaphore.**

   Initial value of semaphore=7

   Then 10 times wait, for 7 times,the value of semaphore is decremented by 1

   So semaphore =0, blocked=3

   Then 4 signal():

   1:wake a process

   2:wake a process

   3:wake a process

   4:No process to wake so increment by 1.

   Semaphore value=1

2. **A semaphore starts with value 3. If 5 wait() and 6 signal() operations occur, calculate the resulting semaphore value.**

   When 5 wait() called, semaphore has value 3. So, 5-3=2,**2 are BLOCKED.**

   Value =0.

   When **6** times **Signal()** occur:

   1st signal: 1st block wake.

   2nd signal: 2nd block wake

   6-2=4.No process to wake anymore so increment by 4.

   Value=0+4

3. **A semaphore is initialized to 0. If 8 signal() followed by 3 wait() operations are executed, find the final value.**

   When semaphore starts with value **0**.When **8 signal()** called, semaphore increases by 8.

   Value = **8**.Then **3 wait()** are called:

   $8 - 3 = \mathbf{5}$, no process is blocked.**Value = 5**.

4. **A semaphore is initialized to 2. If 5 wait() operations are executed:**

a) **How many processes enter the critical section?**

   2 processes will enter the critical section

b) **How many processes are blocked?**

   3 processes are blocked

5. A semaphore starts at 1. If 3 wait() and 1 signal() operations are performed:

a) **How many processes remain blocked?**

   1 process remains blocked.

b) **What is the final semaphore value?**

When semaphore starts with value **1**.When **3 wait()** called, semaphore has value 1.
So, **3 − 1 = 2**, **2 are BLOCKED**.
Value = **-2**.When **1 signal()** occurs:
1st signal: **1 blocked process wakes up**.Value=-1
No increment happens.Value = **-1**.

6.
semaphore S = 3;
wait(S); wait(S);
signal(S);
wait(S); wait(S);

a) **How many processes enter the critical section?**
   4 processes enter the wait section
b) **What is the final value of S?**
   The final value is:3-2+1-2=0

7.
semaphore S = 1;
wait(S); wait(S);
signal(S);
signal(S);

a) **How many processes are blocked?**
   0 processes blocked
b) **What is the final value of S?**

Semaphore **S starts with value 1**.

wait(S) called first time:
S = **1 − 1 = 0**, no process blocked.

wait(S) called second time:
S = **0**, 1 process is BLOCKED**.**
Value = **-1**.

signal(S) called first time:
1 blocked process wakes up.
Value becomes **0**.

signal(S) called second time:
No blocked process to wake, so semaphore increments.
Value = **1**.

8. A binary semaphore is initialized to 1. Five wait() operations are executed without any signal(). How many processes enter the critical section and how many are blocked?
   1 process enters a critical section.The other 4 are blocked.

9. A counting semaphore is initialized to 4. If 6 processes execute wait() simultaneously, how many proceed and how many are blocked?

6-4=2,2 processes are blocked and 4 processes proceed.

10. A semaphore S is initialized to 2. wait(S); wait(S); wait(S); signal(S); signal(S); wait(S);

a) **Track the semaphore value after each operation.**

For first wait(S): value=2-1=1
For second wait(S): value=1-1=0
For third wait(S): value=0-1=-1
For First Signal:value=-1+1=0
For Second signal:0+1=1
For 4^{th} wait:1-1=0

b) **How many processes were blocked at any time?**

1 process

11. A semaphore is initialized to 0. Three processes execute wait() before any signal(). Later, 5 signal() operations are executed.

a) How many processes wake up?

All processes wake up

b) What is the final semaphore value?

Final semaphore value:0-3=5=2

Part 2: Semaphore Coding

Consider the Producer–Consumer problem using semaphores as implemented in Lab-10 (Lab-plan attached). Rewrite the program in your own coding style, compile and execute it successfully, and explain the working of the code in your own words.

Submission Requirements:

- Your rewritten source code
- A brief description of how the code works
- Screenshots of the program output showing successful execution

```
#include <stdio.h>
#include <pthread.h>
#include <semaphore.h>
```

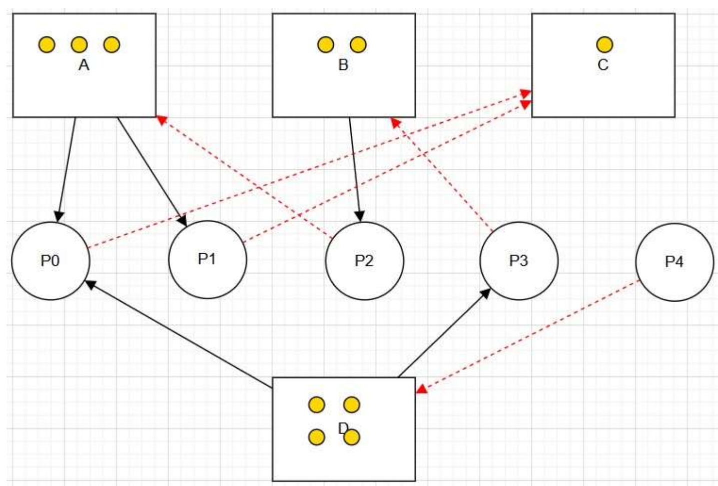
```c
#include <unistd.h>

sem_t slots;

void* vehicle(void* arg) {
    int num = *(int*)arg;

    printf("Vehicle %d wants to enter.\n", num);

    sem_wait(&slots);

    printf("Vehicle %d entered successfully.\n", num);

    sleep(2);

    printf("Vehicle %d exited.\n", num);

    sem_post(&slots);

    return NULL;
}

int main() {
    pthread_t threads[10];
    int numbers[10];

    sem_init(&slots, 0, 3);

    for (int i = 0; i < 10; i++) {
        numbers[i] = i + 1;
        pthread_create(&threads[i], NULL, vehicle, &numbers[i]);
    }

    for (int i = 0; i < 10; i++) {
        pthread_join(threads[i], NULL);
    }
    sem_destroy(&slots);
    return 0;
}
```

EXPALNATION:

- A semaphore ‘slots’ represents the available parking spaces,its value is 3.
- Multiple threads are created, where each thread represents a vehicle.
- Each vehicle tries to enter by calling sem_wait().
- If a slot is available, the vehicle enters; otherwise, it waits (blocks) until a slot is free.
- The slot is free,when semaphore has value greater than 0.
- Once inside, the vehicle stays for some time using sleep().
- After leaving, the vehicle calls sem_post() to free the slot.
- This ensures that only 3 vehicles can enter at a time, even though 10 threads are running.

Part 3: RAG (Recourse Allocation Graph)

- Convert the following graph into matrix table ,



Allocation matrix:

Process	A	B	C	D
P0	1	0	0	1
P1	1	0	0	0
P2	0	1	0	0
P3	0	0	0	1
P4	0	0	0	0

Resource Matrix:

Process	A	B	C	D
P0	0	0	1	0

P1	0	0	1	0
P2	1	0	0	0
P3	0	1	0	0
P4	0	0	0	1

Part 4: Banker's Algorithm

System Description:

- The system comprises five processes (P0–P3) and four resources (A,B,C,D).

- Total Existing Resources:

Total			
A	B	C	D
6	4	4	2

- Snapshot at the initial time stage:

	Allocation				Max				Need			
	A	B	C	D	A	B	C	D	A	B	C	D
P0	2	0	1	1	3	2	1	1				
P1	1	1	0	0	1	2	0	2				
P2	1	0	1	0	3	2	1	0				
P3	0	1	0	1	2	1	0	1				

-

Questions:

1. **Compute the Available Vector**:

   - Calculate the available resources for each type of resource.

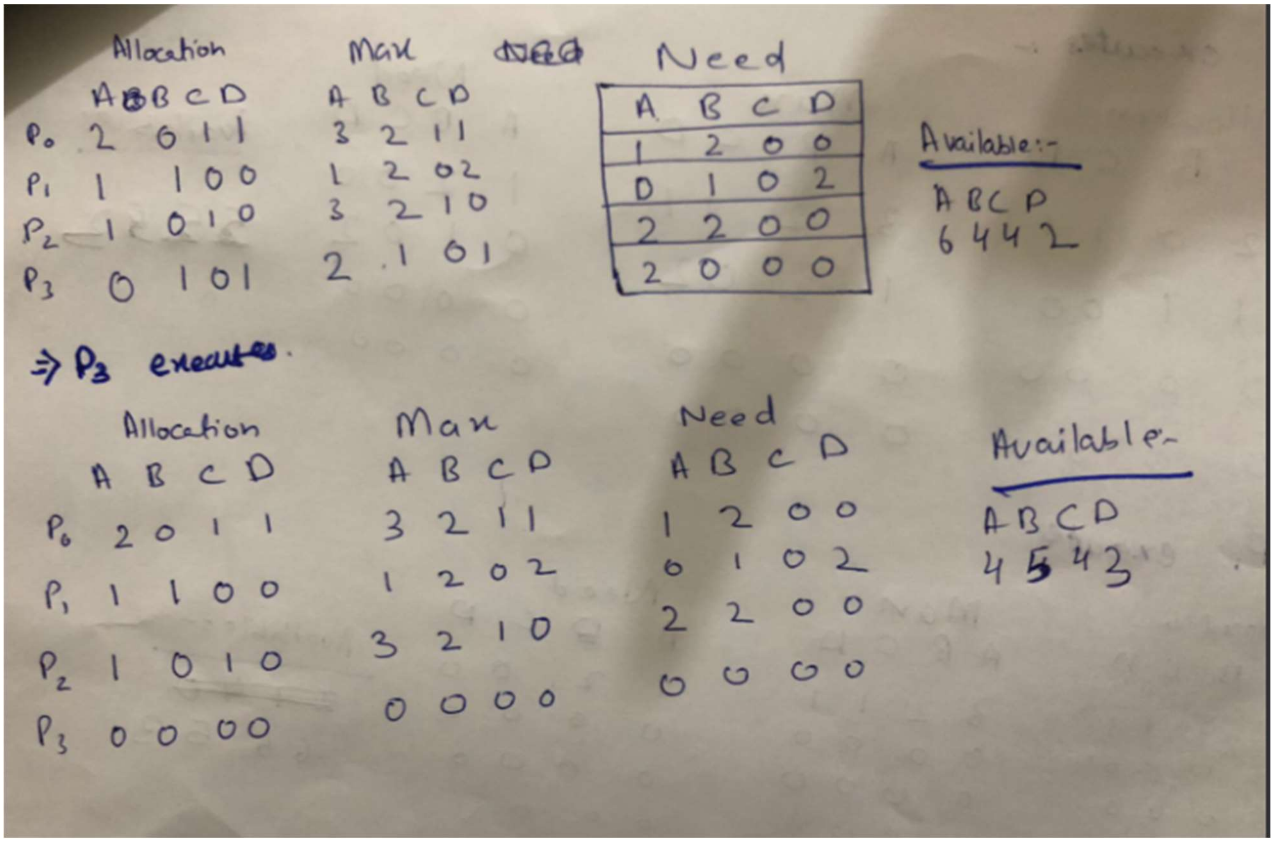
2. **Compute the Need Matrix**:

   - Determine the need matrix by subtracting the allocation matrix from the maximum matrix.

3. **Safety Check**:

- Determine if the current allocation state is safe. If so, provide a safe sequence of the processes.

- Show how the Available (working array) changes as each process terminates.



	Allocation A B C D	Max A B C D	Need A B C D
P_0	2 0 1 1	3 2 1 1	1 2 0 0
P_1	1 1 0 0	1 2 0 2	0 1 0 2
P_2	1 0 1 0	3 2 1 0	2 2 0 0
P_3	0 1 0 1	2 1 0 1	2 0 0 0

Available:-

A B C D
6 4 4 2

⇒ P_3 executes.

	Allocation A B C D	Max A B C D	Need A B C D
P_0	2 0 1 1	3 2 1 1	1 2 0 0
P_1	1 1 0 0	1 2 0 2	0 1 0 2
P_2	1 0 1 0	3 2 1 0	2 2 0 0
P_3	0 0 0 0	0 0 0 0	0 0 0 0

Available-

A B C D
4 5 4 3

⇒ P_2 executes :-

	Allocation				Max				Need				Available :-
	A	B	C	D	A	B	C	D	A	B	C	D	
P_0	2	0	1	1	3	2	1	1	1	2	0	0	5552
P_1	1	1	0	0	1	2	0	2	0	1	0	2	
P_2	0	0	0	0	0	0	0	0	0	0	0	0	
P_3	0	0	0	0	0	0	0	0	0	0	0	0	

⇒ P_1 executes :-

	Allocation				Max				Need				Available :-
	A	B	C	D	A	B	C	D	A	B	C	D	
P_0	2	0	1	1	3	2	1	1	1	2	0	0	~~2140~~
P_1	0	0	0	0	0	0	0	0	0	0	0	0	6653
P_2	0	0	0	0	0	0	0	0	0	0	0	0	
P_3	0	0	0	0	0	0	0	0	0	0	0	0	

⇒ P_0 Executes :-

	Allocation				Max				Need				Available
	A	B	C	D	A	B	C	D	A	B	C	D	
P_0	0	0	0	0	0	0	0	0	0	0	0	0	7464
P_1	0	0	0	0	0	0	0	0	0	0	0	0	
P_2	0	0	0	0	0	0	0	0	0	0	0	0	
P_3	0	0	0	0	0	0	0	0	0	0	0	0	

$P_3 \rightarrow P_2 \rightarrow P_1 \rightarrow P_0$ Safe ✓

Submission Guidelines:

- Ensure all answers are well-explained and calculations are shown step-by-step.
- Submit your assignment on MS Team and GitHub in a PDF format.
- VIVA based Evaluation so Develop your own solution after getting help.