# United International University

## Department of Computer Science and Engineering

CSE 4509/CSI 309: Operating System Concepts/Operating Systems

Mid Term Examination, Spring 2025 Time: 1.5 Hours

*Any examinee found adopting unfair means will be expelled from the trimester / program as per UIU disciplinary rules.*

*N.B. Answer all the questions*

---

1. (a) "In the life cycle of a process, there is a continuous loop between the Ready and Running state"- argue for or against the statement. [2]

   (b) Simulate the code snippet given below (in details), and include and identify the corresponding output(s) generated. [6]

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/wait.h>
int main() {
    pid_t pid1, pid2, pid3;
    pid1 = fork();
    if (pid1 == 0) {
        printf("Horcrux destroyed: Tom Riddle's Diary!\n");
        pid3 = fork();
        if (pid3 == 0) {
            printf("Horcrux destroyed: Hufflepuff's Cup!\n");
            exit(0);
        } else if (pid3 > 0) {
            wait(NULL);
            exit(0);
        }
    }
    else if (pid1 > 0) {
        pid2 = fork();

        if (pid2 == 0) {
            printf("Horcrux destroyed: Slytherin's Ring!\n");
            exit(0);
        }
        else if (pid2 > 0) {
            wait(NULL);
            printf("At least one Horcrux gone!\n");
        }
    }
    return 0;
}
```

   (c) Differentiate between the contents of the data and stack portion corresponding to the memory layout of a process. [2]

2. (a) What is caching in storage-device hierarchy? [2]

   (b) Which one is better - Polling or interrupt driven I/O ? And why? [2]

3. (a) Five processes (P1, P2, P3, P4, and P5) arrive at time 0 with the following run times: [3]
P1: 6 ms, P2: 8 ms, P3: 7 ms, P4: 3 ms, P5: 4 ms
calculate the turnaroud time for each process if Shortest Job First (SJF) scheduling is used.

(b) Now assume the processes arrived at the following times: [3]
P1: 3 ms, P2: 0 ms, P3: 2 ms, P4: 0 ms, P5: 4 ms
The run times of the processess are same as in question (a).
Calculate the average response time if Shortest Time To Completion First (STCF) scheduling is used.

4. (a) Consider the following pseudocode and write the output: [3]

```
mutex_t m;
int done[3]={0};
cond_t c[3];
int i=0;

void *thread_serial(void *args) {
    int k = i%3;
    mutex_lock(&m);
    if(!done[k]) {
       wait(&c[k], &m);
    }
    printf("thread %d\n", k);
    done[k] = 0;
    i += 2;
    k = i%3;
    done[k] = 1;

   cond_signal(&c[k]);
   mutex_unlock(&m);
}

int main() {
    pthread_t t[3];
    done[0] = 1;
    for(int i=0; i<3; i++) {
        pthread_create(t[i], thread_serial);
    }

    for(int i=0; i<3; i++) {
        pthread_join(t[i]);
    }
}
```

(b) Identify the problem and write the solution in the following implementation of producer-consumer problem. Note that, you do not need to write the full code. Only write the modified line(s). [3]

```
void *producer(void *arg) {
    for (int i = 0; i < loops; i++) {
        Mutex_lock(&m); // p1
        if (numfull == max) // p2
            Cond_wait(&empty, &m); // p3
        do_fill(i); // p4
        Cond_signal(&fill); // p5
        Mutex_unlock(&m); // p6
    }
}
void *consumer(void *arg) {
    while (1) {
        Mutex_lock(&m); // c1
        if (numfull == 0) // c2
            Cond_wait(&fill, &m); // c3
        int tmp = do_get(); // c4
        Cond_signal(&empty); // c5
        Mutex_unlock(&m); // c6
    }
}
```

(c) A host of a party has invited $N > 2$ guests to his house. Due to fear of Covid-19 exposure, the host does not wish to open the door of his house multiple times to let guests in. Instead, he wishes that all N guests, even though they may arrive at different times to his door, wait for each other and enter the house all at once. The host and guests are represented by threads in a multi threaded program. Given below is the pseudocode for the host thread, where the host waits for all guests to arrive, then calls openDoor(), and signals a condition variable once. You must write the corresponding code for the guest threads. The guests must wait for all N of them to arrive and for the host to open the door, and must call enterHouse() only after that. You must ensure that all N waiting guests enter the house after the door is opened. You must use only locks and condition variables for synchronization. [4]

The following variables are used in this solution: lock m, condition variables cv_host and cv_guest, and integer guest_count (initialized to 0). You must not use any other variables in the guest for synchronization.

```
mutex_t m;
cond_t cv_host, cv_guest;

void *host(void *args) {
    lock(m);
    while(guest_count < N) {
        wait(cv_host, m);
    }
    openDoor();
    signal(cv_guest);
    unlock(m);
}

void *guest(void *args) {

    // You need to implement this part only

}
```