Department of Computer Science and Engineering
Indian Institute of Technology Patna.
Bihta, Patna, Bihar - 801106

Operating Systems (CS341)

Duration: 120 Mins | Mid-Sem Examination | Marks: 60

Instructions:

- **All questions are compulsory.**
- **No doubt will be entertained during the examination.**

1. Explain the mechanism of multitasking in Andriod-based mobile phones. Discuss any three system calls related to process management in the Windows operating system. [2+2=4]
2. Briefly explain the dual-mode operation of the operating system with a suitable diagram. Further, mention one benefit of dual-mode operation. [2+2+2=6]
3. Define the process control block (PCB) and explain any of its four attributes(or fields). Further, list any four important PCB's attributes required to support multi-threading explicitly. [2+2+2=6]
4. The microkernel structure-based operating systems are more secure and easy to extend. Justify. Further, points out one major drawback of this structure. [2+2+2=6]
5. Define the condition variable and its associated functions in the context of monitors. Suggest an implementation of binary semaphore that eliminates busy waiting in the entry section of the code. [3+3=6]
6. Discuss different CPU scheduling criteria. How is the scheduler different from the dispatcher? Prove formally that the Shortest Job First scheduling algorithm is optimal in that it minimizes the average waiting time for a given set of processes. [1+1+4=6]
7. What is the priority inversion problem? Give an example in which a lower-priority process can prevent a higher-priority process from running. Further, mention how priority inversion handles this. [2+2+2=6]
8. The arrival and burst times for six processes ($A - F$) are given below. Assume that the scheduler takes one unit of time as overhead for handling the process arrival. Compute the average waiting and turnaround times for First-come-first-serve (FCFS) and preemptive Shortest-job-first scheduling algorithms. Which of the algorithms is a good choice for the given scenario and why? [4+4+2=10]

Process ID	Arrival time (msec)	Burst time (msec)
A	0	3
B	1	2
C	2	1
D	3	4
E	4	5
F	5	2

9. Discuss the readers/writers problem in detail. Provide a semaphore solution for the same. You should write the pseudo-code with justification of each possible line. [2+4+4=10]

Department of Computer Science and Engineering
Indian Institute of Technology Patna
CS341: Operating System

Duration: 180 Mins | Marks: 100

1. (a) The `signal()` operations associated with semaphores and monitors are persistent and non-persistent, respectively. Justify with an example. [3]

   (b) A file is to be shared among different processes identified by a unique id, namely, *pid*. The file can be accessed simultaneously by several processes if the sum of all *pid*s associated with all the processes currently accessing the file must be less than *thres*. For the given monitor, complete the procedure `file_access()` and `finish_access()` to coordinate the file access. You may assume `c.signal()` wakes up valid process(s). [4+3]

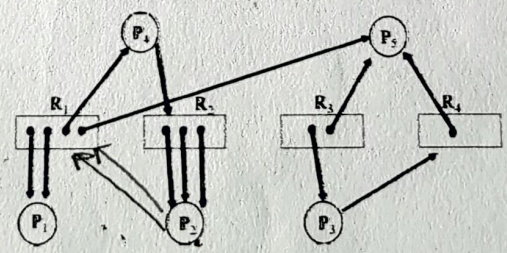
```
monitor_file_access {
    int curr_pid_sum = 0; \\ to store the current sum of pids
    int thres;
    condition c; \\ condition variable for process synchronization
    void file_access(int pid) {
        ..........................
    }
    void finish_access(int pid) {
        ..........................
    }
}
```

2. (a) How ordinary pipes are different from named pipes. Explain the mechanism of enabling two-way communication using ordinary pipes. [2+1]

   (b) Explain the context-switching mechanism using a suitable diagram. Consider a system running ten I/O-bound tasks and one CPU-bound task. Assume that the I/O-bound tasks issue an I/O operation once for every millisecond of CPU computing and that each I/O operation takes 10 milliseconds to complete. Also, assume that the context switching overhead is 0.1 millisecond and that all processes are long-running tasks. What is the CPU utilization for a round-robin scheduler if the time quantum is 10 milliseconds? [4+3]

3. (a) Explain the process life cycle with a suitable diagram and explicitly mention the event that causes the process to move from one state to another. [3]

   (b) Given the following resource allocation graph. [3+3+1]



   (i) Check whether the system is in a deadlock state or not. If not, find the safe sequence.

   (ii) Let the process P_2 also request for 2 instances of resource R_1, does the system enter a deadlock? Why?

   (iii) If at all deadlock occurs in parts (i) and/or (ii), find the set of the process(s) to be killed for deadlock resolution.

4. (a) Explain four necessary conditions for deadlock to occur. Which of these four conditions cannot be denied to prevent deadlock? Explain briefly. [3+2]

   (b) Write the banker's safety algorithm and discuss its time complexity. [3+2]

5. (a) What is the cause of thrashing? How does the system detect thrashing? Once it detects thrashing, what can the system do to eliminate this problem? [2+2+1]

   (b) Assume we have a demand-paged memory. The page table is held in registers. It takes 8 milliseconds to service a page fault if an empty page is available or the replaced page is not modified and 20 milliseconds if the replaced page is modified. Memory access time is 100 nanoseconds. Assume that the page to be replaced is modified 70 percent of the time. What is the maximum acceptable page-fault rate for an effective access time of no more than 100 nanoseconds? [5]

6. Suppose that a disk drive has 5000 cylinders numbered 0 to 4999. The drive currently serves a request at cylinder 143, and the previous request was at cylinder 125. The queue of pending requests, in FIFO order, is 86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130. Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests for each of the following disk-scheduling algorithms? [2*5]

   (i) FCFS (ii) SSTF (iii) SCAN (iv) LOOK (v) C-SCAN

7. (a) What is the copy-on-write features; Under what circumstances is it beneficial to use this feature? Differentiate between `fork()` and `vfork()` system call. [3+2]

   (b) Consider the following segment table; what are the physical addresses for the following logical addresses? [1*5]

Segment	Base	Length
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

   (i) 0, 430 (ii) 1, 10 (iii) 2, 500 (iv) 3, 400 (v) 4, 112

8. (a) Consider the following page reference string:
   7, 2, 3, 1, 2, 5, 3, 4, 6, 7, 7, 1, 0, 5, 4, 6, 2, 3, 0, 1. Assuming the process starts executing after loading page 7 followed by page 1.Further, three frames are allocated to the process. How many page faults would occur for the LRU and Second Chance page replacement algorithms? [3+3]

   (b) Consider a paging system with the page table stored in memory. [1+3]

   (i) If a memory reference takes 200 nanoseconds, how long does a paged memory reference take?

   (ii) If we add associative registers and 75 percent of all page-table references are found in the associative registers, what is the effective memory reference time? (Assume that finding a page-table entry in the associative registers takes zero time if the entry is there.)

Operating System Lab

End Semester Exam (SET-1)

Write a C program to implement the concept of multithreading. In this program you have to create two threads, the first thread intended to calculate the average waiting time (AWT) and average turnaround time (ATT) using First Come First Serve (FCFS) scheduling algorithm. Further, the second thread implement the Shortest Remaining Time First (SRTF) scheduling algorithm to compute AWT and ATT. Finally, both the threads return the computed AWT to the main process for display the final outputs.

Input:

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

Desired Output Format:

```
PS C:\Users\Dell\Desktop\Endsem_OS> gcc -pthread q1.c -o  q1
PS C:\Users\Dell\Desktop\Endsem_OS> ./q1
Enter the number of processes: 5
Enter arrival time and burst time for each process:
Process 1 Arrival Time: 2
Process 1 Burst Time: 6
Process 2 Arrival Time: 5
Process 2 Burst Time: 2
Process 3 Arrival Time: 1
Process 3 Burst Time: 8
Process 4 Arrival Time: 0
Process 4 Burst Time: 3
Process 5 Arrival Time: 4
Process 5 Burst Time: 4
Average Waiting Time: 8.000000
Average Turnaround Time: 12.600000
Average Waiting Time: 4.600000
Average Turnaround Time: 9.200000
```

Code Skeleton:

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
// Process structure to store information about each process
struct Process {
    int id;
    int arrivalTime;
    int burstTime;
};
// Structure to pass data to the thread
struct ThreadData {
    struct Process *processes;
    int n;
};
// Function to calculate average waiting time and average turnaround time
void calculateAverages(struct Process *processes, int *waitingTime, int *turnaroundTime, int
    ____________________
}
// Function to implement FCFS (First-Come-First-Serve) scheduling algorithm
void *fcfs(void *arg) {
    ____________________
    calculateAverages(processes, waitingTime, turnaroundTime, n);
    ____________________
    ____________________
}
// Function to implement SRTF (Shortest Remaining Time First) scheduling algorithm
void *srtf(void *arg) {
        ____________________
}
// Function to implement FCFS (First-Come-First-Serve) scheduling algorithm
void *fcfs(void *arg) {
    ____________________
    ____________________
    calculateAverages(processes, waitingTime, turnaroundTime, n);
    ____________________
  // Free allocated memory
    free(remainingTime);
    free(waitingTime);
    free(turnaroundTime);
    pthread_exit(NULL);
}
int main() {
    int n;
    printf("Enter the number of processes: ");
    scanf("%d", &n);
        ____________________
}
```

Mid-Term Test

CS342: Operating Systems; Set-2

Q1> Write a C program to search an element in an array. If the element is found, in the array then program should print that number otherwise the associated process should be killed using SIGTERM. The input must be taken from command line.

Q2> Write a program that computes the cube roots of the integers from 0 to 999 in a separate thread and returns an array of doubles containing the results. In the meantime the main thread should display a short message "Computation under process. Please wait ..." to the user and then display the results of the computation when they are ready.

Q3> Write a C program mthread.c to compute the sum for the first N natural numbers. The user provides the *N as a command line argument*. In the program, create two threads, namely t1 and t2. Both the threads should execute the same function add( ). The t1 thread should compute the sum of the first half of natural numbers and update the global variable result; similarly, t2 should compute the sum of the second half of natural numbers and update the global variable result. The parent process should wait for the completion of both the threads and print the final result as output. To avoid the overwriting issue among the threads, use MUTEX.

- The function prototype for SIGINT signal is as follows:

  void signal_handler(int signum);
  To Register the SIGINT signal handler:
  signal(SIGINT, signal_handler);

  This prototype defines a function named signal_handler that takes an integer argument signum, which represents the signal number. In the case of SIGINT (interrupt signal, typically generated by pressing Ctrl+C in the terminal), signum will be set to SIGINT (which is usually defined to 2).

  void signal_handler(int signum);
  To Register the SIGTERM signal handle:
  signal(SIGTERM, signal_handler);

- The function prototype for pthread_create is as follows:

```
#include <pthread.h>

int pthread_create(pthread_t *thread, const pthread_attr_t *attr,

        void *(*start_routine) (void *), void *arg);
```

- The function prototype for pthread_join is as follows:

```
#include <pthread.h>


    int pthread_join(pthread_t thread, void **retval);
```