

Course Code: CS2006	Course Name: Operating System
Instructor Names: Dr. Ghufraan Ahmed, Anaam Hamid and Safia	
Student Roll No:	Section No:

Instructions:

- Return the question paper.
- Read each question completely before answering it. There are **4 questions** and **2 pages**.
- All the answers must be solved according to the sequence given in the question paper.

Time: 60 minutes.

Max Marks: 50

Question no. 01

[Marks: 4+4+5 = 13]

- a. Should an operating system include applications like web browsers and mail programs? Give reasons that support your answers: why such programs should/ shouldn't be part of an OS?
- b. How will you differentiate between a kernel mode and a user mode in terms of the basic principle of protection (security)? Identify the mode in which the following instructions should run.
 1. Set value of timer.
 2. Read the clock.
 3. Clear memory.
 4. Issue a trap instruction.
 5. Turn off interrupts.
 6. Modify entries in device-status table.
 7. Switch from user to kernel mode.
 8. Access I/O device
- c. Differentiate between a client-server model and a peer-to-peer model in a distributed system.

Question no. 02

[Marks:4+4+4=12]

- a. What are the advantages and disadvantages of using a layered approach in designing an operating system?
- b. What system call(s) must be executed by a command line interpreter or shell to start a new process on a UNIX system?
- c. Explain Asymmetric and Symmetric multiprocessing with examples.

Asymmetric Multiprocessing system is a multiprocessor computer system where not all the multiple interconnected central processing units (CPUs) are treated equally. In

asymmetric multiprocessing, only a master processor runs the tasks of the operating system.

For example, AMP can be used in assigning specific tasks to CPU based on priority and importance of task completion.

Symmetric Multiprocessing is a type of multiprocessing where each processor is self-scheduling.

For example, SMP applies multiple processors to that one problem, known as parallel programming.

Question no. 03

[Marks:10]

Consider the following program code. Answer the following questions:

```
1 #include<sys/types.h>
2 #include <stdio.h>
3 #include <unistd.h>
4 #include <sys/wait.h>
5 int main() {
6
7 pid_t pid;
8 pid = fork();
9     if (pid < 0) {
10         fprintf(stderr, "Fork Failed");
11         return 1;
12     }
13     else if (pid == 0) {
14         execlp("/bin/ls", "ls", NULL);
15         printf("LINE J");
16     }
17     else {
18         wait(NULL);
19         printf("Child Complete");
20     }
21     return 0;
22 }
```

- a. Identify ONLY the system call(s) with line number(s).
- Line 8 -fork()
 - Line 10-fprintf()
 - Line 11-return
 - Line 14 -execlp()
 - Line 15 -printf()
 - Line 18 -wait()
 - Line 19 -printf()
 - Line 21-return
- b. What will be the output of the code?
- Print
1. List of all files and directories of present working directory.
 2. "Child Completed"
- c. What will be the output if we remove the wait (NULL) command from line 18?
- Parent process execute itself earlier and then child executed.
- Output:
- Print
1. "Child Completed".
 2. List of all files and directories of present working directory.

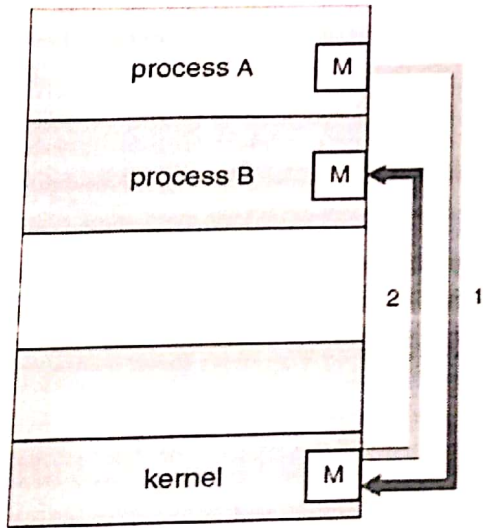
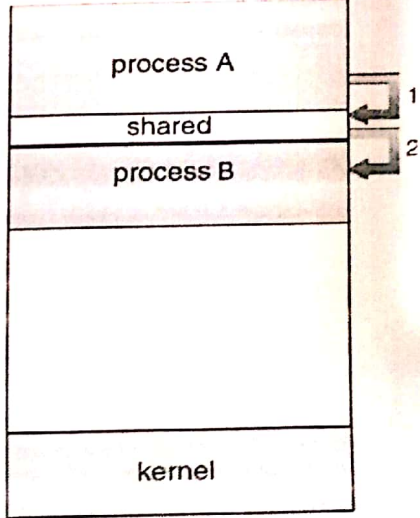
- d. Which lines of code will be executed by the child process and the parent process?
 Line 14 and 15 – Executed by Child.
 Line 18 and 19 – Executed by Parent.
- e. When will "LINE J" be printed?
 It is executed in the child process if the `execvp()` system call fails. And if it is executed, it never print.

Question no. 04

[Marks:10+5=15]

- a. Describe the following. You may elaborate your answer via diagram.

1. Preemptive and Non-Preemptive Scheduling.
2. Inter process Communication models.
3. Short Term Scheduling vs Long Term Scheduling.
4. PCB and Context Switching.
5. Dispatch Latency vs throughput.

Preemptive and Non-Preemptive Scheduling	
Preemptive scheduling allows a running process to be interrupted by a high priority process	non-preemptive scheduling, any new process must wait until the running process finishes its CPU cycle.
Inter process Communication models	
(a) Message passing. (b) shared memory.	
<div style="display: flex; justify-content: space-around; align-items: flex-start;"> <div style="text-align: center;">  <p>(a)</p> </div> <div style="text-align: center;">  <p>(b)</p> </div> </div>	
Short Term Scheduling vs Long Term Scheduling	
Short-term scheduler selects from among the processes in ready queue, and allocates the CPU to one of them	Long-term scheduler (or job scheduler) – selects which processes should be brought into the ready queue
PCB and Context Switching	
A process control block (PCB) is a data structure used by computer operating systems to store all the information about a process	When CPU switches to another process, the system must save the state of the old process and load the saved state for the new process via a context switch

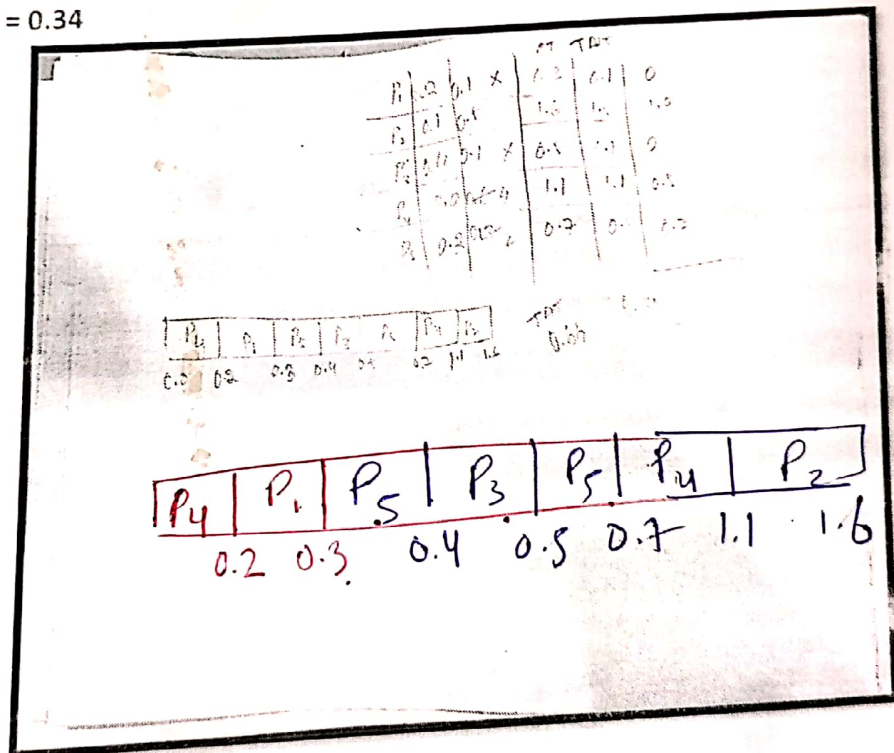
Context of a process represented in the PCB	
Dispatch Latency vs throughput	
Dispatch latency – time it takes for the dispatcher to stop one process and start another running	Throughput – Number of processes completed per unit time. May range from 10 / second to 1 / hour depending on the specific processes

- b. Draw Gantt charts that illustrate the execution of the processes listed in Table. Calculate the **average turnaround time** and the **average waiting time** by Preemptive version of the **SJF algorithm**.

Process Burst Time						
Process	Arrival Time	Burst Time	Priority	CT	TAT	WT
P11	0.2	0.1	2	0.3	0.1	0.0
P12	0.1	0.5	4	1.6	1.5	1.0
P13	0.4	0.1	3	0.5	0.1	0.0
P14	0.0	0.6	1	1.1	1.1	0.5
P15	0.2	0.3	5	0.7	0.5	0.2

Avg TAT = 0.66

Avg WT = 0.34



*****Best of Luck*****

9/30

