ISLAMIC UNIVERSITY OF TECHNOLOGY (IUT)
ORGANISATION OF ISLAMIC COOPERATION (OIC)
Department of Computer Science and Engineering (CSE)

SEMESTER FINAL EXAMINATION SUMMER SEMESTER, 2018-2019
DURATION: 3 Hours FULL MARKS: 150

CSE 4673: Operating Systems

Programmable calculators are not allowed. Do not write anything on the question paper.
There are **8 (eight)** questions. Answer any **6 (six)** of them.
Figures in the right margin indicate marks.

---

1. a) What are the conditions of deadlock? Deadlock prevention works by making sure that certain conditions necessary for deadlock do not arise. Explain each possible method of deadlock prevention along with its associated deadlock causing condition. 16

   b) Consider a scenario where a program has approximately 60% serial and 40% parallel processing code. What is the highest speedup we can achieve using a multicore processing architecture? How many processor cores do we need to achieve a speedup of 1.25 times? 3+6

2. a) Consider the following snapshot of the system: 14

Table 1: Snapshot of the system

Process	Allocation				Max				Available			
	A	B	C	D	A	B	C	D	A	B	C	D
P_0	2	0	0	1	4	2	1	2	3	3	2	1
P_1	3	1	2	1	5	2	5	2				
P_2	2	1	0	3	2	3	1	6				
P_3	1	3	1	2	1	4	2	4				
P_4	1	4	3	2	3	6	6	5				

Use Banker's algorithm to answer the following questions:

i. Create and show the NEED matrix for the snapshot of the system available.
ii. Illustrate that the system is in a safe state by demonstrating an order in which the processes may complete.
iii. If a request from process P_1 arrives for (1,1,0,0), can the request be granted immediately?
iv. If a request from process P_4 arrives for (0,0,2,0), can the request be granted immediately?

   b) With the help of figures, show the components of a process control block along with a short description of each of their functionalities. 11

3. a) Consider the following scenario where there are 4 processes that appear at the same time to be processed: 18

Table 2: Process along with burst-times

Process	Burst Time
P_1	8
P_2	6
P_3	10
P_4	5

For this scenario, select an appropriate time quantum for a Round Robin scheduling algorithm. Justify your choice of a time quantum. Show the scheduling process for the Round Robin algorithm using a Gantt chart. Do the same for a Shortest Job first scheduling. Calculate the average waiting time and average turnaround times for both of the algorithms.

   b) How does thread creation work in UNIX? How does process termination work? 3+4

4. a) Draw resource allocation graphs to show an instance where there is no deadlock in the system. Draw another graph that shows a deadlocked system. 4+4

b) "The many to many multithreading model provides sufficient concurrency without overcrowding the system with threads"- justify this statement. 6

c) "A large time quantum essentially turns a Round Robin scheduling into a First Come First Serve scheduling"- do you agree with this statement? With the help of pictures justify your choice. 8

d) Why are multicore processing systems better than multiprocessor systems? 3

5. a) With the help of figures, explain Mac OSX's hybrid structure. 10

b) With the help of figure, show the individual parts of dispatch latency. 6

c) Suppose that a system is in an unsafe state. Show that it is possible for the processes to complete their execution without entering a deadlock state. 9

6. Consider two processes P_1 and P_2, where $p_1=50$, $t_1=25$, $p_2=75$, $t_2=30$.

a) Can these two processes be scheduled using rate-monotonic scheduling? Illustrate your answer using a Gantt chart. 8

b) Illustrate the scheduling of these processes using Earliest Deadline First scheduling. 8

c) For this scenario, under what circumstances is rate-monotonic scheduling inferior to earliest-deadline-first scheduling in meeting the deadlines associated with processes? 9

7. a) Distinguish between the client–server and peer-to-peer models of distributed systems. 9

b) Take a look at the following code: 6

```
#include <stdio.h>
#include <unistd.h>
int main()
{
/* fork a child process */
fork();
/* fork another child process */
fork();
/* and fork another */
fork();
return 0;
}
```

Figure 1: Code listing 1

Including the initial parent process, how many processes are created by the program shown in Figure 1. Justify your answer by drawing a tree to show the process creation process.

c) Write short notes on the following: 2×5

i. Thread Pools ii. Multilevel feedback queue

8. a) Briefly explain the advantages of Multiprocessing. Distinguish between Symmetric and Asymmetric Multiprocessing. 4+4

b) The traditional UNIX scheduler enforces an inverse relationship between priority numbers and priorities: the higher the number, the lower the priority. The scheduler recalculates process priorities once per second using the following function: 6

$$\text{Priority} = (\text{recent CPU usage} / 2) + \text{base}$$

where base = 60 and recent CPU usage refers to a value indicating how often a process has used the CPU since priorities were last recalculated. Assume that recent CPU usage for process P_1 is 40, for process P_2 is 18, and for process P_3 is 10. What will be the new priorities for these three processes when priorities are recalculated? Based on this information, does the traditional UNIX scheduler raise or lower the relative priority of a CPU-bound process?

c) What is storage hierarchy? Explain with the help of figures. 4

d) Write the pseudo-code for the safety algorithm. 7