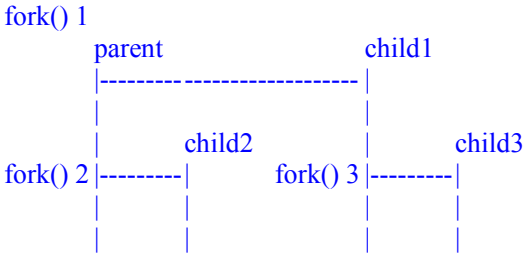
KING SAUD UNIVERSITY
COLLEGE OF COMPUTER AND INFORMATION SCIENCES
COMPUTER SCIENCE DEPARTMENT

CSC 227: Operating Systems **Mid Term 2 Exam** **Summer 2009**
Date: Aug. 22, 2009 **Time: 14:30 – 16:00**
Student Name: …… **ID#: ……** **Section#: ……**
This examination is close notes and close book.

QUESTION 1 [7 marks]

1. [2 marks] Draw the process tree generated by the code segment in this box.

```
int k;
 k = fork (); /* 1 */
 if (k > 0)
   k = fork ( ); /* 2 */
 if (k = = 0) fork ( ); /* 3 */
```



1. [0.5 mark] What is an orphaned process?

An orphaned process is a process whose parent process has terminated.

2. [0.5 mark] What is a zombie process?

A zombie process is one that is waiting for its parent to accept its termination code. Zombie processes might accumulate if the program running in the parent process is poorly written and never calls wait () to pick up the child's termination signal.

3. [1mark] How can a parent find out how its children died?

The wait () system call returns the exit code of the child process.

4. [1 mark] Give one item of a PCB of a process that is specifically needed by a time-sharing system

Memory management information, CPU scheduling information …

5. [2 marks] Consider two scenarios: a) two user threads are mapped into one kernel thread, and b) each of the two user threads is mapped into a unique kernel thread.
   Which achieves better concurrency in execution? Why?

The b) achieves better concurrency since two threads can execute concurrently.
In a), when one thread is blocked or in waiting state, the other thread can not be scheduled to run.

QUESTION 2 [6 marks]

1. [1 mark] What is a critical section?

Each of N processes has a segment of code, called critical section, in which the process may change common variables, updating a table, writing a file …

2. [1 mark] Peterson's solution, TestAndSet and Swap instructions can be used to protect a critical section. What is their main problem?

These solutions to the critical-section problem rely on busy-waiting loops.

3. [2 marks] Consider two processes Pa and Pb using two semaphores S and Q initialized to 1. S and Q are implemented with waiting queues.

Pa	Pb
wait (S);	wait (Q);
wait (Q);	wait (S);
…	…
signal (S);	signal (Q);
signal (Q);	signal (S);

What situation may occur when Pa and Pb are running?

Pa may wait in the queue Q for a signal from Pb which may be blocked in the queue S … Deadlock.

4. [2 marks] Consider the producer-consumer problem with a bounded buffer. Solve this problem using a semaphore S. Give a pseudo-code for the producer and the consumer.

```
/* Producer */
 while (true) {

        /*  produce an item and put in nextProduced  */
             while (count == BUFFER_SIZE)
                        ; // do nothing
                  buffer [in] = nextProduced;
                  in = (in + 1) % BUFFER_SIZE;
                  Wait (S)
                  count++;
                  Signal (S)
}

/* Consumer */
while (true)  {
              while (count == 0)
                       ; // do nothing
                       nextConsumed =  buffer[out];
                        out = (out + 1) % BUFFER_SIZE;
                    Wait (S)
                    count--;
                    Signal (S)

          /*  consume the item in nextConsumed
          }
```

QUESTION 3 [7 marks]

1. Five processes, A trough E, arrive at the same time. They have estimated running times of 10, 6, 2, 4, and 8 milliseconds, respectively. Their priorities are 3, 5, 2, 1, and 4, respectively, with 5 being the highest priority. Assume that all processes are completely CPU bound.
   For each of the following scheduling algorithms, draw the Gantt chart and determine the average process turnaround time. Ignore process switching overhead.

1.
   (a) 30, 22, 6, 16, 28. avg(turnaround time)= 20.4 milliseconds
   (b) 6, 14, 24, 26, 30. avg(turnaround time)= 20 milliseconds
   (c) 10, 16, 18, 22, 30. avg(turnaround time)= 19.2 milliseconds
   (d) 2, 6, 12, 20, 30. avg(turnaround time)= 14 milliseconds

2. [1 mark] What are the functions of long-term and short-term schedulers?
Job Scheduler – CPU Scheduler.

3. [1 mark] In what way is SJF Optimal?
2. Shortest remaining Time First (Preemptive SJF).

4. [1 mark] Why is it hard to implement SJF as CPU-scheduler?
Prediction of the next CPU burst.