Midterm Test

(200 points)

This exam has a total of 10 questions, spread over 7 pages, including this cover page. Page8 is provided for rough work.

Date: October 14, 2014

Student Name: .....

Student ID: .....

You are allocated a maximum amount of space to answer each question. (We have provided sufficient lines.) Adhere to those limitations when you formulate your answers. Page8 is provided as extra space for rough work as needed. It will not be graded.
Make an effort to write in a readable fashion. We will skip over (and therefore not grade) non-readable portions.

"I have adhered to the Aggie Code of Honor."

Signature:
.....

1	(33)
2	(12)
3	(15)
4	(20)
5	(15)
6	(10)
7	(30)
8	(20)
9	(15)
10	(30)
T	(200)

1. [33 points: 3 points for each item] Match each term in the left column to the definition/description in the right column that **fits best**. Do this by filling in the void entries on the left. If you are not sure about your choice, be sure to state your assumption and reasoning carefully in space provided.

( )	Critical section	(A)	Coordination between threads
( )	Semaphore	(B)	Could result in starvation
( )	Shortest job first scheduling	(C)	Routine in the Kernel to process an Interrupt
( )	Context switching	(D)	Refers to the number of processes in memory
( )	Asynchronous	(E)	Its timing is not controlled by the system
( )	Ready queue	(F)	executes as a single, uninterruptible unit
( )	Round Robin Scheduling	(G)	Used as a primitive for controlling access to a critical section
( )	Degree of multiprogramming	(H)	refers to where a process is accessing/updating shared data
( )	Handler	(I)	saves the state of the currently running process and restores the state of the next process to run
( )	Synchronization	(J)	contains threads that are waiting to execute on CPU
( )	Atomic instruction	(K)	CPU allocation with time slices

2. [12 points: 3 points for each item] Circle the following statements TRUE or FALSE. If you are not sure about your choice, be sure to state your assumption and reasoning carefully in space provided.

T F A race condition results when several threads try to access and modify the same data concurrently.

T F Priority inversion occurs when a higher-priority process needs to access a data structure that is currently being accessed by a lower-priority process.

T F The stack of a process contains temporary data such as function parameters, return addresses, and local variables.

T F The exec() system call creates a new process.

3. [15 points: 3 points for each item] Which of the following actions should be allowed only in kernel privileged mode? (circle correct answer)

(a) Masking of an Interrupt	YES NO
(b) Exception Handling	YES NO
(c) String search in a file	YES NO
(d) Creation of a file	YES NO
(e) OUTS (output string) instruction	YES NO

4. [20 points] In the following piece of code, what will the output be at Lines X and Y?

```
#include  <sys/types.h>
#include  <stdio.h>
#include  <unistd.h>
#define SIZE 5
int nums[SIZE] = {0,1,2,3,4};
int main()
{
   int i;
   pid_t pid;
   pid = fork();
   if (pid == 0) {
   for (i = 0; i < SIZE; i++) {
      nums[i] *= -i;
      printf("CHILD: %d ",nums[i]); /* LINE X */
      }
   }
   else if (pid > 0) {
   wait(NULL);
   for (i = 0; i < SIZE; i++)
      printf("PARENT: %d ",nums[i]); /* LINE Y */
   }
   return 0;
}
```

4a. [10 points] The output at Line (X) is _____

Note: Space left for explanation if needed

4b. [10 points] The output at Line (Y) is _____

Note: Space provided for explanation if needed

5. [15 points] Explain the steps that an operating system goes through when the CPU receives an interrupt. Be careful in explicitly listing the steps in the right sequence.

6. [10 points] What advantage is there in having different time-slice sizes at different levels of a multi-level feedback queue scheduling system? Illustrate your answer with an example.

7. [30 POINTS] Consider the following set of processes, with the length of the CPU burst time given in milliseconds:

Process	BurstTime	Priority
P1	2	2
P2	1	1
P3	8	4
P4	4	2
P5	5	3

The processes are assumed to have arrived in the order P1, P2, P3, P4, P5 all at time 0. We are going to schedule the execution of these processes using the following scheduling algorithms: First Come First Served (FCFS), Shortest Job First (SJF), Round Robin (RR), and a new type of scheduling algorithm called non-preemptive priority scheduling.

Assumptions and Clarifications:

a. Assume the quantum (aka time slice) of 2 for RR algorithm
b. Non-Preemptive means once scheduled, a process cannot be preempted (i.e. it runs to completion). Assume a larger priority number implies a higher priority.

7a. [10points] What is the turnaround (completion) time of each process for each of the scheduling algorithms. Write the times in the table below.

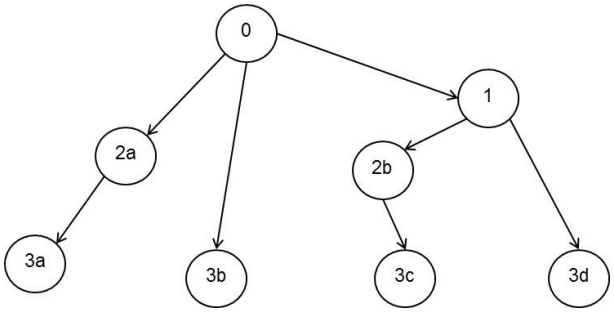
	FCFS	SJF	Priority	RR
P1				
P2				
P3				
P4				
P5				

7b. [10 points] What is the waiting time of each process for each of these scheduling algorithms?

	FCFS	SJF	Priority	RR
P1				
P2				
P3				
P4				
P5				

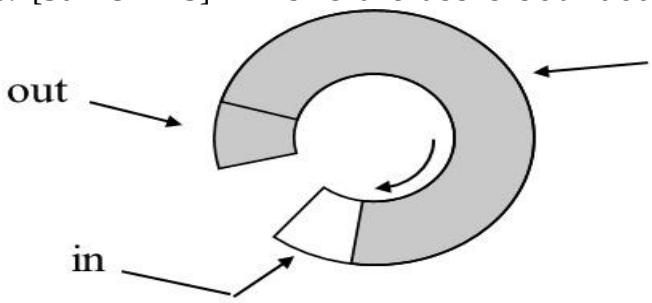
7c. [10 points] Which of the algorithms results in the min. avg. waiting time (over all processes)? Why?

8. [20 points] Write a program to build the process tree shown in the figure below. Pseudo code is acceptable.



9. [15 points] Race conditions are possible in many computer systems. Consider a banking system that maintains an account balance with two functions: *deposit(amount)* and *withdraw(amount).* These two functions are passed the amount that is to be deposited or withdrawn from the bank account balance. Assume that a husband and wife share a bank account. Concurrently, the husband calls the *withdraw()* function and the wife calls *deposit().* **Describe how a race condition is possible and propose a solution with your favorite synchronization primitive discussed in class to prevent the race condition from occurring**.

10. [30 POINTS] This is a classic bounded-buffer producer-consumer problem.



Producer:

```
void deposit(Item * next) {
  while (counter == n) no_op;
  buffer[in] = next;
  in = (in+1) MOD n;
  counter = counter + 1;
}
```

Consumer:

```
Item * remove() {
  while (counter == 0) no_op;
  next = buffer[out];
  out = (out+1) MOD n;
  counter = counter - 1;
  return next;
}
```

10a. [10pts] Identify the race condition in above implementation

10b. [20pts] Propose a solution for the above problem using Semaphores.

THIS PAGE INTENTIONALLY LEFT BLANK TO SERVE AS SPACE FOR ROUGH WORK. DO NOT USE THIS PAGE TO WRITE ANSWERS.