CIS*3110 – Operating Systems
Mid-Term Exam
Winter 2016

Name: _____

ID #: _____

Instructions:

- This is a closed book exam.
- Please write all answers in the spaces provided. If you use the back of a page please indicate this clearly for the marker.
- There are 7 pages (including this cover page) in this exam and 4 sections. Please check that your exam booklet is complete.
- Please write clearly. If the marker cannot read the answer then the grade will be 0.

Question Section	Marks	Out Of
1		25
2		25
3		25
4		25
Total		100

Question Section 1 [25 marks total] Short Answers

(1.1) [5 marks] What are the key benefits of threads?

- Far less time to create/terminate
- Switching between threads is faster
- No memory management issues
- Can enhance communication efficiency

[2 marks for 2 correct benefits plus 1 mark for a third]

(1.2) [5 marks] In operating system design, what is the difference between policy and mechanism? Give an example.

Mechanisms determine ***how*** to do something; policies determine ***what*** will be done.

Example: using a timer in a scheduling algorithm for a wait is policy but the actual timing used is a mechanism. (others can be appropriate – use judgment as to whether the example is reasonable)

]3 marks for the definition and 2 for an example]

(1.3) [5 marks] What are the three requirements a solution to the critical-section problem must satisfy?

- Mutual Exclusion
- Progress
- Bounded Waiting

[2 marks for 2 correct requirements plus 1 mark for a third]

(1.4) [5 marks] What are the two operations that can be performed on a semaphore?

A **semaphore** S is an integer variable that is accessed only through two standard atomic operations: wait() and signal().

The definition of wait() is as follows:

```
wait(S) {
  while (S <= 0) ;  // busy wait
  S--;
}
```

The definition of signal() is as follows:

```
signal(S) {
  S++;
}
```

[4 marks for identifying wait and signal and 1 more mark for any type of explanation]

(1.5) [5 marks] What is a Process Control Block (PCB)?

A Process Control Block is a collection of process attributes including **[2 marks]**

- Process identification **[1 mark]**
- Process state information **[1 mark]**
- Process control information including **[1 mark]**
  - Scheduling and state information
  - Privileges
  - Memory management
  - Resource ownership and utilization

Question Section 2: Programming [25 marks total]

Fork is the primary method of process creation on POSIX operating systems.

(a) [10 marks] Describe the behavior of *fork()*.

The call to the fork() system call **makes a complete copy of the running (parent) process**. **[6 marks]**

The only difference between the calling parent process and the newly-created child is the fork() return value

- in the parent process, the return value is the process ID of the child **[2 marks]**
- in the child process, the return value is 0 **[2 marks]**

(b) [15 marks] Write a short C program to create (*fork()*) processes which represent a grandfather-father-son relationship.

```
#include <stdlib.h>

main(){
  int pid;

  if  ( (pid = fork()) != 0 ) {          /* grandfather process */
        printf ( "grandfather\n" );      /* or do something */
        wait ( 0 );
  } else if  ( (pid = fork()) != 0 ) {   /* father process */
        printf ( "father\n" );           /* or do something */
        wait ( 0 );
  } else {                               /* son process  */
        printf ( "son\n" );              /* or do something */
  }
  exit ( 0 );
}
```

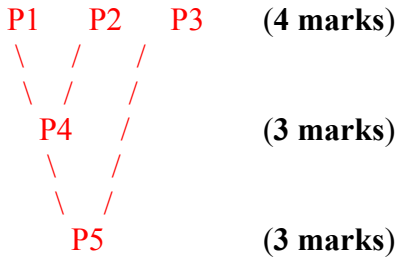
1) **A correct C program [3 marks]**
2) **Correct relationship [9 marks]**
3) **Proper waiting [3 marks]**

Question Section 3: Semaphores [25 marks total]

Consider the following five processes. Assume that *a* to *f* and *temp1* to *temp5* are all shared variables.

P_1: temp1 = a + b;
P_2: temp2 = c + d;
P_3: temp3 = e / f;
P_4: temp4 = temp1 * temp2;
P_5: temp5 = temp4 - temp3;

(a) [10 marks] Draw the process flow graph for these five processes, such that concurrency is maximized.



(**4 marks**)

(**3 marks**)

(**3 marks**)

(b) [15 marks] Assume that all five processes are executing concurrently. Use semaphores to synchronize these processes (as necessary) according to your process flow graph. In other words, add **wait / signal** operations (as needed) around *temp1* to *temp5*, in order to ensure correct execution of the concurrent processes. For each semaphore used, state the initialization value.

Semaphore S1 = S2 = 0;

(**3 marks**) P1: temp1 = a + b; **Signal (S1)**
(**3 marks**) P2: temp2 = c + d; **Signal (S1)**
(**3 marks**) P3: temp3 = e/f; **Signal (S2)**
(**3 marks**) P4: **Wait (S1) Wait (S1)** temp4 = temp1 + temp 2; **Signal (S2)**
(**3 marks**) P5: **Wait (S2) Wait (S2)** temp5 = temp4 – temp 3;

OR

Semaphore S1 = S2 = S3 = S4 = 0;

P1: temp1 = a + b; **Signal (S1)**
P2: temp2 = c + d; **Signal (S2)**
P3: temp3 = e/f; **Signal (S3)**
P4: **Wait (S1) Wait (S2)** temp4 = temp1 + temp 2; **Signal (S4)**
P5: **Wait (S3) Wait (S4)** temp5 = temp4 – temp 3;

Question Section 4: Scheduling [25 marks total]

Consider the following set of processes, with the length of the CPU-burst time given in some time unit:

Process	**Burst Time**	**Priority**
P_1	10	3
P_2	1	1
P_3	2	3
P_4	1	4
P_5	5	2

The processes are assumed to have arrived in the order P_1, P_2, P_3, P_4, P_5 all at time 0.

(a) [15 marks] Draw where the processes execute on the four Gantt charts using:

- First Come First Serve (FCFS)
- Shortest Process Next (SPN) [non-preemptive]
- PRIORITY scheduling (a smaller priority number implies a higher priority) [non-preemptive]
- Round Robin (RR quantum = 1 time unit) scheduling

First Come First Serve

P1	P1	P1	P1	P1	P1	P1	P1	P1	P1	P2	P3	P3	P4	P5	P5	P5	P5	P5
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

Shortest Process Next

P2	P4	P3	P3	P5	P5	P5	P5	P5	P1	P1	P1	P1	P1	P1	P1	P1	P1	P1
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

Priority

P2	P5	P5	P5	P5	P5	P1	P1	P1	P1	P1	P1	P1	P1	P1	P1	P3	P3	P4
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

Round Robin

P1	P2	P3	P4	P5	P1	P3	P5	P1	P5	P1	P5	P1	P5	P1	P1	P1	P1	P1
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19

-5 for incorrect sequence to a minimum of 0

(b) [10 marks] What are the turnaround times of each process for each of the scheduling algorithms in part **(a)**?

	FCFS	SPN	Priority	RR
P_1	10	19	16	19
P_2	11	1	1	2
P_3	13	4	18	7
P_4	14	2	19	4
P_5	19	9	6	14

-0.5 for any wrong answer

Extra Space