CSC-246: Concepts and Facilities of Operating Systems for Computer Scientists (Section 001)

Sample Midterm Exam

Name _____ Unity ID _____

Please read the following instructions carefully:

- This exam is closed book and closed notes.
- Your work must be individual. Cheating will be punished according to university academic integrity code.

About this exam:

- The total point value of the exam is 100.
- Problem 1 is a list of multiple-choice questions.
- Problem 2 is a list of short answer questions.
- Problems 3-5 are problems require longer answers.

Problem	Possible Points	Your Points
1. Multiple Choice	10	
2. Short Answer	20	
3. Processes	20	
4. Threads	20	
5. Scheduling	30	
Total	100	

1. Multiple Choice Questions (10 pts, 2 pts each)

Mark the **best** answer for each of the following questions.

a) In a modern operating system, the kernel is _____

A. the collection of library code that's stored in a process' memory.
B. the layer of software that can access hardware directly.
C. the program that runs at boot time to load the rest of the OS into memory.
D. the portion of the OS that's stored in ROM or EEPROM.

b) On an interactive computer system, a modern operating system would normally depend on _____ to ensure that one process doesn't use all the CPU time.

A. a timer interrupt
B. periodic calls to sched_yield()
C. user-level threads
D. availability of multiple cores

c) A system might use a multilevel queue with feedback in order to _____

A. prevent starvation.
B. schedule processes differently based on their behavior.
C. keep multiple CPU cores busy.
D. be able to quickly select a runnable process.

d) The contention scope for a thread identifies _____

A. when other threads can terminate it.
B. how its stack space is related to that of other threads in the same process.
C. what portions of memory may be accessed by other threads at the same time.
D. what threads it competes with for CPU time.

e) In general, it's impossible to implement a SJF scheduler. In practice, if we want SJF behavior, we must _____

A. use I/O burst length as a predictor for CPU burst length.
B. force user processes to exhibit particular burst lengths.
C. estimate burst length based on previous CPU bursts.
D. first measure the next burst length, then run it.

2. Short answer questions (20 pts, 5 pts each)

Provide brief answers to the following questions. Be as concise as possible. You should be able to answer each question with 2-4 sentences.

a) What is temporal locality? What is spatial locality?

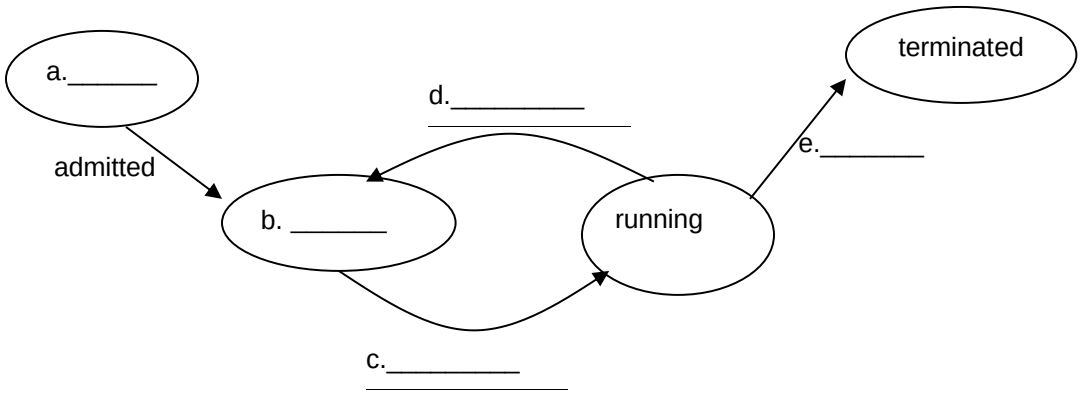
b) At this point in the semester, we are assuming that the hardware provides a very simple mechanism for memory protection. Briefly describe 1) what hardware elements are required for this mechanism and 2) how these are used during execution to provide memory protection.

c) The shortest-job-first (SJF) and shortest-time-to-completion-first (STCF) policies are both unrealistic policies to implement in a general-purpose operating system. Why?

d) List four fields that would normally appear in a Process Control Block (PCB). Mark at least one of these with a star to indicate a field that would not be in the PCB if the OS supported multi-threaded processes.

3. Processes (20 pts)

a) Fill the missing two states and three transition events of the process state chart (10 pts)



b) What is the output of the C program below?

```
1  #include <stdio.h>
2  #include <unistd.h>
3  #include <sys/types.h>
4  #include <sys/wait.h>
5  #include <stdlib.h>
6
7  int main() {
8          int a = 12;
9          int b = 9;
10         int fid = fork();
11
12         if( fid == 0) {
13                 a++;
14         }
15         else {
16                 wait(NULL);
17                 b = b - 5;
18         }
19
20         printf("program exit. a = %d, b = %d\n", a, b);
21         return 0;
22 }
```

Output (8 pts):

_____

After the parent process makes the wait() call, it will be blocked until _____ . (2 pts)

4. Threads (20 pts)

Rewrite the program given in the last question by performing the same tasks with multithreading, using the pthread library. The code framework is given below. The code should produce the same output as program of previous question (question 3). Please fill in the missing parts with numbered comments.

//----------Start of Code----------

```
#include <pthread.h>
#include <stdio.h>

typedef struct {
          int a;
          int b;
          } local_data;

void *foo(void *arg);

int main()
{
   int a = 12;
   int b = 9;
   pthread_t tid;
   pthread_attr_t attr;

   local_data local;
   local.a = a;
   local.b = b;

   pthread_attr_init(&attr);
```

// 1) create and wait for the child thread

```
    b = b - 5;
    printf("program exit. a = %d, b = %d \n", a, b);
    return 0;
}

void *foo (void *arg)
{
    int a, b;
    local_data *local = (local_data*)arg;
```

// 2) implement the rest of the foo function

```
     printf("program exit. a = %d, b = %d \n", a, b);
     pthread_exit(0);
}
```

//----------End of Code----------

5. Scheduling (30 pts)

Consider the following set of processes, with arrival times and service times provided below. CPU serves these processes either by following SJF or priority algorithm. **Assume all algorithms are preemptive**. For SJF, if two processes have the same amount of remaining service time, use priority as the deciding factor as to which process to schedule. The lower priority number, the higher priority a process has. Also, if a process arrives at time *t*, it may begin executing at time *t*.

Process	Arrival Time	Burst Time	Priority
P1	0	9	3
P2	5	4	2
P3	8	3	1

a) Draw the Gantt charts that illustrate the execution of these processes using the SJF and priority algorithms. (10 pts)

b) For each algorithm, what is the turnaround time of each process? What is the average turnaround time for each algorithm? (10 pts)

c) For each algorithm, what is the waiting time of each process? What is the average waiting time for each algorithm? (10 pts)

This is one scratch page.