CSCI 3453 – Operating System Concepts

Spring 2021
Midterm Exam

Name:

Score: / 119

This is a take home test. Please make a note of the following:

- This is open book / open notes exam.
- You can use a calculator.
- You must justify all your answers to get full credit
- You must show all your work to get full credit
- If you make any assumptions to solve a problem, clearly state your assumptions.
- Add additional sheets as needed to complete your work.

Due Date: March 14, 2021 @ 11:55 PM – You must submit the test before this deadline. I am not going to make any exception to this rule.

GOOD LUCK!

1) Multiple Choice Questions: Please select the best answer from the given list for the following statements/questions. **(1 point each)**

A) The goal of an operating system is to:
   i) Make it harder for a user to use the computer.
   ii) Minimize resource utilization.
   iii) Make the computer system convenient to use.
   iv) Use the computer hardware in an inefficient manner.

B) A ___ provides the programmer with an API for creating and managing threads.
   i) thread library
   ii) thread GUI
   iii) thread keeper
   iv) thread maker

C) Information associated with Process control block is:
   i) process state
   ii) process number
   iii) program counter
   iv) memory limits
   v) all of the above

D) If the parent process terminates before the termination of the child process then the child process becomes a(n):
   i) Stranger
   ii) Zombie
   iii) Orphan
   iv) Parent process

E) refers to the POSIX standard defining API for thread creation and synchronization.
   i) Jthread
   ii) MS Thread
   iii) Apple Thread
   iv) Pthread

F) The interval from the time of submission of a process to the time of completion is the:
   i) Throughput
   ii) Turnaround time
   iii) Waiting time
   iv) Response time.

G) Which of the following is true about the dual mode operation (user mode and kernel mode):
   i) It provides means to protect operating system from errant user.
   ii) Prevents the user application from executing privileged instructions.
   iii) Hardware devices can be accessed only when the program is in kernel mode,
   iv) All of the above.

H) Which of the following is a factor that influences the design of an OS?
   i) Error free
   ii) Performance
   iii) Protection
   iv) Maintainability
   v) All of the above

I) Which of the following component of program state is not shared across threads in a multithreaded process?
   i) Register values
   ii) Code ?
   iii) Global variables
   iv) Open files

J) Which of the following CPU scheduling algorithm does not risk starvation of processes?
   i) Preemptive Priority
   ii) RR
   iii) Preemptive Shortest Job First
   iv) Preemptive Shortest Remaining Time First

K) A deadlocked state occurs whenever:
   i) a process is waiting for an I/O device that does not exist
   ii) the system has no available free resources
   iii) every process in a set is waiting for an event that can only be caused by another process in the set
   iv) a process is unable to release its request for a resource after use

L) One necessary condition for deadlock is ____, which states that a process must be holding one resource and waiting to acquire additional resources.
   i) hold and wait
   ii) mutual exclusion
   iii) circular wait
   iv) no preemption

M) Which of the following statements is true?
   i) A safe state is a deadlocked state.
   ii) A safe state may lead to a deadlocked state.
   iii) An unsafe state is necessarily, and by definition, always a deadlocked state.
   iv) An unsafe state may lead to a deadlocked state.

2) True or False: Please indicate if the following statements are True or False: **(1 point each)**

A) An operating system is a program that acts as an intermediary between a user of a computer and the computer hardware. - T

B) A process can create other processes using `knife()` system call. - - F
fork()

C) A priority scheduling algorithm can leave some low priority jobs waiting indefinitely. - T

D) The difference between a program and a process is that a program is an active entity while a process is a passive entity. - F

E) A cycle in a resource-allocation graph is necessary for a deadlock. - T

3) Please fill in the blanks in the following statements: **(1 point each)**

A) One program that runs at all time on the computer is OPERATING SYSTEM .

B) Process SCHEDULING selects among available processes for execution on CPU.

C) The parent process may wait for termination of child process by using the WAIT() system call.

D) The two models for interprocess communication are SHARED MEMORY and MESSAGE PASSING .

E) Most of the first generation operating system were implemented using MACHINE LANGUAGE
programming language.

4) A process can create child processes or threads. Which is better? Why? **(10 Points)**

Neither are better, it depends on what it is being used for, ideally threads would be better for general use, but depending on what you need to do, a child might be the way to go. If you want speed or space, threads are better.

Process - Child processes get their own copy of the parents existing resource, so no race conditions. A process can also run a program whereas it will be more difficult on a thread. A child process also gets its own time quantum equivalent to the parent whereas the thread shares the parents time slots. This allows for greater parallelism than with simple threads. A process also can be assigned priority. I would also say a process is a bit more secure since it does not share data with one another. However, threads would be superior since

Thread - threads are lightweight, takes less time to terminate, they share resource of process, it is cheaper than process creation and can be easily scalable. Once the number of thread grows, multiple tasks can be done.

5) What is a microkernel? What are the advantages and disadvantages of using microkernel?

(10 Points)

A microkernel is the minimum amount of software that can provide the mechanisms needed to operate an Operating System.

Advantages:

- Because it is small and isolated, it can perform better
- Can be added to system application without disturbing kernel
- more flexibility, extensibility, portability and reliability
- Easier to port OS to new architectures

Disadvantages:

- Performance penalty caused by replacing service calls with message exchange between process
- Performance overhead of user space to kernel space communication

6) Consider the following code and answer the following questions.

```
#include <stdio.h>
#include <unistd.h>
#include <sys/wait.h>
int x, y;
void cal();

int main()
{
   int pid;
   int x=1;
   int y=2;
   pid = (int) fork(); /* create a process */
   if (pid == 0) {
      x = x +y; // 1+2 = 3
      y++; // 2 + 1 = 3
      printf("(at A) x is %d, y is %d\n", x , y); /* (A) */
      return (0);
   }

   else if (pid > 0) {
      wait(NULL);
      y--; // 2-1 = 1
      x = x + 10; // 1+10 = 11
      printf("(at B) x is %d, y is %d\n", x, y); /* (B) */
      return (0);
   }
}
```

A) Assume a process will be successfully created. What would be the value of x and y at (A)? Justify your answer. x is 3 and y is 3. I ran it in the compiler **(5 Points)**

And I also wrote out the math in bold in the code

Both statements will run since it was forked. Both will run through its x/y math and output the statements. Pid does not have any current values so both outputs will print

B) Assume a process will be successfully created. What would be the value of x and y at (B)? Justify your answer. x is 11, y is 1. I ran it in the compiler **(5 Points)**

Both statements will run since it was forked. Both will run through its x/y math and output the statements. Pid does not have any current values so both outputs will print

And I also wrote out the math in bold in the code

7) Using Amdahl's Law, calculate the speedup gain for the following applications: 60 percent parallel with:
= 40% serial

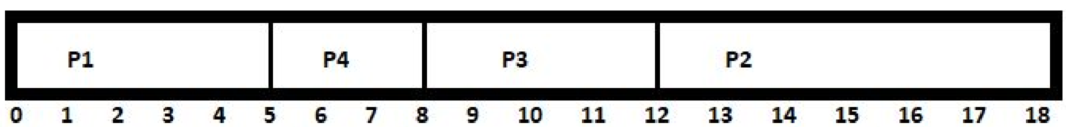
A) 4 processing cores **(5 Points)**
Amdahl's Law = 1/ (S + (1-S)/N) = 1/ (.4 + (1-.4)/4) = 1.81818

B) 8 processing cores **(5 Points)**
Amdahl's Law = 1/ (S + (1-S)/N) = 1/ (.4 + (1-.4)/8) = 2.10526

8) Consider the following set of processes with the length of the CPU-burst time given in milliseconds.

Process	Arrival Time	Burst Time
P1	0	5
P2	1	6
P3	2	4
P4	3	3

A) Draw the Gantt chart illustrating the execution of these processes using preemptive shortest remaining time first. **(10 points)**



B) Calculate the wait time and turnaround time for each of the process. **(10 points)**

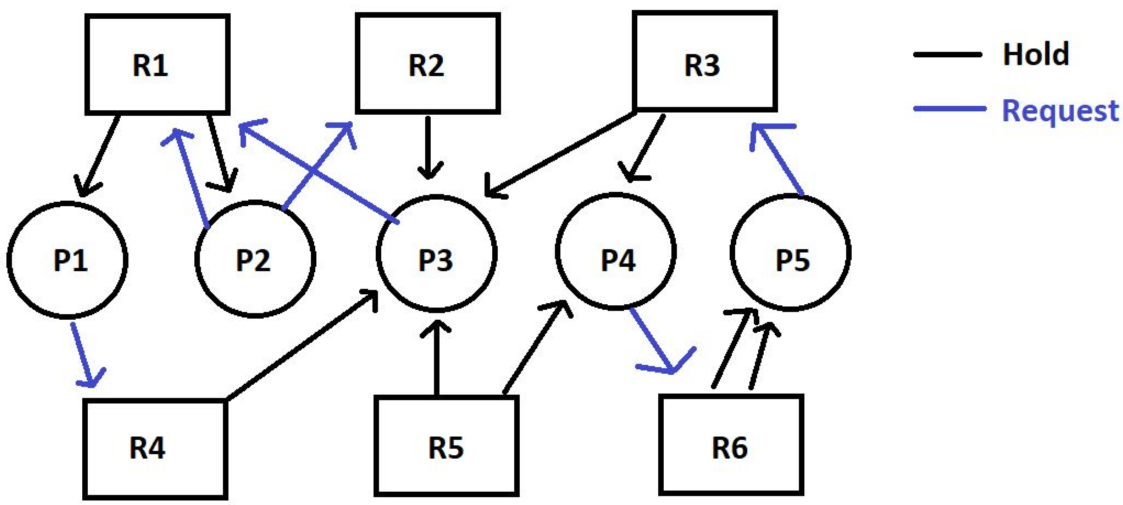
Turnaround = completion - arrival
Waiting = turnaround - burst
* I swapped the order of the table. Turnaround first then wait later

Process	Turnaround	Waiting Time
P1	5	0
P2	17	11
P3	10	6
P4	5	2

9) Consider a system with five processes P1, P2, P3, P4, P5 and resources R1, R2, R3, R4, R5, R6. Also assume that there are two units of each resource type. Now consider the following status:

- Process P1 holds 1 unit of R1 and requests 1 unit of R4.
- Process P2 holds 1 unit of R1 and requests 1 unit each of R1 and R2.
- Process P3 holds 1 unit of each R2, R3 R4 and R5, and requests 1 unit of R1.
- Process P4 holds 1 unit of each R3 and R5, and requests 1 unit of R6.
- Process P5 holds 2 units of R6 and requests 1 unit of R3.

A) Show the resource allocation graph that represents this system state. **(10 points)**



B) Is this system deadlocked? If so, state which processes are involve. If not, give an execution sequence that eventually ends, showing resource acquisition and release at each step. **(5 points)**

We are not in a deadlock. If each resource has two units then we look at what is currently

being held	Requests
P1 - R1	R4
P2 - R1	R1,R2
P3 - R2, R3, R4, R5	R1
P4 - R3, R5	R6
P5 - R6, R6	R3

So currently R1, R3, R5 and R6 are full and being used. R2, R4 are the current free resource. As it continues down the process release and fulfilling requests, it will grab onto the next free resource that a process has let go. P4 has to wait, but once the process finishes, P4 can obtain its resource, so no deadlock occurs

As it goes through the processes, P1 would release R1 and hold onto R4.
New available: R1, R2
P2 would release R1 and hold onto R1 and R2
New available: R1
P3 would release R2, R3, R4, R5 and hold R1
New available R2, R3, R4, R5
P4 has to wait for P5
P5 releases R6, R6 and holds R3
New available R2, R4, R5, R6, R6
P4 continues, releasing R3, R5, holds R6
New available R2, R3 R4, R5, R5, R6

This would be the new setup after all of the holding and releasing of processes
P1 - R4
P2 - R1, R2
P3 - R1

P4 - R6
P5 - R3

All processes are good to go, not all resources are tied up

10) Consider the following snapshot of system that allocates resources using the Banker's algorithm.

Process	Allocation A B C D	Max A B C D	Available A B C D	Need A B C D
P0	3 1 4 1	6 4 7 3	2 2 2 4	3 3 3 2
P1	2 1 0 2	4 2 3 2		2 1 3 0
P2	2 4 1 3	2 5 3 3		0 1 2 0
P3	4 1 1 0	6 3 3 2		2 2 2 2
P4	2 2 2 1	5 6 7 5		3 4 5 4

A) Is the system in a safe state? Show your work for credit. **(10 points)**

It is in a safe state.

So after calculating all the needs, we cannot run P0 since need <= available, so we move down to the next i

At P1, not all entities are smaller than available, so we will move to the next available i, P2

We will be running P2 first since 0120 < 2224

Work/Available is now going to be 2224 + 2413 = 4637

Finish = [false
false
true
false
false]

P3: 2222 < 4637

New work = 4637 + 4110 = 8747

Finish = [false
false
true
true
false]

We skip P4 since not all entities are smaller

P0 : 3332 < 8747

new work = 8747 + 3141 = 11888

Finish = [true
true
true
false
false]

P1 : 3332 < 11888
new work = 11888 + 2102 = 13990
Finish = [true
true
true
true
false]
P4 : 3454 < 13990
new work = 139902221 + 2221 = 16221
Finish = [true
true
true
true
true]

Order is <P2, P3,P0, P1, P4>, Finish == true so system is in safe state

B) If a request from process P2 arrives for (0, 1, 1, 0), can the request be granted immediately? Show your work for credit. **(10 points)**

P2 Original

Allocation	Max	Available	Need
2 4 1 3	2 5 3 3	2 2 2 4	0 1 2 0

Under his notes is true, request can be granted immediately
Need > Request
0120 > 0110 True

Available > Request
2224 > 0110 True

New available -> Available - Request = 2224 - 0110 = 2114
New Allocation = 2413 + 0110 = 2523
New Need = 0120 - 0110 = 0010

P2 Updated

Allocation	Max	Available	Need
2 5 2 3	2 5 3 3	2 1 1 4	0 0 1 0

Another assumption that I want to explore is if we're talking about immediate requests. As the request is coming in as P2 is currently processing its request
current need + new need <= Available/Work for it to move forward (2224)
0120 + 0110 = 0230 > Available.
Then under this assumption, it cannot be immediately granted since the current need is higher than what is available. So it can start once P3 runs and it's resources are released back to available. If

this works then that’s cool, so it would go onto asking, when would you use one method of processing requests over another?