CS3204 Operating Systems - Fall 2000
Instructor: Dr. Craig A. Struble
Final
Length: Take Home
Points: 150

Date: Due: Dec. 13, 2000 by 5:00 p.m. EST Name: _____

Student ID: _____

This is a take home exam. You may use a calculator, computer, or other electronic devices to help with any calculations. Your answers should be concise and well written. Write your answers on a piece of scrap paper first and, after revision, place your final solution in the area provided on the test. Extra pages may be used if necessary, but must be placed with the question being answered.

For computational questions, be sure to include intermediate computations. **Solutions without intermediate computations will be assigned a grade of zero (0).** Include any assumptions you make in your answer, as your answer will be graded against any valid assumptions you make.

The exam may be turned in directly to the instructor or placed in the exam collection box outside of the instructor's office (521 McBryde). If your exam is placed in the collection box, it must be contained in a sealed envelop or it will not be accepted.

You may not discuss this exam with anyone except the instructor until after Dec. 13 at 5:00 p.m. EST. Any discussion about or related to this exam with any person (except the instructor) in any form (e.g., face to face, email, electronic chat, phone, or the listserv) is considered a violation of the Virginia Tech undergraduate or graduate honor code, whichever applies. You must sign below for your exam to be accepted.

I understand that this exam is work governed by the undergraduate or graduate honor code, whichever applies. I also understand that discussing this exam with anyone except the instructor, until after the exam due date and time shown above, is a violation of the governing honor code.

_____

Signature

1. *[10 pts.]* Identify whether **physical** or **virtual** (relative) addresses are used by **processes** for each of the following memory management techniques.

1. single contiguous memory
2. demand paging
3. overlays
4. fixed partitioning with absolute translation
5. dynamic partitioning

---

2. *[20 pts.]* Use the table below containing process service times and deadlines to answer the following questions.

1. Schedule the processes so that each process meets its deadline.
2. Give the average wait time.
3. Give the average turnaround time.
4. Give the average weighted turnaround time.

i	$\tau(P_i)$	Deadline
0	210	350
1	85	405
2	100	300
3	130	600
4	10	320

3. *[10 pts.]* Suppose processes are executing on an operating system using demand paging. The page and block sizes are 2k. Use the PMT/FMT for process 1 and MBT below to answer the following questions.

1. What is the physical address corresponding to compiled address 3356 in process 1? Clearly identify the steps taken during address translation and update the tables as needed.

2. What compiled address corresponds to physical address 5394? Clearly identify which process' address space is being referenced.

Page	Block	Disk	Disk Address
0	3	n	100
1	2	y	101
2	4	n	102

Block	Process	Page	Modified
0	2	1	y
1			n
2	3	0	n
3	1	0	y
4	1	2	n

4. *[20 pts.]* Suppose that a process P is running on an OS using demand paging. Assume that page and block sizes are 4k. Process P contains an array A storing 10000 32-bit integers. Assume that the storage for A starts on a page boundary.

1. Discuss how each of the following affects spatial locality:

   - a linear search on A,
   - a binary search on A,
   - using A as a hash table.

2. Using a linear search for a random element in A performs approximately $n/2 = 5000$ comparisons/accesses, on average. Assuming that only 1 block is allocated to store A (i.e., only one page of A can be in memory at a time), how many page faults, on average, are incurred during a linear search for a random element in A?

3. Assume that only 1 block is allocated for storing A. How many page faults are incurred, on average, searching for a random element in A, if A is a hash table? (Hint: How many comparisons/accesses in A are performed on average?)

4. **10 points extra credit** Assume that only 1 block is allocated for storing A. How many page faults, on average, are incurred during a binary search for a random element in A?

---

5. *[20 pts.]* Assume that 3 processes are executing on a timesharing OS. There exists 2 reusable resources R_1 and R_2 with 2 units of R_1 and 3 units of R_2. The maximum claims for each process are contained in the following table.

Process	R_1	R_2
P_1	1	2
P_2	2	0
P_3	0	3

Assume that the following sequence of resource requests occur.

P_1 requests 1 unit of R_1
P_2 requests 1 unit of R_1
P_1 requests 1 unit of R_2
P_3 requests 2 units of R_2
P_1 requests 1 unit of R_2
P_3 requests 1 unit of R_2
P_2 requests 1 unit of R_1

1. If an unsafe state exists, identify when the system transitions from a safe to unsafe state. If an unsafe state does not exist, justify why not.

2. If deadlock exists, identify when the system would recognize deadlock assuming that a deadlock detection algorithm is executed after each request. If deadlock does not exist, justify why not.

Be sure to show your work.

---

6. *[20 pts.]* Use the following **sequential** statements to answer the questions below.

```
A = Z + X
B = A + 10
C = A + B
D = A + 20
E = B + C
F = X + A
```

1. What are the read and write sets of each statement?
2. Draw a precedence graph for the set of statements above.
3. What is the maximum number of statements that can be executed concurrently?
4. Using `parbegin/parend` blocks, rewrite the statements above to achieve the maximum number of concurrently executing statements.

7. *[20 pts.]* With a pure or demand paging system, extra hardware is used to perform address translation. The PMT and FMT are often loaded into special caches (e.g., the translation lookaside buffer) for direct access by the CPU. When a user program executes, the CPU performs address translation for every address used.

1. What would the impact be on the fetch-execute cycle if the extra hardware did not exist?

2. Some operating systems turn off the use of address translation while executing kernel code. What does this imply about the addresses generated for kernel code in this scenario?

3. At what point (or points) during the execution of kernel code would the automatic address translation be turned off?

4. Describe a simple mechanism that could be used to indicate to the CPU that address translation should not be used.

---

8. *[30 pts.]* Suppose you have two kinds processes, `gatekeeper` and `horse`. One copy of `gatekeeper` and `NUMHORSES` copies of `horse` are executing in the system. Each `horse` enters a stall and waits for the gate in front of it to open. The `gatekeeper` waits for each `horse` to enter a stall. When all `horse`s are in a stall, the `gatekeeper` opens the gate in front of each `horse`.

1. Using shared variables and the `enter()` and `exit()` functions described in the book and in class, write skeleton code that shows how the `gatekeeper` and `horse` processes synchronize with each other.

2. Using semaphores, write skeleton code that shows how the `gatekeeper` and `horse` processes synchronize with each other.
