Answer 100 points worth of questions; if you answer more than that, I will throw away/scale down your worst answers. Please try to be concise; this is an open book exam – you will get no credit for repeating information when it is not relevant.

Problem 1: (10 pts)
For each of the following OS subsystems, (1) process management/scheduling; (2) memory management; and (3) file system.
(a) (6 pts) describe what you think is the most important concept you learned about it in this class.
(b) (4 pts) for any two of the above systems, describe the relationship between them. Give examples of how events/policy choices in one would affect the other

Problem 2: (10 pts)
Provide short explanations for the following potentially wrong statements.
1. User level threads do not need separate stacks.
2. If disks were organized into pages of the same size as the physical memory frames, buffering for disk accesses would not be necessary.
3. A new magnetic disk device with two independently controllable read/write heads has been introduced. Suggest a disk scheduling algorithm for it. What if only one head can read?
4. In class, we discussed how semaphores do not really eliminate busy waiting. The only reason to use semaphores over locks is therefore to get the flexibility of multiple values that semaphores provide.
5. If programs had a random reference pattern to their pages (no locality), virtual memory would be useless unless we had optimal page replacement

Problem 3: (20 pts)
Briefly answer the following questions:
1. Some modern processors support two physical page sizes; one 4-16k, and one 1-4Mbyte. The physical memory is pre-devided into a fixed number of each. Discuss the advantages/disadvantages of this scheme.
2. Suggest two modifications to Round Robin scheduling to provide more fairness to I/O bound processes.
3. For a RAID disk array (say level 0) with the data striped (distributed) across the array disks for faster access, evaluate the file allocation policies we discussed in class (contiguous; linked; indexed). Suggest one.
4. Can the compiler, linker and loader help reduce the number of page faults? Discuss each individually.
5. Construct an unsafe resource allocation state in which there are three processes, there is one class of resources with 4 resources and each process has a claim of 3 resources.

Problem 4: (15 points) Consider the case of passengers standing in the check-in line at the airport. Every passenger stands in line until they are at the top of the line. There are 5 airline employees. Each one of them takes care of a passenger, then calls for the next one when they are free.
(a) (8 points) Write the pseudo-code simulating the passengers and the airline employees using semaphores and/or condition locks. Be careful of the case when multiple employees become free at the same time
(b) (7 points) First class passengers get preferential treatment and do not stand in line. Update your implementation to account for first class passengers.

Problem 5: (15 pts)
(a) (6 points) Identify three factors that make caches an effective techniques for performance optimization.
(b) (9 points) Give three examples of caches in a computer system (caches for three different things). For your examples, suggest of a good replacement policy and a bad replacement policy. For each case, explain what makes the policy good or bad.

Problem 6: (10 points) Two processes share two buffers through which they exchange data. Thread 1 places work for Thread 2 in the “input” buffer. Thread 2 removes one item from the input buffer, processes it,
and places the result back to the output buffer where it is eventually removed by the other thread. There
is a fixed amount of memory available in which both buffers reside. The two buffers can grow dynamically
in size as needed. Explain how deadlock can happen. Suggest two ways to prevent it and compare their
relative merits.

**Problem 7:** (15 points) The following is a brief description of the NT File System (NTFS) (“Operating
Systems, 5th Ed.; 2000” by Rajkumar):

“NTFS works with disk volumes. NTFS allocates the volume in clusters, where a cluster is one or more
(generally, a power of two) contiguous sectors. The cluster size is defined when the disk is formatted. NTFS
uses logical cluster numbers as disk addresses. Multiplying the logical cluster number by the cluster size
yields a physical disk (or volume) address.

Each volume is organized into four regions. The first few sectors contain the partition boot sector
which holds information about the volume layout and the file system structures. Next is the Master File
Table (MFT), essentially a master directory. The MFT entries contain information on each file on the
volume. Following the MFT is a system area which contains a duplicate of portions of the MFT (to provide
redundancy), log files used for NTFS for recoverability, a cluster bit map, and an attribute definition table.
The rest of the volume is the file data area.

The master file table is the main object NTFS uses to access files. It consists of an array of variable
length records. The first 16 records describe the MFT itself, followed by records for each file or directory. If
the file is small enough (less than 1500 bytes) it is written directly in the MFT entry. Otherwise, the MFT
entry contains index pointers to the clusters that contain the actual data.”

(a) (10 points) Compare this file system to the Unix Inode scheme – what are the relative advantages
and disadvantages

(b) (5 points) Comment on the choice of using clusters as the unit of allocation (rather than blocks)

**Problem 8:** (15 pts)
Implement the readers-writers problem using conditional critical regions, making sure that readers and
writers never starve.

**Problem 9:** (15 pts)
Consider NFU, clock algorithm and LRU.

(a) (7 points) Construct 3 different page reference strings that will each have a different algorithm perform
best.

(b) (8 points) Suggest a replacement policy for a segmented virtual memory system