1. (5%) What is the most common approach to prevent deadlocks by ensuring that the circular-wait condition never holds?

2. (10%) (a) Grouping is a modification of the linked-list approach for maintaining the free blocks of a disk. As compared with the standard linked-list approach, what is the main advantage of grouping? (4%)
(b) Give the other two common approaches for maintaining the free blocks of a disk. (6%)

3. (10%) (a) What property must be satisfied for a page-replacement algorithm to be a stack algorithm? (5%)
(b) Prove or disprove that FIFO replacement algorithm is a stack algorithm. (5%)

4. (20%) Consider the following processes:

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

(a) Suppose we are using a non-preemptive priority scheduler in which the priority value of each process is the inverse of its burst time. A larger priority value implies a higher priority.
(i) Write down the execution order of these processes. (4%)
(ii) What is the average waiting time? (4%)
(b) Now suppose we are using a Round-Robin scheduler with a time quantum of 2 units.
(i) Write down the execution order of these processes. (4%)
(ii) Write down the content of the first-in-first-out ready queue whenever it changes. (4%)
(ii) What is the average waiting time? (4%)

5. (5%) What are the drawbacks of the client-server architecture?

6. (4%) Memory hierarchy design takes advantage of the principle of locality. Name and briefly describe the two types of locality exploited in memory hierarchy design.

7. (10%) Consider a 256-byte 2-way set-associative write-back cache with 16-byte blocks. Assume that the cache does not store any data at the beginning, and the least recently used replacement policy is used. Consider a sequence of memory references whose byte addresses (in base 10) are 0, 1, 256, 128, and 0. The type of each memory reference (ordered from left to right) is indicated below.

address	0	1	256	128	0
type	write	write	read	read	read

(a) Determine whether there is a cache miss or hit for each memory reference. (5%)

(b) Determine whether a write-back to main memory occurs for each memory reference. (5%)

8. (7%) Consider an instruction set consisting of h different instruction classes, with the execution time of class-i instructions being $3+i$ ns, $1 \le i \le h$.

(a) What would be the smallest possible clock cycle time for a single-cycle control? (3%)

(b) Consider a multi-cycle control implementation with a clock cycle of 0.5ns and assume that class-i instructions can be executed in $7+2i$ clock cycles. What is the speedup factor of the multi-cycle control relative to the single-cycle control in part (a), assuming that the various instruction classes are used with the same frequency? (4%)

9. (5%) Consider a 5-stage pipelined datapath where ALU computation is performed at the third stage and data memory read is performed at the fourth stage. We have a program of 10^3 instructions in the format of "load, add, load, add, …". Each add instruction is of the form "add *$a, $b, $c*" which performs *$a* = *$b* + *$c*. Each load instruction is of the form "load *$t, k($s)*" which performs *$t* = memory[*$s*+*k*]. Suppose each add instruction depends (and only depends on) the load instruction right before it. Each load instruction also depends (and only depends) on the add instruction right before it. With forwarding, what would be the actual CPI?

10. (4%) Explain the ideas behind the following instruction-set-architecture design choices.
(a) Keep all instructions a single size. (2%)
(b) Have 32 registers rather than many more. (2%)

11. (8%) Consider an 8 bits floating point format: The first bit is for the sign (1 for negative numbers and 0 for positive numbers); the next three bits are for the exponent (with exponent-bias 3); and the last four bits are for the mantissa, which uses the normalization scheme in IEEE 754.
(a) What is the number of the bit pattern represented by (10101100)? (4%)
(b) What is the bit pattern of the number that is closest to but less than 0.3? (4%)

12. (12%) The function szero(void *s, size_t len) clears a string s of len bytes. If len is 0, szero() does nothing. The following MIPS assembly code implements szero(). The words to the right of the sharp symbol(#) are comments. The value in register *$a* is the address of s; the value in *$b* is len; the value in *$zero* is 0; and the value in *$ra* is the return address.

```
szero:  beq   $b, $zero, end   # branch to [end:] if $b=$zero
loop:   sb    $zero, 0($a)     # store $zero to $a
        add   $a, $a, 1        # $a = $a + 1
        sub   $b, $b, 1        # $b = $b - 1
        bne   $b, $zero, loop  # branch to [loop:] if $b≠$zero
end:    jr    $ra              # jump to $ra
```

Suppose you want to run szero() to clear a string of 80 bytes on a 2.5 GHz processor with the following CPI table.

Instruction type	CPI
Arithmetic (add, sub)	3
Memory (load, store)	5
Control (branch, jump)	4

(a) How much CPU time does it take? (4%)

(b) What is the average CPI? (4%)

(c) If one reduces the CPI of the arithmetic instructions to 2, what will be the speedup of the above code for a string length of 80 compared to before? (4%)