CMSC 330 Practice Final Examination

Show all work necessary to justify your answers! There are 7 questions.

1. (15 points) Assuming that static scoping rules are used, what will the following program, in C++ syntax, output if parameters are passed by:

   (a) value?

   (b) value-result?

   (c) reference?

```
#include <iostream.h>
int k = 5;

void f(int i, int j) {
  if (i < 4) i = i + 1;
  else       i = i - 1;
  j = j + 2;
  k = k - 1;
}

main() {
  int j = 3;
  f(j, k);
  f(k, j);
  cout << j << " " << k << '\n';
}
```

   Should print:
   value: 3 3
   value-result: 6 6
   reference: 6 4

2. (15 points)
Give an unambiguous grammar for the language:
L = $\{ w \mid w \in \{0,1\}^*$ and w contains exactly one occurrence of the substring 10 $\}$

A $\rightarrow$ 1B | 0A
B $\rightarrow$ 1B | 0C | 0
C $\rightarrow$ 1D | 0C | 0 | 1
D $\rightarrow$ 1D | 1

3. (15 points)
You are given an array **A** whose elements are structures of type **S**.

```
struct S {
  char c[2];
  float f;
  double d[2];
  int i;
} A[30,20,10];
```

For this problem assume that *chars* take up one byte, *ints* and *floats* take up 4 bytes each and *doubles* take up 8 bytes, and that the fields in a structure must have their *natural* alignment.

(a) (5 points)
Show the layout of variables of type S, with the offset of the first field in the structure being 0.

S is 28 bytes: field c starts at 0, f at 4, d at 8, i at 24

(b) (10 points)
Assuming that array indices always start at 0 and the start address for array **A** is 1000, what will be the address of A[20,5,5], if **A** is stored in:

i. row major order?
answer: 1000 + (20*20*10 + 5*10 + 5) * 28 = 1000 + 4055*28 = 114540

ii. column major order?
answer: 1000 + (5*20*30 + 5*30 + 20) * 28 = 1000 + 3170*28 = 89760

4. (15 points)
Variables of type **set of Z** are usually stored as bit vectors, with 1 bit for each element. For example, if **Z** is the set, [*orange, kiwi, blueberry, grapefruit, pear, banana, strawberry, plum*], then a variable of the type **set of Z** would be represented as an 8-bit vector. The first bit represents the element *orange*, the second bit represents *kiwi*, etc. If a bit is set (1), then the corresponding element is a member of the set. Otherwise the element is not a member of the set.

(a) What is the representation for the set A = [*banana, kiwi*], and the set B = [*strawberry, kiwi, plum, grapefruit*]?

A = 01000100 , B = 01010011

(b) What machine instruction could be used to perform the operation $C = A \cup B$, and what is the representation for C?

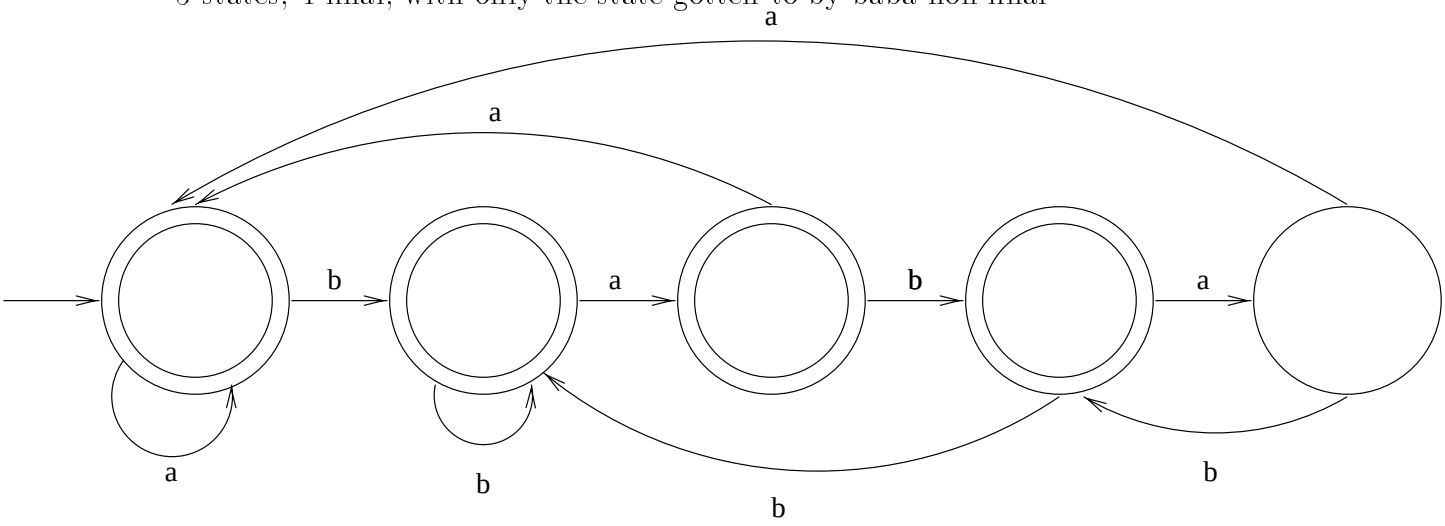
Use C = A OR B, union is bitwise OR, and C = 01010111

(c) What operations must be performed to test whether the element *blueberry* is in the set C? Show both the set operations and machine instruction(s) required.

Represent blueberry as A = 00100000, and use A $\cap$ C (bitwise AND)

5. (15 points) Produce a deterministic finite automaton (DFA) that recognizes the language:
   L = $\{ w \mid w \in \{a, b\}^*$ and w doesn't end with $baba \}$

   5 states, 4 final, with only the state gotten to by baba non-final



6. (15 points)

This question requires you to write pseudocode for the MIPS assembly language instructions that must be executed to run the following program. The pseudocode should be as detailed as the assembly instructions, but we are not concerned about syntax. For example, the instruction **bgt $t1,$t0,L1** can be written as *branch to label L1 if the contents of register t1 is greater than the contents of register t0.* Remember that the MIPS instruction set includes branches on many different conditions (e.g., less than (**blt**), equal (**beq**), greater than or equal (**bge**), etc.)

```
program Exam;
   var A,B,C,D: integer;

   if (A > 0) and (B <> C) then
      while (A >= B) or (C < D) do
         A := A - D
end.
```

Show the pseudocode necessary to perform short circuit evaluation of the conditional expressions for the **if** and **while** statements. Show all the branch and test code, along with any other code required for the **if** and **while** statement bodies. You are *not* required to show the runtime stack manipulation code for the program, but you can assume that the local variables for *Exam* are stored in the order declared, starting at -56 from the global pointer.

```
    lw $t0, -56($gp)  # A
    li $t1, 0
    ble t0, t1, L1  # if A <= 0, short circuit

    lw $t1, -60($gp)  # B
    lw $t2, -64($gp)  # C
    beq $t1, $t2, L1  # if fails

L3: lw $t0, -56($gp)  # A
    lw $t1, -60($gp)  # B
    bge $t0, $t1, L2  # if A >= B, short circuit

    lw $t2, -64($gp)  # C
    lw $t3, -68($gp)  # D
    bge $t2, $t3, L1  # loop terminates

L2: sub $t0, $t0, $t3 # A - D
    sw $t0, -56($gp)  # store back into A
    j L3  # while loop
L1:
```

7. (10 points)
Write a Java class, named **Barrier**, that performs barrier synchronization across N Java threads. The constructor for a **Barrier** object takes N as an argument, and is called by only one thread (e.g., the main program thread). **Barrier** should have a synchronized method, called **bar**, that is called by a thread to enter the barrier. The exact syntax of the Java code is not important, but it should be close.

```
public class Barrier {
  int num_threads, cur_threads;

  public Barrier(int N) {
    num_threads = N;
    cur_threads = 0;
  }

  public synchronized void bar() {
    cur_threads++;
    if (cur_threads >= N) {
      notifyAll();
    }
    else {
      while (cur_threads < N) {
        wait();
      }
    }
  }
}
```