General Information:
Final time: 8:30am-11pm, Monday May 6, 2013.
Location: MSC, W306 (usual room)
The exam is cumulative and will cover material from the entire semester. However, it will focus more on material covered since the midterm. There will be a portion of the exam that is scored via a scantron, so please bring a pencil or two and a good eraser with you. I will not have extras for you to borrow. The exam will have 120 points and will be designed so that it can be completed in 2 hours (although you are free to take the entire exam period). Again, allot approximately 1 minute per point so that you do not get bogged down on any one question.

The below topics and questions are not intended to be exhaustive. They merely provide a starting point.

Topics (since midterm only):
- Stack architecture and manipulation in assembly language - stack pointer, frame pointer, associated special addressing modes
- Subroutines (methods or functions) – parameter passing, local variable allocation, branch and return instructions and effects
- Recursion – how recursive functions effect use of stack in assembly programming
- Linked list manipulation in assembly (specifically in the context of recursive functions)
- C topics:
  - Reading input from terminal/user and writing input to terminal/user
  - Operators: basic mathematic & logical operators, bitwise operators: | & ~ ^, bitwise shift operators: >> << and how they work
  - Pass by value vs. pass by reference and how C handles parameters of different datatypes
  - How C handles defining vs. declaration of functions/variables/etc and pitfalls
  - Arrays: 1D and 2D (and as parameters)
  - Lifetime & scope of variables
  - Reference and de-reference operators (*, &, ->)
  - User defined datatypes: particularly structs
  - Linked lists in C
  - Recursion in C
  - Memory management (allocation and de-allocation/freeing of memory for user-defined variables)
  - Strings

Topics that will not be on the final:
- Lecture 15: link and unlk instructions
- Lecture 16: conditional operator (?), comma operator, unix utility make (other then a general idea of what a makefile can/should do)
- Lecture 17: Bit arrays
- Lecture 18, Enumeration datatype, Union datatype, bit fields, dynamic arrays, equivalence of arrays and pointers,
- Lecture 19: All
Study questions:

1. You learned in CS171 about Quicksort. The algorithm is given below (via Wikipedia which also has psuedocode):

   **QuickSort is a divide and conquer algorithm. QuickSort first divides a large array/list into two smaller sub-arrays/sub-lists: the low elements and the high elements. QuickSort can then recursively sort the sub-arrays/sub-lists.**

   The steps are:
   1. Pick an element, called a **pivot**, from the array/list.
   2. Reorder the array/list so that all elements with values less than the pivot come before the pivot, while all elements with values greater than the pivot come after it (equal values can go either way). After this partitioning, the pivot is in its final position. This is called the **partition** operation.
   3. Recursively apply the above steps to the sub-list of elements with smaller values and separately the sub-array of elements with greater values.
   4. The base case of the recursion are arrays/lists of size zero or one, which never need to be sorted.

   Implement the recursive solution to this problem in M68000 assembly language. Then do it in C.

2. You are given a `getTemps()` function returns an integer composed of: the daily high temperature in bits 20-29, the daily low temperature in bits 10-19, and the current temperature in bits 0-9, all as 2's complement 10-bit integers.
   Write a C program which extracts the high, low, and current temperature and prints the values.

3. Take any of your CS171 homeworks and implement them in C instead of Java. Be on the lookout for efficiency improvements you can make.

4. **Thinking in C** by Bruce Eckels/Chuck Alison. Some good basic exercises. Chapter 7 (Pointers 101) may be particularly useful