Final Exam Topics

• Names, Scopes, and Bindings
  - Names and binding.
  - Potential binding time.
  - Object lifetime and binding lifetime.
  - Deactivation/reactivation of bindings.
  - Dangling references/memory leaks.
  - Object storage management schemes.
  - The memory layout of a process/program.
• Static allocation
  • Example of objects with static allocation.
  • Can static allocation be used for local variables?
• Subroutine frame/activation record format.
• Stack allocation.
• Heap allocation
  • Internal and external fragmentation.
  • Block selection: first fit/best fit
  • Garbage collection, what does it do?
• Scope definition.
• Dynamic scoping and static scoping.
  • Be able to do example problems with these like on the homework.
• Static and dynamic chains
  • Be able to show these on a stack.

• Expression and assignment.
  • Infix, prefix, postfix, Cambridge Polish notations.
  • Operator Precedence and Associativity.
  • Evaluation order of expressions.
    • Why this is not specified by precedence and associativity?
    • Why it is important?
    • Design choices for different languages.
  • Short circuit evaluation for boolean expressions.
  • L-values and R-values.
  • Value model and reference model for assignments.

• Structured and Unstructured Control Flow
  • The concepts of structured and unstructured control flow.
  • Goto’s.
  • Sequencing.
  • Selection and different forms of selection.
- Iteration and iterators.
- Enumeration-controlled loops.
- Logically-controlled loops.
  - Pretest
  - Posttest
  - Midtest
- Recursion.
- Tail-recursive functions and their optimization.

- Subroutines and Parameter Passing
  - Subroutine frame (activation record) format.
  - Stack layout for dynamic and static binding languages.
  - Calling sequences.
  - Prologue and Epilogue.
  - Issues of/solutions to saving and restoring registers.
  - Describe a typical calling sequence.
  - Hardware/software support for efficient subroutine execution.
    - Register windows.
    - Inline functions.
  - Parameter passing.
    - Call-by-value.
    - Call-by-reference.
    - Call-by-result, call-by-value/result.
    - Call-by-name.
    - Need to be able to apply these in programs.

- Functional Programming
  - What is functional programming?
  - Pros and cons of functional programming.
  - Functional features used in other languages.
  - Be able to write a small Scheme function (e.g. summing elements of a list, appending to a list, etc.).

- Logic Programming
  - Need to know the background of logic programming and programming in Prolog.
  - Horn clauses for rules and facts.
  - Logical resolution strategies: Backward chaining and forward chaining.
  - Backtracking.
  - Be able to write a small set of prolog clauses and axioms (e.g. removing element from list, appending to a list, …)