**B1. Compilers (25 pts) Answer all questions.**

(a) ( 6 points)
Construct an operator precedence table for parsing Boolean expressions composed of the following terminals: **id, (, ), not, and, or**. Be sure that a parser using this table willl be able to handle an expression like **id and id or id and not id** properly. Assume that both the **and** and **or** operators are left associative.

(b) (5 points)
Write an algorithm to do operator precedence parsing using an operator precedence table.

(c) (6 points)
Explain two ways that an operator precedence parser can detect and recover from errors.

(d) (4 points)
Write a grammar for Boolean expressions composed of terminals **id**, **(**, **)**, **not**, **and**, and **or** that can be used to construct a top-down parser for such expressions.

(e) (4 points)
Write a grammar for Boolean expressions composed of terminals **id**, **(**, **)**, **not**, **and** and **or** that can be used to construct an LR parser and that preserves the left-associativity of operators **and** and **or**.

**B2. Compilers (25 points) Answer all questions.**

When compiling and running an arbitrarily large program written in a general-purpose programming language (e.g., C, Java, Lisp), several different forms of dynamic memory structures and management are needed. For each of the following compiler and program components, describe the common kinds of specialized dynamic memory structures that are used and the way that they are managed (allocated , deallocated).

(a) (3 points) large symbol tables

(b) (4 points) parser

(c) (4 points) attributes used during syntax-directed translation

(d) (4 points) run-time environment

(e) (3 points) code generator

(f) (3 points) code optimizer

(g) (4 points) objects (in an object-oriented language)

**B3. Compilers (25 points) Answer all questions.**

(a) (5 points) Define and describe code optmization. How close to optimal is the resultant code?

(b) (5 points) Define and describe type checking. Include a discussion of how coercion and overloading are dealt with.

(c) (6 points) List the phases of compilation from source code to machine code.

(d) (4 points) Define and describe an Abstract Syntax Tree.

(e) (5 points) Define and describe data flow analysis.

**B4. Compilers (25 points) Answer all questions.**

(a) (8 points) Define and describe peephole optimization. In what phase of the compiler does it occur? Give two examples.

(b) (9 points) Describe the process of register allocation and provide an algorithm for it. How do you think it interacts with architectures that perform register renaming? Defend your answer.

(c) (8 points) Describe two forms of loop transformations in detail.