B1. Compilers (25 points)

1. (10 points) Bernie needs a lexical analyzer. He wants to take input in the form of numbers that will be non-negative integers in either decimal notation or in hexadecimal notation. For example, the number fourteen can be written either as 14 or as 0xE, and one hundred can be written either as 100 or as 0x64. Write a single regular expression that describes exactly the allowed character strings to be accepted by Bernie's lexical analyzer. The empty string of characters, numbers with leading 0's (e.g., 03, 0433) and the string "0x" are not allowed.

2. (15 points) Bernice wants to know whether she can use an LR parser for her grammar:

D --> D D a
D --> D b
D --> 0 | 1

Attempt to construct an LR(1) parse table for Bernice's grammar. Show all your work so that we can see how you are doing it. If it is not possible to construct an LR(1) parse table, explain why it can't be done.

B2. Compilers (25 points)
Bridget is writing a compiler for an object-oriented language that has classes (of methods and attributes) and single inheritance, object instantiation through constructor methods, and polymorphic call sites.

1. (15 points) Describe a symbol table management scheme that will be effective and efficient, including descriptions of the data structures implementing the symbol table(s) (5 points), and descriptions of the implementations of each of the symbol table operations (5 points). Be careful to discuss how the features of the object-oriented language are handled (3 points), and justify the efficiency of the design (2 points).
2. (5 points) How can complete static type checking be done when all types must be declared, but operators may be overloaded?
3. (5 points) What is a polymorphic function? How does the existence of polymorphic functions affect static type checking? How can a type checker be designed to address polymorphic functions?

B3. Compilers (25 points)

1. (9 points) Static vs. Dynamic: Answer the following 3 questions.
What is the difference between
(1) static and dynamic compilation?
(2) static and dynamic binding?
(3) static and dynamic scope?

2. (12 points) Activation Records:
(1) What is an activation record used for and is it a compile-time or a run-time concept?
(2) Name three items stored in an activation record?
(3) Explain when an activation record would need to be stored on the heap.
(4) Describe an optimization that would reduce the number of activation records needed.

3. (4 points) Why is it beneficial to put an object on the stack versus putting it on the heap?

B4. Compilers (25 points)

1. (5 points) Compare and contrast 2 of the following intermediate representations.
CFG, SSA, AST, DAGs

2. (5 points) Why is SSA useful for optimizations?

3. (5 points) What are the key tradeoffs between structural and linear intermediate representations?

4. (10 points) Translate the following code into a CFG.

```
{the following loop looks for a null character}

null_index := -1;          {-1 will mean "not found"}
test_inx := 0;             {index of first char}
done := (len = 0);         {don't loop if no data}

while not done do
  begin
                           {see if current character is null...}
  if buf [test_inx] = chr (0) then
    begin                    {found a null!}
    null_index := test_inx;  {remember location}
    done := true;            {terminate loop}
    end

  else
    begin                  {incr inx, check end}
    test_inx := test_inx + 1;
    if test_inx >= len then    {inx is 0-based}
      done := true;
    end;
  end;                     {while not done}
```