CS320 Principles of Programing Languages
Winter 2018 Midterm

NAME:

READ THESE INSTRUCTIONS VERY CAREFULLY BEFORE YOU BEGIN:

- Do not turn over this page until the test begins.
- Make sure that you have written your name in the space at the top of this page.
- Time allowed is 100 minutes (one hour and forty minutes).
- You may leave whenever you are finished with the exam; please exit quietly to avoid disrupting the other students.
- The only materials allowed are writing tools (pen or pencil) and a single one-sided 8.5"x11" handwritten sheet of notes. If you have a notes sheet, please turn it in with your exam.
- Please do all your work on these sheets; use the back of the paper if you need more space. In general, the space given for each question should be taken as a rough indication of the expected length of a good answer, but you will not lose points for writing too much or too little as long as the substance of your answer is clear and correct.
- Keep your answers as clear, concise, and legible as possible.
- The points for each part of each question are shown in parentheses; these can help you judge the relative difficulty and/or level of effort expected for each part. Credit will be given for partial answers where appropriate.
- Answer as many questions as you can. You may not be able to answer all of the questions in full; try to prioritize effectively.
- Read the questions carefully. You may ask for clarification of questions at any time during the exam. If you feel that you need to make additional assumptions in order to answer a question, be sure to explain that as part of your answer.
- Discussion of this exam with colleagues or classmates is not allowed until all solutions have been turned in.
- Breaches of academic integrity will be treated very seriously.

1.

a) Explain why it is important for the designers/implementers of a programming language to define both a *concrete syntax* and an *abstract syntax* for the language. **(2)**

b) For each of the following language features, identify whether the feature falls under *static semantics* or *dynamic semantics* and give a brief explanation why. **(2 each)**

i) The C++ specification requires that all function calls in a program have the correct number of arguments in order for a program to be considered legal C++.

ii) The Haskell specification defines an error mechanism that triggers when a program attempts to access an out-of-bounds index in a list at runtime.

iii) The Python specification defines an error mechanism that triggers when a program attempts to perform a function call with an incorrect number of arguments at runtime.

2.

Imagine you're given a buggy compiler ("86toARM") that translates from x86 bytecode to ARM bytecode. (ARM is another processor architecture, with a different instruction set than x86).

a) What does it mean to claim that 86toARM is *incorrect*? **(2)**

b) What kind of evidence could you construct to prove this claim? **(1)**

Now imagine you're given another compiler ("HSto86") that translates from Haskell to x86 bytecode, and you construct a compiler ("HStoARM") that translates from Haskell to ARM bytecode by feeding the output of HSto86 into 86toARM.

c) Assume HSto86 has been proven *correct*, and 86toARM is still incorrect. What can be said about the correctness of HStoARM? Justify your answer. **(3)**

3.

Consider the following Haskell code:

```
reverse :: [Int] -> [Int]
reverse [] = []
reverse (x:xs) = reverse xs ++ [x]
```

For each of the following modifications to the code (highlighted in gray), identify which stage of compilation would first report an error and give a brief justification why. **(2 each)**

a)
```
reverse :: [Int] -> [Int]
reverse [] = []
reverse (x:xs) = reverse xs && [x]
```

b)
```
reverse :: [Int] -> [Int]
reverse [] == []
reverse (x:xs) = reverse xs ++ [x]
```

c)
```
reverse :: [Int] -> [Int]
reverse [] = []
reverse (x:xs) = reverse 'xs' ++ [x]
```

d)
```
reverse :: [Int] -> [Int]
reverse [] = []
reverse (x:xs) = reverse xs ++ [y]
```

4.

For each of the following regular expressions, give a short English description of the language specified by the regex and identify whether each of the given strings is a member of that language. **(4 each)**

a) a* • b* • c*

i)	"aabbcc"	yes | no
ii)	"acba"	yes | no
iii)	""	yes | no

b) (a • b+)* • (c | d)+

i)	"abbabc"	yes | no
ii)	"abcddc"	yes | no
iii)	"aacdcd"	yes | no

5.

Give regular expressions for the following languages. **(2 each)**

a) Non-negative even integer literals (e.g. "0", "4", "038", "123456")

b) Boolean literals separated by & (e.g. "true", "false & true", "true & true & false")

6.

Consider the following context-free grammar for a tiny imperative language L, where `S` is the start symbol, `c` is any lower-case letter, `n` is any non-negative integer, and the other terminals are `:=`, `print`, `;`, and `+`.

```
S → c := E
S → print E
S → S ; S
E → c
E → n
E → E + E
```

Here are some example strings in L:

```
"print 100"
"x := 3 + 4 + 5"
"x := 1 ; y := x + x ; print y"
```

For each of the following strings, identify whether the string is in L; if it is in L give a parse tree for it, and if it isn't in L give an argument that there can't be a parse tree for it. **(2 each)**

a) `"10 := 11"`

b) `"x := 5 ; print x ; print y"`

c) `"x := 1 ; y := 2 ; x + y"`

d) Is this grammar for L ambiguous? Justify your answer. **(3)**

6.

For each of the following scenarios, argue in a couple sentences whether a class-based object-oriented programming language or a functional programming language would be more appropriate for writing the program in question.

Focus on the paradigms, not on specific languages/implementations - for example, don't say that an OOP language is more appropriate for a task because GCC compiles C++ to very efficient code and the task requires high performance, or that an FP language is more appropriate because there are high-quality Haskell libraries that are relevant to the task.

(Hint: recall the discussion in week 5 about the expression problem, and the relative strengths and weaknesses of OOP and FP when extending a program with new code.)

These are subjective questions with no strictly right or wrong answers - your responses will be graded on the strength of your justifications.

(2 each)

a) A video game company is building a racing game with a variety of cars to choose from. Each car has the same set of functionality (such as accelerating and steering), but unique values for several parameters (such as weight and horsepower). The team plans to add more cars to the game over time.

b) A team is implementing a compiler for FORTRAN 77, an old language with a published formal specification, in order to enable legacy code to run more efficiently on modern platforms. The code will deal mostly with a single AST type, which will never change: the team does not plan to support newer versions of FORTRAN. However, they do plan to extend the codebase in the future to support additional features such as automatic code formatting and static analysis to detect memory leaks.

c) A web development startup is contracted to build a web store for a retail corporation that sells a wide variety of products. The application will initially only allow users to search for products by name and make purchases, but the retail corporation has indicated that they may want to expand the capabilities of the website in several different ways if the initial launch is a success. For a couple examples, they're interested in allowing customers to track the availability of selected products on a wishlist; categorizing the products into a structured hierarchy that can be browsed visually; and offering a rewards system that tracks the purchases each customer makes and gives them discounts.