# Montana State University

## CSCI 305 - Programming Languages

### Midterm Exam

### SPRING, 2018

**Name:** _____

**Student Id:** _____

| Question | Points | Score |
|---|---|---|
| 1 | 30 | |
| 2 | 2 | |
| 3 | 2 | |
| 4 | 2 | |
| 5 | 2 | |
| 6 | 2 | |
| 7 | 10 | |
| 8 | 15 | |
| 9 | 15 | |
| 10 | 20 | |
| Total: | 100 | |

(1) (30 points) Mark each question as either True (T) or False (F) by circling either T or F.

a. **T** **F** Modula was one of the first languages to incorporate namespaces.

b. **T** **F** COBOL was designed as a programming language for scientific applications.

c. **T** **F** Plankakul was originally developed for Artificial Intelligence problems.

d. **T** **F** A weakly typed programming language is one that detects all type errors at compile time.

e. **T** **F** A non-terminal that occurs on the far right of any production rule is right associative.

f. **T** **F** Parameter coercion is not a type of simple polymorphism.

g. **T** **F** A language system involving the classical sequence typically has a step involving the loading of the source code into an interpreter.

h. **T** **F** Ruby is a statically typed language.

i. **T** **F** BNF is weaker in descriptive power than EBNF.

j. **T** **F** There is no direct link between a rule's associativity and its precedence.

k. **T** **F** It is impossible to define a grammar which is inherently ambiguous.

l. **T** **F** Overloading functions and operators can only be done in dynamically type languages because of their greater flexibility.

m. **T** **F** Delayed linking, interpreters, and virtual machines are all features of language systems which deviate from the classical sequence.

n. **T** **F** A function which exhibits parametric polymorphism if it has a type that contains one or more type variables.

o. **T** **F** ML allows for implicit type coercion.

(2) (2 Points) Give the ML type corresponding to the following set: `{(true, true), (true, false), (false, true), (false, false)}`

(3) (2 points) Context free grammars are used to capture which aspect of a programming language?

```
A. Lexical Structure

B. Syntax

C. Type System

D. Operational Semantics

E. Axiomatic Semantics
```

(4) (2 points) Circle all of the following types that are primitive types:

```
A. Java int

B. ML String

C. C float

D. ML char

E. ML int list
```

(5) (2 points) Circle all of the following things that types are useful for:

```
A. Improving code readability and understandability through type annotations

B. The value of the evaluation of the expression last evaluated in the method

C. Both A and B

D. None of the above.
```

(6) (2 points) Circle all of the following that are mainly Imperative Languages.

```
A. C
```

```
B. Fortran 77

C. Java

D. LISP

E. Perl
```

(7) (10 points) What is the binding time for each of the following in a C/C++ program? State the binding time as precisely as possible (language-definition time, language-implementation time, compile time, link time, load time, or runtime)

```
    a. The meaning of the `while` keyword.



    b. The location in memory of a local variable in a function.



    c. The type of a local variable in a function.
```

(8) Answer these questions about the following grammar. The starting symbol is `<S>`

```
<S> ::= <S1> + <S> | <S1>
<S1> ::= <S2> | <S1> * <S2>
<S2> ::= ( <S> ) | a | b
```

```
    a. (2 points) Which operator has higher precedence?



    b. (2 points) What is the associativity of the `*` operator?
```

```
c. (2 points) Does this grammar describe a finite or infinite language?



d. (2 points) Is this grammar ambiguous?



e. (4 points) Rewrite this grammar using EBNF, considering the version of EBNF
provided in the book which adds only the grouping symbols `[ ]`, `{ }`, and
`( )`.







f. (3 points) Draw a parse tree to explain the string: `b + a * (a + b) + a`
```

(9) Suppose there are three variables X, Y, and Z with these types:

```
X: integer that is divisible by 3
Y: integer that is divisible by 12
Z: integer
```

For each of the following assignments, knowing nothing about the values of the variables except their types, answer whether a language system can tell before running the program whether the assignment is safe? Answer Yes or No and explain why.

```
a. (5 points) `Z := X`
```

```
b. (5 points) `Y := Y + 1`


c. (5 points) `X := X + 3`
```

(10) Consider an unknown language with a right-associative `*` operator that is overloaded to have the following types: `int * real -> real`, `int * int -> int`, `real*int -> real`, and `real*real -> real`. Suppose the variable `i` has type `int` and the variable `r` has type `real`. For each `*` operator in each of the following expressions, say which type of `*` is used:

```
a. (5 points) `r * i`



b. (5 points) `r * i * r`



c. (5 points) `r * (i * r)`



d. (5 points) `i * r * i * (i * r)`
```