Spring 2017 (Thursday, March 9) Name: _____

CS 3360 — Design and Implementation of Programming Languages

Exam 1

This test has 8 questions and pages numbered 1 through 7.

Reminders

This test is closed-notes and closed-book. However, you are allowed to bring 1 page (8.5" × 11") of notes (both sides). Your notes must be of your own, they must be hand written, and they must be turned in with your test. This test is to be done individually, and you are not to exchange or share materials with other students during the test.

If you need more space, use the back of a page. Note when you do that on the front.

This test is timed. Your test will not be graded if you try to take more than the time allowed. Therefore, before you begin, please take a moment to look over the entire test so that you can budget your time.

Clarity is important; if your writing or code is sloppy or hard to read, you will lose points. Correct syntax also makes some difference.

There are 100 points all.

---

1. (total 20 points) Short answers: multiple choice and fill-in-the-blank.

   (a) (4 points) There are many different criteria for evaluating programming languages, including readability, writability, reliability, cost, and so on. Among these, the most important criterion for judging a programming languages is _____. Justify your answer by stating the reason.

   Readability, because a language that cannot be easily comprehended or understood cannot be maintained, expanded, or for that matter debugged.

   (b) (2 points) The design and evaluation of a programming language is highly dependent on the domain in which it is to be used. Which of the followings is an incorrect association between a programming language and its application area?

   i. Fortran for scientific applications
   ii. COBOL for business applications
   iii. C for systems programming
   iv. Lisp for artificial intelligence
   v. Java for scripting

   (c) (2 points) A language is said to be **Orthogonal** if it has a relatively small set of primitive constructs that can be combined in a relatively small number of ways to build the control and data structures of the language. Such a language is easy to learn and read because every possible combination is legal without an exception and the meaning is context independent.

(d) (2 points) The major methods of implementing programming languages are compilation, interpretation, and hybrid implementation. Java uses _____Hybrid_____.

(e) (2 points) An attribute grammar is a BNF with three additions: attributes, attribute computation functions, and predicate functions. Among the three additions, the role of __Predicate Functions__ is to state the semantic rules or constraints of a grammar rule.

(f) (2 points) The address of a variable (l-value) is the machine memory address with which it is associated. Which of the following statements is incorrect about the address of a variable?

i. A variable may have different addresses at different times during execution.
ii. The same name may have different addresses at different places in a program.
iii. If two variable names can be used to access the same memory location, they are called aliases.
iv. Aliases are harmful to program efficiency.

(g) (2 points) PHP supports several different types of values including Booleans, integers, floating-point numbers, strings, objects, and resources. All these PHP values are either true or false value.

i. True
ii. False

(h) (2 points) Which of the following statements are incorrect about PHP strings?

i. A string can be expressed by enclosing it in a pair of single or double quotes, e.g., 'Hello' and "Hello".
ii. There is no semantic difference between the use of single and double quotes in expressing a string.
iii. A special character can be contained in a string by preceding it with a escape character.
iv. Two strings can be concatenated using the dot (.) operator.

(i) (2 points) There are five different subscript and storage bindings possible for arrays. Java uses __Fixed Heap-Dynamic__

2. (6 points) State the difference between *static binding* and *dynamic binding*, and give an example of each in terms of bindings of variable attributes occurring in the following Java local variable declaration.

```
if (...)  { int x = 10; ...  }.
```

The difference between static and dynamic bindings is that in static binding the binding does not change, where as in dynamic binding the binding of any variable is changed during execution.

3. (6 points) This question is about the ambiguity of a grammar.

   - When a grammar is said to be ambiguous? I.e., give the definition of the ambiguity of a grammar.
   - Prove or disprove the ambiguity of the following grammar.

     `<assign-expr> -> <assign-expr> = <assign-expr>`

4. (10 points) Write a BNF grammar for the language recursively defined as follows.

   - A symbol `a` is in the language.
   - For any two sentences x and y in the language, x `<<` y is also in the language.
   - For any two sentences x and y in the language, x `&` y is also in the language.
   - Nothing else is in the language.
   - Both operators (`<<` and `&`) are left associative.
   - The operator `<<` has precedence over the operator `&`.

   Example sentences of the language include `a`, `a << a`, `a & a`, `a << a & a`, etc.

5. (8 points) There can be different PHP variables and values depending on their scopes and lifetimes, including the following. For each, state the storage binding scheme used; if needed, justify your answer by stating the reason.

   - Local variables
   - Formal parameters
   - Global variables
   - Static local variables
   - Objects (fields)
   - Arrays

6. (10 points) Suppose a programming language PL_m that supports multi-dimensional arrays. Your program written in PL_m contains a two-dimensional array, say $A[100, 50]$ (100 rows $\times$ 50 columns). You learned that (i) each element of the array requires 4 bytes of memory, and (ii) $A[0, 0]$ and $A[10, 1]$ are stored at memory addresses 1000 and 1440, respectively, on a byte-addressable machine.

   (a) Does PL_m use row major order or column major order?

   (b) Write a formula that, given valid indices i and j, gives the memory address where $A[i, j]$ is stored.

   (c) Calculate the memory address where $A[3, 5]$ is stored?

7. (10 points) PHP provides two different equality operators, == and ===.

   (a) Describe the two operators by showing examples to highlight the difference.

   (b) Relate the two equality operators to different kinds of assignments for objects (=, &=, `clone`). E.g., what're the impacts of the different assignments on the equalities of objects?

8. (total 30 points) Consider the Battleship web service that you wrote as part of your PHP assignment. A Battleship board consists of $n \times n$ places, and each place of a board is uniquely identified by a pair of 1-based column and row indices. A set of ships can be placed in a board. Below you are to implement a couple of features of the Battleship game by writing partial code for three classes: Place, Ship, and Board.

   (a) (5 points) Write a constructor for the Place class given below. Given 1-based column and row indices ($x and $y), the constructor initializes the newly created instance. Provide reasonable (default) values for all the fields.

```
class Place {
  var $x;     // 1−based column index of this place
  var $y;     // 1−based row index of this place
  var $ship;  // null or occupant of this place
  var $isHit; // true if this place is hit

  function isHit() { ... } // return true if this place is hit.
  ...
}
```

   (b) (5 points) A ship can be placed on a consecutive sequence of places in a board, either horizontally or vertically. A ship placed on a board becomes sunk if all its places are hit. Write the `isSunk()` function of the Ship class given below.

```
class Ship {
  var $places; // array of the places this ship is placed on

  function isSunk() { ... } // is this ship sunk?
  ...
}
```

(c) (15 points) A board consists of $n \times n$ places, where n is the size (dimension) of the board. A board also knows all the ships placed on it. Write the following three functions of the Board class (see the skeletal code below).

- A constructor that takes an optional board size ($size) and initializes a newly created board instance. If the optional board size is missing, the constructor creates a board of size 10; e.g., `new Board()` creates a board of size 10.
- The `getPlace()` function returning the place of the given 1-based column and row indices ($x and $y) or null if no such place exists.
- The `isGameOver()` function returning true if the game is over, i.e., all the ships placed on the board are sunk.

```
class Board {
  var $size;    // dimension of this board
  var $places; // 1 dimensional array containing all the places of this board
  var $ships;  // array of ships placed in this board

  function getPlace($x, $y) { ... }
  function isGameOver() { ... }
  ...
}
```

(d) (5 points) Explain the function `places()` defined in the Board class below. What value will be returned by the calls `places(1)` and `places(2)`, respectively, and why?

```
class Board {
  var $places; // array of places

  function places($delta) {
    $i = 0;
    while ($i < sizeof($this->places)) {
      yield $this->places[$i];
      $i += $delta;
    }
  }
  ...
}
```