Computer Science Comprehensive Exam—Spring 2014
Programming Languages

Instructions: Do *not* put your name on the exam, please answer all the questions directly on the exam itself. You have 90 minutes. Explain answers as fully as possible; if appropriate give examples or define terms. Answer as many questions as you have time for.

1. C++11 has lambdas, a construction with which one can write anonymous functions which depend on parameters (`n` in the example below) and capture non-local variables (`x` in the example below). In this example, a lambda is used to remove elements less than 5 from a C++ vector.

```
#include <vector>
#include <iostream>
#include <algorithm>
#include <functional>
int main() {
    std::vector<int> c { 1,2,3,4,5,6,7 };
    int x = 5;
    c.erase(std::remove_if(c.begin(), c.end(),
        [x](int n) { return n < x; } ), c.end());
    std::cout << "c: ";
    for (auto i: c) {
        std::cout << i << ' ';
    }
    std::cout << '\n';
}
```

   What implementation challenges does this new construct pose for C++? What approach would you take in its implementation?

2. Consider the following program with a loop. By <> we mean "not equals."

```
i := 1; f := 1;
while (i <> n) {
   i := i+1; f := f*i;
}
```

(a) What does this program compute?

(b) What is the (best) loop invariant for the loop?

(c) Let I be your loop invariant in part (b). Prove the following Hoare triple is valid:

$$\{I\}\ \mathtt{i} := \mathtt{i} + 1; \mathtt{f} := \mathtt{f} * \mathtt{i}\ \{I\}$$

3. The programming language Java, like all object-oriented languages, has subtype polymorphism. Write a subprocedure in Java (or C#, C++) that exhibits subtype polymorphism and explain.

4. Java and C# have bounded-quantification polymorphism, an interesting new form of polymorphism which combines subtype and parametric polymorphism. Give an example and explain.

5. Consider two separate, independent executions of the following Ada-like program. Assuming that `X` is passed by copy-in/copy-out, what are the values of `I` and `A` after the call? Assuming that `X` is passed by reference, what are the values of `I` and `A` after the call?

```
PP: declare
    -- declare an array of 5 elements
    A: array (1..5) of Integer := (1,2,3,4,5);
    I: Integer := 1;
    procedure P (X: Integer) is
    begin
        X := 18;  I := 2;  X := 10;
    end P;
begin
    P (A[I]); -- call P
    -- value of "I", values of "A"?
end PP;
```

6. Given the following Haskell data type:

```
data Tree = Leaf Int | Node2 Tree Tree | Node3 Tree Tree Tree
```

write the following functions.

(a) Write a Haskell function `count` of type

```
Tree -> Int
```

which counts the number of leaves in the tree.

(b) Write a Haskell function `binary` of type

```
Tree -> Boolean
```

which returns true if all the internal nodes of the tree have two sub-trees.

(c) Write a Haskell function `split` of type

```
Tree -> Tree
```

which returns a tree in which all internal nodes have branch exactly two sub-trees. (The tree is constructed without `Node3`.) The resulting tree must have the same leaves as the input.

7. How would you write function `split` of the previous problem in Prolog? You will need to invent your own functors.

8. Find the (most general) unifying substitution for each of the following pairs of terms (x, y, and z are variables), if it exists. To the right of each pair, write "no unifier" if none exists, otherwise give the unifying substitution.

(a)	$g(a, c)$	$g(a, d)$
(b)	$h(a, x, c)$	$h(a, x, d)$
(c)	$h(c, a, x)$	$h(c, a, y)$
(d)	$g(a, c)$	$g(c, a)$
(e)	$g(a, b)$	$g(a, b)$
(f)	$g(a, x)$	$g(a, h(b, c, x))$
(g)	$g(y, x)$	$g(a, h(b, c, y))$
(h)	$g(a, y)$	$g(a, y)$
(i)	$g(g(a, b), h(x, a, y))$	$g(g(z, b), h(b, a, b))$
(j)	$g(g(a, x), h(a, x, b))$	$g(g(a, b), h(a, a, b))$
(k)	$h(z, z, z)$	$h(x, b, y)$
(l)	$g(g(a, x), h(y, a, b))$	$g(y, x)$
(m)	$g(g(a, x), h(y, a, b))$	$g(z, x)$

9. Consider the following PROLOG database of nullary predicates:

```
A.
A :- B.
A :- C.
B.
B :- C, B.
C.
```

Draw a representation of the entire search space for the query `A?`. Is the search space finite? Please circle the answer: yes / no. How many solutions are there? [ ]