CS 345 - Programming Languages
Fall 2010

FINAL

December 9, 2010

DO NOT OPEN UNTIL INSTRUCTED

YOUR NAME: _____

Collaboration policy

No collaboration is permitted on this exam. Any cheating (*e.g.*, submitting another person's work as your own, or permitting your work to be copied) will automatically result in a failing grade. The Computer Sciences department code of conduct can be found at `http://www.cs.utexas.edu/academics/conduct/`.

Final (85 points)

Problem 1 (15 points)

Circle only one of the choices **(3 points each)**.

1. **TRUE** **FALSE** Prototype objects in JavaScript allow new methods and members to be added at runtime to all objects created by a given constructor.
2. **TRUE** **FALSE** SQL injection is caused by poorly written scripts which do not properly validate and/or sanitize user input.
3. **TRUE** **FALSE** When the program calls a virtual function `foo` and the class hierarchy contains several functions called `foo`, the compiler determines which of them to call.
4. **TRUE** **FALSE** In Smalltalk, subtyping is related to inheritance: an object of class B is a subtype of A only if B inherits from A.
5. **TRUE** **FALSE** C++ allows inheritance without subtyping.

Problem 2 (20 points)

Define the following terms:

Same-origin policy:

Encapsulation:

Subtyping:

Dynamic lookup:

Message (in Smalltalk):

Problem 3 (6 points)

JavaScript has higher-order functions and lexical scoping. List two mechanisms needed to support these language features and explain why.

Problem 4 (5 points)

Explain the difference between `this<class>` in Simula and `this` in Java or C++.

Problem 5 (5 points)

Explain, in detail, how the following Smalltalk expression is evaluated. Assume that no optimizations are done.

```
i<j ifTrue: [... code 1 ...]  ifFalse: [... code 2 ...]
```

Problem 6

Problem 6a (4 points)

If a type A is known to be a subtype of a type B, what does this enable programs to do that they could not do in a non-object-oriented language?

Problem 6b (4 points)

What is the difference between message passing and overloading?

Problem 6c (4 points)

What is the difference between a virtual and a non-virtual function in C++?

Problem 7 (8 points)

Assume that that `A` is a subtype of `B` (denoted as `A <: B`) and `B <: C`. In the following list, circle all subtype relationships that hold in principle:

```
B->A  <:  B->B

A->B  <:  B->B

A->A  <:  B->B

C->A  <:  B->B
```

Problem 8

Consider the following C++ code:

```
class Point {
  public:
    int x;
    int y;
    Point(int x1, int y1) { x = x1; y = y1; }
    Point* move(int dx, int dy) {
      return new Point(x+dx,y+dy); }
};
class Point3D: public Point {
  public:
    int z;
    Point3D(int x1, int y1, int z1): Point(x1,y1) { z=z1; }
    Point3D* move_3D(int dx, int dy, int dz) {
      return new Point3D(x+dx,y+dy,z+dz); }
};
Point *p = new Point(3,4);
Point3D *p1 = new Point3D(3,4,5);
```

Problem 8a (10 points)

For each of the following statements, indicate whether it is statically type-correct (under the type checking rules of C++) and, if not, explain why not. Each statement should be considered to follow the previous statements.

```
Point *p2 = new Point3D(3,4,5);
```

```
Point3D* p3 = p2;
```

```
Point3D* p4 = p2->move_3D(7,6,5);
```

```
Point* p5 = p1->move_3D(-1,-2,4);
```

```
Point* p6 = p1->move(1,-1);
```

Problem 8b (4 points)

Consider adding the following method to the `Point3D` class:

```
class Point3D: public Point {
  ...
  void move(int dx, int dy) {
    x+=dx; y+=dy; }
}
```

How would the C++ static type checker react to this new method? Why? If there are any problems, how could we change this code to fix them?