CS1110 Prelim 2 **Sample Answers** (yours may be different) **12 March 2009**

Have a good break!!!

Question 0. (answer omitted)

Question 1.

```
/** = the number of times the first char of s appears
        at the beginning of s.
        Precondition: s has at least one char. */
public static int firstC(String s) {
      if (s.length() <= 1 || s.charAt(0) != s.charAt(1))
         return 1;

      // first and second chars are the same
      return 1 + firstC(s.substring(1));
   }
```

Question 2(a). Make a class abstract so instances of it cannot be created. To make it abstract, insert keyword **abstract** after **public**.

2(b)

```
/** = "p is an object of class PhD, and it has the
            same name as this PhD". */
public boolean equals(Object p) {
      if (!(p instanceof PhD))
         return false;
      PhD phd= (PhD) p;
      return name.equals(phd.name);

}
```

2(c).

```
/** = the job this PhD now holds */
public abstract String hasJob();
```

The purpose of making hasJob abstract is to ensure that any non-abstract subclass overrides it.

Question 3(a).

```
public Faculty(String n, int y, String u, String uw,
                  String r, PhD a1, PhD a2) {
      super(n, y, u, a1, a2);
      univ= uw;
      rank= r;
}

public boolean equals(Object f) {
      if (!(f instanceof Faculty))
         return false;
      Faculty fac= (Faculty) f;
      return super.equals(f) &&
            univ.equals(fac.univ) &&
            rank.equals(fac.rank);
}
```

3(b) Remove hasJob, and it won't compile because hasJob is declared as abstract in class PhD. Remove rank, and it will still compile.

3(c) p.hasJob() is syntactically legal, because hasJob is declared in superclass PhD. The function called is the one found, will be the overriding one in the object whose name is in p.

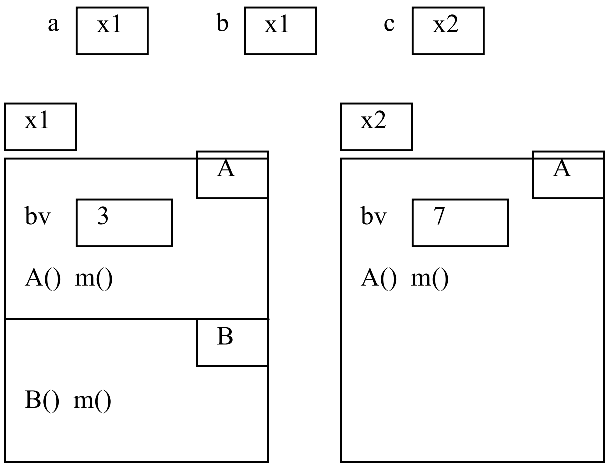
p.rank() is syntactically illegal because rank is not declared in or inherited by class PhD.

Question 4. We give two answers. The first is the more traditional one developed by looking first for the base case, etc. The second uses conditional expressions to shorten the function body immensely, without losing readability.

```
/** = Total no. of ancestral advisors of this PhD */
public int ancestors() {
      if (advisor1 == null &&  advisor2 == null)
        return 0;

      // this PhD has at least 1 advisor
      if (advisor1 == null)
         return 1 + advisor2.ancestors();
      if (advisor2 == null)
         return 1 + advisor1.ancestors();

     // this PhD has 2 advisors
      return 2 + advisor1.ancestors() +
                  advisor2.ancestors();
}

/** same spec as ancestors */
public int ancestorsb() {
   return (advisor1 == null ? 0 :
                        (1 + advisor1.ancestorsb())) +
            (advisor2 == null ? 0 :
                        (1 + advisor2.ancestorsb()));
}
```

Question 4.(a).
1. true
2. true
3. true
4. false
5. 6
6. 17

(b) 1. Apparent and real classes of a are A and B.
2. Apparent and real classes of b are A and A.



a
x1
b
x1
c
x2
x1
A
bv
3
A() m()
B
B() m()
x2
A
bv
7
A() m()