LAST NAME______________________________first name___________________________ Cornell Net id_____________

CS1110 Prelim 2 8 November 2011

This 90-minute exam has 6 questions (numbered 0..5) worth a total of 100 points. Scan the whole test before starting. Budget your time wisely. Use the back of these pages if you need more space. You may separate the pages; we have a stapler at the front of the room.

Question 0 (2 pts). Write your last name, first name, and Cornell NetId, legibly, at the top of each page.

Question 1 (16 points) while-loops. Complete the body of the following function —the repetend and all the underlined places. Note that the invariant is given and must be used, so study it carefully. Examples: `eq("aaaxxyx",0)` is `3` and `eq("aaaxxyx",5)` is `1`. Be careful with the loop condition; there are two situations in which execution of the loop should terminate.

```
/** = the length of the sequence of equal characters beginning at s[i].
      Precondition: 0 <= i < s.length(). */
public static int eqChars(String s, int i) {

    int k= ____________________________;
    /** invariant: characters in s[i..i+k-1] are the same. */

    while (_____________________________________________________________) {


    }
    return _______________________________________;
}
}
```

Question 2 (20 points) recursion. We want to compress strings that have long sequences of equal characters. For example, we want to compress `"bbbbaaa$$$$$$$$$$$$$$$$d"` to `"b4a3$16d1"`. In the compression, each sequence of equal characters is given by the character followed by the length of the sequence. Write function compress to do this. Use no loops; use only recursion. You may use function `eqChars` of the previous question 1.

```
/** = the compression of s, as explained above.
      Precondition: s contains no digits 0..9. */
public static String compress(String s) {









}
```

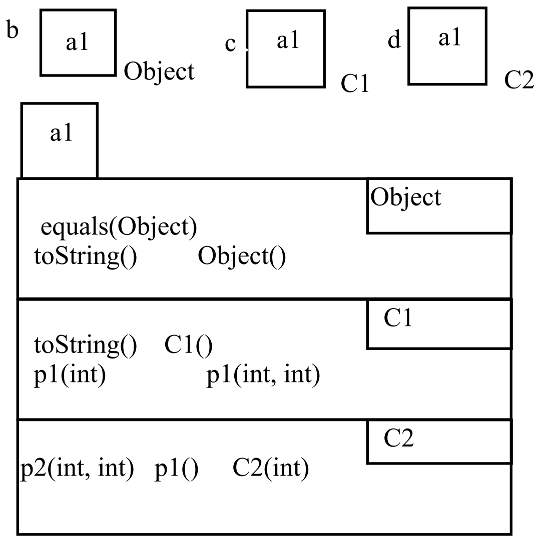
LAST NAME______________________________first name____________________________ Cornell Net id_____________

Question 3 (20 pts) OO: casting, instanceof, etc.

(a) Consider the three variables and object, to the right. Each variable is annotated with its apparent class. Below are six method calls; circle those that are legal.

b.p1(5) c.p1(5) d.p1(5)

b.p1() c.p1() d.p1()



(b) Using variable b, write a legal expression that calls method `p2` of the object.

(c) Write the body of function `equals` in class `Planet`, below. You may not write any methods in class `CelestialBody`, and you may not assume that others exist. A table on page 3 has methods of class `Vector` that might be useful.

```
/** An instance maintains information about a celestial body */
public class CelestialBody {
   private String bodyName; // Name of the body
   private boolean life; // True if life exists here

   /** Constructor: A Celestial Body with life li, name n*/
   public CelestialBody(boolean li, String n) { ... }

   /** = “b is a CelestialBody and has the same body name and life property as this celestial body” */
   public boolean equals(Object b) { ... }
}
```

```
/** An instance maintains information about a planet */
public class Planet extends CelestialBody {
   private Vector<CelestialBody> ms; // The moons of the planet, in alphabetical order

   /** Constructor: a Planet with life l, body name n, and no moons. */
   public Planet(boolean l, String n) { ... }

   /** = "b is a Planet and has the same body name, life property, and moons as this Planet */
   public boolean equals(Object b) {










   }
}
```

LAST NAME____________________________first name__________________________ Cornell Net id____________

Question 4 (22 pts) Abstract classes, abstract methods.

(a) What is the effect of adding the keyword **abstract** to a class definition?

(b) What is the effect of adding the keyword **abstract** to a method definition?

(c) Solve this little puzzle, using your knowledge of abstract classes, casting, etc. Below are classes `A`, `B`, `C`, and `D` for you to complete, filling in the underlined places and declaring exactly **four** methods (no constructors). Variables `a`, `b`, `c`, and `d` have types `A`, `B`, `C`, and `D`, respectively. Complete the four classes so that the four points 1..5 listed below are satisfied. The five points could be listed in any order; the ordering below may help you draw conclusions from each point as you read it.

We suggest you solve the puzzle by drawing the hierarchy (or objects) as you study 1..5 below and complete the classes only when you have the solution.

1. Each of the following statements produces an error at compile time:

   a= **new** A(); // compile-time error
   a.h(); // compile-time error

2. The following sequence of 6 statements compiles and runs without error:

   d= **new** D(); a= d; b= (B)a;
   a.a(); b.h(); d.h();

3. **If** we were to add the following declaration, it would give a compile-time error:

   public class E **extends** A { } // compile-time error

4. No object has partitions for all of A, B, C, and D.

5. The following statements compile, but the second produces an error at runtime.

   a= **new** C();
   c= (B)a; // run-time error

____________ **class** A _______________ {

}

____________ **class** B _______________ {

}

____________ **class** C _________________ {

}

____________ **class** D_________________ {

}

Vector methods		
	Vector()	Constructor for an empty Vector —no objects in it
void	v.add(p)	Append object p to Vector v's list of objects
int	v.size()	The length of Vector v's list of objects
Object	v.get(i)	Return the object at position i in v
boolean	v.equals(Object ob)	= "ob is a Vector, lists v and ob are the same size, and corresponding elements of v and ob are equal". Two elements e1 and e2 are equal if (e1==**null** ? e2==**null** : e1.equals(e2))
boolean	**boolean**	= "Vector v's list contains ob, according to method ob.equals"

LAST NAME_____________________________first name___________________________ Cornell Net id____________

0	___________	out of 02
1	___________	out of 16
2	___________	out of 20
3	___________	out of 20
4	___________	out of 22
5	___________	out of 20
Total	________	out of 100

Question 5 (20 pts) Arrays and loops.

(a) Write a declaration with an array initializer for an array `bb` that contains the five strings `"a"`, `"e"`, `"i"`, `"o"`, and `"u"`.

(b) Array `b`, with element type `CelestialBody` (see question 3) contains no **null** elements. However, `b` may contains duplicates. For example, `b` might be {a1, a2, a2, a3, a5, a4, a4, a0}, where each element is the name of a `CelestialBody` object. The function given below is supposed to extract elements from `b` and put them in `Vector v`, but without duplicates. For example, given `b` as just shown, `v` will contain {a1, a2, a3, a5, a4, a0}. Write the body of the function given below as follows:

1. Notice that a range of integers has to be processed. Write that range here:
2. Under the first underline is a command: "Store in v a list ..." Rewrite that as the postcondition in the appropriate place.
3. Fill in the header of the for-loop —between "**for** (" and ");".
4. Write the loop invariant in the appropriate place.
5. Write any necessary initialization.
6. Write the repetend.

Note: "all duplicates appear next to each other" means that situations like the following won't happen, because the a1's are not next to each other.

b = {a1, a2, a1, a1}

```
/** = a list of elements of b with no duplicates.
      Precondition: the elements of b are not null, and all duplicates appear next to each other */
public static Vector<CelestialBody> extract(CelestialBody[] b) {

    __________________________________________________________;
    // Store in v a list of all elements of b[0..length-1] but with no duplicates

    // invariant: __________________________________________________________

    for (__________________________________________________________) {

    }

    // Postcondition: __________________________________________________________
    return v;
  }
```