NAME (LEGIBLY): _____ **CORNELL NETID:**_____
(last, first)

This 90-minute exam has 6 questions worth a total of 100 points. Spend a few minutes looking at all questions before beginning to see what is expected. Budget your time wisely. Use the back of these pages if you need more space. You can tear the pages apart; we have a stapler at the front of the room.

Question 0		**/02**
Question 1		**/22**
Question 2		**/20**
Question 3		**/15**
Question 4		**/21**
Question 5		**/20**
Total		**/100**

Question 0 (2 points): Write your name and NetID, legibly, at the top of each page.

Question 1 (22 points): Answer the following questions concisely:

(a) What is a class? An object?

(b) Where does a static variable go? A non-static variable?

(c) What is a parameter? An argument?

(d) Name the three kinds of methods and state the use of each.

(e) Below each of these 3 expressions, write its value. If evaluation leads to an error, write 'BAD'.

```
true || (5/0 < 1)        (5/0 < 1) || true        3/2
```

NAME:_____ **NETID:**_____

Question 2 (20 points): At the bottom of the page are two class definitions. Draw one folder (object, instance) of each:

```
public class Vehicle {
    private int weight= 0; // Weight of vehicle

    /** = v1 and v2 are the same folder */
    public static boolean isEqual(Vehicle v1,
                                  Vehicle v2) {
        return v1 == v2;
    }

    /** = this vehicle's weight */
    public int getWeight() {
        return weight;
    }

    /** Set this vehicle's weight to w */
    public void setWeight(int w) {
        weight= w;
    }

    /** = "this vehicle can cross a bridge
       that can bear a max weight of w" */
    public boolean canCross(int w) {
        return weight <= w;
    }
}
```

```
public class Truck extends Vehicle {

    // Weight of load truck can carry
    private int capacity= 0;

    /** = this truck's capacity */
    public int getCapacity() {
        return capacity;
    }

    /** Set this truck's capacity to c*/
    public void setCapacity(int c) {
        capacity= c;
    }

    /** = "this truck can cross a bridge
      that can bear a max weight of w" */
    public boolean canCross(int w) {
        return getWeight() + capacity < w;
    }
}
```

Question 3 (15 points): Assume that the following two variables have been initialized to contain (the names of) folders:

```
Vehicle x;          Truck y;
```

(a) Write down the names of the methods and fields that `Truck` inherits from `Vehicle`.

(b) Write down the names of the methods that `Truck` overrides.

(c) Suppose `y` contains (the name of) a `Truck` with weight 100 and capacity 100. What is the value of the expression `y.canCross(150)` ?

NAME:_____ **NETID:**_____

Question 4 (21 points): This question refers to classes `Vehicle` and `Truck` defined in Question 2.

(a) Write a subclass `Airplane` of `Vehicle`.

1. Class `Airplane` should contain a field for the number of passengers on the plane.
2. Its constructor should let one give both the plane's weight and the number of passengers.
3. It should have a method `flyWeight()` that calculates the combined weight of the plane and its passengers (assume each passenger weighs 150).
4. It should have a method `canCross()`that does something reasonable (there are times when a plane has to cross a bridge).

(b) You set out to test your `flyWeight()` method. You write the following JUnit test case:

```
public void testFly() {
    Airplane a= new Airplane(1000, n);
    assertEquals(100+150*n, a.fly weight());
}
```

You get the following errors when compiling the program. What do you need to do in order to fix the errors?

1. cannot resolve symbol : variable n

2. cannot resolve symbol : method flyweight ()

Question 5 (20 points): A method name in Java has to have a certain format. Among other things:

- It must start with a letter or an underscore ('_').
- It cannot contain spaces (' ').
- It must have fewer than 65536 characters.

Write a method that converts its only parameter, a String, to a String that follows the above rules. The following methods are available to you:

`s.length()`	= the length of the string (as an **`int`**)
`s.charAt(k)`	= the character at position `k of String s`
`s.substring(h,k)`	= a String consisting of characters in `s[h..k-1]`, i.e. `s[h],s[h+1],...,s[k-1]`
`s.substring(h)`	= a String consisting of characters `s[h..s[s.length()-1]`
`s.replaceAll(n,h)`	= `s` but with all occurrences of `String n` replaced by `String h`
`Character.isLetter(char c)`	= "`char c` is a letter"

```
/** = String s converted as follows:
   · If s does not start with a letter or '_', add '_' to the beginning of s.
   · Remove all spaces from s.
   · If s has more than 65535 chars, delete those past the first 65535 chars.
*/
public String makeValidMethodName(String s) {




}
```