Cornell net id _____ **Name** _____

Section day _____ **Section time** _____

CS 100J Prelim 2 — 16 March 2006

Have a good break!!!

This 90-minute exam has 6 questions (numbered 0..5) worth a total of 100 points. Spend a few minutes looking at all questions before beginning. Use the back of the pages, if you need more space.

0	_____	out of 02
1	_____	out of 10
2	_____	out of 15
3	_____	out of 14
4	_____	out of 22
5	_____	out of 37
Total	_____	out of 100

Question 0 (2 points). Fill in the information, legibly, at the top of each page (Hint: do it now.)

Question 1 (10 points). (a) Consider the call C.m(5) on the following static method m, declared in class C.

```
public static void m(int p) {
    for (int k= 0; k != p; k= k+1) {
        int x= p*k;
        p= p + x;
    }
}
```

(a1) When is p created during execution of the call?

(a2) When is k created during execution of the call?

(a3) When is x created during execution of the call?

(b) Below, fill in the assignment so that the following assertion is true —i.e. if execution starts with assertion P true, then after the assignment to y, R is true. Be careful.

```
// {P: y  is the sum of  h..100}
y =                                   ;
// {R: y is the sum of  (h–1)..101}
```

(c) State the four loopy questions used to develop a loop.

Cornell net id _____ **Name** _____

Section day _____ **Section time** _____

Question 2 (15 points). At the bottom of the page are definitions of three classes: Animal, Cat, and Dog. Assume that these three statements have been executed:

```
Animal a= new Cat("A", 1);
Cat c= new Cat("C", 2);
Dog d= new Dog("D", 3);
```

Write the value of each expression given below; if one leads to an error, write "ERROR". It may help you to draw the objects that are created by execution of the three statements above.

(2a) c.isOlder(a);

(2b) a.getWeight();

(2c) c.getWeight();

(2d) d.getWeight();

(2e) c.getNoise();

(2f) d.getNoise();

(2g) ((Cat)a).getNoise();

(2h) ((Cat)d).getNoise();

(2i) ((Dog)d).getNoise();

(2j) ((Dog)a).getNoise();

```
public class Animal  {
    private String name; // name of the animal
    private int age; // age of the animal

    /** Constructor: an Animal with name n,
        age a */
    public Animal(String n, int a)  {
        name= n;
        age= a;
    }

    /** = "this Animal is older than h"
          Precondition: h is not null */
    public boolean isOlder(Animal h)  {
        return this.age > h.age;
    }

    /** = noise this animal makes */
    public String getNoise()    {
        return "";
    }

    /** = name of this Animal */
    public String getName() { return name; }

    /** age of this Animal */
    public int getAge() { return age; }
}
```

```
public class Cat extends Animal  {
    /** Constructor: a Cat with name n, age a */
    public Cat(String n, int a) {
        super(n, a);
    }

    /** = noise this cat makes */
    public String getNoise()  {
        return "meow";
    }

    /** = weight of this Cat */
    public int getWeight()  {
        return 20;
    }
}

public class Dog extends Animal  {
    private String breed= “”; // breed, eg. “terrier”

    /** Constructor: a Dog with name n, age a */
    public Dog(String n, int a)  {
        super(n, a);
    }

    /** = weight of this Dog */
    public int getWeight()  {
        return 30;
    }
}
```

Cornell net id _____ **Name** _____

Section day _____ **Section time** _____

Question 3 (14 points). Write the bodies of the two methods whose specifications and headers are given below, assuming that these methods are in class Dog.

```
/** = ob is a non-null dog with the same fields as this Dog */
public boolean equals(Object ob) {




}
```

```
/** Constructor: a Dog that is a terrier, has name Spot, and is y years old */
public Dog(int y) {




}
```

Cornell net id _____ **Name** _____

Section day _____ **Section time** _____

Question 4 (22 points). The Ithaca Animal Spa needs your help! They have two separate rooms, one for cats and one for all other animals. But right now, their website just collects one list of all the animals who have been signed up for treatment. We need you to write a function removeCats in class Cats, whose specification and header is given below. This function removes all the Cats from its parameter and returns a new vector of type Vector<Cat> that contains all the Cats in the original Vector (in the same order).

For example, suppose woof and rover are instances of class Dog and furball and meowy are instances of class Cat, and parameter v is

[woof, furball, rover, meowy]

Then, execution of the call removeCats(v) changes v to [woof, rover] and returns a new Vector that contains [furball, meowy].

You will probably write a loop; you need not write a loop invariant, but it may help you.

Assume v has type Vector<C> for some class C		
Return	**Method**	**Purpose**
C	`v.get(int k)`	= `v[k]`
void	`v.set(int k, C c)`	replace `v[k]` by `c`
int	`v.size()`	= the number of elements in `v`'s list
int	`v.indexOf(C c)`	= `i`, where `v[i]` is `c; -1 if c not in v`
int	`v.add(C c)`	Add object c to v and return the index (position) of the newly added object in v.
	`v.remove(int k)`	Remove element v[k], changing v so that it contains v[0..k–1] followed by v[k+1..]

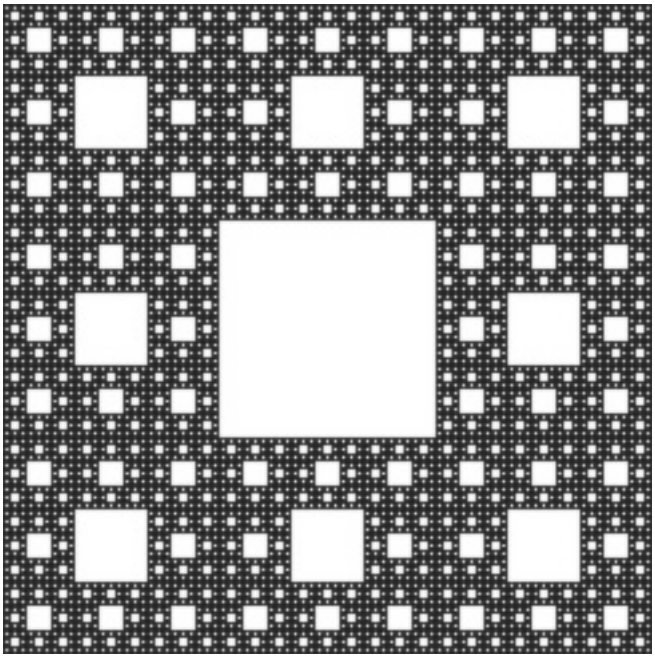
```
/** Remove all Cats from v and return a Vector that contains the removed Cats */
   public static Vector<Cat>  remCats( Vector<Animal> v ) {




}
```

Cornell net id _____ **Name** _____

Section day _____ **Section time** _____



Question 5 (37 points). Write a **recursive procedure** scarpet, which will contain a loop, that draws a "Sierpinski Carpet", named after the person who conceived of the idea (his last name was Sierpinski, not Carpet). An example appears to the right. The specification and header of scarpet appears on the next page. From the spec, you can see that the procedure draws *something* in a square of width and height w whose top-left corner is (x, y). Note that w is a power of 3.

The square to be drawn is originally black, and the pen color is white. You don't have to do anything to change the pen color.

Here's how procedure scarpet should work.

If w < 3, scarpet draws nothing.
If w ≥ 3, the square is viewed as a grid of 9 squares, each of width and height w/3, as pictured to the right. We describe what goes into each of the 9 squares.

sc 0	sc 1	sc 2
sc 3	4	sc 5
sc 6	sc 7	sc 8

The center square gets a white rectangle. Do this using a procedure call like this:

```
graphics.fillRect(x1, y1, w1, h1);
```

where (x1, y1) gives the coordinates of the top-left pixel and the rectangle has width w1 and height h1 (you have to figure out what values to use for the arguments).

Each of the other 8 squares (those with "sc" in them) gets a Sierpinski Carpet.

Solving the problem There are three pieces to procedure scarpet, and we will attempt to grade you separately on each. How well we are able to do on this depends on *your* ability to write your code in such a way that the three pieces are easily seen. Here are the three pieces.

1. **The recursive concept**, with the idea of base and recursive cases.
2. **One loop that processes each of the 9 squares** of the grid in turn, drawing what it has to in that square. We have numbered the squares in their lower right corners, and the loop should process them in that order.
3. **Getting the coordinates and lengths of the squares in the grid correct**. What is the top-left pixel of each square given that the upper-left corner of the original square is (x, y) and that its height and length are w? This may help: If k is the control variable for the loop that processes the squares, expressions k/3 and k%3 may be useful. It may help you to pencil in coordinates and lengths of some of the squares of the grid that appears above and to the right.

Cornell net id _____ **Name** _____

Section day _____ **Section time** _____

```
/** Draw a Sierpinski Carpet of width and height w with top left corner (x, y).
      Precondition: w is a power of 3.
      Precondition: Place to be drawn is black, and the pen color is white. */
public static void scarpet(int x, int y, int w) {




}
```