NAME (PRINT LEGIBLY!)_____
(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 so that you can see what is expected. Budget your time wisely. Use the back of the pages, if you need more space.

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

Question 1 (15 points). Write the body of method `fixZip`,
specified below. The following methods of class `String` may be
useful:

- s.length()   = the number of characters in s.
- s.charAt(k)   = the character at position (or index) k of string s
- s.substring(h, k)   = a string consisting of characters s[h], s[h+1], …, s[k–1]
- s.substring(h)   = a string consisting of characters s[h], s[h+1], …, s[s.length()–1]
- s.indexOf(c)   = index of first occurrence of character c in s (-1 if none)
- s.indexOf(s1)   = index of first char of the first occurrence of String s1 in s (-1 if none)

/** s, an address, may be a Cornell address with the wrong zip number. If s contains the substring
   "Cornell" and also the substring "14850", then return s but with "14850" replaced by "14853".
   Otherwise, return s.
   E.g. for s = "Gries; Olin 167; Cornell; Ithaca, NY 14850; USA",
        return "Gries; Olin 167; Cornell; Ithaca, NY 14853; USA". */
public String fixZip(String s) {

}

Question 2 (20 points). Below are two class definitions, for classes P and S.

(a) Draw a manilla folder (instance, object) of class P.

(b) Draw a manilla folder of class S. Note that both fields are given by initializing declarations. You have to draw the values in these fields correctly. If this requires drawing new folders, do so.

```
public class P {                          public class S extends P {
    public static int x= 3;                   public static double pi= 3.14;
    private S s= null;                         private P one= new P();
    public void m(String p) {
        x= p.length();                         private P two= null;
    }                                     }

    public void setX(int v) {
        x= v;
        x= x + 1;
    }
}
```

Question 3 (23 points). Use the back of the previous page for your answers, if you need more space. This question deals with classes Student and Undergrad, shown below, and two variables x and y declared as:

    Student x;   Undergrad y;

(a) Write down the names of methods and fields that class Undergrad inherits.

(b) Write down the names of methods in class Undergrad that override other methods.

(c) Suppose y contains an Undergrad with name "Johnny", netId "JD123", and college "A&S". Write down the value of expression y.toString() .

(d) Write a subclass Freshman of Undergrad with one field: the number of AP credits that the freshman has. Write a constructor as well as getter and setter methods for the field. Write method toString() —it should do something reasonable, based on function toString in other classes.

```
/** An instance is a student with name and netid. */
public class Student {
  // name of this Cornell student
  private String name= "";

  // netId of this student
  private String netId= "";

  /** Constructor: student with name n, netid id */
  public Student(String n, String id) {
    name= n;
    netId= id;
  }

  /** = this student's name. */
  public String getName() {
    return name;
  }

  /** = this student's net id. */
  public String getNetId() { return netId; }

  /** Set this student's netId to n. */
  public void setNetId(String n) {
    netId= n;
  }

  /** = description of this student. */
  public String toString() {
    return name + ", " + netId;
  }
}
```

```
/** An instance is an undergrad. */
public class Undergrad extends Student {
  // the student's college
  private String college;

  /** Constructor: a student with name n, netId id,
      and college c*/
  public Undergrad(String n, String id, String c) {
    super(n, id);
    college= c;
  }

  /** = the noise this undergrad makes */
  public String noise() {
    if (college.equals("Business School"))
      return "money!";
    return "education!";
  }

  /** = description of this student */
  public String toString() {
    return super.toString() + ", " + college;
  }
}
```

3

Question 4 (20 points). Below is a class Point as well as three variables. Execute the following statements, one after the other, and show what is printed by each println statement in the space provided afer it. **Hint**: You won't be able to do this properly unless you draw all the objects that are created and assign properly to the variables. We suggest using the back of the previous page for this.

b | **null**

c | **null**

d | **null**

```
b= new Point(5);
c= new Point(6, 7);
d= b;
System.out.println("d " + d);
```

```
c.setX(b);
System.out.println("b " + b);
```

```
System.out.println("c " + c);
```

```
System.out.println("d " + d);
```

```
b= new Point();
System.out.println("b " + b);
```

```
System.out.println("d " + d);
```

```java
/** A point (x, y) in the plane. */
public class Point {
    private int x= 0;
    private int y= 0;

    /** Constructor: The point (0,0). */
    public Point() {}

    /** Constructor: The point (0,y). */
    public Point(int y) {
        this.y= y;
    }

    /** Constructor: the point (x,y). */
    public Point(int x, int y) {
        this.x= x; this.y= y;
    }

    /** = a representation of this Point. */
    public String toString() {
        return "(" + x + "," + y + ")";
    }

    /** = the x-coordinate of this point. */
    public int getX() {
        return x;
    }

    /** = the y-coordinate of this point. */
    public int getY() {
        return y;
    }
    /** Set P's x-coordinate to
        this object's x-coordinate + 3. */
    public void setX(Point p) {
        p.x= this.x + 3;
    }
}
```

Cornell net id _____   **Name** _____

Question 5 (20 points).

(a) Define the term "parameter".

(b) Explain how a new-expression like **new** C(5, 3); is executed.

(c) Explain how to execute an assignment statement *<variable>*= *<expression>*;

(d) What is a local variable, and what is its scope?

(e) Suppose a function body is being executed because of a function call. What kind of statement causes the function body to terminate?