You may use one 8.5"x11" page of handwritten notes (dual-sided ok), but no books, computers, calculators, or cell phones during the exam. (Use this to help you *prepare* for the exam! Include details you may not remember during the exam)

CIS 200 Practice Problems for Final

This is NOT a past final exam. This is simply several practice problems to review *some* of the concepts that may appear on the exam. These practice problems cover only the concepts since Exam 3. I would strongly suggest you also carefully review all practice and actual given exams (1-3)

1) Write the **Java** class `Dictionary`. This class should contain the following:
   - An array of strings (words) as an **instance variable**
   - A **constructor** that takes an array of strings as a parameter, creates the necessary space for the instance variable array, and copies the values from the parameter into the instance variable.
   - A **`containsWord` method** that takes a string as a parameter. This method should return true if the parameter is in the instance variable array, and false otherwise.
   - A **`startsWith` method** that takes a character as a parameter. This method should return a count of how many words in its instance variable array start with that parameter character.

2) Create a Java class called **`TestDictionary`** that contains a **`main`** method. Inside the `main` method, declare a `Dictionary` object with the words “apple”, “banana”, and “avocado”, and “carrot”. Print how many words begin with the letter ‘a’. You must call the `startsWith` method when making your calculations.

3) Write the following **Java** method:

```
public static int indexOf(String[] words, String s)
```

This method should return the index of `s` within the `words` array. If `s` is not an element in `words`, you should **throw an appropriate exception**.

4) Write the **C#** class `Point`, which represents an `(x,y)` coordinate. It should have (double) **instance variables** for the `x` and `y` values, and a **constructor** that takes parameters for the `x` and `y` values and initializes the instance variables.

It should have the method **`Print`**, which prints this point in the format **`(x,y)`**. Finally, it should have the method **`Midpoint`**. The `Midpoint` method should take another `Point` as a parameter, and should return a new `Point` object that is the midpoint between THIS point and the parameter. For example, the midpoint between (1,3) and (5,2) is ([1+5]/2,[3+2]/2) = (3, 2.5).

5) Write a **C#** class called `TestPoint`. This class should contain a `Main` method. Inside `Main`, do the following:

   a) Declare an array of type `Point` that can hold 10 points.

   b) Use a loop to ask the user to enter `(x,y)` coordinates for each point. For each one, create a new `Point` object with that information and store it at the current array location.

   c) Print the first point in the array by calling its `Print` method.

   d) Get the midpoint between the second and last points by calling the `Midpoint` method.

   e) Print that midpoint by calling its `Print` method.

6) Consider the C# program below. This program is supposed to get 10 numbers from the user, sort them (from smallest to largest), and then print them out in sorted order. However, the program has a mistake that prevents it from working properly. What is the mistake? What effect will this mistake have on the final list of numbers that is printed? How might the program be fixed to work correctly?

```
using System;
public class FinalExam {
    public static void Main() {
        int[] nums = new int[10];
        for (int i = 0; i < nums.Length; i++) {
            Console.Write(“Enter number: “);
            nums[i] =
                Convert.ToInt32(Console.ReadLine());
        }

        int pos;
        for (int i = 0; i < nums.Length; i++) {
            pos = i;
            for (int j = i+1; j < nums.Length; j++){
                if (nums[j] < nums[pos]) pos = j;
            }
            Swap(nums[i], nums[pos]);
        }

        for (int i = 0; i < nums.Length; i++) {
            Console.Write(nums[i] + “ “);
        }
        Console.WriteLine();
    }

    public static void Swap(int x, int y) {
        int a = x;
        x = y;
        y = a;
    }
}
```

7) Consider the `Point` class from #4. In each of the following statements, explain what it does, what is printed, or if there is an error. Assume the statements are executed in order.

a)
```
Point p1 = null;
```

b)
```
Point p2 = new Point(3.0,4.2);
```

c)
```
double[] vals = {2.5,3.1};
Point p3 = new Point(vals);
```

d)
```
Point p4 = new Point(0.0,0.0);
```

e)
```
p2.Print();
```

f)
```
Console.WriteLine(p2);
```

g)
```
Point p5 = p2;
p5.Print();
```

h) `p5 = new Point(4.8,5,6);`

i) `p2.Print();`

j) `p5.Print();`

k)
```
p2 = p2.Midpoint(p5);
p2.Print();
```

l) `Console.WriteLine(p4.Midpoint(p2));`

m) `p1 = p5.Midpoint(vals);`

n) `p1.Print();`