1. What is your course section?

   Answer: ____________

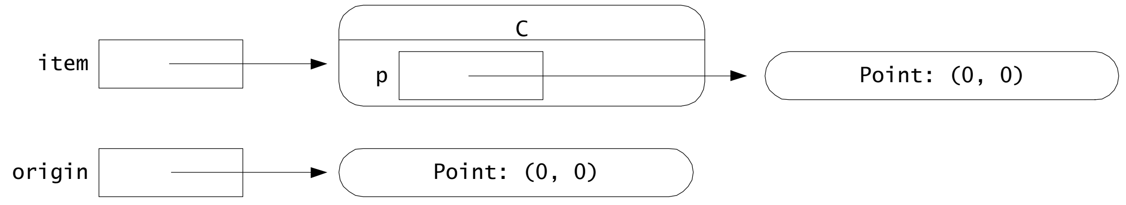
Questions 2 – 8 consider the following class `C`.

```
public class C {
    private Point p;                 // instance variable

    public C(Point v) {              // specific constructor
        p = new Point(v.getX(), v.getY());
    }

    public void set(Point v) {       // instance method
        p = v;
    }

    public void perform(Point v) {   // instance method
        v.setX(10);
        p.setX(10);
    }

    public void do(Point v) {        // instance method
        v = new Point(10, 0);
        p = v;
    }

    public static int f(int x) {     // class method
        return x + 7;
    }

    public static void g(int x) {    // class method
        System.out.println( x + 7 );
    }

}
```

2. Consider the following code segment.

```
Point origin = new Point(0, 0);
C item = new C(origin);
```

After it completes, memory has the following depiction.



Do `item`'s instance variable `p` (i.e., `item.p`) and `origin` refer to the *same* `Point` object after the preceding code segment?

YES ______ NO _____

3. Consider the same code segment. Do `item.p` and `origin` refer to *equivalent* `Point` objects after the code segment finishes?

```
Point origin = new Point(0, 0);
C item = new C(origin);
```

YES ______ NO _____

4. Consider the following code segment. Do `item.p` and `origin` refer to the *same* `Point` object after the code segment finishes?

```
Point origin = new Point(0, 0);
C item = new C(origin);
item.set(origin);
```

YES ______ NO _____

5. Consider the following code segment. Do `item.p` and `origin` refer to the *same* `Point` object after the code segment finishes?

```
Point origin = new Point(0, 0);
C item = new C(origin);
item.perform(origin);
```

YES ______ NO _____

6. Consider the following code segment. Do `item.p` and `origin` refer to the *same* `Point` object after the code segment finishes?

```
Point origin = new Point(0, 0);
C item = new C(origin);
item.do(origin);
```

YES ______ NO _____

7. Consider the following code segment. Does it compile successfully?

```
int y = C.f(4);
```

YES ______ NO _____

8. Consider the following code segment. Does it compile successfully?

```
int y = C.g(4);
```

YES ______ NO _____

9. Suppose `b` is an *already* defined and initialized `int` array with 5 elements. Write a *single statement* that defines and initializes an `int` variable `i` to the value of the first element in `b`.

10. Suppose `b` is an *already* defined and initialized `int` array with 5 elements. Write a *single statement* that defines and initializes an `int` variable `i` to the value of the last element in `b`.

11. If it is possible, write a `static void` method `swap()` that takes two formal `int` parameters `x` and `y`. When invoked, the method is to interchange the values of its actual parameters. For example, with this method the following code segment

    ```
    int a = 10;
    int b = 11;
    swap(a, b);
    System.out.println(a + "   " + b);
    ```

    should display

    ```
    11   10
    ```

    If it is not possible to write the method explain why?

12. If it is possible write a `static void` method `swap()` that takes three formal parameters: an `int` array `a` and two `int` variables `i` and `j`. When invoked, the method is to interchange the values of the `i`th and `j`th elements of the actual array parameter. For example, with this method the following code segment

    ```
    int[] list = new list[3];
    int m = 1;
    int n = 2;
    a[m] = 10;
    a[n] = 11;
    swap(list, m, n);
    System.out.println(list[m] + "   " + list[n]);
    ```

    should display

    ```
    11   10
    ```

    If it is not possible to write the method explain why?

Questions 13 – 16 have you complete and use the following class `State`, where class `State` provides a representation of two values interest in regard to a state — its name and population size.

```
1.    public class State {
2.        private String name;    // represents name of the state
3.        private int size;       // represents number of people in the state
4.
5.        public State() {
6.            // TO BE FILLED IN **************
7.        }
8.
9.        public State(String s, int n) {
10.           name = s;
11.           size = n;
12.       }
13.
14.       public String toString() {
15.           return "(" + name + ": " + size + ")");
16.           p = v;
17.       }
18.
19.       public Object clone() {
20.           // TO BE FILLED IN **************
21.           return result;
22.       }
23.
24.       public boolean equals(Object v) {
25.           // TO BE FILLED IN **************
26.       }
27.   }
```

13. Write a *single statement* that can replace line 6 so that the default constructor initializes a `State` object to represent Virginia with its population of 7,078,515 people.

14. Write a *single statement* that can replace line 20 so that the `clone()` method produces a new object that is equivalent to the invoking (`this`) object.

15. Write a *code fragment* that can replace line 25 so that the `equals()` returns `true` if and only if `Object` parameter `v` is a `State` object equivalent to the invoking (`this`) object.

16. Write a *single statement* that defines a `State` array variable named `usa`. The definition should cause `usa` to reference a new `State` array with fifty elements.

17. Write a static `int[]` method `makeCopy()` that takes a single formal parameter `a`, where parameter `a` is an `int` array `a`. When invoked, the method returns a new `int` array of size `a.length`. The elements of this new array are equal to the corresponding elements of `a`.

18. Because an array is an object, it has a `clone()`, `equals()`, and `toString()` methods.

    YES ______ NO _____

19. Write a *single statement* that defines and initializes an `int` array variable `unit` that is initialized to represent four 8s.

20. Write a static `boolean` method `isSorted()` that takes a single formal parameter `a`, where parameter `a` is an `int` array `a`. When invoked, the method returns whether the elements of the list are in sorted order. Hint: an array `a` is sorted if and only if `a[i]` $\leq$ `a[i+1]` for all valid `i` and `i+1`.

PLEDGE