Name: KEY Section: ___________

CSSE 220—Object-Oriented Software Development

Exam 1 – Part 1, Sep. 20, 2016

This exam consists of two parts. Part 1 is to be solved on these pages. If you need more space, please ask your instructor for blank paper. After you finish Part 1, please turn in your Part 1 answers and then open your computers and wait quietly for the programming exam review section of class to begin.

Allowed Resources on Part 1: You are allowed one 8.5" by 11" sheet of paper with notes of your choice. This section is *not* open book or open notes; and you are not allowed to use your computer for this part.

You will have 50 minutes (the first Rose hour of class) to complete Part 1. Part 2 will be completed in the next class.

Please, begin by writing your name on every page of the exam. We encourage you to skim the entire exam before answering any questions.

Problem	Poss. Pts.	Earned
1	10	______
2	12	______
3	8	______
4	5	______
Paper Part Subtotal	**35**	______
Computer Part Subtotal	**65**	______
Total	**100**	______

Part 1—Paper Part

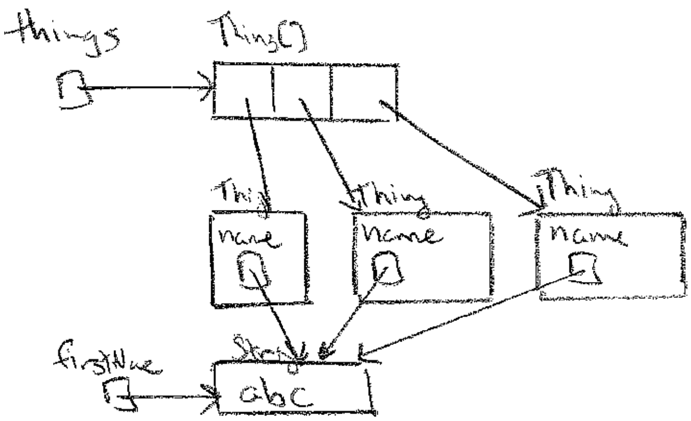
```
public static class Thing {
  private String name;

  public Thing(String name) {
    this.name = name;
  }

  public String getString() {
    return this.name;
  }

  public void append(String string) {
    this.name = this.name + string;
  }
}

public static class Holder {
  private Thing held;

  public Holder(Thing abc) {
    this.held = abc;
  }

  public void combineA(Thing other) {
    this.held = other;
  }

  public Thing combineB(Thing other) {
    Thing alt = new Thing(other.getString());
    this.held.append(other.getString());
    return alt;
  }

}
```

The next question refers to the classes on this page. The javadocs are omitted to save space. DO NOT TYPE THIS CLASS IN ECLIPSE.

1. (10 points, 3 each for a & b, 4 points for c) Below are several code snippets that use the Thing and Holder classes. For each snippet, first *draw a box-and-pointer diagram* (in the blank area below the snippet) showing the *final* result of executing it (i.e., you do not have to show any of the intermediate steps of any of the temporary variables created in the various methods).

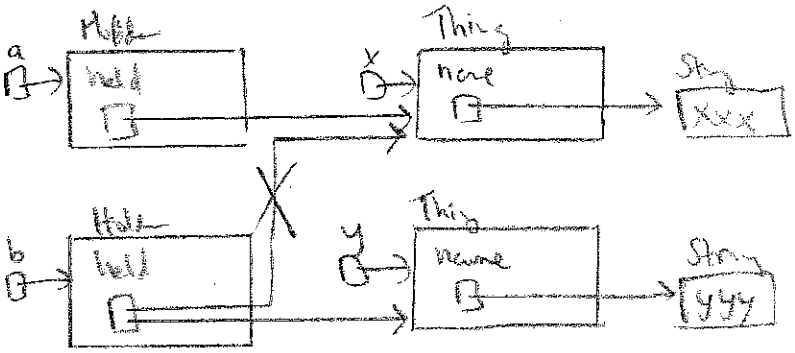
```
Thing[] things = new Thing[3];
String firstName = "abc";
for(int i = 0; i < things.length; i++) {
  Thing tempThing = new Thing(firstName);
  things[i] = tempThing;
}
```

(a) **Diagram:**



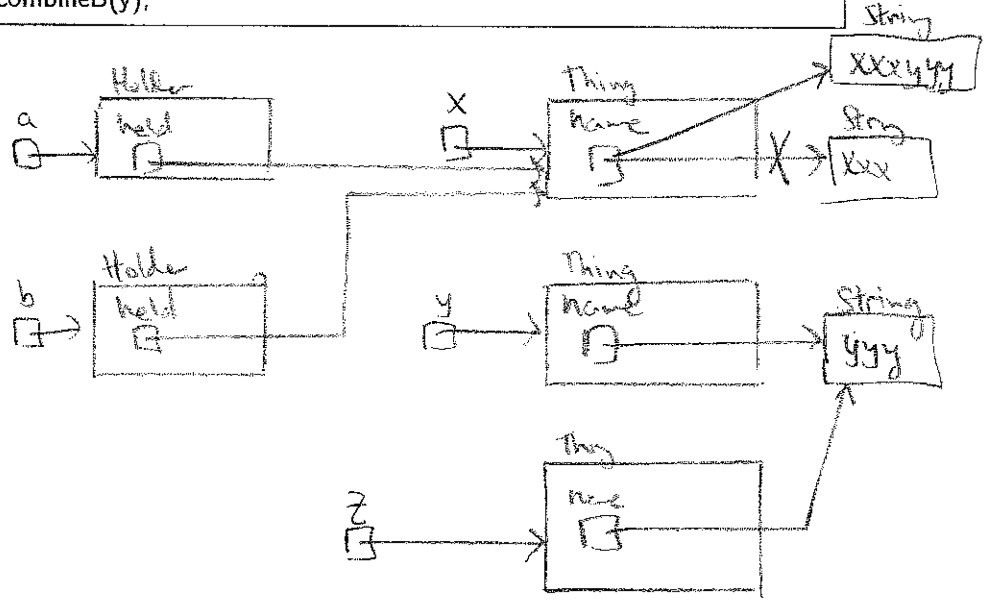
```
Thing x = new Thing("xxx");
Thing y = new Thing("yyy");

Holder a = new Holder(x);
Holder b = new Holder(x);

b.combineA(y);
```

(b) **Diagram:**



```
Thing x = new Thing("xxx");
Thing y = new Thing("yyy");

Holder a = new Holder(x);
Holder b = new Holder(x);
Thing z = b.combineB(y);
```

(c) **Diagram:**



2. (12 points) Predict the output for each code snippet below. (You do *not* need to draw a diagram, but you may if it might help you.) DO NOT TYPE THE CODE SNIPPETS FOR THIS QUESTION IN ECLIPSE. If output spans multiple lines, write additional lines below the Output: line.

```
ArrayList<String> l = new ArrayList<String>();
l.add("a"); l.add("b"); l.add("c"); l.add("d");
l.add("e"); l.add("f"); l.add("g"); l.add("h");
l.remove(6);
l.set(3, "Z");
l.add(0,"Q");
System.out.println(l); // prints out list in brackets [ ]
```

(a) Output: [Q, a, b, c, Z, e, f, h]

	0	1	2	3	4	5	6	7
Q	a	b	c	~~x~~ Z	e	f	~~g~~	h

```
HashMap<String, Integer> h =
  new HashMap<String, Integer>();
for(int i = 0; i < 3; i++) {
  h.put("QQQ", i);
}
System.out.println(h.get("QQQ"));
```

QQQ → ~~0~~ ~~1~~ 2

(b) Output: 2

```
int a1 = 11;
int b1 = 4;
double c1 = 4.9;
System.out.println(a1/b1);
System.out.println((int) c1);
```

(c) Output: 2
4

```
String q = "abc";
String w = "barber";
for(int i = 0; i < q.length(); i++) {
  w.replace(q.charAt(i), 'Q');
}
System.out.println(w);
```

3

(d) Output: barber

(strings are immutable - not replaced)
if not re-assigned

3. (8 points) For each loop below, write down one of the following: how many times its body will execute, infinity, or indicate that we can't tell from the information given. DO NOT TYPE THE CODE SNIPPETS FOR THIS QUESTION IN ECLIPSE.

var
36
12
4

```
int var = 36;
while(var % 3 == 0) {
  var = var / 3;
  System.out.println("in loop");
}
```

(a) Answer: 2

```
HashMap<String, Integer> h2 =
  new HashMap<String, Integer>();
h2.put("A",100);
h2.put("B",100);
h2.put("B",100);
for(String s : h2.keySet()) {
  System.out.println("in loop");
}
```

(b) Answer: ~~1~~ 2

```
for(int i = 10; i >= 0; i--) {
  System.out.println("in loop");
}
```

(c) Answer: 11

len
8

s1
a

s2
a

```
String s = "abXYQZba";
while(true) {
  System.out.println("in loop");
  int len = s.length();
  String s1 = s.substring(0,1);
  String s2 = s.substring(len - 1, len);
  if(!s1.equals(s2)) {
    break;
  }
  s = s.substring(1,len - 1);
}
```

(d) Answer: 3

s
bxYQZb

4. (5 points) *W*rite T next to the statements that are true, F next to the statements that are false.

T If this code compiles "MyClass.doMethod()" then the method doMethod in class MyClass must be static.

F A non–static method in a class cannot call a static method in the same class.

F The method replace in the standard java class String is static.

F This code calls the constructor of the class MyClass 5 times:

MyClass[] foo = new MyClass[5];.

T Standard java style suggests that fields of a class should be private.