Here are some possible questions. They are not necessarily indicative of the questions you will get on the exam, nor are they necessarily representative of the range of topics to be tested on the exam.

THESE QUESTIONS REPRESENT JUST A SMALL SAMPLING OF THE KINDS OF QUESTIONS THAT MAY BE ENCOUNTERED ON THE EXAM. IT IS YOUR RESPONSIBILITY TO PREPARE TO ANSWER QUESTIONS FROM ALL THE MATERIAL COVERED FOR THIS EXAM, INCLUDING THOSE CONCEPTS COVERED IN LABS.

- Identify parts of code.

See answer on page 3.

- Write code for relationship – see sample question on page 3. Relevant relationships are association (via constructor only, via accessor and mutator only, or via constructor, accessor and mutator), composition, and realization.

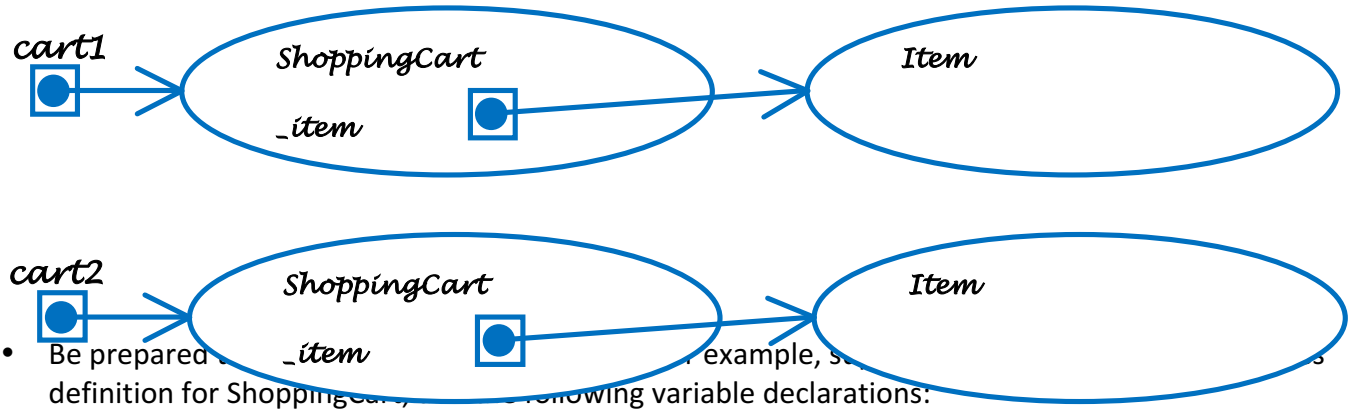
See answer on page 4.

- Given a small bit of code, draw an object diagram to show the resulting program state. For example, given this class definition:

```
public class ShoppingCart {
    private Item _item;
    public ShoppingCart() {
        _item = null;
    }
    public void addItem(Item i) {
        _item = i;
    }
    public Item removeItem () {
        Item temp = _item;
        _item = null;
        return temp;
    }
}
```

draw an object diagram to show the resulting program state

```
ShoppingCart cart1 = new ShoppingCart();
cart1.addItem(new Item());
ShoppingCart cart2 = new ShoppingCart();
cart2.addItem(new Item());
```



- Be prepared [illegible] example, [illegible] definition for Shopping[illegible] following variable declarations:

```
ShoppingCart cart1 = new ShoppingCart();
cart1.addItem(new Item());
ShoppingCart cart2 = new ShoppingCart();
cart2.addItem(new Item());
```

Write code to interchange the items in the two ShoppingCart objects referred two by cart1 and cart2. Do NOT just interchange the value of cart1 and cart2.

here's one possibility - there are others:

```
Item temp = cart1.removeItem();
cart1.addItem(cart2.removeItem());
cart2.addItem(temp);
```

- You must understand the difference between a local variable and an instance variable, in the following ways:
  - you must know how to declare each kind of variable
  - you must know the differences in scope and lifetime between them

Here's a possible question: assuming the following declaration has been done,

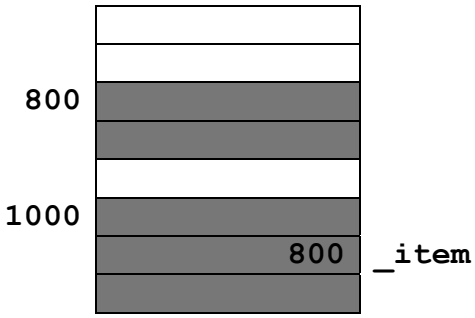
```
ShoppingCart cart = new ShoppingCart();
```

draw a memory diagram showing a possible snapshot of memory during the execution of the following method call:

```
cart.addItem(new Item());
```

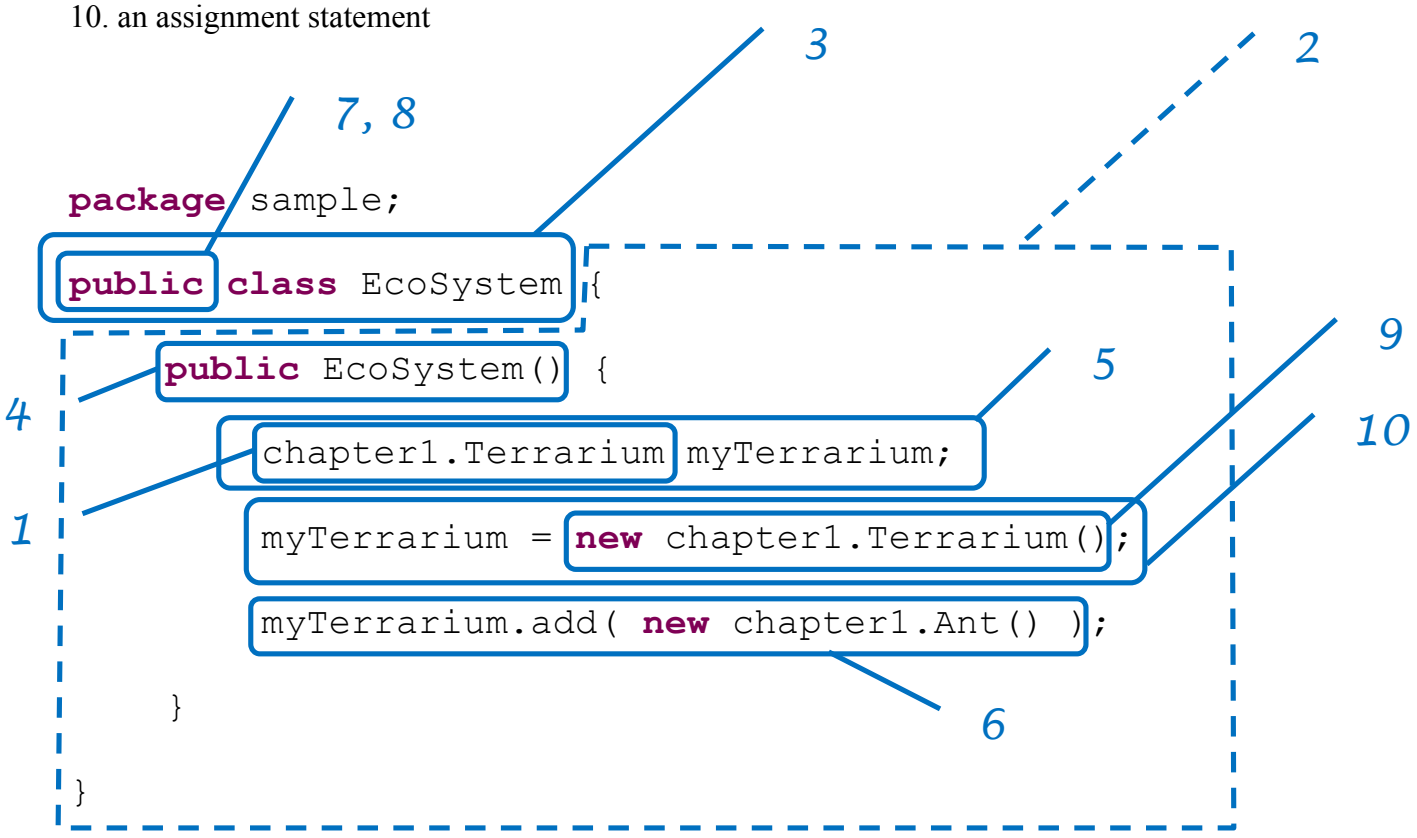
By "possible" here I am referring to the fact that the absolute location in memory of the objects/variables is not important, but rather how they are related. Assume that 401 and 402 are part of the invocation record of the cart.addItem(…) call, but that 400 is not.

400	1000	cart
401	1000	this
402	800	i

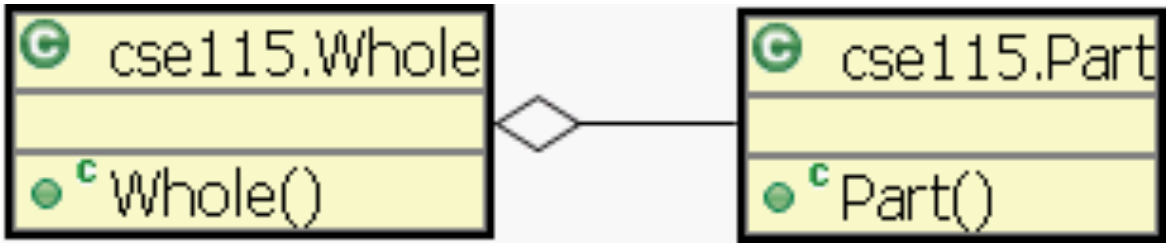


- In the code below, clearly circle and identify by number one (and only one) example of each of the following. If you believe no example exists in the given code, write \no example" next to that item.

1. fully qualified class name
2. class body
3. class header
4. constructor header
5. (local) variable declaration
6. method call
7. Java reserved word (any one will do)
8. access control modifier
9. an expression which instantiates a class
10. an assignment statement



- Write out the minimal code for the source class needed to show the following relationship:




```
package cse115;
public class Whole {
   private Part _p;
   public Whole() {
      _p = new Part();
   }
}
```

- Demonstrate an understanding of how control statements work (if, if-else, while, enhanced for), and also how collections function. For example, state what is printed when the following code executes:

```
HashSet<String> set = new HashSet<String>();
set.add("all");
set.add("cows");
set.add("eat");
set.add("grass");
for (String s : set) {
     if (s.length() == 3) {
          System.out.println("Passes: "+s);
     }
     else {
          System.out.println("Fails: "+s);
     }
}
```

```
Passes: all
Fails: cows
Fails: grass
Passes: eat
```