1. double length = 44.0;

int width =13;

Rectangle myRect = new Rectangle(length, width);

1. Identify the class

Rectangle

1. Identify the object

myRect

1. What type of parameter(s) are passed to the constructor?

double and int

2. Write out the signature for the constructor of the *Rectangle* class from #1 above.

public Rectangle (double l, int w) {

length = l;

width = w;

}

3. Suppose a constructor for the *Lunch* class is as follows:

public Lunch(boolean diet, int cal)

{

diet\_yes\_no = diet;

calories = cal;

}

Write appropriate code that will create a *Lunch* object called *yummy5*. Tell the constructor that, yes, you are on a diet, and the number of calories should be 900.

Lunch yummy5 = new Lunch (true, 900);

4. BankAccount account39 = new BankAccount(500.43);

a. Identify the class

BankAccount

b. Identify the object

account39

1. What type of parameter(s) are passed to the constructor?

double

5. A class is like a . An object is like a .

cookie cutter, cookie

Fill in the blanks above using the word “cookie” and “cookie cutter”.

6. What’s wrong (if anything) with the following constructor for the *School* class?

public void school(int d, String m)

{ … some code … }

constructors don’t need a return type

7. Which of the following is a correct association?

a. One class, many objects

b. One object, many classes

8. Which must exist first?

a. The class

b. The object

9. Is the following legal? If not, why?

//Constructor //This code is in *main* of *Tester* class

public House(int j, boolean k) int p = 3, q = 9;

{ …some code… } House myHouse = new House(p, q);

No, the constructor needs and int and a boolean. myHouse passed two ints as the arguments.

10. //Constructor

public Band(int numMembers, int numInstruments, String director, double amount)

{ …code…}

Band ourBnd = new Band(mem, instrmnts, “Mr. Perkins”, budget);

15-6

What should be the data types of:

1. mem

int

1. instrmnts

int

1. budget

double

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class BibleStory

{

public int var1;

public double var2;

public String sss;

public void Samson(double zorro) { …some code…}

public String getDelilah( ) { …some code…}

public BibleStory(String x, int y, double z) { …some code… }

}

11. From the *BibleStory* class above, write the signature of the constructor.

public BibleStory(String x, int y, double z) {

var1 = y;

var2 = z;

sss = x;

}

12. From the *BibleStory* class above, what is/are the instance field(s).

var1, var2, sss

13. From the *BibleStory* class above, write the signature(s) of the all the method(s).

public void Samson (double zorro)

public String getDelilah ( )

public BibleStory (String x, int y, double z)

14. Write code that instantiates an object called *philistine* from the *BibleStory* class. Pass

the following parameters to the constructor:

The integer should be 19, the *String* “Ralph”, and the *double* 24.18.

BibleStory philistine = new BibleStory(“Ralph”, 19, 24.18);

15. Assume an object called *gravy* has been created from the *BibleStory* class. Write code that will set the state variable *var2* to 106.9 for the *gravy* object.

grave.var2 = 106.9;

16. Write code that will print the value of the *BibleStory* data member, *sss*. Assume you

have already created an object called *bart*.

System.out.println(bart.sss);

17. Again, assume we have an object called *bart* instantiated from the *BibleStory* class.

What should you fill in for **<#1>** below in order that *sss* be stored in the variable *jj*?

**<#1>** jj = bart.sss;

String

18. Create a class called *Trail*. It should have instance fields *x* and *y* that are integers.

Instance field *s* should be a *String*. The constructor should receive a *String* which is

used to initialize *s*. The constructor should automatically set *x* and *y* both equal to 10.

There should be a method called *met* that returns a *String* that is the hex equivalent of

*x\* y*. This method receives no parameters.

public Trail {

int x, y;

String s;

Trail (String arg) {

s = arg;

x = 10; y = 10;

}

public String met () {

return Integer.toHexString(x\*y);

}

}

19. Consider a method whose signature is: *public double peachyDandy(int z)*

Write code that would call this method (assume we have an object name *zippo*). Also

assume that this code will be placed in the *main* method of a *Tester* class and that the

*peachyDandy* method is in some other class.

double d = zippo.peachyDandy (1);

20. Refer to the information in 19 above. What’s wrong with trying to call this method in

the following fashion? *double hamburger = zippo.peachyDandy(127.31);*

The method calls for an int and a double is passed.