CSSE 220 – Object-Oriented Software Development
Rose-Hulman Institute of Technology

Worksheet 10

Name (Print):________________________________ Section:______________

1. Null vs Empty Reference Checks

Type	Check for null	Check for empty	Create an empty instance
String			
Array			
ArrayList			
HashMap			

2. Identify whether each declaration is *null*, *empty*, or *valid (non-empty)*

```
String s = null;   ___

String s = "";   ___

String s = "hi";   ___

int[] nums = null;   ___

int[] nums = new int[0];   ___

int[] nums = {1,2,3};   ___

ArrayList<String> list = new ArrayList<>();   ___

ArrayList<String> list = null;   ___

ArrayList<String> list = new ArrayList<>(Arrays.asList("A"));   ___

HashMap<String,Integer> map = new HashMap<>();   ___

HashMap<String,Integer> map = null;   ___

HashMap<String,Integer> map = new HashMap<>(); map.put("X",1);   ___
```

3. Review - Calculate the length of each object:

```
String s = "Hello";  ______

int[] nums = {1,2,3,4,5};  ______

ArrayList<String> list = new ArrayList<>(); list.add("A"); list.add("B
   ");  ______

HashMap<String,Integer> map = new HashMap<>(); map.put("X",10);
   map.put("Y",20);  ______
```

4. Graphics: 1) ____________________ = components

   2) ____________________ = drawing + events + geometry

5. ____________________ = A top-level container

   ____________________ = The surface inside the frame where drawing happens

6. What is the alternative statement to if-else: ____________________

7. The keyword _______________ will stop the execution and break out of the switch block

8. The keyword _______________ specifies what to do if there is no case match

9. Complete the code using the alternative to if-else statements:

```

 _____________(month)  {

 _________    1:
 # code block

 _________   ;

 _________    2:
 # code block

 _________   ;

default:
 # code block
}
```

10. You are going to use **this** to call another constructor

```java
// Constructor that accepts all parameters
    public Book(String title, String author, int year) {
        this.title = title;
        this.author = author;
        this.year = year;
    }
  // write a constructor without parameters that must invoke  the
    above constructor and add some default values





```

11. What is the output?

```java
public class Example {
    int x;

    public Example() {
        this(10);
        System.out.println("Default Constructor");
    }

    public Example(int x) {
        this.x = x;
        System.out.println("Parameterized Constructor: " + x);
    }

    public static void main(String[] args) {
        Example e = new Example();
    }
}

```

Your answer:

12. What is the output?

```java
public class Rectangle {
    private int width;
    private int height;

    public Rectangle(int width, int height) {
        this.width = width;
        this.height = height;
    }

    public Rectangle(int side) {
        this(side, side);
    }

    public int area() {
        return width * height;
    }

    public static void main(String[] args) {
        Rectangle r = new Rectangle(4);
        System.out.println("Area: " + r.area());
    }
}
```

13. Select all that apply:

A. this can only be used in constructors
B. this is used to refer to the current object's instance variables and methods
C. this() (constructor invocation) must be the first statement in a constructor
D. this is especially necessary when there is a naming conflict between instance variables and parameters

14. What would you like to practice more or want to revisit in class?