1. (6 points) Fill in the blanks! A beautiful program is simple and understandable.

2. (3 points) What are variables used for in a Java program? To store information.

3. (3 points) What is the syntax (BNF) of an assignment statement?

\[ <\text{ass't stmt}> ::= <\text{variable}> = <\text{expression}> \]

4. (3 points) What is the semantics of an assignment statement?

a) Evaluate the expression. b) assign that value to the variable

5. (3 points) Assume there is a class named Halloween. Declare a variable of type Halloween named trickOrTreat. Halloween trickOrTreat;

6. (4 points) Assume you have a variable of type Halloween named trickOrTreat (see above!); write a line of code to instantiate a Halloween and store that new Halloween in the variable trickOrTreat. trickOrTreat = new Halloween();

7. (3 points) Write a statement to send your Halloween instance a getCandy() message which has no parameters. trickOrTreat.getCandy();

8. (3 points) Assuming it exists, what method in the Halloween class is invoked by System.out.println(trickOrTreat);? toString()

9. (3 points) What are parameters used for? to send information to methods

10. (3 points) Here are two legal names: baz, and Baz. These are different, because Java is case sensitive. If you are following the Java naming convention, which of these is the name of a class? Baz

11. (3 points) Why are initializing constructors useful? To simply initialize instance variables.

12. (3 points) In a Java method, what is this? The object which was sent the message which invoked the current method. Or… the current object.

13. (4 points) Why are good names important? Easy to understand!

14. (6 points) What are the three attributes of every variable? name, type, value

15. (6 points) Ruby (!) is facing east. Write an instruction that will make her face west if there are an odd number of coins at her corner, or east if an even number (make her end up with all the coins).

```java
void foo() {
    while (nextToACoin()) {
        turnLeft(); turnLeft();
        takeCoin();
    }
}
```
16. (3 points) How do you fix a bug you can't find? You can't!

17. (5 points) Assume you have a Circle class, with an int variable, x (the x-coord of the circle).
   Write a method, moveRight() which will increase x by its parameter.

   void moveRight(int howFar) {x = x + howFar;}

18. (5 points) Write a method, named product, which returns the product of its two int parameters.

   int product(int x, int y) {return x*y;}

19. (21 points) Write a complete class named Nap with just one int variable, length (in minutes), and
   three methods: accessors for the length variable, and public String toString() (that returns
   the class name and the value of its instance variable using getLength() -- this last so we know getLength() works!).

   class Nap {
       int length;
       public Nap(int length) {   //initializing constructor
           this.length = length;
       }
       public int getLength() {return length;}
       public void setLength(int length) { this.length = length;}
       public String toString() {
           String returnMe = "I am a Nap: ";
           returnMe += "\tlength= " + getLength();
           return returnMe;
       } // toString()
   } // Nap

20. (10 points) Complete this class named Test. Fill in main so it creates two Naps, named catNap
    and longNap, sets their lengths to 20 and 120, respectively, and then souts them each to the Out-
    put window.

   class Test {
       public static void main(String [] args) {
           Nap catNap = new Nap(20), longNap = new Nap(120);
           System.out.println(catNap + longNap);
       }
   }

21. (Extra credit 10 points) Write a complete DeceptiveNap class which is identical with Nap, except
    it pretends the person didn't sleep, by overriding the getLength() method so it returns half the
    length. DeceptiveNap is only allowed to have one method.

   class DeceptiveNap extends Nap {
       int getLength() {return length/2;}
   }