1. Answer each in complete sentences with proper grammar. Points will be given for neatness.

    a) In the context of object-oriented programming, what is a ***class***?

    _____

    _____

    b) In the context of object-oriented programming, what is a ***field***?

    _____

    _____

    c) In the context of object-oriented programming, what is a ***method***?

    _____

    _____

    d) Explain how the `private` modifier affects access to *fields*?

    _____

    _____

    _____

    e) Explain how the `private` modifier affects access to *methods*?

    _____

    _____

    f) What is the purpose of the ***constructor*** in Java?

    _____

    _____

    g) What is the purpose of a `toString` method in Java?

    _____

    _____

    h) What is the purpose of an ***accessor*** method in Java?

    _____

    _____

    i) What is the purpose of an ***mutator*** method in Java?

    _____

    _____

j) What is the benefit of a ***mutator*** method over making a field `public`?

_____

_____

_____

k) When reading the declaration of a field in Java, how are *class* fields distinguished from *instance* fields?

_____

_____

2. Consider the following class definition:

```
1  public class Vector {
2      public double x;
3      public double y;
4  }
```

a) How does Java treat classes that have no explicit constructor defined?

_____

_____

_____

b) Given the above `Vector` class, write a line of code that will create a new object, `v`, of type `Vector`.

_____

c) If a constructor for the `Vector` class were to be explicitly defined, and take two parameters, each parameter of type `double`, in order to initialize the fields `x` and `y`, the line of code written in part (b) would no longer work. Why?

_____

_____

_____

_____

d) If a two-parameter constructor were explicitly defined, as in part (c), but we still wish to be able to call a zero-parameter constructor, as in part (b), what is the solution?

_____

_____

_____