1. What's the difference between **member variables** (also called **instance variables**) and **class variables** (with the static keyword)? Which can be accessed without creating an instance of the class?

Instance variables are created when an object is created with the use of the keyword 'new' and destroyed when the object is destroyed.

Instance variables can be accessed directly by calling the variable name inside the class. However, within static methods (when instance variables are given accessibility), they should be called using the fully qualified name. *ObjectReference.VariableName*.

|  |
| --- |
| Static variables are created when the program starts and destroyed when the program stops. |
|  |

Static variables can be accessed by calling with the class name *ClassName.VariableName*.

1. Does it make sense to write **getter** and **setter** methods for a public member variable? What about private variables?

in a class, but outside a method, constructor or a block.

private data members are generally considered good because they provide encapsulation.

The variable firstName is declared as [public](https://www.codejava.net/java-core/the-java-language/public-keyword), so it can be accessed using the dot (.) operator directly, making the setter and getter useless. A workaround for this case is using more restricted access modifier such as [protected](https://www.codejava.net/java-core/the-java-language/protected-keyword) and [private](https://www.codejava.net/java-core/the-java-language/private-keyword):

* A a = new B();
* B b = (B) a;

Appliance myAppliance = new Refrigerator();

double temperature = (Refrigerator)myAppliance.getTemperature();