C# OOP: Concepts/Jargon Reference Sheet

Classes:
- are the building blocks of C# software applications.
- encapsulate data (stored in fields) and behavior (defined by methods).

Objects:
- are an instance of a class. We can create an object using the new operator.
- User bob = new User();

Constructors:
- are methods that are called when an instance of a class is created.
- should have the exact same name as the class.
- can be overloaded.*

Methods:
- provide functions which perform some kind of behavior.

  Method Signature:
  - consists of the number, type, & order of parameters.

  Method Overloading:
  - creating additional methods with the same name but with different signatures.

Fields:
- store data/values for a class.
- can be initialized either with a constructor or directly upon declaration.

Properties:
- can have different access levels (e.g. ‘public get’ and ‘private set’).
- Established via keywords “Get” and “Set”:
  - "GET" --> used to return a field value
  - "SET" --> used to assign a field value

  Descriptions attributed to property access levels:

  - read-and-write: has both a getter and a setter
  - read-only: has only a getter and no setter
  - write-only: has only a setter and no getter(rare)

Simple properties that require no custom access modification can be implemented as an auto-implemented property:

```
public int number { get; set; }
```

Access Modifiers:
C# has 6 access modifiers:
- public, private, protected, internal, protected internal, private protected
- the most important thing is to be comfortable with implementing ***PUBLIC*** and ***PRIVATE*** access modifiers.
- A class member declared with public is accessible *everywhere*.
- A class member declared with private is accessible *ONLY* from inside the class.

The traditional advice is that we want to declare fields as private and create public properties to provide access to them, however, this has more to do with the use of an application than a blanket principle that always must be adhered to.