nihhfkerer

- A **class** is a way to define a custom data type by bundling related data and methods into a single unit (like a blueprint)  
- **Abstraction** is the process of pulling related data and methods into a logical, reusable implementation, such as a class  
- Using classes, we can observe the attributes and behaviors of real-world objects, like forests, and model data and methods that represent them, respectively.  
- Think of a class as a blueprint from which multiple similar units, like Forests, can be created. Any given forest may have different values for various data points, but they’ll all have a name, a number of trees, and so on.

A microphone and cat head

Description automatically generated

**Defining Classes:**

- Defined used the class name and PascalCase naming convention

A screen shot of a computer code

Description automatically generated - The code for a class is usually put into a file named with the name of the class. In this case, the file name would be Gear.cs.

- **Instantiation** – The process of creating an instance of a class in the program, done using the ***new*** keyword

 A computer screen shot of text

Description automatically generated

- Here, we could say g is an instance, or an **object** of the Gear class, or g is of type Gear

**Fields:**

- A type of class member (component of a class) that are associated with different pieces of data in a class  
- Variables in a class  
A computer code with text

Description automatically generated with medium confidence - set to automatic defaults (null for string and 0 for int)

- Once created, they can be accessed and modified using bag notation

A screen shot of a computer code

Description automatically generated

**Methods:**

- Functions defined within a class that can be called in other parts of the program

A computer screen shot of a code

Description automatically generated A computer screen shot of white text

Description automatically generated

A screenshot of a computer code

Description automatically generated A computer code with text

Description automatically generated

**this:**

- Allows us to refer to the current instance of a class inside its methods  
- In most cases, including this one, this is not necessary and does not alter the behavior of the code. For this example, it only clarifies visually that area is an instance variable.

A computer screen shot of a code

Description automatically generated A computer code with text

Description automatically generated A computer code with text

Description automatically generated

**Constructors:**

- A **constructor**, that runs every time an object is created. Has the same name as the class it’s in  
- If no constructor is defined, C# provides a default **parameterless constructor**, allowing us to instantiate objects without errors:



- A constructor looks like a method but lacks a return type and has the same name as its class. Constructors can take parameters, just like other methods and we can also add code within the constructor body to set instance values:

A computer screen shot of a program code

Description automatically generated

- This is where this is necessary! When a value that is passed into the constructor will be assigned directly to an instance variable, it is common practice to give the parameter and the instance variable the same name.  
- In this example, within the body of the constructor, the name sides refers to the value that is passed as an argument when the constructor function is called, NOT the instance variable. We must use this along with the instance variable in order to assign the argument’s value to the correct place.  
- Assigning sides = sides instead would rewrite the value of the parameter sides with its own argument’s value, leaving the instance variable sides with its default value, which is 0 for an int.

A black background with white text and red and blue letters

Description automatically generated

**Overloading Constructors:**

- Just like other methods, constructors can be overloaded  
A computer code with text

Description automatically generated

- The first constructor sets values for both fields, and the second provides a default value for when the author is not specified. We can create Book instances in two ways:

A screenshot of a computer

Description automatically generated

- Remove duplicated code by putting it all in one

A computer screen shot of text

Description automatically generated

Can also use : this() to refer to another constructor of the same class.

A computer screen shot of text

Description automatically generated

**Completed Code:**

Program.cs

A screen shot of a computer program

Description automatically generated

Forest.cs

A computer screen shot of code

Description automatically generated A screen shot of a computer code

Description automatically generated