In this module we will be covering the differences in a procedural and object-oriented programming language. We will look at classes and instances in O-O.

Procedural and Object-oriented Programming

Procedural programming is a method of writing software. It centers on the actions that take place in a program - most early programming languages were procedural. Procedures (functions) operate on data items that are separate from the procedures.

Object-oriented programming is centered on objects. Objects are created from abstract data types that encapsulate data and functions together. An OBJECT is a software entty that contains both data and procedures wrapped together. Data that is contained in an object is known as the object's DATA ATTRIBUTES. Attributes are simply variables that reference data. The procedures that an object performs are known as METHODS. Methods are functions that perform operations on the object's data attributes. The object is a self-contained unit consisting of data attributes and methods.

Thus, code and data is not separated - this is called ENCAPSULATION. Data hiding refers to an object's ability to hide its data attributes from code that is outside the object. Only the object's Methods may directly access and make changes to the object's data attributes. An object typically hides its data, but allows outside code to access its Methods.

An object is not a stand alone program, but is used by programs that need its services.

You MUST define both the attributes and methods for an object.

Class

A class is code that specifies the data attributes and methods for a particular type of object. A class is a definition of the attributes and methods that the class will have. You need to create a class with a *class definition.* We use the PYTHON method __init__ as our initializer method because it initializes the object's data atrributes. This method will be the first one inside a class definition.

Instance

An instance of a class will be an object. When we want to hide our attributes from outside code, we need to "hide" or make private the attributes. We can do that by using the two _ _ characters at the beginning of the attribute name.

Storing Classes in Modules

We normally store our classes in modules so we can then reuse them as necessary. We simply import the classes into the programs as we need to.

The _ _ str_ _ Method

We might need to display a message that indicates an object's state. An object's "state" is simply the values of the object's attributes at any given moment.

Instance

Each instance of a class has its own set of data attributes.

Accessor and Mutator Methods

A method that returns a value from a class's attributes but does not change it is known as an *accessor method.* A method that stores a value in a data attribute or changes the value of a data attribute in some way is known as a *mutator method.*

Passing Objects as an Argument

We can pass objects to functions and methods as arguments.

Techniques for Designing Classes

When designing a class it is helpful to draw a UML diagram - Unified Modeling Language. A UML Diagram uses the format:

Class

Attributes

Methods

To help in identifying the classes you need for an application:

1. Define the problem domain (what is it you need to solve)

2. Identify all nouns in the description - these will be a potential class

3. refine list to those nouns (classes) relevant to the problem

What must a class know - or what data do we need to support this class? These are the attributes of the class.

What responsibilities does a class have? 1) what are the things that the class is responsible for knowing; 2) what are the actions that the class is responsible for doing? These are your Methods.