P5
Ryan Krage

Software structure

A software structure is a commonly used programming concept, usually with a lot of support from the language.

One of the most common software structures is the **function**, which a block of code that can be jumped to from anywhere else in the program.

The programmer defines the function, and writes the code it contains. Then, they call the function they defined at any time. This is useful when the program is not large/complex enough to warrant using object-orientation, but procedural programming is not advanced enough.

A **procedure** is very similar to a function, except functions can only output values to pass to other segments of code. A procedure can accept and inputs and outputs.

Classes are a software representation of a category of objects. An object is a software representation of something in the real world. For example, there might a class of cars, and the object wold be a specific instance of a car – perhaps a Ford. The Ford object inherits all of the properties of the Car class.

Objects are software representations of real things. They can pass values and be passed to, allowing objects to interact with each other, just like in the real world.

An **attribute** is a property of an object. Some are inherited from their class, and some are unique to the object.

A **method** is a defined way for objects to interact with other code. To protect objects, other objects and code cannot access them directly. Instead, they must use a method that belongs to the object. An analogy would be eating – you don’t put food directly into your stomach, instead you use your mouth to chew and swallow. The mouth is like a method of the stomach object.

All of these software structures help a developer to write clear, efficient code, and solve a problem in the best way possible.