**Week 1 Notes**

**Memory**

**Computer Memory**

For the purpose of this course, you may think of *computer memory* as a long list of storage locations where each location is identified with a unique number and each location houses a value. This unique number is called a *memory address*. Typically, we will write memory addresses as a number with an "id" as a prefix to distinguish them from other numbers (for example, id 201 is memory address 201).

Variables are a way to keep track of values stored in computer memory. A *variable* is a named location in computer memory. Python keeps variables in a separate list from values. A variable will contain a memory address, and that memory address contains the value. The variable then refers to the value. Python will pick the memory addresses for you.

**Terminology**

* A value has a memory address.  
  A variable contains a memory address.  
  A variable refers to a value.  
  A variable points to a value.

Example:

* Value 8.5 has memory address id x34.  
  Variable shoe\_size contains memory address id x34.  
  The value of shoe\_size is 8.5.  
  shoe\_size refers to value 8.5.  
  shoe\_size points to value 8.5.

**Variables**

**Assignment statements**

The general form of an assignment statement:

* variable = expression

Example assignment statements:

>>> base = 20

>>> height = 12

>>> area = base \* height / 2

>>> area

120.0

The rules for executing an assignment statement:

* Evaluate the expression. This produces a memory address.
* Store the memory address in the variable.

**Variable names**

The rules for legal Python names:

* Names must start with a letter or \_.
* Names must contain only letters, digits, and \_.

For Python, in most situations, the convention is to use pothole\_case.

**Built-in Functions**

**Function Call** – call the function and pass arguments to return a value

The general form of a function call:

* function\_name(arguments)

The rules for executing a function call:

* Evaluate the arguments.
* Call the function, passing in the argument values.

**Terminology:**

* Argument: a value given to a function
* Pass: to provide to a function
* Call: ask Python to evaluate a function
* Return: pass back a value

**Defining Functions**

**Function Definitions**

The general form of a function definition:

* def function\_name(parameters):
  + body
* def: a keyword indicating a function definition
* function\_name: the function name
* parameters:
  + the parameter(s) of the function, 0 or more and are separated by a comma
  + a parameter is a variable whose value will be supplied when the function is called
* body:
  + 1 or more statements, often ending with a return statement

Example of a function definition:

def f(x):

return x \*\* 2

return statement

The rules for executing a return statement:

* Evaluate the expression. This produces a memory address.
* Pass back that memory address to the caller. Exit the function.

The rules for executing a function call:

* Evaluate the arguments to produce memory addresses.
* Store those memory addresses in the corresponding parameters.
* Execute the body of the function.