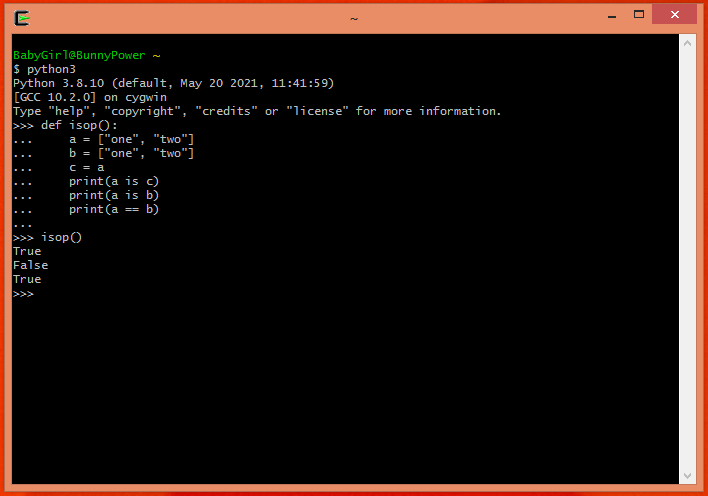
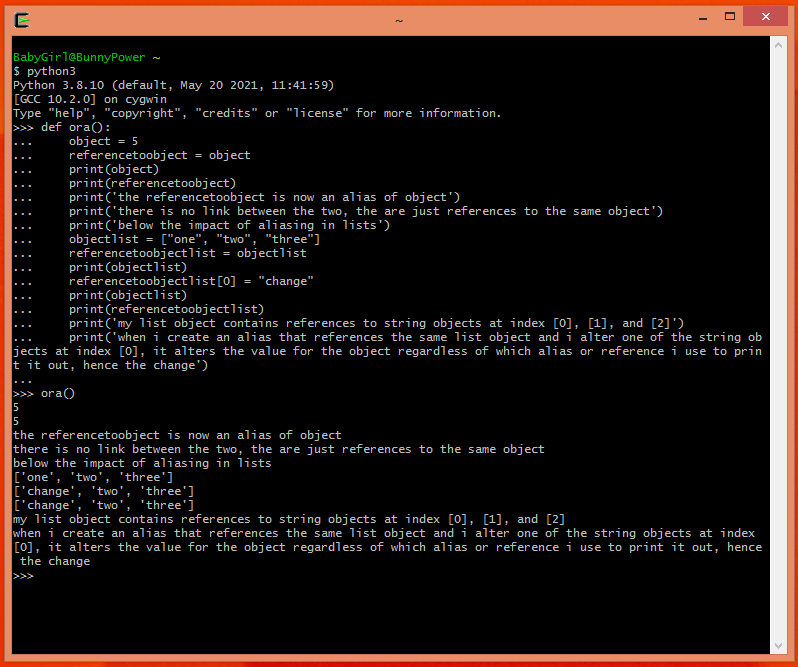
A python operation produces an object with a value associated with it as a result. For example, the python operation 2 x 2 produces a return object with a value equivalent to the value 4. Two objects can have the same value, meaning they are equivalent, but at the same time may not be identical as they remain different objects. For example, variable a and b may share the same value of 4, and are thus equivalent, but are not identical as they are still separate objects a and b. In the coding example below I created a python list that contains different objects with equivalent values and use the is and == operators. The is operator is an identity operator used to check if two values are located in the same part of the memory, and as shown by the results, just because two values are equivalent, does not mean they are identical. The first result returns True because a is the same object as c. The second result returns False because a is not the same object as b, they just have equivalent values. The third result returns True because a and b have the same value, and the == operator compares value while the is operator compares identity (i.e. if they are the same object).



In the screenshot below I added printout commentary to describe the differences between references, objects and aliases and their effect on one another:



In the screenshot below I created a function that takes a list as an argument and changes the value of the list object’s first string object using a reference to a string object I called object to overwrite the value. When I call the function I pass it an object called newlist as the parameter and the result is the first object in the list being changed from “hello” to “Hello”.

