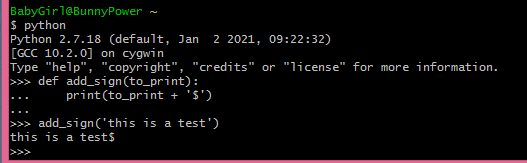
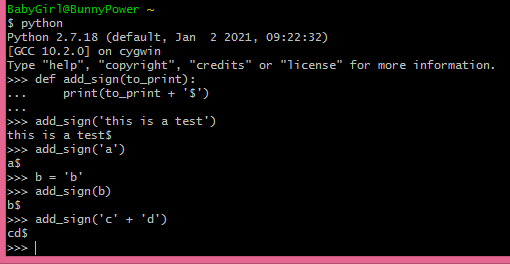
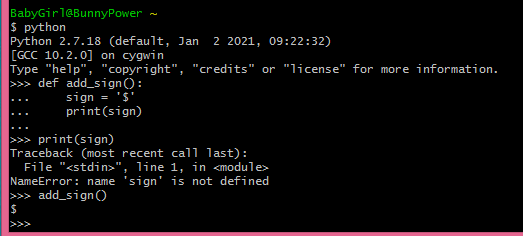
Example 1: In the screenshot below, I have defined a function called add\_sign that adds a dollar sign to the end of the input given. This function takes an argument called to\_print, which is a variable passed as a parameter when the function is called. The argument is the variable to\_print where it is defined in the function definition line “def add\_sign(to\_print)”. The parameter is the variable to\_print where it is assigned in the function call line “add\_sign(‘this is a test’)” where the value is assigned to the parameter and passed as input to the function for execution. 

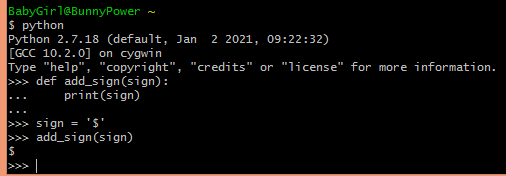
Example 2: In the screenshot below, I have called my function I created in example 1 with 3 different arguments as inputs. The first input passed (‘a’) is a value. The second input passed (b) is a variable assigned the value of ‘b’. The third input passed (‘c’ + ‘d’) is an expression resulting in the concatenation of the characters c and d.



Example 3: In the screenshot below, I created a function called add\_sign. Inside the function, I defined a variable called sign that is assigned a value of ‘$’. When calling the function, it will print out the variable. When calling the variable in a print statement outside the function python returns a NameError stating my variable is not defined. This is because variables are local and my variable is assigned inside my function. It is created when my function is called, but destroyed when my function call is completed which is why it won’t print outside of my function, but will print with the exact same statement when called inside my function.



Example 4: In the screenshot below, I have created a function called add\_sign that expects an argument called sign. Oddly, I could use the same name outside the function to assign a variable by the same name a value and pass that value to the function. I think this is because variables are local, so the variable sign created and called outside the function and the variable by the same name sign created and called inside the function do not overwrite one another but simply pass a value to and from one another.



Example 5: In the screenshot below, I have created two variables with the name n, one outside the function, and one inside the function add\_sign. I have assigned two different descriptive sentences to each variable n, and then printed out each variable. As in the previous example, I do not see any issue here, as each variable prints out the value assigned to it correctly, and I think this is as mentioned above because variables are local and the variables, even though they have the same name, never overwrite each other’s value.

