PYTHON

```python
# the '#' character makes a COMMENT separating Python from English
x = 3 # create the VARIABLE with NAME x and STORE INT VALUE 3
Sebastien_Score = 9001 # variable names can be long, but no spaces!
y = 1.0 * 3 # y stores EXPRESSION's RETURN FLOAT value 3.0
z = "Hi There!" # z stores a STRING value
w = False # w stores a BOOLEAN value
v = [3, 30, "Hello World"] # v stores a LIST of values
print(z) # print function displays output ("Hi There!")

# Maths
a = 3
b = 3.0 # b stores 3.0 (float values are decimal approximations)
c = 7 // 2 # c stores 3 (int division always gives ints)
d = 7 % 2 # d stores 1 (Mod or Remainder of the division)
a = 5 # change the value of a to 5
a += 1 # INCREMENT the value of a by 1 (to 6)

# Boolean Operators
a = (3 > 2) # a stores True because 3 is greater than 2
a = (2 >= 2) # a stores True because 2 is greater than or equal to 2
a = (3 < 2) # a stores False because 3 is not less than 2
a = (2 <= 2) # a stores True because 2 is less than or equal to 2
a = (3 != 2) # a stores True because 3 is not equal to 2
a = (3 == 3) # a stores True because 3 is equal to 3
a = (True and False) # a stores False, AND returns True only when both sides are True
a = (True or False) # a stores True, OR returns True if at least 1 side is True
a = (not False) # a stores True, NOT returns opposite

# BLOCKS are sections of any code chunked together with INDENTATION
# BLOCKS start with a ':' and continue with each INDENTED line
x = 7
if x > 8: # if CONDITION is True, then execute block, otherwise skip block.
    print("Hello") # since x stores 7, this will skip
    print("I Am Sam.") # since x stores 7, this will skip
elif x > 2: # elif condition is True AND previous if was False, execute block
    print("Hi") # since x stores 7, this will execute
    print("I am Sally.") # since x stores 7, this will execute
else: # if all previous conditions are False, executer block.
    print("Yo") # since x stores 7, this will skip
    print("I'm Bob.") # since x stores 7, this will skip

while x > 3: # repeat a block until condition becomes False
    print("Apples")
    x += -1

# Lists store multiple values
a = [10, 30, 20, 90] # create a new list
x = len(a) # x stores 4 (the length)
b = a[0] # INDEX into the list, 0 is first value, b stores 10
c = a[3] # c stores 90
d = a[-1] # -1 is last value, d stores 90
a[1] = 50 # modify the second element in the list, a is now [10, 50, 20, 90]
f = a + [5, 15] # f stores [10, 30, 20, 90, 5, 15], CONCATENATION not addition
g = range(0, 4) # range function returns list  0 up to 4, g stores [0, 1, 2, 3]

# For Loops
for c in "Elephant!": # repeat block with c storing each character 1 at a time
    print(c) # prints one letter per line

for x in [10, 30, 20]:
    print(x) # prints one number per line

# Custom Functions
def myfunc(a, b): # DEFINES a new function that takes 2 INPUT PARAMETER values
    c = 2 * a + b # executes only when function is called
    return c # RETURNS a value back to the calling code

x = myfunc(10, 5) # Calls the myfunc() function, x stores return value 25
y = myfunc(1, 3)  # Calls the myfunc() function, x stores return value 5
```