**Python Library**

**Trinket Code**

* #concatenating strings
* house\_number = "4060"
* street = "W Wildcat Blvd"
* city = "Charleston"
* state = "SC"
* zip = "29414"
* full\_address = house\_number + " " + street + " " + city + ", " + state + " " + zip
* print(full\_address)
* print("Welcome to " + city + "!")
* #some basic math
* total = 10
* discount = 0.20
* total = total - discount
* total -= discount #exact same result as previous line just shorthand
* g = 9 % 4 #stores the remainder only of 9/4
* #conditional operators (if goes first, elif for middle conditions, else for last)
* household\_income = 110000
* if household\_income > 200000:
* print("good for you")
* elif household\_income >= 100000:
* print("not too shabby")
* elif household\_income < 100000 and household\_income >= 50000:
* print("eh")
* else:
* print("sucks for you")
* #nested conditions
* letter\_a = "a"
* letter\_b = "b"
* letter\_c = "c"
* number\_1 = 1
* number\_2 = 2
* number\_3 = 3
* if letter\_a == "a":
* if letter\_b == "c":
* print("b equals c?")
* elif number\_3 == 2:
* print("3 equals 2?")
* elif number\_2 == 2:
* if number\_3 == 3:
* if number\_1 != 2:
* print("a = a, 2 = 2, 3 = 3, and 1 doesn't equal 2")
* else:
* print("What a crazy world")
* '''
* btw enclosing multiple lines with three single quotation marks makes for long
* comments. then you can type a bunch.
* '''
* print("It ignored those comments!") #it ignored this comment too btw
* #lists
* band = ["singer", "guitar", "bass", "drums"]
* print("Original Band Listed Below")
* print(band)
* print("Chuck plays the " + band[1])
* print("Chuck owns two " + band[1] + "s")
* nums = [1, 2, 3, 4, 5]
* print(nums[4])
* band.append("keyboard")
* print(band[4])
* nums.insert(0, 0)
* nums.insert(0, -1)
* nums.insert(3, 1.5)
* print(nums)
* new\_band = band + ["rhythm guitar", "backup singers"]
* print("New Band Listed Below")
* print(new\_band)
* mids = new\_band[3:5] # see comment below for syntax
* print("Mids Listed Below")
* print(mids)
* '''
* btw those are single quotation marks, not periods to separate this text.
* for mids, I added the 4th through the 5th element. the 3 represents
* element 3 (0, 1, 2, 3). the 5 represents the element after the last one I need.
* '''
* brand\_new\_band = new\_band[:5] # this copies the 0, 1, 2, 3, & 4 entries
* print("Brand New Band Listed Below")
* print(brand\_new\_band)
* duet = new\_band[:2]
* print("Duet Listed Below")
* print(duet)
* new\_guys = new\_band[5:]
* print("New Guys Listed Below")
* print(new\_guys)
* print("I wish I could buy another " + brand\_new\_band[1] + " and maybe even a " + brand\_new\_band[4] + "!")
* print("We had to fire the backup singers and the rhythm guitarist from the new band")
* new\_band.remove("rhythm guitar") # command to remove list item by value
* del new\_band[5] # command to remove the 5th list element (remember to start w/ 0)
* print(new\_band)
* # popping and moving list elements to other lists
* new\_band.append("rhythm guitar")
* new\_band.append("backup singers")
* print("Then we rehired the rhythm guitar and the backup singers")
* print("But then they quit and formed the revenge band below")
* quitter1 = new\_band.pop(5) # removes the 6th entry and adds to a variable
* revenge\_band = [] # to append to this list, I had to create it first
* revenge\_band.append(quitter1) # added quitter1 to my list
* revenge\_band.append(new\_band.pop()) # removes the last entry and adds to my list
* print(revenge\_band)
* # tuples are lists that cannot be modified in any way, except to redefine completely
* # a tuple differs from a list in coding by () rather than []
* # tuples can include numbers or strings
* tuple\_example = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
* list\_example = [0, 0.5, 1.5, 2.5]
* list\_example.insert(2, tuple\_example[0])
* list\_example.insert(4, tuple\_example[1])
* print(list\_example)
* s = tuple\_example[9] + list\_example[4]
* print(s)
* # the print code below didn't work when I tried to use tuple\_example (str and int)
* one\_with\_strings = ("1", "2", "3", "4", "5")
* print("My cat is almost " + one\_with\_strings[4] + " years old")
* print("My cat is almost " + str(tuple\_example[4]))
* # use str, int, or float to convert (float is a decimal number)
* # for loops
* best\_dive\_computers = ["Perdix", "Perdix AI", "Eon Steel", "Eon Core"]
* #one\_I\_want = input("Which dive computer do you want?")
* one\_I\_want = "Perdix AI" # could be assigned by data aquisition
* for temp\_variable in best\_dive\_computers: # temp\_variable holds each list value
* if one\_I\_want == temp\_variable: # can use any operators here
* print("You want one of the best dive computers")
* for temp\_variable in best\_dive\_computers:
* print(temp\_variable)
* #one\_he\_wants = input("What dive computer does he want?")
* one\_he\_wants = "Suunto D6i"
* for temp2\_variable in best\_dive\_computers:
* if one\_he\_wants == temp2\_variable:
* print("He wants one of the best, too")
* elif not one\_he\_wants in best\_dive\_computers: #works best to check if something not in list
* print("He should consider checking out other computers")
* break # necessary to put a break here
* # no break on previous line results in 4 repetitions of the print line
* # that's because there are 4 list items to check
* # if checking if elements are or aren't in a list, this isn't the best way
* '''
* best way to check if elements are or are not in a list
* if value\_to\_check in list\_name:
* do so and so
* if value\_to\_check not in list\_name:
* do so and so
* '''
* # nested for loops below will loop for every element in both lists/tuples
* guitars = ["Les Paul", "SG", "Stratocaster", "Telecaster", "Black Panther"]
* amps = ["Marshall", "Fender", "Mesa Boogie", "Line 6"]
* guitar\_amp\_combos = [] # create your empty list first (if needed)
* for a\_guitar in guitars:
* for an\_amp in amps:
* guitar\_amp\_combos.append(a\_guitar + " " + an\_amp) # new list will all permutations
* print(guitar\_amp\_combos) # indenting this line will print the list after each append
* # input commands need a variable
* #email\_address = input("What is your email address?")
* #print(email\_address)
* # input info is a string by default. use int (integer) or float (decimal) to convert
* # use str to convert back to string
* '''
* birth\_month = input("What month were you born?")
* birth\_day = input("What day of the month were you born?")
* if birth\_month == "March" and int(birth\_day) >= 21: # and statements here
* print("Your sign is Aries")
* if birth\_month == "April" and int(birth\_day) <= 19: # input needs converting
* print("Your sign is Aries")
* if birth\_month == "April": # nested if statements here
* if int(birth\_day) >= 20:
* print("Your sign is Taurus")
* if birth\_month == "May":
* if int(birth\_day) <= 20:
* print("Your sign is Taurus")
* if (birth\_month == "July" and int(birth\_day) >= 23) or (birth\_month == "August"
* and int(birth\_day) <= 22):
* print("Your sign is Leo") # Leo code split over two lines using or statements
* '''
* #dealing with cases
* best\_guitar\_brands = ["gibson", "fender", "gretsch"]
* print(best\_guitar\_brands)
* fav\_guitar = input("What is your favorite guitar brand?")
* fav2\_guitar = fav\_guitar.lower() #converts their input to all lower case
* print(fav2\_guitar)
* #tried using a for loop to check each list item against input element.
* #this did not receive desired effect. code below works best for checking lists for elements.
* if fav2\_guitar in best\_guitar\_brands:
* print(fav2\_guitar.title() + " is one of the best guitar brands!")
* elif not fav2\_guitar in best\_guitar\_brands:
* print(fav\_guitar.title() + " is not one of the best guitar brands.")
* #dictionaries: create an index where each "item" consists of a key and a value
* scuba\_gear = {"regulator": "scuba pro mk25/s600", "bc": "scuba pro knighthawk", "computer": "perdix ai"}
* print(scuba\_gear)
* # keys or values can be strings or numbers