***Intro to Python Overview of Code***

***Section 1:***

- Inputs, If else, Sums, Print()

total\_msrp = 0

total\_cost = 0

total\_potential\_gross\_profit = 0

total\_adjusted\_gross\_profit = 0

\*\* We are creating a value for these amount while using an input\*\*

print("Enter details for car #1:")

car\_name1 = input("Car name 1 (e.g., Red 1969 Triumph TR6): ")

msrp1 = int(input("MSRP: "))

cost1 = int(input("Cost: "))

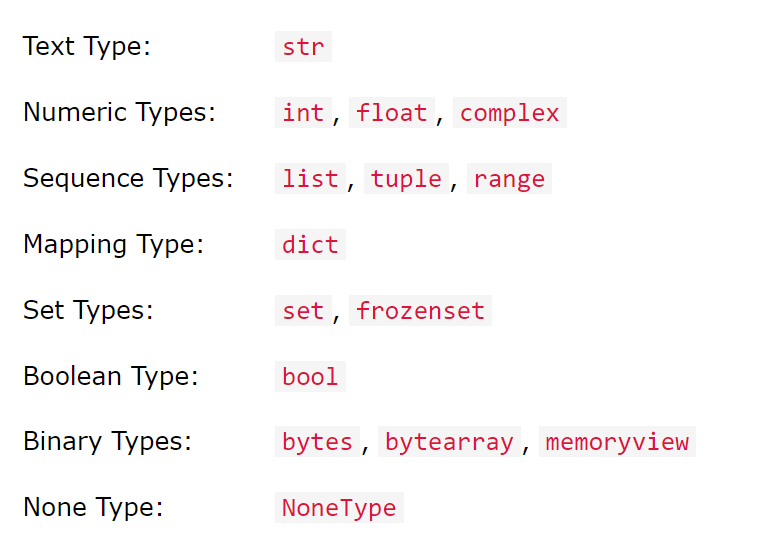
print("Type (0=domestic, 1=import): ")

car\_type\_code1 = int(input())

\*\* Creating an placeholder for the input

- creating an input()

- We are then stating it is an integer to create a number\*\*



#Determine car type

car\_type1 = "domestic" if car\_type\_code1 == 0 else "import"

\*\* This involves identifying the to use domestic or import using and if else statement\*\*

#Calculate gross profit and adjusted gross profit

gross\_profit1 = msrp1 - cost1

adjusted\_gross\_profit1 = gross\_profit1 \* (0.9825 if car\_type\_code1 == 1 else 1)

\*\* Creating new variables that will calculate the information from the inputs that were collected\*\*

#Totals for all cars

total\_cost = sum([cost1, cost2, cost3, cost4, cost5])

total\_msrp = sum([msrp1, msrp2, msrp3, msrp4, msrp5])

total\_potential\_gross\_profit = sum([gross\_profit1, gross\_profit2, gross\_profit3, gross\_profit4, gross\_profit5])

total\_adjusted\_gross\_profit = sum([adjusted\_gross\_profit1, adjusted\_gross\_profit2, adjusted\_gross\_profit3, adjusted\_gross\_profit4, adjusted\_gross\_profit5])

\*\* This was to create a sum at the end without using a loop\*\*

#Totals Car 1

print("Details for Car 1:")

print("Car Name:", car\_name1)

print("MSRP:", msrp1)

print("Cost:", cost1)

print("Potential Gross Profit:", gross\_profit1)

print("Adjusted Gross Profit:", adjusted\_gross\_profit1)

#Total

print("Totals for the shipment:")

print("Total MSRP:", total\_msrp)

print("Total Cost:", total\_cost)

print("Total Potential Gross Profit:", total\_potential\_gross\_profit)

print("Total Adjusted Gross Profit:", total\_adjusted\_gross\_profit)

\*\* Showcasing the print from car one. This includes the key and the information from the input that was collected\*\*

**Section 2**

Functions, If else within a function, While loop, float, and f string

#Called function to determine user's monthly payment based on credit score

def monthly\_payment\_percentage (passedCreditScore, passedMonthlyGrossIncome):

if int(passedCreditScore) < 530:

#15% of gross monthly income

monthly\_payment = float(passedMonthlyGrossIncome) \* float(0.15)

elif int(passedCreditScore) < 590:

#15% of gross monthly income

monthly\_payment = float(passedMonthlyGrossIncome) \* float(0.15)

elif int(passedCreditScore) < 640:

#17% of gross monthly income

monthly\_payment = float(passedMonthlyGrossIncome) \* float(0.17)

elif int(passedCreditScore) < 721:

#20% of gross monthly income

monthly\_payment = float(passedMonthlyGrossIncome) \* float(0.20)

else:

#30% of gross monthly income

monthly\_payment = float(passedMonthlyGrossIncome) \* float(0.30)

return str(monthly\_payment)

\*\* The function name is Monthly\_payment\_percentage

- (passedCreditScore, passedMonthlyGrossIncome) = parameters used to create the function

- if int(passedCreditScore) < 530 = stored as int for passedcreditscore

- monthly\_payment = float(passedMonthlyGrossIncome) \* float(0.20) =

*monthly\_payment:* This is a variable where the calculated monthly payment amount is stored.

*float(passedMonthlyGrossIncome):* This converts the passedMonthlyGrossIncome parameter to a floating-point number. It's assumed that passedMonthlyGrossIncome is either a string that represents a number or an integer that needs to be converted to a float for decimal accuracy in calculations.

*float(0.30):* This represents the percentage (30%) as a floating-point number. Multiplying the gross monthly income by 0.30 calculates 30% of the income.

## *The float()* function in Python is used to convert a given value into a floating-point number. A floating-point number is a way to represent real numbers within a certain degree of precision that can include a fractional part, such as 3.14, -0.001, or 2.0. This function is versatile and can handle various types of input:

## *numbers that have a fractional component. float can handle decimal points (e.g., 3.14, -0.001), while int is limited to whole numbers (e.g., 3, -1).\*\*\**

##return str(monthly\_payment)

\*\* str() Function: The str() function in Python converts the argument passed into it into a string type. When str() is applied to a floating-point number, it creates a textual representation of that number. For example, if monthly\_payment were 123.45, str(monthly\_payment) would convert it to the string "123.45".

return: This keyword is used to exit the function, sending back a value to the place where the function was called. In this case, the function returns the string representation of monthly\_payment.\*\*\*