**COMPETENCY 2: The graduate constructs functions and control structures (if/elif/else, for, while) to interact with data structures (lists, sets, dictionaries, tupes) and direct program flow.**

*# create a function with receives two integers as input, adds them and returns the sum*

def sum(a, b):

return a + b

*# run your function and print the result with integers 7 and 9*

*# expected outcome: 16*

print(sum(7, 9))

*# run your function and print the result with integers 20 and 49*

*# expected outcome: 69*

print(sum(20, 49))

*# Run your function with integers 2 and 8, and save the output to a new variable called myNewSum. Print myNewSum.*

*# expected outcome: 10*

myNewSum = sum(2, 8)

print(myNewSum)

*# You are provided a student's score on the recent exam.*

*# Create a function that will print a reply based on the score.*

*# Students who score 90 points or more receive an A and pass the course.*

*# Students receiving 80 points or more receive a B and pass the course.*

*# Students receiving 79 points or less do not pass and need to retake the exam.*

*# Students receiving a score of 0 have not attempted the exam and need instructions to schedule.*

def student\_reply(score = 0):

if score == 0:

return 'You have not attempted the exam and need instructions to schedule.'

elif score <= 79:

return 'You have not passed and need retake the exam.'

elif (score >= 80) and (score <= 89):

return 'You received a B and pass the course.'

elif score >= 90:

return 'You received an A and pass the course.'

*# Run the function with a score of 90 and print the result*

*# expected outcome: You received an A and have passed the course.*

print(grade\_report(90))

*# Run the function with a score of 70 and print the result*

*# expected outcome: You have not passed the course. Please retake the exam.*

print(grade\_report(70))

*# use the range method to print out numbers 6-12*

for i in range(6, 12):

print(i, end=" ")

*# create a list containing the names Dana, Cemal, Jessica, Mike, and Dana*

names\_list = ['Dana', 'Cemal', 'Jessica', 'Mike', 'Dana']

*# Print the length of the list.*

*# expected outcome: 5*

Print(len(names\_list))

*# Check to see if David is in the list. If not in the list, add her and print the list.*

*# expected outcome: ['Dana', 'Cemal', 'Jessica', 'Mike', 'Dana', 'David']*

if names\_list.count('David') == 0:

names\_list.append('David')

print(names\_list)

*# Print a single string containing all of the names separated by commas*

*# expected outcome: Dana, Cemal, Jessica, Mike, Dana, David*

print(', '.join(names\_list))

*# Print only the names Dana and Mike from myNames*

*# expected outcome: ["Mike","Dana"]*

print(names\_list[3:5])

*# ensure that each name is only listed once and print the list of unique values*

*# expected outcome: ['Dana', 'Cemal', 'Jessica', 'Mike', 'David'] \*note: order of items in list may vary*

names\_list\_unique = list(set(names\_list))

print(names\_list\_unique)

*# create an individual message for each unique name and welcome them to WGU*

*# expected outcome: Welcome to WGU, Dana*

*# Welcome to WGU, Jessica*

*# Welcome to WGU, Mike*

*# Welcome to WGU, David*

*# Welcome to WGU, Cemal*

for student in names\_list:

print('Welcome to WGU, {}'.format(student))

*# given the following dictionary of employees and salaries, create an personalized salary message letting each employee know they have been given a 2% raise and the new total of their salary.*

*# expected outcome: John, your current salary is 54000.00. You received a 2% raise. This makes your new salary 55080.0*

*# Judy, your current salary is 71000.00. You received a 2% raise. This makes your new salary 72420.0*

*# Albert, your current salary is 38000.00. You received a 2% raise. This makes your new salary 38760.0*

*# Alfonzo, your current salary is 42000.00. You received a 2% raise. This makes your new salary 42840.0*

employeeDatabase = {

**'John'**: 54000.00,

**'Judy'**: 71000.00,

**'Albert'**: 38000.00,

**'Alfonzo'**: 42000.00

}

for emp, sal in employeeDatabase.items():

newSal = (sal \* .02) + sal

message = '{}, your current salary is {}. You received a 2% raise. This makes your new salary {}'.format(emp, sal, newSal)

print(message)

*# starting with year 2000, create a list containing 5 leap years*

*# when the list is complete, print the full list with a message*

*# expected outcome: These are the leap years: [2000, 2004, 2008, 2012, 2016]*

leap\_years = []

i = 2000

while len(leap\_years) < 5:

leap\_years.append(i)

i += 4

print(leap\_years)

*# A nurse is monitoring a patient's rising temperature. The temp is rising in increments of .5 degrees continually.*

*# The nurse needs to be shown a message when the temp reaches 104 and the monitoring should end at that time.*

*# expected outcome: The temp has reached 104.0*

temp = 99.5

while temp <= 104.0:

# extra practice

#print('The temp has now reached {}'.format(temp))

if temp == 104.0:

print('The temp has reached 104.0')

temp += .5

*# create a tuple to store the WGU phone number 877-435-7948. Store the phone number as three groups of numbers without the hyphens.*

phone = ("877", "435", "7948")

print(phone)

*# use the tuple to print the last four digits of the phone number*

*# expected outcome: 7948*

print(str().join(phone[-1:]))

*# use the tuple to print the entire phone number with the message to Call WGU now*

*# expected outcome: Call WGU now at 877-435-7948*

print("Call WGU now at " + str('-').join(phone))

*#Finish the fruitFunction to take as input a list of fruits and return a string value containing the first two fruits from the list*

**def** fruitFunction(fruits):

return fruits[:2]

print(fruitFunction([**'banana'**,**'apple'**,**'orange'**,**'grapes'**,**'pineapple'**])) *#expected outcome: banana apple* print(fruitFunction([**'mango'**,**'avocado'**,**'pear'**])) *#expected outcome: mango avocado*

*#Finish the fruitFunction2 to take as input a list of fruits and return a string value letting us know if the avocado is in the list or not. If so, state that the avocado is in the list. If not, state avocado not found.*

**def** fruitFunction2(fruits):

if 'avocado' in fruits:

return 'avocado not found'

else:

return 'avocado is in the list'

print(fruitFunction2([**'banana'**,**'apple'**,**'orange'**,**'grapes'**,**'pineapple'**])) *#expected outcome: avocado not found*

print(fruitFunction2([**'mango'**,**'avocado'**,**'pear'**])) *#expected outcome: avocado is in the list*

*#Finish the favFoods function that takes as input a list of foods and returns a count of the number of times pizza is included in the list of favorite foods*

**def** favFoods(x):

x = [item.lower() for item in x]

for i in x:

return x.count('pizza')

print(favFoods([**'apple'**,**'banana'**,**'pizza'**,**'Pizza'**,**'ice cream'**,**'cupcakes'**])) *#expected output: 2*

print(favFoods([**'almonds'**,**'spaghetti'**,**'pizza'**,**'kombucha'**,**'Pizza'**,**'pizza'**])) *#expected output: 3*

*#Completed the makeList function that takes as input a string value of names and returns each name as an individual item in a list*

**def** makeList(names):

namesList = names.split()

return namesList

print(makeList(**'Jessica John Paul Grace Ginger Billy Arlene'**)) *#expected output: ['Jessica', 'John', 'Paul', 'Grace', 'Ginger', 'Billy', 'Arlene']*

print(makeList(**'David Cemal Dana Rodger Jerry Jessica Mike'**)) *#expected output: ['David', 'Cemal', 'Dana', 'Rodger', 'Jerry', 'Jessica', 'Mike']*

*#Complete costCount that takes one argument as a list of expenses and returns the total cost of all purchases*

**def** costCount(purchases):

total = sum(purchases)

print(total)

print(costCount([39.90, 40.21, 8.73, 9.57])) *#expected output: 98.41*

print(costCount([139.90, 10.11, 1.53, 3.57])) *#expected output: 155.10999999999999*