In [1]:
```
# PYTHON DICTIONARIES
```

In [ ]:
```
# Scenario 1: We want to store information about student names in a class and their

# Scenariuo 2: holding iventory of products in a store and their quantity or price
```

In [5]:
```
chemStuds = ['Florence', 'Onyeagoro','Aina','Elizabeth','Okafor', 'Thelma', 'Blessi
chemScores = [87,89,95,67,90,45,35, 83]
```

In [ ]:
```
# Challenge:Given the name Aina, find his exam score
```

In [7]:
```
# Step 1: Find the index position of the name in the chemStuds list

idxPos = chemStuds.index('Aina')

print(idxPos)

# Step 2: get Aina's score from the chemScores list using the index position

score = chemScores[idxPos]

print(score)
```

```
2
95
```

In [ ]:
```
# Dictionaries


1. Dictioanries are a container type that hold items in pairs of Kes: Values
2. Dictionaries do not allow duplicate keys. There can be duplicate values
3. Keys must immutable
4. Dictionaries are denoted by opening an d closing curly brace
5. Each KEY: VALUE pair must be separated from the next by a comma
```

In [ ]:
```
# SYNTAX of a dictionay


myDict = {KEY1:VALUE1, KEY2:VALUE2, KEY3:VALUE3,,,,,,,,KEYn:VALUEn}
```

In [ ]:
```
# Example of a dictionar

Tunde has thre bank accounts in ABC Bank. savings with 5900, current acct with 6000

Put in a dictionary
```

In [9]:
```
tundeAcct = {'savings':5900, 'current':6000, 'investment':400}
```

In [ ]:
```
acctType = ['savings', 'current', 'investments']
bals = [5900, 6000, 400]
```

In [10]:
```
# How to get a vlue from a disctionary when we have the key


tundeAcct['current']
```

Out[10]: 6000

In [11]:
```
bal = tundeAcct['current']
```

In [12]:
```
bal
```

Out[12]: 6000

In [13]:
```
# Fetch all the keys in a dictionary


tundeAcct.keys()
```

Out[13]: dict_keys(['savings', 'current', 'investment'])

In [14]:
```
# Convert to a list

list(tundeAcct.keys())
```

Out[14]: ['savings', 'current', 'investment']

In [15]:
```
# fetch all the values in a dictionry

tundeAcct.values()
```

Out[15]: dict_values([5900, 6000, 400])

In [16]:
```
# Convert to a list

list(tundeAcct.values())
```

Out[16]: [5900, 6000, 400]

In [17]:
```
# Add another type of account and balance to the dictionary

'FixedDep'

tundeAcct['FixedDep'] = 7600
```

In [18]:
```
tundeAcct
```

Out[18]: {'savings': 5900, 'current': 6000, 'investment': 400, 'FixedDep': 7600}

In [19]:
```
chemStuds = ['Florence', 'Onyeagoro','Aina','Elizabeth','Okafor', 'Thelma', 'Blessi
chemScores = [87,89,95,67,90,45,35, 83]
```

In [20]:
```
# For Loop Tutoria


# For loops enable us to go through the content of any container and fetch each ite
```

In [21]:
```
acctType = ['savings', 'current', 'investments']

for acct in acctType:
    print(acct)
```

```
savings
current
investments
```

In [22]:
```
chemStuds = ['Florence', 'Onyeagoro','Aina','Elizabeth','Okafor', 'Thelma', 'Blessi
chemScores = [87,89,95,67,90,45,35, 83]


for name, score in zip(chemStuds,chemScores ):
    print(name, score)
```

```
Florence 87
Onyeagoro 89
Aina 95
Elizabeth 67
Okafor 90
Thelma 45
Blessing 35
Goodness 83
```

In [25]:
```
chemStuds = ['Florence', 'Onyeagoro','Aina','Elizabeth','Okafor', 'Thelma', 'Blessi
chemScores = [87,89,95,67,90,45,35, 83]

n = 0
for name, score in zip(chemStuds,chemScores ):
    print('########################################')
    print('entering the containers')
    print('current index position being fetched is: ', n)

    print( "fetched ", name, score , ' at index position ', n)

    print(name, score)
    print('########################################')
    n = n + 1
```

```
########################################
entering the containers
current index position being fetched is:  0
fetched  Florence 87  at index position  0
Florence 87
########################################
########################################
entering the containers
current index position being fetched is:  1
fetched  Onyeagoro 89  at index position  1
Onyeagoro 89
########################################
########################################
entering the containers
current index position being fetched is:  2
fetched  Aina 95  at index position  2
Aina 95
########################################
########################################
entering the containers
current index position being fetched is:  3
fetched  Elizabeth 67  at index position  3
Elizabeth 67
########################################
########################################
entering the containers
current index position being fetched is:  4
fetched  Okafor 90  at index position  4
Okafor 90
########################################
########################################
entering the containers
current index position being fetched is:  5
fetched  Thelma 45  at index position  5
Thelma 45
########################################
########################################
entering the containers
current index position being fetched is:  6
fetched  Blessing 35  at index position  6
Blessing 35
########################################
########################################
entering the containers
current index position being fetched is:  7
fetched  Goodness 83  at index position  7
Goodness 83
########################################
```

In [37]:
```
# construct the dictionary

chemStuds = ['Florence', 'Onyeagoro','Aina','Elizabeth','Okafor', 'Thelma', 'Blessi
chemScores = [87,89,95,67,90,45,35, 83]

# create empty dictionary
```

```
examDict = dict()

for name, score in zip(chemStuds,chemScores ):
    # print(name, score)
    print(name, score)
    examDict[name]  = score
    print(examDict)


#print(examDict)
```

```
Florence 87
{'Florence': 87}
Onyeagoro 89
{'Florence': 87, 'Onyeagoro': 89}
Aina 95
{'Florence': 87, 'Onyeagoro': 89, 'Aina': 95}
Elizabeth 67
{'Florence': 87, 'Onyeagoro': 89, 'Aina': 95, 'Elizabeth': 67}
Okafor 90
{'Florence': 87, 'Onyeagoro': 89, 'Aina': 95, 'Elizabeth': 67, 'Okafor': 90}
Thelma 45
{'Florence': 87, 'Onyeagoro': 89, 'Aina': 95, 'Elizabeth': 67, 'Okafor': 90, 'Thelm
a': 45}
Blessing 35
{'Florence': 87, 'Onyeagoro': 89, 'Aina': 95, 'Elizabeth': 67, 'Okafor': 90, 'Thelm
a': 45, 'Blessing': 35}
Goodness 83
{'Florence': 87, 'Onyeagoro': 89, 'Aina': 95, 'Elizabeth': 67, 'Okafor': 90, 'Thelm
a': 45, 'Blessing': 35, 'Goodness': 83}
Aina 50
{'Florence': 87, 'Onyeagoro': 89, 'Aina': 50, 'Elizabeth': 67, 'Okafor': 90, 'Thelm
a': 45, 'Blessing': 35, 'Goodness': 83}
```

In [28]:
```
chemStuds = ['Florence', 'Onyeagoro','Aina','Elizabeth','Okafor', 'Thelma', 'Blessi

for name in chemStuds:
    print(name)
```

```
Florence
Onyeagoro
Aina
Elizabeth
Okafor
Thelma
Blessing
Goodness
```

In [ ]:
```
# NEVER use sets for Dictionary as below!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

 # construct the dictionary

chemStuds = { 'Florence', 'Onyeagoro','Aina','Elizabeth','Okafor', 'Thelma', 'Bless
chemScores = { 87,89,95,67,90,45,35, 83}

# create empty dictionary
```

```
examDict = dict()

for name, score in zip(list(chemStuds),list(chemScores) ):
    # print(name, score)
    print(name, score)
    examDict[name]  = score
    print(examDict)


#print(examDict)
```

In [40]:
```
newDict = {'Florence': 87, 'Onyeagoro': 89, 'Aina': 50,
           'Elizabeth': 67, 'Okafor': 90, 'Thelma': 45,
           'Blessing': 35, 'Goodness': 83}
```

In [41]:
```
# unpack a dictionary to get each key and its corresponding value

for name, score in newDict.items():
    print(name, score)
```

```
Florence 87
Onyeagoro 89
Aina 50
Elizabeth 67
Okafor 90
Thelma 45
Blessing 35
Goodness 83
```

In [42]:
```
newDict.items()
```

Out[42]:
```
dict_items([('Florence', 87), ('Onyeagoro', 89), ('Aina', 50), ('Elizabeth', 67),
('Okafor', 90), ('Thelma', 45), ('Blessing', 35), ('Goodness', 83)])
```

In [43]:
```
x , y = ('Florence', 87)
```

In [44]:
```
x
```

Out[44]:
```
'Florence'
```

In [45]:
```
y
```

Out[45]:
```
87
```

In [ ]:
```
# Python Functions

Functions are a specialized block of code that is designed to do one thing only
```

In [47]:
```
# Function Syntax

def currencyConverter(amt):
    value = amt * 11.97

    print(value)
```

In [48]:
```
currencyConverter(2000)
```

```
23940.0
```

In [ ]:
```
# things we should know about functions

1. functions can receieve valies as inputs or they also not be designed to recieve

2. function can return a value to its caller or not
3. If a function returns a value, you must create a new variable to recieve the val
```

In [49]:
```
# Function Syntax

def currencyConverter1(amt):
    value = amt * 11.97

    return value
```

In [51]:
```
res = currencyConverter1(2000)
```

In [52]:
```
print(res)
```

```
23940.0
```

In [ ]:
```

```

In [ ]:
```
# Challenge:

#  tunde wants a function that will return the grade of sa student when the functio
```

In [53]:
```
chemStuds = ['Florence', 'Onyeagoro','Aina','Elizabeth','Okafor', 'Thelma', 'Blessi
chemScores = [87,89,95,67,90,45,35, 83]
```

In [54]:
```
def scoreGrader(score, name, subject):

    if score >=90:
        grade = 'A'
    elif score >=80:
        grade = 'B'
    elif score >= 70:
        grade = 'C'
    elif score >= 60:
        grade = 'D'
    elif score >= 50:
        grade = 'E'
    else:
        grade = 'F'

    return name, subject, score, grade
```

In [58]:
```
score = 30
name = 'Akintola'
```

```
subject = 'Chem101'

res = scoreGrader(score, name, subject)
print(res)
```

```
('Akintola', 'Chem101', 30, 'F')
```

In [61]:
```
import pandas as pd

chemStuds = ['Florence', 'Onyeagoro','Aina','Elizabeth','Okafor', 'Thelma', 'Blessi
chemScores = [87,89,95,67,90,45,35, 83]
subj = 'Chem101'

resList = []
for name, score in zip(chemStuds, chemScores):
    res = scoreGrader(score, name, subj)
    resList.append(res)
resDF = pd.DataFrame(resList, columns =['name', 'subject', 'score', 'Grade'])
```

In [60]:
```
resList
```

Out[60]:
```
[('Florence', 'Chem101', 87, 'B'),
 ('Onyeagoro', 'Chem101', 89, 'B'),
 ('Aina', 'Chem101', 95, 'A'),
 ('Elizabeth', 'Chem101', 67, 'D'),
 ('Okafor', 'Chem101', 90, 'A'),
 ('Thelma', 'Chem101', 45, 'F'),
 ('Blessing', 'Chem101', 35, 'F'),
 ('Goodness', 'Chem101', 83, 'B')]
```

In [62]:
```
resDF
```

Out[62]:

	name	subject	score	Grade
0	Florence	Chem101	87	B
1	Onyeagoro	Chem101	89	B
2	Aina	Chem101	95	A
3	Elizabeth	Chem101	67	D
4	Okafor	Chem101	90	A
5	Thelma	Chem101	45	F
6	Blessing	Chem101	35	F
7	Goodness	Chem101	83	B

In [63]:
```
resDF.to_csv('examRes.csv')
```

In [ ]:
```

```

In [ ]:
```
# Creating a function from a simple script
```

```
 # construct the dictionary

chemStuds = ['Florence', 'Onyeagoro','Aina','Elizabeth','Okafor', 'Thelma', 'Blessi
chemScores = [87,89,95,67,90,45,35, 83]

# create empty dictionary

examDict = dict()

for name, score in zip(chemStuds,chemScores ):
    # print(name, score)
    print(name, score)
    examDict[name]  = score
    print(examDict)


#print(examDict)
```

In [ ]:
```
# Function Purpose:
# it recieves two lists and converts them into a single dictionary
```

In [65]:
```
def dictGen(keyList, valList):

    examDict = dict()

    for name, score in zip(keyList,valList ):
        # print(name, score)
        #print(name, score)
        examDict[name]  = score
        #print(examDict)

    return examDict
```

In [66]:
```
chemStuds = ['Florence', 'Onyeagoro','Aina','Elizabeth','Okafor', 'Thelma', 'Blessi
chemScores = [87,89,95,67,90,45,35, 83]


res = dictGen(chemStuds, chemScores)
```

In [68]:
```
print(res)
```

```
{'Florence': 87, 'Onyeagoro': 89, 'Aina': 95, 'Elizabeth': 67, 'Okafor': 90, 'Thelm
a': 45, 'Blessing': 35, 'Goodness': 83}
```

In [69]:
```
prdts = ['tea', 'toothpast', 'milk', 'bread','coke']

invList = [56,43,  12,20,100]
```

In [70]:
```
res = dictGen(prdts, invList)
print(res)
```

```
{'tea': 56, 'toothpast': 43, 'milk': 12, 'bread': 20, 'coke': 100}
```

In [71]:
```
# How much toothpast is in inventory


```

```
res['toothpast']
```

Out[71]: 43

In [ ]:
```
# Assignment

# James teaches 4 subjects, Chem, Bio, Math and Physics. He has presented you with
# of students for each subject and seperate lists for their corresponding exam scor
# Using the information provided, answer the questions below

chemStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Belonwu' ]
bioStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Samson', 'Chidi
mathStuds = ['David', 'Maryrose', 'Kelechi', 'Mercy', 'Oshinlolu', 'Efe', 'Bello']
phyStuds = ['David', 'samuel', 'Kelechi', 'Pezo', 'Chiamaka', 'Oshinlolu', 'Wale',

chemScores = [46, 86, 57, 87, 98, 45]
bioScores = [57, 89, 56, 34, 85, 78, 90]
mathScore = [79, 67, 45, 87, 76, 59, 87]
phyScores = [69, 87, 87, 78, 45, 46, 89, 56, 91]

subjectList = ['chem', 'bio', 'math', 'phy']

# Task:
1a. Write code to find students that take chemistry and biology but do not take phy
1b. Write code to find the names of all the students in Jame's class

2. Write a function that can be used to add students and their scores to the releva
            wants to add Kenyata to the biology students list and a score of 45 fo
            your function should be able to do that
3. Write code to store each pair of subject student names and subject scores in a d
4. write a function that can fetch a students score for each subject
5. write function to grade the student scores for each subject and record it into a
6. Write code to store all the student results and grades obtained from Q5 in a sin
7. Write a single function that can be used to do the following
            1. find all the subjects a student offers
            2. fetch a students' exam score for a subject or all the subject he/sh
            3. grade the exam scores of students and store the information in a di
            4. input subject exam scores for students
            5. change or update the exam scores for any subject for a given studen
            6. find all the students whos grade or score is below a specified valu
```

In [1]:
```
# sOLUTIONS
```

In [7]:
```
# 1a. Write code to find students that take chemistry and biology but do not take p

chemStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Belonwu' ]
bioStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Samson', 'Chidi
phyStuds = ['David', 'samuel', 'Kelechi', 'Pezo', 'Chiamaka', 'Oshinlolu', 'Wale',

#chemStuds + bioSTuds Vs phyStuds

c_b_Studs = set(chemStuds).intersection(set(bioStuds)) # List of students that take

noPhyStuds = c_b_Studs.difference(set(phyStuds))
```

In [6]:
```
c_b_Studs
```

Out[6]:
```
{'Chiamaka', 'David', 'Kelechi', 'Oshinlolu', 'samuel'}
```

In [8]:
```
noPhyStuds
```

Out[8]:
```
set()
```

In [13]:
```
# 1b. Write code to find the names of all the students in Jame's class

chemStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Belonwu' ]
bioStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Samson', 'Chidi
mathStuds = ['David', 'Maryrose', 'Kelechi', 'Mercy', 'Oshinlolu', 'Efe', 'Bello']
phyStuds = ['David', 'samuel', 'Kelechi', 'Pezo', 'Chiamaka', 'Oshinlolu', 'Wale',

set(chemStuds+ bioStuds+phyStuds + mathStuds)
```

Out[13]:
```
{'Bello',
 'Belonwu',
 'Chiamaka',
 'Chidiebere',
 'David',
 'Efe',
 'Kelechi',
 'Maryrose',
 'Mercy',
 'Oshinlolu',
 'Pezo',
 'Samson',
 'Wale',
 'samuel'}
```

In [14]:
```
# using Union

((set(chemStuds).union(set(bioStuds))).union(set(phyStuds))).union(set(mathStuds))
```

Out[14]:
```
{'Bello',
 'Belonwu',
 'Chiamaka',
 'Chidiebere',
 'David',
 'Efe',
 'Kelechi',
 'Maryrose',
 'Mercy',
 'Oshinlolu',
 'Pezo',
 'Samson',
 'Wale',
 'samuel'}
```

In [18]:
```
'''
2. Write a function that can be used to add students and their scores to the releva
             wants to add Kenyata to the biology students list and a score of 45 fo
```

```
            your function should be able to do that
'''
```

Out[18]:
```
'\n2. Write a function that can be used to add students and their scores to the re
levant subjects. For example, if the teacher\n             wants to add Kenyata to
the biology students list and a score of 45 for him in the bioScores list, \n
your function should be able to do that\n'
```

In [30]:
```
chemStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Belonwu' ]
bioStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Samson', 'Chidi
mathStuds = ['David', 'Maryrose', 'Kelechi', 'Mercy', 'Oshinlolu', 'Efe', 'Bello']
phyStuds = ['David', 'samuel', 'Kelechi', 'Pezo', 'Chiamaka', 'Oshinlolu', 'Wale',


chemScores = [46, 86, 57, 87, 98, 45]
bioScores = [57, 89, 56, 34, 85, 78, 90]
mathScore = [79, 67, 45, 87, 76, 59, 87]
phyScores = [69, 87, 87, 78, 45, 46, 89, 56, 91]
```

In [77]:
```
def studRecsAdd():
    # recieve user input
    subject = input('please specify subject. Chem, Bio, Math, Phy')
    studName = input('provide student name')
    studScore = float(input(f'provide score for {subject}'))
    print(subject, studName, studScore)

    # Update the subject and score records

    if subject == 'Chem':
        chemStuds.append(studName)
        chemScores.append(studScore)
    elif subject == 'Bio':
        bioStuds.append(studName)
        bioScores.append(studScore)
    elif subject == 'Math':
        mathStuds.append(studName)
        mathScore.append(studScore)
    elif subject == 'Phy':
        phyStuds.append(studName)
        phyScores.append(studScore)
    else:
        print('suject unknown')
```

In [76]:
```
studRecsAdd()
```

```
please specify subject. Chem, Bio, Math, PhyPhy
provide student nameImmaculate
provide score for Phy89
Phy Immaculate 89
```

In [27]:
```
chemStuds
```

Out[27]:
```
['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Belonwu', 'Amuche']
```

In [28]:
```
chemScores
```

Out[28]:
```
[46, 86, 57, 87, 98, 45, 85.0]
```

In [38]:
```
phyStuds
```

Out[38]:
```
['David',
 'samuel',
 'Kelechi',
 'Pezo',
 'Chiamaka',
 'Oshinlolu',
 'Wale',
 'Samson',
 'Chidiebere',
 'Ifeoma']
```

In [ ]:
```

```

In [39]:
```
def studRecsAdd1(subject, studName, studScore):

    # Update the subject and score records

    if subject == 'Chem':
        chemStuds.append(studName)
        chemScores.append(studScore)
    elif subject == 'Bio':
        bioStuds.append(studName)
        bioScores.append(studScore)
    elif subject == 'Math':
        mathStuds.append(studName)
        mathScore.append(studScore)
    elif subject == 'Phy':
        phyStuds.append(studName)
        phyScores.append(studScore)
    else:
        print('suject unknown')
```

In [40]:
```
# recieve user input
subject = input('please specify subject. Chem, Bio, Math, Phy')
studName = input('provide student name')
studScore = float(input(f'provide score for {subject}'))
print(subject, studName, studScore)

studRecsAdd1(subject, studName, studScore)
```

```
please specify subject. Chem, Bio, Math, PhyBio
provide student nameAtoku
provide score for Bio73
Bio Atoku 73.0
```

In [41]:
```
bioStuds
```

Out[41]:
```
['David',
 'samuel',
 'Kelechi',
 'Chiamaka',
 'Oshinlolu',
 'Samson',
 'Chidiebere',
 'Henry',
 'Atoku']
```

In [42]:
```
bioScores
```

Out[42]:
```
[57, 89, 56, 34, 85, 78, 90, 67.0, 73.0]
```

In [ ]:
```
# 3. Write code to store each pair of subject student names and subject scores in a
```

In [ ]:
```
chemRec = {'Atoku':73, 'Samuel':65, ....}
```

In [51]:
```
chemStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Belonwu' ]
bioStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Samson', 'Chidi
mathStuds = ['David', 'Maryrose', 'Kelechi', 'Mercy', 'Oshinlolu', 'Efe', 'Bello']
phyStuds = ['David', 'samuel', 'Kelechi', 'Pezo', 'Chiamaka', 'Oshinlolu', 'Wale',


chemScores = [46, 86, 57, 87, 98, 45]
bioScores = [57, 89, 56, 34, 85, 78, 90]
mathScore = [79, 67, 45, 87, 76, 59, 87]
phyScores = [69, 87, 87, 78, 45, 46, 89, 56, 91]


studsRecs = [chemStuds, bioStuds, mathStuds, phyStuds]
studScores = [chemScores, bioScores,mathScore , phyScores ]
subjectList = ['chem', 'bio', 'math', 'phy']

examsDict = dict()


for nameRecs, scoreRecs, subject in zip(studsRecs, studScores,subjectList):
    #print(nameRecs,scoreRecs,subject)

    # sorting out dictioary for each subject
    tempDict = dict()
    for name, scr in zip(nameRecs,scoreRecs):
        tempDict[name]= scr

    if subject == 'chem':
        examsDict['chem']= tempDict
    elif subject == 'bio':
        examsDict['bio']= tempDict
    elif subject == 'math':
        examsDict['math']= tempDict
    elif subject == 'phy':
        examsDict['phy']= tempDict
```

```
    else:
        print('subject unkown')
```

In [52]:
```
examsDict
```

Out[52]:
```
{'chem': {'David': 46,
  'samuel': 86,
  'Kelechi': 57,
  'Chiamaka': 87,
  'Oshinlolu': 98,
  'Belonwu': 45},
 'bio': {'David': 57,
  'samuel': 89,
  'Kelechi': 56,
  'Chiamaka': 34,
  'Oshinlolu': 85,
  'Samson': 78,
  'Chidiebere': 90},
 'math': {'David': 79,
  'Maryrose': 67,
  'Kelechi': 45,
  'Mercy': 87,
  'Oshinlolu': 76,
  'Efe': 59,
  'Bello': 87},
 'phy': {'David': 69,
  'samuel': 87,
  'Kelechi': 87,
  'Pezo': 78,
  'Chiamaka': 45,
  'Oshinlolu': 46,
  'Wale': 89,
  'Samson': 56,
  'Chidiebere': 91}}
```

In [44]:
```
studsRecs
```

Out[44]:
```
[['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Belonwu'],
 ['David',
  'samuel',
  'Kelechi',
  'Chiamaka',
  'Oshinlolu',
  'Samson',
  'Chidiebere'],
 ['David', 'Maryrose', 'Kelechi', 'Mercy', 'Oshinlolu', 'Efe', 'Bello'],
 ['David',
  'samuel',
  'Kelechi',
  'Pezo',
  'Chiamaka',
  'Oshinlolu',
  'Wale',
  'Samson',
  'Chidiebere']]
```

In [ ]:

In [61]:
```
#  4. write a function that can fetch a students score for each subject


chemStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Belonwu' ]
bioStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Samson', 'Chidi
mathStuds = ['David', 'Maryrose', 'Kelechi', 'Mercy', 'Oshinlolu', 'Efe', 'Bello']
phyStuds = ['David', 'samuel', 'Kelechi', 'Pezo', 'Chiamaka', 'Oshinlolu', 'Wale',


chemScores = [46, 86, 57, 87, 98, 45]
bioScores = [57, 89, 56, 34, 85, 78, 90]
mathScore = [79, 67, 45, 87, 76, 59, 87]
phyScores = [69, 87, 87, 78, 45, 46, 89, 56, 91]


def scoreSearcher():
    # error function to call when name or subject does not exist
    def errorRes():
        print('check the subject or student name. it is also pssible student does n

    # recieve user input
    subj = input('specify subject to search: Chem, Bio, Math, Phy')
    studName = input('student name to search')

    # Logic to search for student score
    score = ''
    if subj == 'Chem':
        if studName in chemStuds:
            idxPos = chemStuds.index(studName)
            score = chemScores[idxPos]
            print(score)
        else:
            errorRes()

    elif subj == 'Bio':
        if studName in bioStuds:
            idxPos = bioStuds.index(studName)
            score = bioScores[idxPos]
            print(score)
        else:
            errorRes()

    elif subj == 'Math':
        if studName in mathStuds:
            idxPos = mathStuds.index(studName)
            score = mathScore[idxPos]
            print(score)
        else:
            errorRes()

    elif subj == 'Phy':
        if studName in phyStuds:
            idxPos = phyStuds.index(studName)
```

```
            score = phyScores[idxPos]
            print(score)
        else:
            errorRes()
```

In [62]:
```
scoreSearcher()
```

```
specify subject to search: Chem, Bio, Math, PhyPhy
student name to searchUju
check the subject or student name. it is also pssible student does not exist in the
records
```

In [73]:
```
!pip install pandas
```

```
Requirement already satisfied: pandas in c:\users\stanl\anaconda3\lib\site-packages
(1.4.4)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\stanl\anaconda3\li
b\site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\stanl\anaconda3\lib\site-pac
kages (from pandas) (2023.3)
Requirement already satisfied: numpy>=1.18.5 in c:\users\stanl\anaconda3\lib\site-pa
ckages (from pandas) (1.21.5)
Requirement already satisfied: six>=1.5 in c:\users\stanl\anaconda3\lib\site-package
s (from python-dateutil>=2.8.1->pandas) (1.16.0)
[notice] A new release of pip is available: 23.2.1 -> 23.3.2
[notice] To update, run: python.exe -m pip install --upgrade pip
```

In [63]:
```
# 5. write function to grade the student scores for each subject and record it into

chemStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Belonwu' ]
bioStuds = ['David', 'samuel', 'Kelechi', 'Chiamaka', 'Oshinlolu', 'Samson', 'Chidi
mathStuds = ['David', 'Maryrose', 'Kelechi', 'Mercy', 'Oshinlolu', 'Efe', 'Bello']
phyStuds = ['David', 'samuel', 'Kelechi', 'Pezo', 'Chiamaka', 'Oshinlolu', 'Wale',


chemScores = [46, 86, 57, 87, 98, 45]
bioScores = [57, 89, 56, 34, 85, 78, 90]
mathScore = [79, 67, 45, 87, 76, 59, 87]
phyScores = [69, 87, 87, 78, 45, 46, 89, 56, 91]


studsRecs = [chemStuds, bioStuds, mathStuds, phyStuds]
studScores = [chemScores, bioScores,mathScore , phyScores ]
subjectList = ['chem', 'bio', 'math', 'phy']

examsDict = dict()

def grader(score):

    if score >=90:
        grade = 'A'
    elif score >=80:
        grade = 'B'
    elif score >=70:
        grade = 'C'
    elif score >= 60:
```

```
        grade = 'D'
    else:
        grade = 'F'
    return grade

studPerfRec = []

for nameRecs, scoreRecs, subject in zip(studsRecs, studScores,subjectList):
    #print(nameRecs,scoreRecs,subject)

    # sorting out dictioary for each subject
    tempDict = dict()
    for name, scr in zip(nameRecs,scoreRecs):

        grade = grader(scr)
        tempDict[name]= [scr,grade]
        studPerfRec.append([name, subject, scr,grade])

    if subject == 'chem':
        examsDict['chem']= tempDict
    elif subject == 'bio':
        examsDict['bio']= tempDict
    elif subject == 'math':
        examsDict['math']= tempDict
    elif subject == 'phy':
        examsDict['phy']= tempDict
    else:
        print('subject unkown')
```

In [72]:
```
studPerfRec
```

Out[72]:
```
[['David', 'chem', 46, 'F'],
 ['samuel', 'chem', 86, 'B'],
 ['Kelechi', 'chem', 57, 'F'],
 ['Chiamaka', 'chem', 87, 'B'],
 ['Oshinlolu', 'chem', 98, 'A'],
 ['Belonwu', 'chem', 45, 'F'],
 ['David', 'bio', 57, 'F'],
 ['samuel', 'bio', 89, 'B'],
 ['Kelechi', 'bio', 56, 'F'],
 ['Chiamaka', 'bio', 34, 'F'],
 ['Oshinlolu', 'bio', 85, 'B'],
 ['Samson', 'bio', 78, 'C'],
 ['Chidiebere', 'bio', 90, 'A'],
 ['David', 'math', 79, 'C'],
 ['Maryrose', 'math', 67, 'D'],
 ['Kelechi', 'math', 45, 'F'],
 ['Mercy', 'math', 87, 'B'],
 ['Oshinlolu', 'math', 76, 'C'],
 ['Efe', 'math', 59, 'F'],
 ['Bello', 'math', 87, 'B'],
 ['David', 'phy', 69, 'D'],
 ['samuel', 'phy', 87, 'B'],
 ['Kelechi', 'phy', 87, 'B'],
 ['Pezo', 'phy', 78, 'C'],
 ['Chiamaka', 'phy', 45, 'F'],
 ['Oshinlolu', 'phy', 46, 'F'],
 ['Wale', 'phy', 89, 'B'],
 ['Samson', 'phy', 56, 'F'],
 ['Chidiebere', 'phy', 91, 'A']]
```

In [64]:
```
examsDict
```

Out[64]:

```
{'chem': {'David': [46, 'F'],
  'samuel': [86, 'B'],
  'Kelechi': [57, 'F'],
  'Chiamaka': [87, 'B'],
  'Oshinlolu': [98, 'A'],
  'Belonwu': [45, 'F']},
 'bio': {'David': [57, 'F'],
  'samuel': [89, 'B'],
  'Kelechi': [56, 'F'],
  'Chiamaka': [34, 'F'],
  'Oshinlolu': [85, 'B'],
  'Samson': [78, 'C'],
  'Chidiebere': [90, 'A']},
 'math': {'David': [79, 'C'],
  'Maryrose': [67, 'D'],
  'Kelechi': [45, 'F'],
  'Mercy': [87, 'B'],
  'Oshinlolu': [76, 'C'],
  'Efe': [59, 'F'],
  'Bello': [87, 'B']},
 'phy': {'David': [69, 'D'],
  'samuel': [87, 'B'],
  'Kelechi': [87, 'B'],
  'Pezo': [78, 'C'],
  'Chiamaka': [45, 'F'],
  'Oshinlolu': [46, 'F'],
  'Wale': [89, 'B'],
  'Samson': [56, 'F'],
  'Chidiebere': [91, 'A']}}
```


In [65]:
```
import pandas as pd

df = pd.DataFrame(studPerfRec, columns=['name','subject','score','grade'])
```

In [66]:
```
df
```

Out[66]:

	name	subject	score	grade
0	David	chem	46	F
1	samuel	chem	86	B
2	Kelechi	chem	57	F
3	Chiamaka	chem	87	B
4	Oshinlolu	chem	98	A
5	Belonwu	chem	45	F
6	David	bio	57	F
7	samuel	bio	89	B
8	Kelechi	bio	56	F
9	Chiamaka	bio	34	F
10	Oshinlolu	bio	85	B
11	Samson	bio	78	C
12	Chidiebere	bio	90	A
13	David	math	79	C
14	Maryrose	math	67	D
15	Kelechi	math	45	F
16	Mercy	math	87	B
17	Oshinlolu	math	76	C
18	Efe	math	59	F
19	Bello	math	87	B
20	David	phy	69	D
21	samuel	phy	87	B
22	Kelechi	phy	87	B
23	Pezo	phy	78	C
24	Chiamaka	phy	45	F
25	Oshinlolu	phy	46	F
26	Wale	phy	89	B
27	Samson	phy	56	F
28	Chidiebere	phy	91	A

In [70]:
```
df.loc[df['score'] >90]
```

Out[70]:

	name	subject	score	grade
4	Oshinlolu	chem	98	A
28	Chidiebere	phy	91	A

In [71]:
```
df.loc[df['grade'] =='F']
```

Out[71]:

	name	subject	score	grade
0	David	chem	46	F
2	Kelechi	chem	57	F
5	Belonwu	chem	45	F
6	David	bio	57	F
8	Kelechi	bio	56	F
9	Chiamaka	bio	34	F
15	Kelechi	math	45	F
18	Efe	math	59	F
24	Chiamaka	phy	45	F
25	Oshinlolu	phy	46	F
27	Samson	phy	56	F

In [78]:
```
df.to_csv("C:/Users/stanl/Documents/workspace/Cohort23-24-demofile.csv")
```

In [ ]:
```
STANLEYOMEIKE@GMAIL.COM
```