Practise resources

https://www.practicepython.org/

https://edabit.com/challenge/PjcKZRx8YE5KzRN63

https://www.w3resource.com/python-exercises/

Variables and datatypes

```
name = "Frodo" #string, whole word
firstLetter = 'F' #Also a string, single letter
age = 12 #integer
age = "12" #string
weight = 7.5 #float
bestFried = "Samwise Gamgee" #string, sentence/ multiple words
hobbit = True #Boolean
hasRing = False #Boolean

weight = 8 #The type can be changed when you reassign a value
```

Type casting

```
item1 = "3"
item2 = 300
item4 = 10.99
item5 = True
item6 = False

print(item5)
print(str(item5)) #prints item5 as a string
```

```
print(item1)
print(int(item1)) #prints item1 as a number
print(item5+item5) #not valid, can not add together booleans
print(str(item5)+str(item5)) #prints 'truetrue'
print(bool(item2)) #prints if the value of item2 is NOT 0, False OR empty
print(int(item6)) #returns the number 0.
```

```
item2 = 300

print(item2 + item2) #prints 600
print(str(item2) + str(item2)) #prints 300300
booleanVal= bool(item2) #stores the boolean value
print(booleanVal) #prints True
```

User inputs

```
#userInput
name = input("What is your name")


#will automatically assume the data is a string
age = input("What is your age?")


# to make an integer as intake use int()
age = int(input("What is your age? "))

#less popular way (extra lines of code):
print("What is your name")
age = input()
```

Mathematical operators

Addition

```
#addition
num1 = 20
num2 = 30
num3 = 3
print(1 + 1)
print(100 + 100)
print(num1 + num2 + num3)

word1 = "Potato"
word2 = "Carrot"
totalLetters = len(word1) + len(word2)
print(totalLetters)
```

Subtraction

```
#subtraction
num1 = 20
num2 = 30
print(1 - 1)
print(100 - 99)
print(num1 - num2)

word1 = "Fantastic"
remainingLetters = 120 - len(word1)
```

Multiplication

```
#multiplication

num1 = 20
num2 = 30
print(1 * 1)
print(100 * 99)
print(num1 * num2)

penPack = 10
cost = 4
totalCost = penPack * cost
```

Division

```
#division
num1 = 20
num2 = 30
print(10 / 2)
print(100 / 99)
print(num1 / num2)

penPack = 10
cost = 4
costPerPen = cost / penPack
```

Exponential (to the power of)

```

# Power of

num1 = 2
num2 = 10
print(10 ** 2)
print(100 ** 8)
print(num2 ** num1)

password = "*******"
allowsChars = 58
totalCombinations = allowsChars ** len(password)
```

Modulus

```
# modulos

remainder = 10 % 2 #answer == 0

busSeats = 42
students = 190
```

```
remainderStudents = students % busSeats
```

Integer Division / Floor division

```
# int/ floor division

wholeDivisions = 10 / 2 #answer == 5

busSeats = 42
students = 190
fullBuses = students / buseSeats
remainderStudents = students % busSeats
```

Comparison operators

```
print(10 == 10) #true
print(10 == '10') #false
print(2 > 10) #false
print(2 < 10) #true
print(10 >= 10) #true
print(9 <= 10) #true

stringEx = "100"
boolEx = True
intEx = 100
floatEx = 100.00

print(intEx == floatEx) #true
print(len(stringEx) > 5) #false
print(stringEx == boolEx) #false
print(bool(stringEx) == boolEx) #true
```

Logical operators

```
##print out all the prime numbers from 1 to 99 (inclusive)

for i in range(1,100):
    if i % i == 0 and i % 1 == 0:
        print(i)
```

```
#write a program to display a menu of options:
# Menu 1.Music  2.History   3. D&T   4.Exit   Please enter your choice:

#the user then enters a choice and the program prints a message such as
#  "You chose...", if they choose 4 the program prints "good bye"

options="1. Music","2. History","3. D&T","4. Exit"

print(options)

choice=input("Choose a subject: ")

if choice == "music" or "history" or "D&T":
    print("You chose", choice)
elif choice == exit:
    print("goodbye")
else:
    print("That was not on the list")
```

String manipulation

```
#string manipulation
```

```
myString = "Welcome to the black parade!"

print(myString[0]) #print W

print(myString[-1]) #prints '!', -1 finds the last index position

print(myString[1:4]) #prints elc

print(myString[:3]) #prints 'Wel'. Starts at 0 up until but no including
the
                    #letter at index position 3

print(myString[5:]) #prints the letters from index position 5 until the
end of the
                    #string.

myString[0] = 'w' #changes the value of the char at index position 1


newString = myString.replace("e","E") #replaces all lowercase e's with
uppercase
print(newString)

newString = myString.replace(" ","_") #replaces spaces with underscores
print(newString)

print(newString.upper()) #prints the string to uppercase

newString.lower() #converts whole string to lowercase

print(newString.isupper()) #prints 'true'
```

Selection (IF, Else)

```
randomNum = random.randint(1,4)
print(randomNum)
```

```
if randomNum == 1:
    suit = "Hearts"
elif randomNum == 2:
    suit = "Diamonds"
elif randomNum == 3:
    suit = "Spade"
elif randomNum == 4:
    suit = "Clubs"
else:
    print("Error")
```

```
                compTurn = random.randint(1,3)
                playerTurn = input("make a choice: 1 for Rock, 2 for Paper
or 3 for Scissors")
                roundCount +=1

                if int(playerTurn) == compTurn:
                    print("Draw")
                    compWins = compWins
                    playerWins = playerWins
                elif playerTurn == "1" and compTurn == 3 or playerTurn ==
"2" and compTurn == "1" or playerTurn == "3" and compTurn ==2:
                    playerWins +=1
                    print("You win this round")
                else:
                    compWins +=1
                    print("Computer wins this round")
```

UK tax calculator

```
pType = input("How often do you get payed?: ")
#WEEKLY WAGE
if pType == "weekly":
    wage = int(input("Enter your weekly wage: "))
    #input of tax
    tax = 0.2
    NI = 0.1
    sLoan = 0.05
```

```
    #Setting rates of tax and national insurance
    if wage < 230:
        tax = 0
    if wage < 192:
        NI = 0
    if wage >= 961:
        tax = 0.4
    if wage < 480:
        sLoan = 0
    print("Your tax total is £",wage*tax)
    print("Your National insurance is £",wage*NI)
    print("Your Student loan payment is £",wage*sLoan)
    TotalTax = tax + NI + sLoan
    if TotalTax == 0:
        print("Your net wage is £",wage)
    else:
        print("Your net wage is £",int(wage-(wage*TotalTax)))
#MONTHLY WAGE
elif pType == "monthly":
    wage = int(input("Enter your monthly wage: "))
    #input of tax
    tax = 0.2
    NI = 0.1
    sLoan = 0.05
    #Setting rates of tax and national insurance
    if wage < 1000:
        tax = 0
    if wage < 833:
        NI = 0
    if wage >= 4166:
        tax = 0.4
    if wage < 2083:
        sLoan = 0
    print("Your tax total is £",wage*tax)
    print("Your National insurance is £",wage*NI)
    print("Your Student loan payment is £",wage*sLoan)
    TotalTax = tax + NI + sLoan
    if TotalTax == 0:
        print("Your net wage is £",wage)
    else:
```

```
        print("Your net wage is £",int(wage-(wage*TotalTax)))


#FOUR WEEKLY WAGE
elif pType == "four weekly":
    wage = int(input("Enter your four weekly wage: "))
    #input of tax
    tax = 0.2
    NI = 0.1
    sLoan = 0.05
    #Setting rates of tax and national insurance
    if wage < 920:
        tax = 0
    if wage < 768:
        NI = 0
    if wage >= 3844:
        tax = 0.4
    if wage < 1920:
        sLoan = 0
    print("Your tax total is £",wage*tax)
    print("Your National insurance is £",wage*NI)
    print("Your Student loan payment is £",wage*sLoan)
    TotalTax = tax + NI + sLoan
    if TotalTax == 0:
        print("Your net wage is £",wage)
    else:
        print("Your net wage is £",int(wage-(wage*TotalTax)))


#YEARLY WAGE
elif pType == "yearly":
    wage = int(input("Enter your yearly wage: "))
    #input of tax
    tax = 0.2
    NI = 0.1
    sLoan = 0.05
    #Setting rates of tax and national insurance
    if wage < 12000:
        tax = 0
    if wage < 10000:
```

```
        NI = 0
    if wage >= 50000:
        tax = 0.4
    if wage < 25000:
        sLoan = 0
    print("Your tax total is £",wage*tax)
    print("Your National insurance is £",wage*NI)
    print("Your Student loan payment is £",wage*sLoan)
    TotalTax = tax + NI + sLoan
    if TotalTax == 0:
        print("Your net wage is £",wage)
    else:
        print("Your net wage is £",int(wage-(wage*TotalTax)))
```

Nested IF Statements

```
      if playerWins > compWins:
            if len(winnerList) < 4:
                winnerList.append("Player")
            else:
                winnerList.pop()
                winnerList.append("Player")
      elif compWins > playerWins:
            if len(winnerList) < 4:
                winnerList.append("Computer")
            else:
                winnerList.pop()
                winnerList.append("Computer")
        else:
            print("Unable to add item to list")
```

For loops

```
#prints 0-99 loops starts at 0 unless
#otherwise declared
```

```
for x in range(100):
   print(x)

#prints 0-3
for num in range(0,4):
   print("This is loop num", num)

#prints 100 down to 1.
for j in range(100,0,-1):
   print(j)


#starting at 1, it prints each increment of 5
#up to but not including 100.
for x in range(1,100,5):
   print(x)

#counts the number of characters in a string, and
#uses that to define how many times the loop
#will repeat
name="Freddy FR07"
for x in range(len(name)):
   print(x)


#takes a user input name. Starts a counter at 0
#checks each letter, and if it equals 'a' then
#the counter increments by 1
name = input("Enter name: ")
countA = 0
for c in name:
   if c == "a":
       countA +=1
   else:
       countA = countA
print(countA)


#takes the word mississippi and a counter of 5.
#the first loop uses each character in the word
```

```
#the variable letter is equal to each character in turn
#for each character, the next loop is repeated 5 times.
#each time printing out the letter and the value of the counter + 1
#once this loop has finished its 5th iteration
#it moves onto the next letter and repeats the process.

word = "Mississippi"
count = 5
for letter in word:
    for x in range(count):
        print(letter, x+1, "\n")
```

While loops

```
number = 10
while(number > 0):
    print(number)
    number -= 1


wordToGuess = "Potato"
found = False
while(found != True):
    guess = input("Guess the word: ")
    if guess == wordToGuess:
        found = True

print("You guessed right")
```

```
import random

#this loop as it is, will repeat forever
while(1):
    num = random.randint(1,100)
    print(num)
```

```
lives = 5
attempts = 0

while(lives > 0 and attempts < 10):
    gamePlay = True
```

Functions

```
def usernameGen(fName, lName):
    randNum = random.randint(0,9)
    username = fName[:3] + lName[-3:] + str(randNum)

    return username

names = []
for x in range(3):
    fName = input("First name: ")
    lName = input("Last name: ")
    username = usernameGen(fName, lName)
    print(username)
    names.append(username)

print(names)
```

```
import random

randomNum = random.randint(1,4)
print(randomNum)

def chooseASuit(randomNum):
    if randomNum == 1:
        suit = "Hearts"
    elif randomNum == 2:
        suit = "Diamonds"
    elif randomNum == 3:
```

```
        suit = "Spade"
    elif randomNum == 4:
        suit = "Clubs"
    else:
        print("Error")

    return suit

def main():
    suit = chooseASuit(randomNum)
    print(suit)

main()
```

```
### username creation and password checker

def create_username():
    created = False
    username = input("Please type in a username between 5-10 characters
long")
    while created == False:
        if len(username) > 4 and len(username) < 11:
            username = username
            created = True
        else:
            username = input("Please try again. It must be between 5-10
characters")
            created = False
    return username


def create_password():
    created = False
    password = input("Please type in a password between 5-8 characters
long")
    while created == False:
        if len(password) > 4 and len(password) < 9:
            password = password
```

```
            created = True
        else:
            password = input("Please try again. It must be between 5-8
characters")
            created = False

    return password

def display_username(username):
    print("Your new username is: ", username)


def main():
    username = create_username()
    password = create_password()
    print_username = display_username(username)

main()
```

Categories game - daily opening

```
import categories

lettersUsed = []
topics = categories.topics
count = 0

while count < 10:
    letter = input("What is your letter? ")
    if letter not in lettersUsed:
        lettersUsed.append(letter)
        print(topics[count])
        count += 1
    else:
        print('Try another letter')
    print(count)
```

categories.py

```
topics = ['Four letter word', 'Movies', 'Fruits', 'Cities', 'Girls names',
'Objects found in your house', 'Foods', 'things you can buy in ikea',
'animals', 'countries']
```

Rock paper scissors

```
import random

def computerTurn():
    oList = ['rock', 'paper', 'scissors']
    rNum = random.randint(0,2)
    compTurn = oList[rNum]
    return compTurn

def playerTurn():

    isValid = False
    while isValid == False:
        playerTurn = input('Make your choice: rock, paper, scissors')
        playerTurn = playerTurn.lower()
        if playerTurn == 'rock' or playerTurn == 'paper' or    playerTurn
== 'scissors':
            isValid = True
            return playerTurn
        #     print("Is valid works")
        else:
            print('Choice not valid')



def compareChoices(computerTurn, playerTurn):
    winner = None
    if (computerTurn == playerTurn):
        winner = 'draw'
```

```
    elif (playerTurn == 'rock' and computerTurn == 'paper') or (playerTurn
== 'paper' and computerTurn == 'scissors') or (playerTurn == 'scissors'
and computerTurn == 'rock'):
        winner  = 'computer'
    else:
        winner = 'player'
    print(f'Winner was {winner}')
    return winner


winners = []

def winnersList():
    for person in range(len(winners)):
        print(f'Winner {person+1} was {winners[person]}')



def main():

    for x in range(3):
        playerWins = 0
        computerWins = 0
        overallWinner = None
        while overallWinner != 'Computer' and overallWinner != 'Player':
            compTurn = computerTurn()
            pTurn = playerTurn()
            winner = compareChoices(compTurn , pTurn )
            if winner == 'computer':
                computerWins = computerWins + 1
                if computerWins == 2:
                    overallWinner = 'Computer'
            elif winner == 'player':
                playerWins += 1
                if playerWins == 2:
                    overallWinner = 'Player'
            else:
                print('It was a draw')

            print(f'{overallWinner} is the overall winner for this round')
```

```
        print('Game over')
        winners.append(overallWinner)
    winnersList()

main()
```

Hangman

```
"""Hangman
Standard game of Hangman. A word is chosen at random from a list and the
user must guess the word letter by letter before running out of
attempts."""

import random

def main():
    welcome = ['Welcome to Hangman! A word will be chosen at random and',
               'you must try to guess the word correctly letter by
letter',
               'before you run out of attempts. Good luck!'
               ]

    for line in welcome:
        print(line, sep='\n')

    # setting up the play_again loop

    play_again = True

    while play_again:
        # set up the game loop

        words = ["hangman", "chairs", "backpack", "bodywash", "clothing",
                 "computer", "python", "program", "glasses", "sweatshirt",
                 "sweatpants", "mattress", "friends", "clocks", "biology",
                 "algebra", "suitcase", "knives", "ninjas", "shampoo"
                 ]
```

```
        chosen_word = random.choice(words).lower()
        player_guess = None # will hold the players guess
        guessed_letters = [] # a list of letters guessed so far
        word_guessed = []
        for letter in chosen_word:
            word_guessed.append("-") # create an unguessed, blank version
of the word
        joined_word = None # joins the words in the list word_guessed

        HANGMAN = (
"""
-----
|   |
|
|
|
|
|
|
|
--------
""",
"""
-----
|   |
|   0
|
|
|
|
|
--------
""",
"""
-----
|   |
|   0
|  -+-
```

```
|
|
|
|
|
--------
""",
"""
-----
|   |
|   0
| /-+-
|
|
|
|
|
--------
""",
"""
-----
|   |
|   0
| /-+-\
|
|
|
|
|
--------
""",
"""
-----
|   |
|   0
| /-+-\
|   |
|
|
|
```

```
|
--------
""",
"""
-----
|    |
|    0
|  /-+-\
|    |
|    |
|
|
|
--------
""",
"""
-----
|    |
|    0
|  /-+-\
|    |
|    |
|   |
|
|
--------
""",
"""
-----
|    |
|    0
|  /-+-\
|    |
|    |
|   |
|   |
|
--------
""",
"""
```

```
-----
|   |
|   0
| /-+-\
|   |
|   |
|  | |
|  |
|
--------
""",
"""
-----
|   |
|   0
| /-+-\
|   |
|   |
|  | |
|  | |
|
--------
""")

        print(HANGMAN[0])
        attempts = len(HANGMAN) - 1


        while (attempts != 0 and "-" in word_guessed):
            print(("\nYou have {} attempts remaining").format(attempts))
            joined_word = "".join(word_guessed)
            print(joined_word)

            try:
                player_guess = str(input("\nPlease select a letter between
A-Z" + "\n> ")).lower()
            except: # check valid input
                print("That is not valid input. Please try again.")
                continue
            else:
```

```
                if not player_guess.isalpha(): # check the input is a
letter. Also checks an input has been made.
                    print("That is not a letter. Please try again.")
                    continue
                elif len(player_guess) > 1: # check the input is only one
letter
                    print("That is more than one letter. Please try
again.")
                    continue
                elif player_guess in guessed_letters: # check it letter
hasn't been guessed already
                    print("You have already guessed that letter. Please
try again.")
                    continue
                else:
                    pass

            guessed_letters.append(player_guess)

            for letter in range(len(chosen_word)):
                if player_guess == chosen_word[letter]:
                    word_guessed[letter] = player_guess # replace all
letters in the chosen word that match the players guess

            if player_guess not in chosen_word:
                attempts -= 1
                print(HANGMAN[(len(HANGMAN) - 1) - attempts])

        if "-" not in word_guessed: # no blanks remaining
            print(("\nCongratulations! {} was the
word").format(chosen_word))
        else: # loop must have ended because attempts reached 0
            print(("\nUnlucky! The word was {}.").format(chosen_word))

        print("\nWould you like to play again?")

        response = input("> ").lower()
        if response not in ("yes", "y"):
            play_again = False
```

```
if __name__ == "__main__":
    main()
```

Lists (1D)

```


#using lists
names = ['john', 'peter', 'joseph', 'paul', 'simon']
print(names)

print(names[0] + " Hello")

#accessing items
#first item
print(names[0])
# #second item
print(names[1])
# #last item
print(names[-1])
# #select afew
print(names[0:2])

# #display everyother names
for x in range(0, len(names),2):
    print(x, " ", names[x])


# #methods

print(names.pop()) #removes the last item
# print(names)
names.append('joseph') #adds the item to the end
# print(names)
names.remove('paul') #finds 'paul' and removes it
# print(names)
names.insert(2, 'mary')#inserts #mary' into position 2
# print(names)
```

```
names.sort() #sorts the names in alpha/ numerical order
print(names)
names.sort(reverse=True) #sorts the names in reverse
print(names)

jIndex = names.index('joseph') #finds the index position of the first
joseph
print(jIndex)

indexPos = []
for x in range(len(names)):
    if names[x] == 'joseph':
        indexJ = x
        indexPos.append(indexJ)

# print(indexPos)


print('for in range \n')

for name in range(len(names)):
    print(names[name])

print('for in list \n')
for orange in names:
    print(orange)


names.clear() #deletes the whole list
```

```
# 1) Write an algorithm that takes 5 user input names and stores them into
a list.

names = []
for x in range(5):
    name = input("What is your name: ")
    names.append(name)
```

```
print(names)

#2. Using this list, print out the name of the person who is first
alphabetically.

names.sort()
firstAlph = names[0]
print(firstAlph)


# 3. Ask the user to choose a name to replace, take a new name from the
user and store this in the same index location as the one being removed.

nameOut = input("Which name would you like to replace? ")
nameIn = input("Which name would you like to add? ")
for x in range(len(names)):
    if names[x] == nameOut:
        index = x

names[index] = nameIn
print(names)
```

Lists (2D)

```
results = [] #empty 2D list


totalPlayers = 2

for p in range(totalPlayers):
    name = input("What is your name")
    age = int(input("What is your age"))
    score = float(input("What score did you get"))
    currentList = [name,age,score]
    results.append(currentList)

print(results)
```

```
print(results[1][1])

for row in results:
    for col in row:
        print(col)
```

```
# Write an algorithm that takes a name, the age and height (in cm) of 5
people. The age and height should be integers. Each set of data should be
stored within a 2D list.

# 1. ask a user for a name and return the name, age and height for that
person in a formatted string.

data = []
for x in range(3):
    name = input("What is your name: ")
    age = int(input("What is your age? "))
    height = int(input("What is your height? "))
    data.append([name, age, height])

print(data)

nameSearch = input("Whos data would you like? ")
for row in range(len(data)):
    for item in data[row]:
        if item == nameSearch:
            index = row
            currentRow = data[row]
            print(f'This person is called {currentRow[0]}, they are
{currentRow[1]} years old and their height is {currentRow[2]}cm.')
```

```
# # 2. Calculate the average age for all the users.

ageTotal = 0
for row in data:
    ageTotal += row[1]

averageAge = ageTotal / len(data)
print(f'The average age is {averageAge}')
```

```
# 3. Calculate the difference between the tallest person and the shortest
person.

tallestP = data[0][2]
shortestP = data[0][2]

for row in data:
    if row[2] > tallestP:
        tallestP = row[2]
        tallestPData = row

    if row[2] < shortestP:
        shortestP = row[2]
        shortestPData = row

print(tallestP, shortestP)
difference = tallestP - shortestP
print(f'The difference between the two people is: {difference} ')
```

Dictionaries

```
myDictionary = {
    'key1' : 'Grace',
    'key2' : 30,
    'key3' : '5ft 9'
```

```
    }

#add new key:value
Key = ‘name’
Name = ‘loki’
myDictionary[f’{key}’] = name
#return a single value
print(myDictionary.get('key1'))
#update a value
myDictionary['key3'] = 'Ablus'
#prints the items as a list of tuples
print(myDictionary.items())
#prints the items in their key: value pair format
print(myDictionary)
#delete an item
del myDictionary['key1']
#or using pop
myDictionary.pop('key3')
print(myDictionary.items())

list(myDictionary)

sorted(myDictionary)
print(myDictionary)

# #returns true or false
isLoki = 'loki' in myDictionary
print(f'Is loki in the list: {isLoki}')

#search for key
searchFor = input('What would you like to search for? ')
isFound = searchFor in myDictionary
print(f'Is {searchFor} in the list? {isFound}')

##iterating through
for key, value in myDictionary.items():
    print(f'The key is {key}, the value is {value}')
```

```
questionsD = {
    'q1' : 'What is your name? ',
    'q2' : 'Whhat is your age?',
    'q3' : 'What is your height?',
    'q4' : 'What is your favourite pet?'
}

answersD = {
    'a1' : 'Grace ',
    'a2' : 30,
    'a3' : '5ft 9',
    'a4' : 'Loki'
}


#little adaption to the Python documentation
for q, a in zip(questionsD.items(), answersD.items()):
    print(f'{q[1]} {a[1]}')


questions = ['name', 'quest', 'favorite color']
answers = ['lancelot', 'the holy grail', 'blue']
for q, a in zip(questions, answers):
    print(f'What is your {q}?  It is {a}.')
```

```
Important for group task

# iterating through and storing in a list

keyName = None

value = None

values = []

for k, v in myDictionary.items():

    value = v

    key = k

    values.append([key,value])

print(values)
```

```
# #creating a dictionary from a list

def createDictionaryItem(incomingList):
    myDictionary = {}
    for row in incomingList:
        key = row[0]
        value = row[0]
        myDictionary[f'{key}'] = value

    return myDictionary


myList = [['q1' , 'What is your name? '], ['q2' ,'Whhat is your
age?'],['q3', 'What is your height?'],
    ['q4', 'What is your favourite pet?']]
```

```
myNewDictionary = createDictionaryItem(myList)
print(myDictionary.items())
```

Deleting a ‘column’

```
students = [['Alfie', 15, 'Networking'], #length of the list - 1

            ['Harrison', 18, 'Programming'],

            ['Jakub', 15, 'Networking'],

            ['Brandon', 16, 'Programming', 'Art'],

            ['Kyle', 17, 'Hacking the school network'],

            ['Matilda', 17, None]]



#assume user input where users start counting at 1

colNum = int(input('Enter a column number to delete, e.g. 1,2,3 '))

colNum -= 1

for row in students:

    if colNum <= len(row)-1:

        del row[colNum]



print(students)
```

Regular expressions

Identifiers:

\ used to escape a character

\d any number

\D anything but a number
\s space
\S anything but a space
\w any character
\W anything but a character
. any character except a new line
\. actually a period
\b whitespace around words

Modifiers:
{1,3} we're expecting 1-3
+ Match 1 or more
? Match 0 or 1
* Match 0 or more
$ match the end of a string
^ match the beginning of a string
| matches either or e.g. \d{1-3}|\w{5-6}
[] Match range or "variance" e.g. [A-Za-z] or [1-5a-qA-Z]
{x} expecting "x" amount

White Space Characters:
\n new line
\s space
\t tab
\e escape (rare)
\f form feed (rare)
\r return

DON'T FORGET!:
. + * ? [ ] $ ^ ( ) { } \ |

```
import re


#########################################################################


name = input("Enter your name: ")
valid = re.match("[A-Z]",name)
if valid:
    print("That looks OK")
else:
    print("Invalid, no capital")

#########################################################################


numText = 'My telephone number is 07598345903. My mums numbers
07756235412.'
# phoneNumber = input('Enter your number: ')
numbers = re.search(r'07\d{9}', numText)
print(numbers)
# number = numText[23:34]
# print(number)

#########################################################################


## match() will start looking at the start of the string
numText = 'My telephone number is 07598345903. My mums numbers
07756235412.'
isNumValid = re.match(r'07\d{09}', numText) #will return false
print(bool(isNumValid))

isNumValid = re.match(r'.+07\d{09}', numText) #will return true
print(str(bool(isNumValid)))


#########################################################################

##findall()
```

```
numText = 'My telephone number is 07598345903. My mums numbers
07756235412.'
# phoneNumber = input('Enter your number: ')
isNumValid = re.findall(r'07\d{09}', numText)
print(isNumValid)
#output will be - ['07598345903', '07756235412']


###########################################################################

fileName = 'file1, file2, file33, 3files, 13_files'
fileFound = re.findall(r'file', fileName)
fileFound = re.findall(r'.file', fileName)
fileFound = re.findall(r'\wfile', fileName)
fileFound = re.findall(r'\w+file', fileName)
fileFound = re.findall(r'[a-z0-9_]*file', fileName)
fileFound = re.findall(r'\dfile', fileName)
fileFound = re.findall(r'file\d+', fileName)
fileFound = re.findall(r'file\d{2}', fileName)
fileFound = re.findall(r'.files?', fileName)
fileFound = re.findall(r'.+files?', fileName)

print(fileFound)

###########################################################################

##valid characters a-z, A-Z, 0-9 !$%?@ - 6-10 characters in length

while(1):
    passwordIn = input('What password would you like to add? ')
    isValid = re.match(r'^[a-zA-Z0-9!$%?@]{6,10}$', passwordIn)

    if isValid:
        print('Valid')
    else:
        print('Not valid')

###########################################################################
```

```
testString = 'This is a standard string. It has all the usual punctuation
marks! I should probably delete them...'
# regEx = re.findall(r'[^!_.?]+', testString)
# print(regEx)

#remove the spaces
clean = re.findall(r'[^!_.?]+', testString) #with space
print(clean)
newstring = ' '.join(clean) #rejoins the words
print(newstring)

##############################################################################
```

```
finalPin = ''

while 1:
    count = 1
    while count != 5:
        pin = input(f'Enter pin number {count}: ')
        valid = re.match('^[0-9]{1}$', pin)
        if valid:
            print('Match')
            finalPin += pin
            count += 1
        else:
            print('Invalid')

    print(f'your pin is: {finalPin}')
```

```
##4 digit pin

while 1:

    pin = input('Enter a pin: ')

    valid = re.match('^[0-9]{4}$', pin)

    if valid:

        print('Match')

    else:

        print('Invalid')
```

```
#any name between 1-10 characters long, upper or lower

while True:
    name = input("Enter your name: ")
    valid = re.match('^[a-zA-Z]{1,10}$', name)
    if valid:
        print('Match')
    else:
        print('Invalid')
```

```
# 3 args: regex, replace string, target string
re.sub(r'[aeiou]+', '-', wood)
#output
'H-w m-ch w-d w-ld - w-dch-ck ch-ck -f - w-dch-ck c-ld ch-ck w-d?'
```

```
wood = 'How much wood would a woodchuck chuck if a woodchuck could chuck
wood?'
re.sub(r'[aeiou]+', '', wood)  # substitute with an empty string
#output
'Hw mch wd wld  wdchck chck f  wdchck cld chck wd?'
```

```
name = input("Enter your name: ")
valid = re.match("[A-Z] ",name)
if valid:
    print("That looks OK")
else:
    print("Invalid, no capital")
```

Error handling (exceptions)

```
def askForInt():
    while True:
        try:
            result = int(input("Please provide a number: "))
        except:
            print("Whoops, this is not a number")
            continue
        else:
            print("Yes, thank you")
            break
        finally:
            print("I am going to ask you again")
```

```
# try - this is the block to be attempted
# except - will execute if there is an error in the try block
# finally - will always be executed
# else - will be run if there is not error in the try block
```

```
#########################################################################
##########



def add(n1,n2):
    print(n1+n2)

add(10,20)
#This will provide an error
number1 = 10
number2 = input("Please provide a number")
# add(number1, number2)


#########################################################################
#########

def add(n1,n2):
    sum = n1+n2
    return sum

try:
    #want to attempt this code
    number1 = 10
    number2 = int(input("Please provide a number: "))
except:
    #what happens if there is an error
    print("This is the except handler, your try statement did not work.")
else:
    #runs if there is not an error
    sumOut = add(number1, number2)
    print("If there is no error, this will run in addition to the try")
    print(sumOut)
```

```
#########################################################################
##########


##will try and open a file called text.txt in write mode. If it does not
exist it will create a new one.
try:
    f = open('text.txt', 'w')
    f.write(90)
except TypeError:
    print("Type Error")
except OSError:
    print("OS error")
except:
    print("All other exeptions")
finally:#optional
    print("I always run")
```

File Handling

```

with open ('testfile.txt', mode = 'w') as f:
    f.write('i created this file')
    print (f)
```

```
rj="rj.txt"
with open(rj,"wt") as file_object:
    lines = file_object.readlines()

for line in lines:
    print(line)
```

```

file = open("invictus.txt","rt")
```

```
contents = file.read()
LINE_BREAK= '\n'
COMMA = ","
SPACE = " "
rows=contents.split(LINE_BREAK)
words=contents.split(SPACE)

for i in rows:
    print(i)

print("\n" * 10)

for i in rows[0]:
    print(i)

print("\n" * 10)

count = 0

for w in words:
    print(w)

for w in words:
    if w.lower() == "the":
        count +=1

print(count)
```

```
import re

#opening a file, reading each line and splitting it up to search each
item/ display only certain items at the end

file = open("people.txt", "r")
for loop in range(4):
    # wholeFile = file.read()
    line = file.readline()
    itemsInLine = line.split(",")
```

```
    # print(line)
    # print(itemsInLine)

    isMusician = re.match(r'Musician', itemsInLine[2])
    if isMusician:
        print(itemsInLine[0],itemsInLine[3])
file.close()

##########################################################

##opening a file, reading each line, searching for musician and printing
the whole line

file = open("people.txt", "r")
for loop in range(4):
    line = file.readline()
    isMusician = re.search(r'Musician', line)
    if isMusician:
        print(line)
##end of for loop
file.close()

##########################################################

##counting how many lines are in the file by reading whole file, splitting
at \n and counting the total lines

with open('people.txt') as f:
    counter = 0
    content = f.read()
    print(content + '\n')
    listOFLines = content.split('\n')
    print(listOFLines)
    # numofLine = len(listOFLines)
    for i in listOFLines:
        if i:
            counter += 1
    print(f'Number of lines in the file {counter}')

print('Connections closed')
```

```
f.write('Test')

#the first algorithm taking counter instead of a hardcoded number.

file = open("people.txt", "r")
for loop in range(counter):
    line = file.readline()
    data = line.split(",")
    isMusician = re.match(r'Musician', data[2])
    if isMusician:
        print(data[0],data[1])
file.close()

############################################################

##counting how many lines are in the files a more simple way

with open('people.txt') as myfile:
    counter = 0
    for line in myfile: #readline()
        if line:
            counter += 1

    print(f'Number of lines in the file {counter}')


#Using line count and combining with search algorithm

with open('people.txt') as myfile:
    counter = 0
    for line in myfile:
        if line:
            counter += 1
            data = line.split(",")
            isMusician = re.match(r'Musician', data[2])
            if isMusician:
                print(data[0],data[3])
                with open('musicians.txt', 'a') as myfile1:
                    myfile1.write(f'{data[0]}, {data[3]}')
```

```
    print(f'Number of lines in the file {counter}')
```

```
# import re

with open('testFile.txt', 'r') as f:
    content = f.read()
    print(content)
    regEx = re.search(r'Grace', content)
    if regEx:
        print('found')
    else:
        print('Not found')

#############################################################################
```

```
with open('demoFile4.txt', 'w') as file4:
    for x in range(3):
        try:
            name=input("Enter a name: ")
            age=int(input("Enter your age: "))
            file4.write(f'{name},{str(age)}\n')
        except:
            print('That was an invalid age ')
```

```
with open('demoFile4.txt', 'w') as file4:
    counter = 0
    for x in range(3):
        try:
            name=input("Enter a name: ")
            age=int(input("Enter your age: "))
            file4.write(f'{name},{str(age)}\n')
        except:
            print('That was an invalid age ')
```

```
#################################################################
```

OOP

```
##teacher class example
import random

class teacher:
    def __init__(self, first, last, con_hours, con_pay):
        self.first = first
        self.last = last
        self.con_hours = con_hours
        self.con_pay = con_pay
        self.idnum = first[0] + last[0:2] + str(random.randint(0,100))

    def pay(self):
        self.payOut = round(self.con_pay / 12, 2)
        return self.payOut



    #what does this do? In real terms?
pay_rise = 1.04
teacher_1 = teacher ('Jake','Gold',35,30000)
print(teacher_1.idnum)
print(teacher_1.pay())
teacher_1.con_pay = teacher_1.con_pay * pay_rise
print(teacher_1.pay())
```

```
##teacher class example
import random

class teacher:

    def __init__(self, first, last, con_hours, con_pay):
        self.first = first
```

```
        self.last = last
        self.con_hours = con_hours
        self.con_pay = con_pay
        self.idnum = first[0] + last[0:2] + str(random.randint(0,100))
        self.email = self.idnum + "@Leventhorpe.net"

    def payMonthly(self):
        self.pay = int(self.con_pay / 12)
        return self.pay

    def pay_rise(self):
        self.con_pay = int(self.con_pay * 1.2)
        self.payMonthly()
        return self.pay, self.con_pay

teacher_1 = teacher ('Jake','Gold',35,30000)
teacher_2 = teacher ('Betty','Fry',45,25000)


print(teacher_1.email)
print(teacher_1.payMonthly())
#teacher_1.pay_rise()
#print(teacher_1.payMonthly())
```

```
class Student():
    def __init__(self, firstName, surname):
        self.firstName = firstName
        self.surname = surname

    def displayDeets(self):
        print(f'My first name is {self.firstName} and my surname is
{self.surname}')


studentObject1 = Student('Alfie', 'Smith')
studentObject1.displayDeets()
```

```python
class Animal:
    def __init__(self, species, name):
        self.species = species
        self.name = name
        # self.age = age
        # self.bornPlace = bornPlace

    def printName(self):
      print(f'My name is {self.name}')

# someAnimal = Animal('Unknown', 'Grogu')
# someAnimal.printName()
# print(someAnimal.name)
########################################

class Cat(Animal):
    def __init__(self, species, name):
        super().__init__(species, name)

    def hiss(self):
      print('HISS!')

    def meow(self):
      print('Meow')

    def purr(self):
      print('Puurr Puuurrrr')

###################################

class Bird(Animal):
    def __init__(self, species, name):
        super().__init__(species, name)

    def squark(self):
      print('SKWARK!!!!!!')

########################################
```

```
class Dog(Animal):
    def __init__(self, species, name):
        super().__init__(species, name)

    def runAndBark(self):
        print("Bark Bark, run run")

#####################################


class Game():
  def __init__(self):
    self.finished = False

  def compareObj(self, object1, object2):
    if object1.species == 'Cat' and object2.species == 'Dog':
      object1.hiss()
      object2.runAndBark()
    elif object1.species == 'Cat' and object2.species == 'Bird':
      object2.squark()
    elif object1.species == 'Cat' and object2.species == 'Cat':
      object1.meow()
      object1.meow()



############################################################

kittie1 = Cat('Cat', 'Loki')
kittie2 = Cat('Cat', 'Albus')
doggo = Dog('Dog', 'Chip')
birdy = Bird('Bird', 'Tweety Pie')
currentGame = Game()

while(currentGame.finished == False):
  currentGame.compareObj(kittie1, kittie2)
  print('Next \n')
  currentGame.compareObj(kittie1, birdy)
  print('Next \n')
```

```
  currentGame.compareObj(kittie1, doggo)
  print('Next \n')
  currentGame.finished = True
```

Sysadmin

```
import os, subprocess


groups = ['sysAdmin', 'DevOps', 'Cloud']
for item in groups:
    try:
        item
        os.system(f'sudo groupadd {item} ')
    except:
        os.system(f'echo "{item} not made" > usernameErrors.txt')

os.system('sudo tail /etc/group')
```

```
import subprocess, os, re

#cat paythonFiles.txt / read file
#remove ./ - left with files name
#pass into chmod <filename> 755

output = os.system("find ./ -name '*.py' > pythonFiles2.txt")
print(output)

with open('pythonFiles2.txt', 'r') as f:
    for line in f:
```

```
        try:
            fileName = line[2:]
            os.system("chmod 755 " + fileName)
        except:
            print(f'Unable to modify file {fileName}')
```

```
import os

path = '/home/admin/python'

files = os.listdir(path)

# os.system('ls -l')

print('\n')
print(files)

for file in files:
    print(file)

print('\n')
print('Hello, all done')
```

```
#script to create usernames and then make new accounts
import os

with open ('names.txt', 'r') as file:
    for line in file:
        items = line.split(',')
        try:
            username = (items[0][-1] + items[1] + str(items[2])).lower()
            print(username)
```

```
            os.system(f'sudo useradd {username}')
        except:
            os.system(f'echo "error in making username" >
failedUserNames')
        else:
            try:
                os.system(f'sudo passwd {username}')
            except:
                print('Issue setting password')


print('done')
```

Practice Q's with answers

```
# q1 - user inputs a number and program reports
# if it is over 5

num = int(input("Enter a number: "))

if num > 5:
    print("This is greater than 5")
else:
    print("This is less than or equal to 5")


# q2 User inputs 3 numbers, program outputs
# in the opposite order. e.g. 1,2,3 is 3,2,1

# beginner

num1 = int(input("Enter number 1: "))
num2 = int(input("Enter number 2: "))
```

```
num3 = int(input("Enter number 3: "))
print(num3,", ", num2,", ", num1)

# advanced

nums = []
for x in range(3):
    num= int(input("Enter a number: "))
    nums.append(num)

for i in range(len(nums)-1,-1,-1):
    print(nums[i])


# q3 user inputs two numbers. If the first is larger
# output sum. If the second is larger
# reports the difference

num1 = int(input("Enter a number: "))
num2 = int(input("Enter a number: "))

if num1 > num2:
    sum = num1 + num2
    print("Number one was larger. The sum of the numbers is ", sum)
elif num2 > num1:
    sum = num2 - num1
    print("Number two was larger, the difference is ", sum)
else:
    print("These are the same number")


#q4a - user inputs a number, the program
# counts up from zero to that number

num = int(input("Enter a number: "))

for x in range(0, num +1):
    print(x)

#q4b - ext 4a, for every other number the program outputs
```

```python
# boo

for x in range(0, num + 1):
    if x % 2 == 0:
        print(x)
    else:
        print("boo")

#5 - Create a needy program that waits for
# the word hug... Who wrote this haha!


hugged = False

while hugged == False:
    wordIn = input("Enter a word or phrase: ")
    if "hug" in wordIn:
        hugged = True

print("Thanks, *hugs!*")

#q6 write a program that can convert percentage change
# to actual change. e.g. user enters old size, %
# change and units and it will output actual
# increase in units



def percentageUnits():
    oldSize = int(input("Enter the old size: "))
    percentageC = int(input("Enter the percentage change: "))
    units = input("Enter the units: ")

    change = oldSize * (percentageC / 100)
    print(f'The change is {change}{units} ')

percentageUnits()
```

```
# q7 User inputs an int, the program outputs a
# half of the integer, rounded down. and if that number is
# odd or even

def numberHalf(num):
    halfNum = num // 2
    if halfNum % 2 == 0:
        print(halfNum, " .This number is even")
    else:
        print(halfNum, " .This number is odd" )

numberHalf(28)



#q8 user inputs the vol of a sphere and program
# outputs the radius

def radius(vol):
    rad = (vol / ((4/3)*3.14)) ** (1/2)
    print(rad)

radius(100)



# q9 A car carrying up to 5 people generates 200g of
# c02 per mile. A bus can carry 40 people
# and generate 1000g of Co2. Write an algorithm to find
# out which is more environmentally friendly

carCo2 = 200
busCo2 = 1000

passengerCar = 200 / 5
passengerBus = 1000 / 40

numberOfPassengers = int(input("How many people are travelling?"))
carsNeeded = 0
```

```
busesNeeded = 0

if numberOfPassengers % 5 != 0:
    carsNeeded = numberOfPassengers // 5 + 1
else:
    carsNeeded = numberOfPassengers // 5

if numberOfPassengers % 40 != 0:
    busesNeeded = numberOfPassengers // 40 + 1
else:
    busesNeeded = numberOfPassengers // 40

totalEmissionBus = busesNeeded * busCo2
totalEmissionCar = carsNeeded * carCo2

print("cars", totalEmissionCar)
print("buses", totalEmissionBus)

if totalEmissionBus > totalEmissionCar:
    print("Car is less co2")
else:
    print("Bus is less co2")



# q10 write an algorithm to count from 1 to 30, outputting
# numbers as it goes along. place an ! next to all numbers
# divisible by both 3 and 4

for x in range(1,31):
    if x % 3 == 0 and x % 4 == 0:
        print(x,"!")
    else:
        print(x)



# q11 - user inputs two numbers, a & b, the program will output
# all the powers of a up to b. B can not be less than 2
```

```
a = int(input("Enter a number: "))
b = int(input("Enter a number of 2 or more: "))
diff = b - a
j = 1
for x in range(a, b+1):
    a = a * j
    j+=1
    print(a)



# q12 user inputs a word and the program finds the middle letter. If the
# word has an even length then round down

word = input("Enter a word to test: ")

if len(word) % 2 == 0:
    midValue = (len(word) // 2 )-1
    letter = word[midValue]
else:
    midValue = (len(word) // 2)
    letter = word[midValue]

print(letter)



#q12 ask a user to add 5 words to a list, each word should be unique so
# do not allow users to enter the same word twice

listFull = False
wordsList = []
listLen = 0

while listFull == False and listLen < 5:
    wordIn = input("Please enter a word: ")
    if wordIn not in wordsList:
        wordsList.append(wordIn)
        listLen +=1
    else:
```

```
        print("The word is already in the list")

    print(listLen)

print(wordsList)
```

```
#q13 Ask the user to input a name, and if the person is guest or not
#keep asking if they want to add another
#once they say no, only print the names of the guests.

partyList = []
another = "y"
def getGuests():
    name = input("Enter a name: ")
    guest = input("Are they a guest of the party, True or False: ")
    guest = guest.lower()
    partyList.append([name, guest])

def showList():
    for x in range (len(partyList)):
        if partyList[x][1] == "true":
            print(partyList[x][0])
        else:
            pass

while another == "y":
    getGuests()
    another = input("again?")

showList()
```

```
#q 14 The user needs to input a message and a letter to begin
#a substring from

message = input("Enter a message")
letter = input("Enter a letter")
count = 0

for x in message:
```

```
    count += 1
    if x == letter:
        position = count
        break

print(message[count-1:])
```

```
#q15 create a 2D list where each row contains two numbers, and the total
of these
#the two numbers should come from input
#ask the user if they would like to add another col, repeat until they say
no
#Then ask them which column would they like to add: 1, 2 or 3.
#display the totals of these cols

""" Example of input and logic
turn1 - 2, 4, 6
turn 2- 10, 11, 21
turn 3- 99, 1, 100

col 1 total = 2 + 10 + 99
col 2 total = 4 + 11 + 1
col 3 total = 6 + 21 + 100

"""

numbers = []
another = "yes"
colTotal = 0

while another == "yes":
    num1 = int(input("What number would you like to add?"))
    num2 = int(input("What number would you like to add?"))
    total = num1 + num2
    numbers.append([num1, num2, total])
    another = input("Would you like to go again?")

columnAdd = int(input("What column would you like to add"))

for x in range (len(numbers)):
```

```
    colTotal = colTotal + numbers[x][columnAdd]

print(colTotal)
```

```
#q16 generate a random number that a user must guess.
#keep track of how many attempts it takes the user
#each time let them know if their guess was too high or too low
#once they guess the number, let me know how many attempts
#it took. e.g. "Well done, that took you 10 attempts"

import random

lownum = int(input("Enter a num"))
highnum = int(input("Enter a highnum"))
num2guess = random.randint(lownum, highnum)
numfound = False
count = 0

while numfound == False:
    guess = int(input("Guess the number"))
    if guess > highnum or guess < lownum:
        print("Try again!")
    else:
        if num2guess == guess:
            count += 1
            print("Well done, that took you ",count, " attempts")
            numfound = True
        elif guess < num2guess:
            print("Too low")
            count +=1
        elif guess > num2guess:
            print("Too high")
            count +=1
```

```
#q17 ask for a users DOB in the following format: 07/10/1982
```

```
#ask for the user's first name and last name.
#create a username that is made up of:
# The first three letters of the  surname, and the last letter of their
#firstname. The numerical sum of their DOB
#e.g. 07/10/1982 == (7 + 1 + 1 + 9 + 8 + 2) = 30.  Display their new
username.
# Grace Shaffi 07/03/1990 becomes Shae29


DOB = input("Enter your date of birth, eg, 07/03/1990: ")
total = 0
firstname = input("What is your first name ")
lastname = input("What is your last name ")


for x in DOB:
    if x != "/":
        item = int(x)
        total = total + item

username =  lastname[:3] + firstname[-1] + str(total)
print(total)
print(username)
```

NEA

```
import random

##nea task
songs =[["Trey Songz", "I Need A Girl"],
        ["T Pain", "Im Sprung"],
        ["Bobby Valentino", "Slow Down"],
        ["Ginuwine", "Differenes"],
        ["Mariah Carey", "We Belong Together"]]
```

```
users=[["user1","pass1"],
       ["user2","pass2"]]

def login(users):
    verified = False
    while verified == False:
        username = input("Enter your username: ")
        password = input("Enter your password: ")
        for x in range(len(users)):
            if username in users[x][x] and password in users[x][x+1]:
                print("found")
                verified = True
                break
            else:
                print("nope!")

def getSong(songs, songsUsed):
    newSong = False
    while newSong == False:
        num = random.randint(0, len(songs)-1)
        if num not in songsUsed:
            song = songs[num][1]
            songsUsed.append(num)
            artist = songs[num][0]
            print(songsUsed)

            newSong = True

    return song, artist

def playGame(song, artist, attempts, score):
        songOut = song
        for i in range(0,len(song)):
            letter = songOut[i]
            if letter.islower():
                songOut = songOut.replace(letter,"_")
        print(artist, " ", songOut)
        userGuess = input("What song is this: ")

        if userGuess == song and attempts == 2:
```

```
            score += 3
            print("Well done, 3 points!")

        elif userGuess == song and attempts == 2:
            score += 1
            print("Well done, 1 points")

        elif userGuess != song and attempts ==2:
            attempts -=1
            print("That was not correct, 1 attempt left")

        elif userGuess != song and attempts ==1:
            attempts -=1
            endGame = True
            print("That was your second attempt, sorry!")

        print("Your total score is ", score)

        return attempts, score


def main():
    login(users)
    songsUsed=[]
    attempts = 2
    score = 0
    endGame = False
    while score < 15 and endGame == False:
        song, artist = getSong(songs, songsUsed)
        attempts, score = playGame(song, artist, attempts, score)


main()
```

```
class Human():

    def __init__(self, fName, lName, age, birthPlace):

        self.firstName = fName

        self.lastName = lName

        self.age = age

        self.birthPlace = birthPlace



    def displayDeets(self):

        print(f'My name is {self.firstName} {self.lastName}. I was born in
{self.birthPlace} and I am {self.age} years old.')



    def speak(self):

        self.displayDeets()

        again = input('Did you want me to say that again? y/n ').lower()

        if again == 'y':

            self.displayDeets()

        else:

            print('Okay, bye!')
```

```
firstHuman = Human('Eve', 'Noname', 35, 'Unknown')


firstHuman.speak()
```

```
class Adult(Human):

    def __init__(self, fName, lName, age, birthPlace, job, canDrive,
married):

        super().__init__(fName, lName, age, birthPlace)

        self.job = job

        self.canDrive = canDrive

        self.married = married




    def changeJob(self):

        newJob = input('What is your new job? ')

        if newJob:

            self.job = newJob

            print(f'Congrats, on the new job as a {self.job}.')

        else:

            print('Sorry, please enter a job title')
```

```
adult1 = Adult('Grace', 'Smith', '30', 'Hackney', 'Instructor', True,
True)



adult1.displayDeets()

adult1.changeJob()









class Child(Human):

    def __init__(self, fName, lName, age, birthPlace, hungry, mother,
canWalk):

        super().__init__(fName, lName, age, birthPlace)

        self.hungry  = hungry

        self.mother = mother

        self.canWalk = canWalk



    def feedMe(self):

        if self.hungry:

            print('Okay, have a potato')
```

```
            self.hungry = False

        else:

            print('Nope, here is a dummy istead')



    def callMother(self):

        print(f'Can you get my mother, her name is
{self.mother.firstName}')




child1 = Child('Grace Jnr', 'Briody', 5, 'Harlow', True, adult1, True)



child1.callMother()

class ListOfItems():

    def __init__(self, maxLength):

        self.myMaxLength = maxLength

        self.myIsEmpty = True

        self.myItems = []

        self.myLength = len(self.myItems)
```

```
    def addItem(self, item):

        if self.myLength < self.myMaxLength:

            self.myItems.append(item)

            # print('Item added')

            # print(self.myItems)

            self.myIsEmpty = False

        else:

            print('Sorry the list is full')



    def displayItems(self):

             for item in self.myItems:

                 print(item)




class Player():

    def __init__(self, name):

        self.name = name

        self.score = 0
```

```
        self.quiestionsAnswers = 0




    def printScore(self):

        print(f'My score is {self.score}')



    def answerQuestion(self, question, answer):

        print(question)

        answerAttempt = input('What is your answer: ')

        if answerAttempt == answer:

            self.score = self.score + 1

            myList.addItem(answerAttempt)

        else:

            print('Sorry that was the wrong answer')



    def printFinalScore(self):

        print(f'My final score is {self.score}')
```

```
myList = ListOfItems(5)


name = input("What is your name: ")
currentPlayer = Player(name)


q = 'What is the capital of Ireland? '
a = 'Dublin'
currentPlayer.answerQuestion(q, a)


q = 'What is the capital of England? '
a = 'London'
currentPlayer.answerQuestion(q, a)


q = 'What is the capital of Wales? '
a = 'Cardiff'
currentPlayer.answerQuestion(q, a)
```

```
q = 'What is the capital of Scotland? '

a = 'Edinburgh'

currentPlayer.answerQuestion(q, a)



q = 'What is the capital of France? '

a = 'Paris'

currentPlayer.answerQuestion(q, a)



print(myList.myItems)

print(currentPlayer.printFinalScore())
```

Databases

```
import mysql.connector

con = mysql.connector.connect(
```

```
  host="127.0.0.1",
  user="admin",
  password="#",
  database="sql_store"
)

cur = con.cursor()

cur.execute("SELECT * FROM customers WHERE customer_id = 1")

result = cur.fetchall()

print(result)
```

```
import mysql.connector

con = mysql.connector.connect(
  host="127.0.0.1",
  user="admin",
  password="#",
)

cur = con.cursor()
cur.execute("SHOW DATABASES")
db = cur.fetchall()
print(db)
```

Encryption (Salt hashing)

```
import hashlib, sys, uuid
goAgain = True

while goAgain == True:
```

```
    password = input("Enter a password: ")
    salt = uuid.uuid4().hex
    hashedP =
hashlib.sha256(salt.encode()+password.encode()).hexdigest()+":"+salt
    print(hashedP)

    f = open("hasing.txt","a")
    f.write("\n" + hashedP)
    f.close()

    again = input("y/n")
    if again =="y":
        goAgain = True
    else:
        goAgain = False
```

Gimme the salt

```
import random
import string

def saltHash(stringLength):
    hashCreate = False
    while hashCreate == False:
        letters = string.ascii_lowercase
        saltKey = ''.join(random.choice(letters) for i in
range(stringLength))
        hashCreate = True
        print(saltKey)
    return saltKey



def main():

    saltKey = saltHash(15)

main()
```

Hash hash hash

```
import hashlib
import uuid



password = "password1"
def hasher(password):    #The hash function used to hash a password
    salt = uuid.uuid4().hex
    return
hashlib.sha256(salt.encode()+password.encode()).hexdigest()+":"+salt


def main():
    passwordOut = hasher(password)
    print(hasher(password))

main()




def verifyhash(self, userpass, storedpass):          #Verifies the hash
    self.userpass = userpass
    try:                                             #Prevents crash in
instance of invalid stored hash
        password,salt=storedpass.split(":")
    except:
        pass
    else:
        data = []
        data.append(password)

data.append(hashlib.sha256(salt.encode()+self.userpass.encode()).hexdigest
())
        return data[0]==data[1]
```

O

Database and python class project 1 - OOP

```
import uuid, hashlib, mysql.connector, os



class DBCon():
    def __init__(self):
        self.host = '127.0.0.1'
        self.user = 'root'
        self.password = os.environ.get('dbPassword')
        self.dataBase = 'DBUsers'
        self.cur = None
        self.con = None

    def getCon(self):
        self.con = mysql.connector.connect(
        host = self.host,
        user = self.user,
        password =f"{self.password}",
        database= self.dataBase
        )

        self.cur = self.con.cursor()

    def closeDB(self):
        self.con.close()



    def showDatabases(self):
        self.cur.execute('SHOW DATABASES')
        dbs = self.cur.fetchall()
        print(dbs)
# dataB.showDatabases()

class DatabaseActions():
    def __init__(self):
        self.dbCur = dataB.cur
        self.dbCon = dataB.con
```

```
    def addToDb(self, username, HashedPassword, email):
        try:
            sql = f"""INSERT INTO `users` VALUES ('{username}',
'{HashedPassword}','{email}')"""
            self.dbCur.execute(sql)
            self.commit1()
        except:
            print('Did not insert')
        else:
            print('All done :) ')


    def commit1(self):
        self.dbCon.commit()

    def execute(self, sqlCommand):
        self.dbCur.execute(sqlCommand)

class HashingPasswords():

    def hashPassword(self, password):
        salt = uuid.uuid4().hex
        hashedPassword =
hashlib.sha256(salt.encode()+password.encode()).hexdigest()+":"+salt

        return hashedPassword


    def verifyhash(self, userpass, storedpass):   #Verifies the hash
        try:   #Prevents crash in instance of invalid stored hash
            password,salt=storedpass.split(":")
        except:
            pass
        else:
            data = []
            data.append(password)

data.append(hashlib.sha256(salt.encode()+userpass.encode()).hexdigest())
```

```
        return data[0]==data[1]

class UserActions():
    def __init__(self):
        self.exitP = 0

    def login(self):
        valid = False
        loggedIn = False
        while not valid:
            username = input("Enter a username: ")
            password = input("Enter a password: ")
            if username and password:
                valid = True
                print('Validated \n')
        while not loggedIn:
            sql = f"""SELECT password FROM users WHERE
username='{username}'"""
            dbActions.execute(sql)
            items = dbActions.dbCur.fetchone()
            print(items)
            if items:
                dbPassword = items[0]
                match = hasher.verifyhash(password, dbPassword)
                if match:
                    loggedIn = True
                    print('You are now logged in. ')
                else:
                    print('The password doesnt match')
            else:
                print('Sorry, I could not find you ')




    def register(self):
        valid = False
        while not valid:
            username = input("Enter a username: ")
```

```
            password = input("Enter a password: ")
            email = input("Enter an email:")
            if username and password and email:
                valid = True

        hashedUserpass = hasher.hashPassword(password)
        dbActions.addToDb(username, hashedUserpass, email)
        ##not yet checked if already exisits

    def exit(self):
        print('Goodbye!')
        self.exitP = 1

class Program():

    def main(self):

        userA.exitP = 0
        while not userA.exitP:
            try:
                choice = int(input('Welcome to the program. Choose form
the following: \n1.Register \n2.Login \n3.Exit \n'))
            except:
                print("Please enter either 1, 2 or 3 to indicate your
choice ")
            else:
                if choice == 1:
                    userA.register()
                elif choice == 2:
                    userA.login()
                elif choice == 3:
                    userA.exit()
                else:
                    print('Please type 1, 2 or 3 to indicate your choice:
')



dataB = DBCon()
```

```
dataB.getCon()
dbActions = DatabaseActions()
hasher = HashingPasswords()
userA = UserActions()
currentP = Program()

currentP.main()
```

Database and python class project 1 - Procedural

```
import uuid, hashlib, mysql.connector, os

SQLpassword = os.environ.get('dbPassword')

con = mysql.connector.connect(
  host="127.0.0.1",
  user="root",
  password=f"{SQLpassword}",
  database="DBusers"
)

cur = con.cursor()

# cur.execute("SHOW DATABASES")
# db = cur.fetchall()
# print(db)

def getUserInfo():
    valid = False
    while not valid:
        username = input("Enter a username: ")
        password = input("Enter a password: ")
        email = input("Enter an email:")
        if username and password and email:
            valid = True
```

```
        #regex username, password, email.
        #
    return username, password, email

def loginGetInfo():
        valid = False
        while not valid:
            username = input("Enter a username: ")
            password = input("Enter a password: ")
            if username and password:
                valid = True
                print('Validated \n')
        #regex username, password, email.

        return username, password

def addToDb(username, HashedPassword, email):
    try:
        sql = f"""INSERT INTO `users` VALUES ('{username}',
'{HashedPassword}','{email}')"""
        cur.execute(sql)
        con.commit()
    except:
        print('Did not insert')
    else:
        print('All done :) ')

def hashPassword(password):
    salt = uuid.uuid4().hex
    hashedPassword =
hashlib.sha256(salt.encode()+password.encode()).hexdigest()+":"+salt

    return hashedPassword

def verifyhash(userpass, storedpass):   #Verifies the hash
    try:   #Prevents crash in instance of invalid stored hash
        password,salt=storedpass.split(":")
    except:
        pass
    else:
```

```
        data = []
        data.append(password)

data.append(hashlib.sha256(salt.encode()+userpass.encode()).hexdigest())

    return data[0]==data[1]

def main():
    x = 0
    while not x:
        try:
            choice = int(input('Welcome to the program. Choose form the
following: \n1.Register \n2.Login 3.Exit '))
        except:
            print("Please enter either 1, 2 or 3 to indicate your choice
")
        else:
            if choice == 1:
                username, password, email = getUserInfo()
                hashedUserpass = hashPassword(password)
                addToDb(username, hashedUserpass, email) #add to database
            elif choice == 2:
                loggedIn = False
                username, password = loginGetInfo()
                while not loggedIn:
                        #get username and password
                        sql = f"""SELECT password FROM users WHERE
username='{username}'"""
                        cur.execute(sql)
                        items = cur.fetchone()
                        print(items)
                        if items:
                            dbPassword = items[0]
                            match = verifyhash(password, dbPassword)
                            if match:
                                loggedIn = True
                                print('You are now logged in. ')
                            else:
                                print('The password doesnt match')
                        else:
```

```
                            print('Sorry, I could not find you ')

            elif choice == 3:
                print('Goodbye!')
                x = 1
            else:
                print('Please type 1, 2 or 3 to indicate your choice: ')


while(1):
    main()
```