"Project 1"

HI welcome to the game ,This is number guessing game,we got 7 chances to guess the number)

Task: Below are the steps:

Build a Number guessing game, in which the user selects a range. Let's say User selected a range, i.e., from A to B, where A and B belong to Integer. Some random integer will be selected by the system and the user has to guess that integer in the minimum number of guesses Analysis:

Explanation 1: If the User inputs range, let's say from 1 to 100. And compiler randomly selected 42 as the integer. And now the guessing game started, so the user entered 50 as his/her first guess. The compiler shows "Try Again! You guessed too high". That's mean the random number (i.e., 42) doesn't fall in the range from 50 to 100. That's the importance of guessing half of the range. And again, the user guesses half of 50 (Could you tell me why?). So the half of 50 is 25. The user enters 25 as his/her second guess. This time compiler will show, "Try Again! You guessed too small". That's mean the integers less than 25 (from 1 to 25) are useless to be guessed. Now the range for user guessing is shorter, i.e., from 25 to 50. Intelligently! The user guessed half of this range, so that, user guessed 37 as his/her third guess. This time again the compiler shows the output, "Try Again! You guessed too small". For the user, the guessing range is getting smaller by each guess. Now, the guessing range for user is from 37 to 50, for which the user guessed 43 as his/her fourth guess. This time the compiler will show an output "Try Again! You guessed too high". So, the new guessing range for users will be from 37 to 43, again for which the user guessed the half of this range, that is, 40 as his/her fifth guess. This time the compiler shows the output, "Try Again! You guessed too small". Leaving the guess even smaller such that from 41 to 43. And now the user guessed 41 as his/her sixth guess. Which is wrong and shows output "Try Again! You guessed too small". And finally, the User Guessed the right number which is 42 as his/her seventh guess.

Total Number of Guesses = 7

In [8]:
```
import random
import math

lower=int(input("Enter the lower bound:"))
upper=int(input("Enter the upper bound:"))

x=random.randint(lower,upper)

total_chances=math.ceil(math.log(upper-lower+1,2))

print("\n\tYou've only",total_chances,"chances to guess the integer!\n")

count=0
flag=False

while count<total_chances:
    count+=1
    guess=int(input("your guess number is:"))
    if x==guess:
        print("Congratulations you did it in",count,"try")
        flag=True
        break
    elif x>guess:
        print("your number smaller to guess number")
    elif x<guess:
        print("your  number greater to guess number")
if not flag:
    print("\n The number is %d" %x)
    print("\t Better Luck Next time !")
```

```
Enter the lower bound:1
Enter the upper bound:100

	You've only 7 chances to guess the integer!

your guess number is:50
your  number greater to guess number
your guess number is:25
your  number greater to guess number
your guess number is:12
your  number greater to guess number
your guess number is:6
Congratulations you did it in 4 try
```

In [4]:
```
import random

print("Hi welcome to the game, This is a number guessing game.\nYou got 7 chances to guess the number. Let's sta

number_to_guess = random.randrange(100)

chances = 7

guess_counter = 0
```

```
while guess_counter < chances:

    guess_counter += 1
    my_guess = int(input('Please Enter your Guess : '))

    if my_guess == number_to_guess:
        print(f'The number is {number_to_guess} and you found it right !! in the {guess_counter} attempt')
        break

    elif guess_counter >= chances and my_guess != number_to_guess:
        print(f'Oops sorry, The number is {number_to_guess} better luck next time')

    elif my_guess > number_to_guess:
        print('Your guess is higher ')

    elif my_guess < number_to_guess:
        print('Your guess is lesser')
```

```
Hi welcome to the game, This is a number guessing game.
You got 7 chances to guess the number. Let's start the game
Your guess is lesser
Your guess is higher
Your guess is lesser
Your guess is lesser
Your guess is lesser
Your guess is higher
Oops sorry, The number is 47 better luck next time
```

Project 2

In [ ]:
```
"Word guessing Game in Python"
```

This program is a simple word-guessing game where the user has to guess the characters in a randomly selected word within a limited number of attempts. The program provides feedback after each guess, helping the user to either complete the word or lose the game based on their guesses.

In [1]:
```
import random

name=input("what is your name?")
print("Good Luck !",name)

words=['rainbow','computer','science','programming','python','mathematics','player','condition','reverse','wate
word=random.choice(words)

print("Guess the characters")

guesses=''
turns=12

while turns>0:
    failed=0
    for char in word:
        if char in guesses:
            print(char,end=" ")
        else:
            print("_")
            failed+=1
    if failed==0:
        print("you won")
        print("The word is :",word)
        break
    print()
    guess=input("guess a charcater")
    guesses+=guess
    if guess not in word:
        turns-=1
        print("wrong")
        print("you have",+turns,'more guesses')
        if turns ==0:
            print("you loose")
```

```
Good Luck ! srinu
Guess the characters
_
_
_
_
_
_
_

wrong
you have 11 more guesses
_
_
_
_
_
_
_

wrong
you have 10 more guesses
_
_
_
_
_
_
_

r _
_
_
r _
_

wrong
you have 9 more guesses
r _
_
_
r _
_

wrong
you have 8 more guesses
r _
_
_
r _
_

r e _
e r _
e
r e v e r _
e
r e v e r s e you won
The word is : reverse
```

Project 3

In [ ]:
```
"Hangman Game in Python"
```

This is a simple Hangman game using Python programming language. Beginners can use this as a small project to boost their programming skills and understanding logic.

The Hangman program randomly selects a secret word from a list of secret words. The random module will provide this ability, so line 1 in program imports it. The Game: Here, a random word (a fruit name) is picked up from our collection and the player gets limited chances to win the game. When a letter in that word is guessed correctly, that letter position in the word is made visible. In this way, all letters of the word are to be guessed before all the chances are over. For convenience, we have given length of word + 2 chances. For example, word to be guessed is mango, then user gets 5 + 2 = 7 chances, as mango is a five-letter word.

In [38]:
```
# Python Program to illustrate
# Hangman Game
import random
from collections import Counter

someWords = '''apple banana mango strawberry
orange grape pineapple apricot lemon coconut watermelon
```

```
cherry papaya berry peach lychee muskmelon'''

someWords = someWords.split(' ')
# randomly choose a secret word from our "someWords" LIST.
word = random.choice(someWords)

if __name__ == '__main__':
    print('Guess the word! HINT: word is a name of a fruit')

    for i in word:
        # For printing the empty spaces for letters of the word
        print('_', end=' ')
    print()

    playing = True
    # List for storing the letters guessed by the player
    letterGuessed = ''
    chances = len(word) + 2
    correct = 0
    flag = 0
    try:
        while (chances != 0) and flag == 0:  # Flag is updated when the word is correctly guessed
            print()
            chances -= 1

            try:
                guess = str(input('Enter a letter to guess: '))
            except:
                print('Enter only a letter!')
                continue

            # Validation of the guess
            if not guess.isalpha():
                print('Enter only a LETTER')
                continue
            elif len(guess) > 1:
                print('Enter only a SINGLE letter')
                continue
            elif guess in letterGuessed:
                print('You have already guessed that letter')
                continue

            # If letter is guessed correctly
            if guess in word:
                # k stores the number of times the guessed letter occurs in the word
                k = word.count(guess)
                for _ in range(k):
                    letterGuessed += guess  # The guessed letter is added as many times as it occurs

            # Print the word
            for char in word:
                if char in letterGuessed and (Counter(letterGuessed) != Counter(word)):
                    print(char, end=' ')
                    correct += 1
                # If user has guessed all the letters
                # Once the correct word is guessed fully,
                elif (Counter(letterGuessed) == Counter(word)):
                    # the game ends, even if chances remain
                    print("The word is: ", end=' ')
                    print(word)
                    flag = 1
                    print('Congratulations, You won!')
                    break  # To break out of the for loop
                    break  # To break out of the while loop
                else:
                    print('_', end=' ')

        # If user has used all of his chances
        if chances <= 0 and (Counter(letterGuessed) != Counter(word)):
            print()
            print('You lost! Try again..')
            print('The word was {}'.format(word))

    except KeyboardInterrupt:
        print()
        print('Bye! Try again.')
        exit()
```

```
Guess the word! HINT: word is a name of a fruit
_ _ _ _ _ _ _ _ _

_ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _
p _ _ _ _ p p _ _
```

```
p i _ _ _ p p _ _
p i _ _ a p p _ _
p i _ _ a p p l _
p i _ e a p p l e
The word is:  pineapple
Congratulations, You won!
```

Project 4

In [ ]:
```
"Number game in Python"
```

21, Bagram, or Twenty plus one is a game which progresses by counting up 1 to 21, with the player who calls “21” is eliminated. It can be played between any number of players. Implementation This is a simple 21 number game using Python programming language. The game illustrated here is between the player and the computer. There can be many variations in the game.

The player can choose to start first or second. The list of numbers is shown before the Player takes his turn so that it becomes convenient. If consecutive numbers are not given in input then the player is automatically disqualified. The player loses if he gets the chance to call 21 and wins otherwise. Winning against the computer can be possible by choosing to play second. The strategy is to call numbers till the multiple of 4 which would eventually lead to 21 on computer hence making the player the winner.

In [ ]:
```
# Python code to play 21 Number game

# returns the nearest multiple to 4
def nearestMultiple(num):
    if num >= 4:
        near = num + (4 - (num % 4))
    else:
        near = 4
    return near

def lose1():
    print ("\n\nYOU LOSE !")
    print("Better luck next time !")
    exit(0)

# checks whether the numbers are consecutive
def check(xyz):
    i = 1
    while i<len(xyz):
        if (xyz[i]-xyz[i-1])!= 1:
            return False
        i = i + 1
    return True

# starts the game
def start1():
    xyz = []
    last = 0
    while True:
        print ("Enter 'F' to take the first chance.")
        print("Enter 'S' to take the second chance.")
        chance = input('> ')

        # player takes the first chance
        if chance == "F":
            while True:
                if last == 20:
                    lose1()
                else:
                    print ("\nYour Turn.")
                    print ("\nHow many numbers do you wish to enter?")
                    inp = int(input('> '))

                    if inp > 0 and inp <= 3:
                        comp = 4 - inp
                    else:
                        print ("Wrong input. You are disqualified from the game.")
                        lose1()

                    i, j = 1, 1

                    print ("Now enter the values")
                    while i <= inp:
                        a = input('> ')
                        a = int(a)
                        xyz.append(a)
                        i = i + 1

                    # store the last element of xyz.
                    last = xyz[-1]
```

```
                    # checks whether the input
                    # numbers are consecutive
                    if check(xyz) == True:
                        if last == 21:
                            lose1()

                        else:
                            #"Computer's turn."
                            while j <= comp:
                                xyz.append(last + j)
                                j = j + 1
                            print ("Order of inputs after computer's turn is: ")
                            print (xyz)
                            last = xyz[-1]
                    else:
                        print ("\nYou did not input consecutive integers.")
                        lose1()

        # player takes the second chance
        elif chance == "S":
            comp = 1
            last = 0
            while last < 20:
                #"Computer's turn"
                j = 1
                while j <= comp:
                    xyz.append(last + j)
                    j = j + 1
                print ("Order of inputs after computer's turn is:")
                print (xyz)
                if xyz[-1] == 20:
                    lose1()

                else:
                    print ("\nYour turn.")
                    print ("\nHow many numbers do you wish to enter?")
                    inp = input('> ')
                    inp = int(inp)
                    i = 1
                    print ("Enter your values")
                    while i <= inp:
                        xyz.append(int(input('> ')))
                        i = i + 1
                    last = xyz[-1]
                    if check(xyz) == True:
                        # print (xyz)
                        near = nearestMultiple(last)
                        comp = near - last
                        if comp == 4:
                            comp = 3
                        else:
                            comp = comp
                    else:
                        # if inputs are not consecutive
                        # automatically disqualified
                        print ("\nYou did not input consecutive integers.")
                        # print ("You are disqualified from the game.")
                        lose1()
            print ("\n\nCONGRATULATIONS !!!")
            print ("YOU WON !")
            exit(0)

        else:
            print ("wrong choice")


game = True
while game == True:
        print ("Player 2 is Computer.")
        print("Do you want to play the 21 number game? (Yes / No)")
        ans = input('> ')
        if ans =='Yes':
            start1()
        else:
            print ("Do you want quit the game?(yes / no)")
            nex = input('> ')
            if nex == "yes":
                print ("You are quitting the game...")
                exit(0)
            elif nex == "no":
                print ("Continuing...")
            else:
```

```
                print ("Wrong choice")
```

```
Player 2 is Computer.
Do you want to play the 21 number game? (Yes / No)
Enter 'F' to take the first chance.
Enter 'S' to take the second chance.
Order of inputs after computer's turn is:
[1]

Your turn.

How many numbers do you wish to enter?
Enter your values
Order of inputs after computer's turn is:
[1, 2, 3, 4, 5, 6, 7]

Your turn.

How many numbers do you wish to enter?
Enter your values
Order of inputs after computer's turn is:
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]

Your turn.

How many numbers do you wish to enter?
Enter your values
Order of inputs after computer's turn is:
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]

Your turn.

How many numbers do you wish to enter?
Enter your values
Order of inputs after computer's turn is:
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]

Your turn.

How many numbers do you wish to enter?
Enter your values

CONGRATULATIONS !!!
YOU WON !
Enter 'F' to take the first chance.
Enter 'S' to take the second chance.
```

Project 5

In [ ]:
```
                "Mastermind Game using Python"
```

Rules of the game
Two players play the game against each other; let's assume Player 1 and Player 2.

Player 1 plays first by setting a multi-digit number.
Player 2 now tries his first attempt at guessing the number.
If Player 2 succeeds in his first attempt (despite odds which are highly unlikely) he wins the game and is crowned Mastermind! If not, then Player 1 hints by revealing which digits or numbers Player 2 got correct.
The game continues till Player 2 eventually is able to guess the number entirely.
Now, Player 2 gets to set the number and Player 1 plays the part of guessing the number.
If Player 1 is able to guess the number within a lesser number of tries than Player 2 took, then Player 1 wins the game and is crowned Mastermind.
If not, then Player 2 wins the game.
The real game, however, has proved aesthetics since the numbers are represented by color-coded buttons.

In [19]:
```
import random

# Generate a random 4-digit number
num = random.randrange(1000, 1010)

# Input from the user
n = int(input("Guess the 4-digit number: "))

# If the user guesses the number on the first try
if n == num:
    print("Great! You guessed the number in just 1 try! You're a Mastermind!")
else:
```

```
    # Initialize counter for number of tries
    ctr = 0

    # Loop until the correct number is guessed
    while n != num:
        ctr += 1
        count = 0

        # Convert both input and generated numbers to strings to compare digits
        n_str = str(n)
        num_str = str(num)

        # Check for correct digits
        for i in range(4):
            if n_str[i] == num_str[i]:
                count += 1

        # Provide feedback
        if count > 0:
            print(f"Not quite the number. But you did get {count} digit(s) correct!")
        else:
            print("None of the numbers in your input match.")

        # Ask for the next guess
        n = int(input("Enter your next guess: "))

    # Final message when the correct number is guessed
    ctr += 1  # Increment for the successful guess
    print("You've become a Mastermind!")
    print(f"It took you only {ctr} tries.")
```

```
Not quite the number. But you did get 2 digit(s) correct!
Not quite the number. But you did get 2 digit(s) correct!
Not quite the number. But you did get 3 digit(s) correct!
Not quite the number. But you did get 3 digit(s) correct!
You've become a Mastermind!
It took you only 5 tries.
```

In [30]:

```
# import random
num=random.randrange(1000,1100)
n=int(input("Enter a 4 digit number:"))

if n==num:
    print("Great ! YOU gueesed within just 1 try!")
else:
    ctr=0
    while (n!=num):
        ctr+=1
        count=0
        n_str=str(n)
        num_str=str(num)
        correct=[]
        for i in range(4):
            if n_str[i]==num_str[i]:
                count+=1
                correct.append(n_str[i])
            else:
                continue
        if count<4  and count!=0:
            print(f"Not quite the number .But you did get {count} digits correct")
            print("Also these numbers in your input were correct.")
            for k in correct:
                print(k,end=" ")
            print('\n')
            print('\n')
            n=int(input("Enter your next choice of numbers:"))
        elif count==0:
            print("Noe of the numbers in your input match ")
            n=int(input("Enter your next choice numbers:"))
    if n==num:
        print("YOu've become mastermind !")
        print(f"It took you only {ctr} tries")
```

```
Not quite the number .But you did get 1 digits correct
Also these numbers in your input were correct.
1
```

```
Not quite the number .But you did get 2 digits correct
Also these numbers in your input were correct.

1 0



Not quite the number .But you did get 3 digits correct
Also these numbers in your input were correct.
1 0 8



Not quite the number .But you did get 3 digits correct
Also these numbers in your input were correct.
1 0 8



Not quite the number .But you did get 3 digits correct
Also these numbers in your input were correct.
1 0 8



Not quite the number .But you did get 3 digits correct
Also these numbers in your input were correct.
1 0 8



Not quite the number .But you did get 3 digits correct
Also these numbers in your input were correct.
1 0 8



YOu've become mastermind !
It took you only 7 tries
```

Project 5

In [ ]:
```
"2048 Game in Python"
```

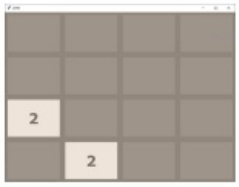
In [35]:
```
from PIL import Image
Image.open('box.png')
```

Out[35]:

In this article we will look python code and logic to design a 2048 game you have played very often in your smartphone. If you are not familiar with the game, it is highly recommended to first play the game so that you can understand the basic functioning of it.

How to play 2048 :

1. There is a 4*4 grid which can be filled with any number. Initially two random cells are filled with 2 in it. Rest cells are empty.

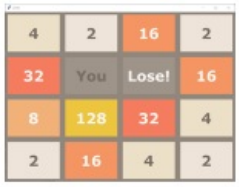


2. we have to press any one of four keys to move up, down, left, or right. When we press any key, the elements of the cell move in that direction such that if any two identical numbers are contained in that particular row (in case of moving left or right) or column (in case of moving up and down) they get add up and extreme cell in that direction fill itself with that number and rest cells goes empty again.

3. After this grid compression any random empty cell gets itself filled with 2.

4. Following the above process we have to double the elements by adding up and make 2048 in any of the cell. If we are able to do that we wins.

5. But if during the game there is no empty cell left to be filled with a new 2, then the game goes over.



In [39]:
```
Image.open('exa.png')
```

Out[39]:

Example :

```
Commands are as follows :
'W' or 'w' : Move Up
'S' or 's' : Move Down
'A' or 'a' : Move Left
'D' or 'd' : Move Right
[0, 0, 0, 0]
[0, 0, 0, 0]
[0, 0, 0, 0]
[0, 0, 2, 0]
Press the command : a
GAME NOT OVER
[0, 0, 0, 2]
[0, 0, 0, 0]
[0, 0, 0, 0]
[2, 0, 0, 0]
Press the command : s
GAME NOT OVER
[0, 0, 0, 0]
[0, 0, 0, 0]
[0, 0, 2, 0]
[2, 0, 0, 2]
Press the command : d
GAME NOT OVER
[0, 0, 0, 0]
[0, 0, 0, 0]
[2, 0, 0, 2]
[0, 0, 0, 4]
Press the command : a
GAME NOT OVER
[0, 2, 0, 0]
[0, 0, 0, 0]
[4, 0, 0, 0]
[4, 0, 0, 0]
Press the command : s
GAME NOT OVER
[0, 0, 0, 0]
[0, 0, 0, 0]
[0, 0, 0, 0]
[8, 2, 0, 2]
.
.
.
And the series of input output will go on till we lose or win!
```

In [102…

```
import random
class logic:
    def start_game():
        mat=[]
        for i in range(4):
            mat.append([0]*4)
        print("Commands are as folows :")
        print("'W' or 'w' : Move UP")
        print("'S' or 's' : Move Down")
        print("'A' or 'a' : Move Left")
        print("'D' or 'd' : Move Right")
        add_new_2(mat)
        return mat


    def add_new_2(mat):
        r=random.randint(0,3)
        c=random.randint(0,3)
        while (mat[r][c]!=0):
            r=random.randint(0,3)
            c=random.randint(0,3)
        mat[r][c]=2
    def get_current_state(mat):
        for i in range(4):
            for j in range(4):
                if mat[i][j]==2048:
                    return 'Won'
        for i in range(4):
            for j in range(4):
                if (mat[i][j]==0):
                    return 'GAME NOT OVER'

        for i in range(3):
            for j in range(3):
                if( mat[i][j]==mat[i+1][j] or mat[i][j]== mat[i][j+1]):
                    return 'GAME NOT OVER'
        for j in range(3):
            if mat[3][j]==mat[3][j+1]:
                return 'GAME NOT OVER'
        for i in range(3):
            if mat[i][3]==mat[i+1][3]:
                return 'GAME NOT OVER'
        return 'lost'

    def compress(mat):
        changed=False
        new_mat=[]
        for i in range(4):
            new_mat.append([0]*4)
        for i in range(4):
            pos=0
            for j in range(4):
                if mat[i][j]!=0:
                    new_mat[i][pos]=mat[i][j]
                    if (j!=pos):
                        changed=True
                    pos+=1

        return new_mat,changed

    def merge(mat):
        changed=False
        for i in range(4):
            for j in range(3):
                if mat[i][j]==mat[i][j+1] and mat[i][j]!=0:
                    mat[i][j]=mat[i][j]*2
                    mat[i][j+1]=0
                    changed=True

        return mat,changed

    def reverse(mat):
        new_mat=[]
        for i in range(4):
            new_mat.append([])
            for j in range(4):
                 new_mat[i].append(mat[i][3-j])
        return new_mat

    def transpose(mat):
        new_mat=[]
        for i in range(4):
            new_mat.append([])
```

```
            for j in range(4):
                new_mat[i].append(mat[j][i])
        return new_mat

    def move_left(grid):

        new_grid,changed1=compress(grid)
        new_grid,changed2=merge(new_grid)
        changed=changed1 or changed2
        new_grid,temp=compress(new_grid)
        return new_grid,changed

    def move_right(grid):
        new_grid=reverse(grid)
        new_grid,changed=move_left(new_grid)
        new_grid=reverse(new_grid)
        return new_grid,changed

    def move_up(grid):
        new_grid=transpose(grid)
        new_grid,changed=move_left(new_grid)
        new_grid=transpose(new_grid)
        return new_grid,changed

    def move_down(grid):
        new_grid=transpose(grid)
        new_grid,changed=move_right(new_grid)
        new_grid=transpose(new_grid)
        return new_grid,changed


logic=logic()
```

In [1]:
```
import logic
if __name__=='__main__':
    mat=logic.start_game()

while (True):
    x=input("Press the command")
    if (x=='W' or x=='w'):
        mat,flag=logic.move_up(mat)
        status=logic.get_current_state(mat)
        if (status=='GAME NOT OVER'):
            logic.add_new_2(mat)
        else:
            break
    elif(x=='A' or x=='a'):
        mat,flag=logic.move_left(mat)
        status=logic.get_current_state(mat)
        print(status)
        if(status== 'GAME NOT OVER'):
            logic.add_new_2(mat)
        else:
            break

    else:
        print("Invalid key Passed")
    print(mat)
```

```
Commands are as follows :
'W' or 'w' : Move Up
'S' or 's' : Move Down
'A' or 'a' : Move Left
'D' or 'd' : Move Right
[[0, 0, 2, 2], [0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
Invalid key Passed
[[0, 0, 2, 2], [0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
GAME NOT OVER
[[4, 0, 0, 0], [0, 0, 2, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
GAME NOT OVER
[[4, 0, 0, 0], [2, 0, 0, 0], [0, 0, 0, 0], [2, 0, 0, 0]]
GAME NOT OVER
[[4, 0, 0, 0], [2, 0, 0, 2], [0, 0, 0, 0], [2, 0, 0, 0]]
GAME NOT OVER
[[4, 0, 0, 0], [4, 0, 0, 0], [0, 0, 0, 0], [2, 0, 2, 0]]
```

```
GAME NOT OVER

[[4, 0, 0, 0], [4, 0, 0, 0], [0, 0, 0, 0], [4, 2, 0, 0]]
Invalid key Passed
[[4, 0, 0, 0], [4, 0, 0, 0], [0, 0, 0, 0], [4, 2, 0, 0]]
[[8, 2, 2, 0], [4, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
GAME NOT OVER
[[8, 4, 2, 0], [4, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
GAME NOT OVER
[[8, 4, 2, 0], [4, 0, 0, 0], [2, 0, 0, 0], [0, 0, 0, 0]]
[[8, 4, 2, 0], [4, 0, 0, 0], [2, 0, 2, 0], [0, 0, 0, 0]]
GAME NOT OVER
[[8, 4, 2, 0], [4, 0, 0, 0], [4, 2, 0, 0], [0, 0, 0, 0]]
[[8, 4, 2, 0], [8, 2, 0, 0], [0, 0, 0, 2], [0, 0, 0, 0]]
GAME NOT OVER
[[8, 4, 2, 0], [8, 2, 0, 0], [2, 0, 0, 2], [0, 0, 0, 0]]
Invalid key Passed
[[8, 4, 2, 0], [8, 2, 0, 0], [2, 0, 0, 2], [0, 0, 0, 0]]
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[1], line 6
      3     mat=logic.start_game()
      5 while (True):
----> 6     x=input("Press the command")
      7     if (x=='W' or x=='w'):
      8         mat,flag=logic.move_up(mat)

File ~\anaconda3\Lib\site-packages\ipykernel\kernelbase.py:1262, in Kernel.raw_input(self, prompt)
   1260     msg = "raw_input was called, but this frontend does not support input requests."
   1261     raise StdinNotImplementedError(msg)
-> 1262 return self._input_request(
   1263     str(prompt),
   1264     self._parent_ident["shell"],
   1265     self.get_parent("shell"),
   1266     password=False,
   1267 )

File ~\anaconda3\Lib\site-packages\ipykernel\kernelbase.py:1305, in Kernel._input_request(self, prompt, ident, p
arent, password)
   1302 except KeyboardInterrupt:
   1303     # re-raise KeyboardInterrupt, to truncate traceback
   1304     msg = "Interrupted by user"
-> 1305     raise KeyboardInterrupt(msg) from None
   1306 except Exception:
   1307     self.log.warning("Invalid Message:", exc_info=True)

KeyboardInterrupt: Interrupted by user
```

Project 6

In [ ]:
```
        "Python Program to implement simple 'FLAMES' game"
```

In [ ]:
```
FLAMES is a popular game named after the acronym: Friends, Lovers, Affectionate, Marriage, Enemies, Sibling. Th

There are two steps in this game:

Take the two names.
Remove the common characters with their respective common occurrences.
Get the count of the characters that are left .
Take FLAMES letters as [“F”, “L”, “A”, “M”, “E”, “S”]
Start removing letter using the count we got.
The letter which last the process is the result.
```

In [34]:
```
#funtion for removing common characters with their respective occureneces

def remove_match_char(list1,list2):
    for i in range(len(list1)):
        for j in range(len(list2)):
            #if common character is found
            #then remove that character
            #and return list of concatenated list with True flag
            if list1[i]==list2[j]:
                c=list1[i]
                #remove character from the list
                list1.remove(c)
                list2.remove(c)
                #concatenation of two list elements with * is act as border mark here
                list3=list1+["*"]+list2
                #return the concatenated list with True Flag
                return [list3,True]
    #no common characters are found
```

```
    # return the concatenated list with False flag
    list3=list1+["*"]+list2
    return [list3,False]


#Driver code
if __name__=="__main__":
    #take first name
    p1=input("Player 1 name:")
    #converted into lower case
    p1.lower()
    #replace any string with empty string
    p1.replace(" ","")
    #make a list of letters or characters
    p1_list=list(p1)

    #take second name
    p2=input("Player 2 name:")
    p2.lower()
    p2.replace(" ","")
    p2_list=list(p2)
    #taking a flag as True initially
    proceed=True
    #keep calling remove_match_char function untill common characters is found or keep looping untill proceed f
    while proceed:
        ret_list=remove_match_char(p1_list,p2_list)
        con_list=ret_list[0]
        proceed=ret_list[1]
        star_index=con_list.index("*")
        p1_list=con_list[:star_index]
        p2_list=con_list[star_index+1:]
    count=len(p1_list)+len(p2_list)
    result=["Friends","Love","Affection","Marriage","Enemy","Sister"]
    while len(result)>1:
        split_index=(count%len(result)-1)
        if split_index>=0:
            right=result[split_index+1:]
            left=result[:split_index]
            result=right+left
        else:
            result=result[:len(result)-1]
    #print final result

    print("Relationship status:",result[0])
```

```
Relationship status: Enemy
```

Project 7

In [ ]:
```
             "Python Pokemon Training Game"
```

You are a Pokémon trainer. Each Pokémon has its own power, described by a positive integer value. As you travel, you watch Pokémon and you catch each of them. After each catch, you have to display maximum and minimum powers of Pokémon caught so far. You must have linear time complexity. So sorting won't help here. Try having minimum extra space complexity.

In [ ]:
```
Examples:
Suppose you catch Pokémon of powers 3 8 9 7. Then the output should be
3 3
3 8
3 9
3 9
```

In [13]:
```
#python code to train pokeman
powers=[3,8,9,7]
mini,maxi=0,0

for power in powers :
    if mini==0 and maxi==0:
        mini,maxi=powers[0],powers[0]
        print(mini,maxi)
    else:
        mini=min(mini,power)
        maxi=max(maxi,power)
        print(mini,maxi)
```

```
3 3
3 8
3 9
3 9
```

"Project 8"

In [ ]:
```
          "Python program to implement rock paper scissior game"
```

Let's create a simple command-line Rock-Paper-Scissor game without using any external game libraries like PyGame. In this game, the user gets the first chance to pick the option between Rock, paper, and scissors. After the computer select from the remaining two choices(randomly), the winner is decided as per the rules.

In [ ]:
```
Winning Rules as follows:
Rock vs paper-> paper wins
Rock vs scissor-> Rock wins
paper vs scissor-> scissor wins.
```

In [5]:
```
import random
#print multiline instruction
print('Winnning rules of the game ROCK PAPER SCISSORS are:\n'
      + "Rock vs Paper -> Paper wins \n"
      + "Rock vs Scissors -> Rock wins \n"
      + "Paper vs Scissors -> Scissors wins \n")
while True:
    print("Enter your choice \n 1 - Rock \n 2 - Paper \n 3 - Scissors \n")
    #take the input from the user
    choice=int(input("Enter your choice: "))
    #looping untill user enters a correct choice valid input
    while choice>3 or choice<1:
        choice=int(input("Please enter a valid choice ☺ :"))
        #Initialise value of the choice_name variable corresponding to the choice value
    if choice==1:
        choice_name='Rock'
    elif choice==2:
        choice_name='Paper'
    else:
        choice_name='Scissors'
    print('User choice is: ',choice_name)
    print("Now it's computer turn...")
    comp_choice=random.randint(1,3)
    #Initialize value of comp_choice_name varibale corresponding to the choicec value.
    if comp_choice==1:
        comp_choice_name='Rock'
    elif comp_choice==2:
        comp_choice_name='Paper'
    else:
        comp_choice_name='Scissors'
    print("Computer choice is :",comp_choice_name)
    print(choice_name,"vs",comp_choice_name)
    #Determine the winnner
    if choice==comp_choice:
        result='DRAW'
    elif (choice==1 and comp_choice==2) or (comp_choice==1 and choice==2):
        result='Paper'
    elif (choice==1 and comp_choice==3) or (comp_choice==1 and choice==3):
        result='Rock'
    elif (choice==2 and comp_choice==3) or (comp_choice==2 and choice==3):
        result='Scissors'
    #print the result
    if result=='DRAW':
        print("<== It's a tie ==>")
    elif result==choice_name:
        print("<== YOU wins ==>")
    else:
        print("<== computer wins ==>")


    #ask player wants to play again
    print("Do you want to play again? (Y/N)")
    ans=input().lower()
    if ans=='n':
        break

print("Thanks for playing")
```

```
Winnning rules of the game ROCK PAPER SCISSORS are:
Rock vs Paper -> Paper wins
Rock vs Scissors -> Rock wins
Paper vs Scissors -> Scissors wins

Enter your choice
 1 - Rock
 2 - Paper
 3 - Scissors

User choice is:  Paper
Now it's computer turn...
Computer choice is : Scissors
Paper vs Scissors
<== computer wins ==>
Do you want to play again? (Y/N)
Thanks for playing
```

"Project 9"

In [ ]:
```
"Taking screesnshots using Pyscreenshot in Python"
```

Python offers multiple libraries to ease our work. Here we will learn how to take a screenshot using Python. Python provides a module called pyscreenshot for this task. It is only a pure Python wrapper, a thin layer over existing backends. Performance and interactivity are not important for this library.

Installation

Install the package pyscreeshoat using the below command.

In [28]:
```
pip install pyscreenshot
```

```
Collecting pyscreenshot
  Downloading pyscreenshot-3.1-py3-none-any.whl.metadata (1.4 kB)
Collecting EasyProcess (from pyscreenshot)
  Downloading EasyProcess-1.1-py3-none-any.whl.metadata (855 bytes)
Collecting entrypoint2 (from pyscreenshot)
  Downloading entrypoint2-1.1-py2.py3-none-any.whl.metadata (1.0 kB)
Collecting mss (from pyscreenshot)
  Downloading mss-9.0.2-py3-none-any.whl.metadata (6.1 kB)
Downloading pyscreenshot-3.1-py3-none-any.whl (28 kB)
Downloading EasyProcess-1.1-py3-none-any.whl (8.7 kB)
Downloading entrypoint2-1.1-py2.py3-none-any.whl (9.9 kB)
Downloading mss-9.0.2-py3-none-any.whl (23 kB)
Installing collected packages: entrypoint2, EasyProcess, mss, pyscreenshot
Successfully installed EasyProcess-1.1 entrypoint2-1.1 mss-9.0.2 pyscreenshot-3.1
Note: you may need to restart the kernel to use updated packages.
```

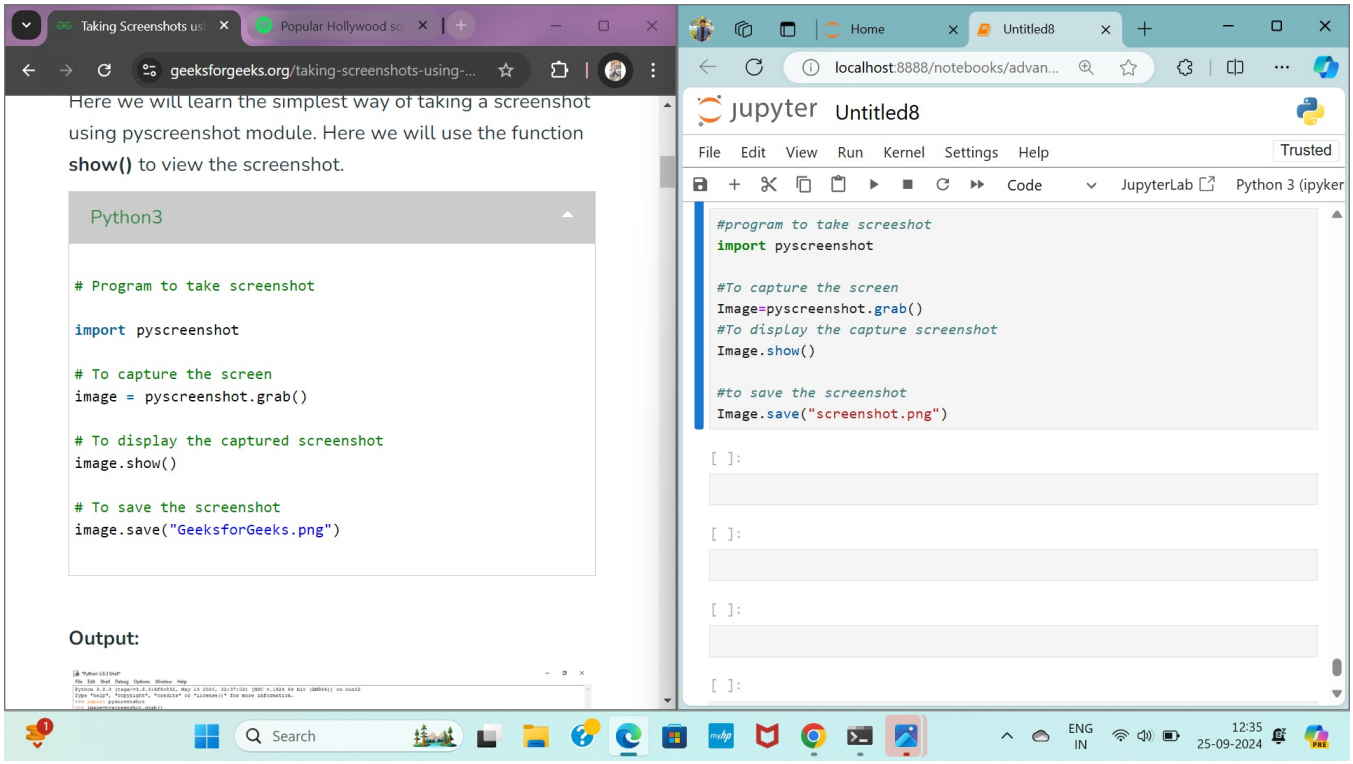
"capturing Full Screen"

Here we will learn the simplest way of taking a screenhot using pyscreenshot module.Here we will use the funtion "show()" to view the screenshot

In [46]:
```
#program to take screeshot
import pyscreenshot

#To capture the screen
Image=pyscreenshot.grab()
#To display the capture screenshot
Image.show()

#to save the screenshot
Image.save("screenshot.png")
```

In [44]:
```
from PIL import Image
Image.open('screenshot.png')
```

Out[44]:



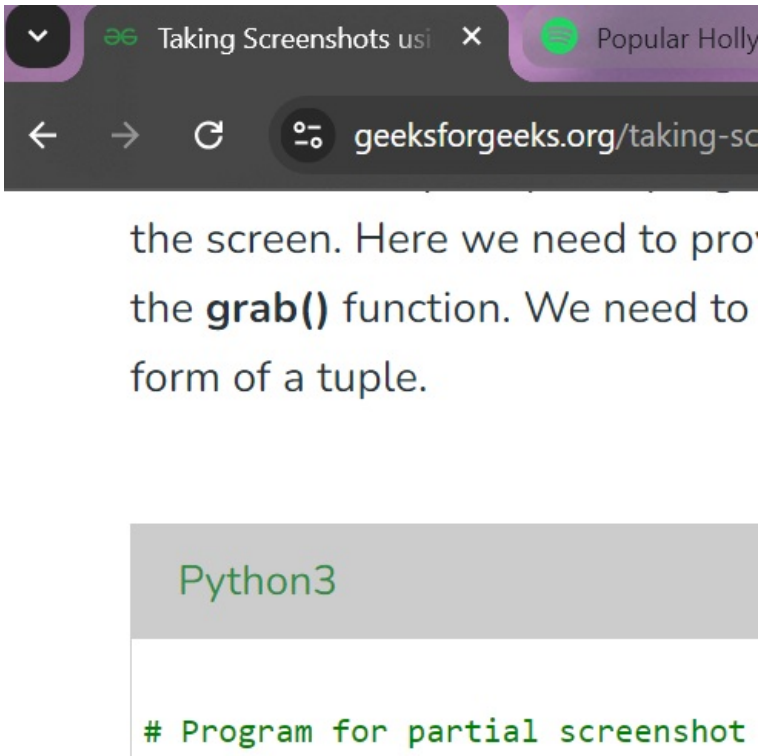
In [ ]:
```
"Capturing a part of screen"
```

Here is the simple Python program to capture the part of the screen.Here we need to provide the pixel positions in the "grab()" function.we need to pass the coordinates in the form of a tuple.

In [49]:
```
import pyscreenshot

# im=pyscreenshot.grab(bbox=(x1,x2,y1,y2))
image=pyscreenshot.grab(bbox=(10,10,500,500))

image.show()
image.save('screenshot1.png')
```

In [53]:
```
from PIL import Image
Image.open('screenshot1.png')
```

Out[53]:



"Project 11"

In [41]:
```
pip install pynput
```

```
Collecting pynput
  Downloading pynput-1.7.7-py2.py3-none-any.whl.metadata (31 kB)
Requirement already satisfied: six in c:\users\gadam\anaconda3\lib\site-packages (from pynput) (1.16.0)
Downloading pynput-1.7.7-py2.py3-none-any.whl (90 kB)
   ---------------------------------------- 0.0/90.2 kB ? eta -:--:--
   ---------------------------------- --- 81.9/90.2 kB 4.5 MB/s eta 0:00:01
   ---------------------------------------- 90.2/90.2 kB 2.5 MB/s eta 0:00:00
Installing collected packages: pynput
Successfully installed pynput-1.7.7
Note: you may need to restart the kernel to use updated packages.
```

In [30]:
```
pip install bs4
```

```
Collecting bs4
  Downloading bs4-0.0.2-py2.py3-none-any.whl.metadata (411 bytes)
Requirement already satisfied: beautifulsoup4 in c:\users\gadam\anaconda3\lib\site-packages (from bs4) (4.12.3)
Requirement already satisfied: soupsieve>1.2 in c:\users\gadam\anaconda3\lib\site-packages (from beautifulsoup4-
>bs4) (2.5)
Downloading bs4-0.0.2-py2.py3-none-any.whl (1.2 kB)
Installing collected packages: bs4
Successfully installed bs4-0.0.2
Note: you may need to restart the kernel to use updated packages.
```

In [32]:
```
pip install win10toast
```

```
Collecting win10toast
  Downloading win10toast-0.9-py2.py3-none-any.whl.metadata (2.1 kB)
Collecting pypiwin32 (from win10toast)
  Downloading pypiwin32-223-py3-none-any.whl.metadata (236 bytes)
Requirement already satisfied: setuptools in c:\users\gadam\anaconda3\lib\site-packages (from win10toast) (69.5.
1)
Requirement already satisfied: pywin32>=223 in c:\users\gadam\anaconda3\lib\site-packages (from pypiwin32->win10
toast) (305.1)
Downloading win10toast-0.9-py2.py3-none-any.whl (21 kB)
Downloading pypiwin32-223-py3-none-any.whl (1.7 kB)
Installing collected packages: pypiwin32, win10toast
Successfully installed pypiwin32-223 win10toast-0.9
Note: you may need to restart the kernel to use updated packages.
```

In [68]:
```
pip install notify2
```

```
Collecting notify2Note: you may need to restart the kernel to use updated packages.

  Downloading notify2-0.3.1-py2.py3-none-any.whl.metadata (2.2 kB)
Downloading notify2-0.3.1-py2.py3-none-any.whl (8.0 kB)
Installing collected packages: notify2
Successfully installed notify2-0.3.1
```

In [20]:
```
pip install requests
```

```
Requirement already satisfied: requests in c:\users\gadam\anaconda3\lib\site-packages (2.32.2)Note: you may need
to restart the kernel to use updated packages.

Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\gadam\anaconda3\lib\site-packages (from requ
ests) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\gadam\anaconda3\lib\site-packages (from requests) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\gadam\anaconda3\lib\site-packages (from requests)
(1.26.19)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\gadam\anaconda3\lib\site-packages (from requests)
(2024.6.2)
```

In [ ]:
```
                            "Cows and Bulls game"
```

Cows and Bulls is a pen and paper code-breaking game usually played between 2 players. In this, a player tries to guess a secret code number chosen by the second player. The rules are as follows:

A player will create a secret code, usually a 4-digit number. This number should have no repeated digits. Another player makes a guess (4 digit number) to crack the secret number. Upon making a guess, 2 hints will be provided- Cows and Bulls. Bulls indicate the number of correct digits in the correct position and cows indicates the number of correct digits in the wrong position. For example, if the secret code is 1234 and the guessed number is 1246 then we have 2 BULLS (for the exact matches of digits 1 and 2) and 1 COW (for the match of digit 4 in the wrong position) The player keeps on guessing until the secret code is cracked. The player who guesses in the minimum number of tries wins.

In [2]:
```
# Import required module
import random

# Returns list of digits
# of a number
def getDigits(num):
    return [int(i) for i in str(num)]


# Returns True if number has
```

```
# no duplicate digits
# otherwise False
def noDuplicates(num):
    num_li = getDigits(num)
    if len(num_li) == len(set(num_li)):
        return True
    else:
        return False


# Generates a 4 digit number
# with no repeated digits
def generateNum():
    while True:
        num = random.randint(1000,9999)
        if noDuplicates(num):
            return num


# Returns common digits with exact
# matches (bulls) and the common
# digits in wrong position (cows)
def numOfBullsCows(num,guess):
    bull_cow = [0,0]
    num_li = getDigits(num)
    guess_li = getDigits(guess)

    for i,j in zip(num_li,guess_li):

        # common digit present
        if j in num_li:

            # common digit exact match
            if j == i:
                bull_cow[0] += 1

            # common digit match but in wrong position
            else:
                bull_cow[1] += 1

    return bull_cow


# Secret Code
num = generateNum()
tries =int(input('Enter number of tries: '))

# Play game until correct guess
# or till no tries left
while tries > 0:
    guess = int(input("Enter your guess: "))

    if not noDuplicates(guess):
        print("Number should not have repeated digits. Try again.")
        continue
    if guess < 1000 or guess > 9999:
        print("Enter 4 digit number only. Try again.")
        continue

    bull_cow = numOfBullsCows(num,guess)
    print(f"{bull_cow[0]} bulls, {bull_cow[1]} cows")
    tries -=1

    if bull_cow[0] == 4:
        print("You guessed right!")
        break
else:
    print(f"You ran out of tries. Number was {num}")
```

```
1 bulls, 0 cows
1 bulls, 1 cows
2 bulls, 1 cows
3 bulls, 0 cows
2 bulls, 0 cows
2 bulls, 0 cows
Number should not have repeated digits. Try again.
2 bulls, 1 cows
3 bulls, 0 cows
4 bulls, 0 cows
You guessed right!
```

In [4]:
```
pip install openpyxl
```

```
Requirement already satisfied: openpyxl in c:\users\gadam\anaconda3\lib\site-packages (3.1.2)
Requirement already satisfied: et-xmlfile in c:\users\gadam\anaconda3\lib\site-packages (from openpyxl) (1.1.0)
Note: you may need to restart the kernel to use updated packages.
```

"Project 11"

In [ ]:
```
"Python Higher - Lower Game in Python"
```

In [11]:
```
pip install replit
```

```
Collecting replit
  Downloading replit-4.1.0-py3-none-any.whl.metadata (2.5 kB)
Requirement already satisfied: Flask>=2.0.0 in c:\users\gadam\anaconda3\lib\site-packages (from replit) (3.0.3)
Requirement already satisfied: Werkzeug<4,>=2 in c:\users\gadam\anaconda3\lib\site-packages (from replit) (3.0.3
)
Requirement already satisfied: aiohttp>=3.6.2 in c:\users\gadam\anaconda3\lib\site-packages (from replit) (3.9.5
)
Collecting aiohttp-retry<3.0.0,>=2.8.3 (from replit)
  Downloading aiohttp_retry-2.8.3-py3-none-any.whl.metadata (8.9 kB)
Collecting protobuf<5.0.0,>=4.21.8 (from replit)
  Downloading protobuf-4.25.5-cp310-abi3-win_amd64.whl.metadata (541 bytes)
Collecting pyseto<2.0.0,>=1.6.11 (from replit)
  Downloading pyseto-1.7.9-py3-none-any.whl.metadata (17 kB)
Requirement already satisfied: requests<3.0.0,>=2.25.1 in c:\users\gadam\anaconda3\lib\site-packages (from repli
t) (2.32.2)
Requirement already satisfied: typing_extensions<5,>=4 in c:\users\gadam\anaconda3\lib\site-packages (from repli
t) (4.11.0)
Requirement already satisfied: urllib3<3,>=1.26 in c:\users\gadam\anaconda3\lib\site-packages (from replit) (1.2
6.19)
Requirement already satisfied: aiosignal>=1.1.2 in c:\users\gadam\anaconda3\lib\site-packages (from aiohttp>=3.6
.2->replit) (1.2.0)
Requirement already satisfied: attrs>=17.3.0 in c:\users\gadam\anaconda3\lib\site-packages (from aiohttp>=3.6.2-
>replit) (23.1.0)
Requirement already satisfied: frozenlist>=1.1.1 in c:\users\gadam\anaconda3\lib\site-packages (from aiohttp>=3.
6.2->replit) (1.4.0)
Requirement already satisfied: multidict<7.0,>=4.5 in c:\users\gadam\anaconda3\lib\site-packages (from aiohttp>=
3.6.2->replit) (6.0.4)
Requirement already satisfied: yarl<2.0,>=1.0 in c:\users\gadam\anaconda3\lib\site-packages (from aiohttp>=3.6.2
->replit) (1.9.3)
Requirement already satisfied: Jinja2>=3.1.2 in c:\users\gadam\anaconda3\lib\site-packages (from Flask>=2.0.0->r
eplit) (3.1.4)
Requirement already satisfied: itsdangerous>=2.1.2 in c:\users\gadam\anaconda3\lib\site-packages (from Flask>=2.
0.0->replit) (2.2.0)
Requirement already satisfied: click>=8.1.3 in c:\users\gadam\anaconda3\lib\site-packages (from Flask>=2.0.0->re
plit) (8.1.7)
Requirement already satisfied: blinker>=1.6.2 in c:\users\gadam\anaconda3\lib\site-packages (from Flask>=2.0.0->
replit) (1.6.2)
Collecting argon2-cffi>=23.1.0 (from pyseto<2.0.0,>=1.6.11->replit)
  Downloading argon2_cffi-23.1.0-py3-none-any.whl.metadata (5.2 kB)
Requirement already satisfied: cryptography<44,>=42.0.1 in c:\users\gadam\anaconda3\lib\site-packages (from pyse
to<2.0.0,>=1.6.11->replit) (42.0.5)
Collecting iso8601<3.0.0,>=1.0.2 (from pyseto<2.0.0,>=1.6.11->replit)
  Downloading iso8601-2.1.0-py3-none-any.whl.metadata (3.7 kB)
Collecting pycryptodomex>=3.18.0 (from pyseto<2.0.0,>=1.6.11->replit)
  Downloading pycryptodomex-3.20.0-cp35-abi3-win_amd64.whl.metadata (3.4 kB)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\gadam\anaconda3\lib\site-packages (from requ
ests<3.0.0,>=2.25.1->replit) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\gadam\anaconda3\lib\site-packages (from requests<3.0.0,>
=2.25.1->replit) (3.7)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\gadam\anaconda3\lib\site-packages (from requests<3
.0.0,>=2.25.1->replit) (2024.6.2)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\gadam\anaconda3\lib\site-packages (from Werkzeug<4,
>=2->replit) (2.1.3)
Requirement already satisfied: argon2-cffi-bindings in c:\users\gadam\anaconda3\lib\site-packages (from argon2-c
ffi>=23.1.0->pyseto<2.0.0,>=1.6.11->replit) (21.2.0)
Requirement already satisfied: colorama in c:\users\gadam\anaconda3\lib\site-packages (from click>=8.1.3->Flask>
=2.0.0->replit) (0.4.6)
Requirement already satisfied: cffi>=1.12 in c:\users\gadam\anaconda3\lib\site-packages (from cryptography<44,>=
42.0.1->pyseto<2.0.0,>=1.6.11->replit) (1.16.0)
Requirement already satisfied: pycparser in c:\users\gadam\anaconda3\lib\site-packages (from cffi>=1.12->cryptog
raphy<44,>=42.0.1->pyseto<2.0.0,>=1.6.11->replit) (2.21)
Downloading replit-4.1.0-py3-none-any.whl (30 kB)
Downloading aiohttp_retry-2.8.3-py3-none-any.whl (9.8 kB)
Downloading protobuf-4.25.5-cp310-abi3-win_amd64.whl (413 kB)
   ---------------------------------------- 0.0/413.4 kB ? eta -:--:--
   ---------------------------------------- - 399.4/413.4 kB 8.3 MB/s eta 0:00:01
   ---------------------------------------- 413.4/413.4 kB 8.6 MB/s eta 0:00:00
Downloading pyseto-1.7.9-py3-none-any.whl (64 kB)
   ---------------------------------------- 0.0/64.0 kB ? eta -:--:--
   ---------------------------------------- 64.0/64.0 kB 1.7 MB/s eta 0:00:00
Downloading argon2_cffi-23.1.0-py3-none-any.whl (15 kB)
Downloading iso8601-2.1.0-py3-none-any.whl (7.5 kB)
```

```
Downloading pycryptodomex-3.20.0-cp35-abi3-win_amd64.whl (1.8 MB)
   ---------------------------------------- 0.0/1.8 MB ? eta -:--:--
   ----- ---------------------------------- 0.2/1.8 MB 6.9 MB/s eta 0:00:01
   ----------------- ---------------------- 0.8/1.8 MB 10.4 MB/s eta 0:00:01
   --------------------------- ------------ 1.3/1.8 MB 10.1 MB/s eta 0:00:01
   ---------------------------------------- - 1.7/1.8 MB 9.7 MB/s eta 0:00:01
   ---------------------------------------- 1.8/1.8 MB 8.0 MB/s eta 0:00:00
Installing collected packages: pycryptodomex, protobuf, iso8601, argon2-cffi, aiohttp-retry, pyseto, replit
  Attempting uninstall: protobuf
    Found existing installation: protobuf 3.20.3
    Uninstalling protobuf-3.20.3:
      Successfully uninstalled protobuf-3.20.3
  Attempting uninstall: argon2-cffi
    Found existing installation: argon2-cffi 21.3.0
    Uninstalling argon2-cffi-21.3.0:
      Successfully uninstalled argon2-cffi-21.3.0
Successfully installed aiohttp-retry-2.8.3 argon2-cffi-23.1.0 iso8601-2.1.0 protobuf-4.25.5 pycryptodomex-3.20.0
pyseto-1.7.9 replit-4.1.0
Note: you may need to restart the kernel to use updated packages.
```

Game Play:

The name of some Instagram accounts will be displayed, you have to guess which has a higher number of followers by typing in the name of that account. Make sure you type the name exactly as shown. If the answer is correct, one of the accounts will stay, and some other account will be displayed to compare to. We will develop the game using basic concepts of Python.

In [ ]:
```
data = [
    {
        'name': 'Instagram',
        'follower_count': 346,
        'description': 'Social media platform',
        'country': 'United States'
    },
    {
        'name': 'Cristiano Ronaldo',
        'follower_count': 215,
        'description': 'Footballer',
        'country': 'Portugal'
    },
    {
        'name': 'Ariana Grande',
        'follower_count': 183,
        'description': 'Musician and actress',
        'country': 'United States'
    }
]

link-->https://media.geeksforgeeks.org/wp-content/cdn-uploads/20220410192821/game_data.txt
```

In [52]:
```
# since we want to randomly select an
# option from the data, we need random
# module
import random

# import the game data
from game_data import data

# import the ASCII art to display.
from art import logo
from art import vs

# IDEs like Replit have their own "clear"
# functions to clear the output console:
from replit import clear

# If you are using some other IDE, this import
# will not work.
# You will have to use clear of that IDE
# For example, in Google Colab Notebook we have:
from IPython.display import clear_output

clear_output(wait=True)

def assign():
    pass

def compare(p1, p2, user_input):
    pass

def play_higher_lower():
    pass
```

```
want_to_play = input("Do you want to play Higher Lower? (y/n)\n").lower()
if want_to_play == 'y':
    clear()
    play_higher_lower()
elif want_to_play == 'n':
    print("Program Exit Successful.")
else:
    print("Invalid Input, Program Exited.")

def assign():
    return random.choice(data)

def compare(p1, p2, user_input):

    # store the follower count of
    # account1 in a variable
    sum1 = p1['follower_count']

    # store the follower count of
    # account2 in a variable
    sum2 = p2['follower_count']

    # make an empty variable max, where
    # we will store the name of account
    # with highest followers, then compare
    # it with user input name.
    max =""

    # if account1 has greater follower count
    if sum1 > sum2:

        # max is name of account1
        max = p1['name']
    elif sum1 < sum2:

        # otherwise, if account2 has higher
        # follower count,
        # max is name of account two.
        max = p2['name']

    # now compare the name of account with greater
    #follower count against the user input name,
    # if user is correct, return True
    if max == user_input:
        return True
    else:
          # otherwise return False
        return False
def play_higher_lower():

    # infinite loop to make user play
    # game again and again.
    playing_game = True
    while playing_game:

        # variable to keep track fo suer's score,
        # i.e., how many times he answers correctly
        score = 0


        # infinite loop to continue the current game play
        still_guessing = True
        while still_guessing:

            # print the logo after clearing the screen.
            clear()
            print(logo)

            # assign two account names to display
            person1 = assign()
            person2 = assign()

            # if we are not in first iteration, and user
            # input corret answer to previous iteration,
            # in that case, make account1 become account2
            # and randomly asing account2 some other account.
            if score > 0:
                person1 = person2
                person2 = assign()

                # we need to make sure that the two accounts
```

```
                # are not same.
                if person1 == person2:
                    person2 = assign()

            # display account1 name and description
            print(f"Name: {person1['name']}, Desc: {person1['description']}")

            # display the "VS" art
            print(vs)

            # display account2 name and description
            print(f"Name: {person2['name']}, Desc: {person2['description']}")

            # display current score
            print("-----------------------------------------------")
            print(f"Your current score is: {score}")
            print("-----------------------------------------------")

            # ask for user input
            guess = input("Enter name of person with Higher Followers: ")

            # see if user is correct
            if compare(person1, person2, guess):

                  # if user is correct continue to next iteration
                # and increase score by 1
                score += 1

            else:
                # if user is wrong, the current game play loop stops
                still_guessing = False

        # since the user was wrong in previous iteration and
        # the game ended, ask him if he want to play again.

        play_again = input("Want to Play Again? (y/n): ").lower()

        # if he want to, continue, otherwise end the outer
        # loop as well. also check if the user entered some
        # other input than what is allowed
        if play_again == 'y':
            continue
        elif play_again == 'n':
            playing_game = False
            clear()
            print("Game Exited Successfully")
        else:
            playing_game = False
            print("Invalid Input Taken as no.")

play_higher_lower()
```

```
 _   _ _       _                 _
| | | (_) __ _| |__   ___ _ __  | |    _____      _____ _ __
| |_| | |/ _` | '_ \ / _ \ '__| | |   / _ \ \ /\ / / _ \ '__|
|  _  | | (_| | | | |  __/ |    | |__| (_) \ V  V /  __/ |
|_| |_|_|\__, |_| |_|\___|_|    |_____\___/ \_/\_/ \___|_|
         |___/


Name: Zendaya, Desc: Actress and musician


__   _____
 \ \ / / __|
  \ V /\__    \_/ |___/


Name: Drake, Desc: Musician
-----------------------------------------------
Your current score is: 0
-----------------------------------------------
```

```
 _   _ _       _                 _
| | | (_) __ _| |__   ___ _ __  | |    _____      _____ _ __
| |_| | |/ _` | '_ \ / _ \ '__| | |   / _ \ \ /\ / / _ \ '__|
|  _  | | (_| | | | |  __/ |    | |__| (_) \ V  V /  __/ |
|_| |_|_|\__, |_| |_|\___|_|    |_____\___/ \_/\_/ \___|_|
         |___/


Name: Justin Bieber, Desc: Musician


__   _____
 \ \ / / __|
  \ V /\__    \_/ |___/


Name: Jennifer Lopez, Desc: Musician and actress
-----------------------------------------------
Your current score is: 0
-----------------------------------------------
 _   _ _       _                 _
| | | (_) __ _| |__   ___ _ __  | |    _____      _____ _ __
| |_| | |/ _` | '_ \ / _ \ '__| | |   / _ \ \ /\ / / _ \ '__|
|  _  | | (_| | | | |  __/ |    | |__| (_) \ V  V /  __/ |
|_| |_|_|\__, |_| |_|\___|_|    |_____\___/ \_/\_/ \___|_|
         |___/


Name: UEFA Champions League, Desc: Club football competition


__   _____
 \ \ / / __|
  \ V /\__    \_/ |___/


Name: Ellen DeGeneres, Desc: Comedian
-----------------------------------------------
Your current score is: 1
-----------------------------------------------
Game Exited Successfully
```

"Project 12"

In this article, we will discuss how to create a Fun Fact Generator Web App in Python using the PyWebio module. Essentially, it will create interesting facts at random and display them on the web interface. This script will retrieve data from uselessfacts.jsph.pl with the help of GET method, and we will use the requests and the JSON module to acquire and load the data respectively. After the data has been loaded, just pass the text within the dictionary so that we can only get the jokes. For creating a simple and interactive interface on the web, we will use the PyWebIO module.

PyWebIO contains functions for getting user input and output on the browser, turning the browser into a “rich text terminal”, and can be used to build simple web applications or browser-based GUI applications. With this module, anyone can produce a web application without any prior knowledge or overhead of HTML and JS.

In [ ]:
```
Before Going to code we have to install one library that is 'pywebio'
```

In [2]:
```
pip install pywebio
```

```
Collecting pywebio
  Downloading pywebio-1.8.3.tar.gz (500 kB)
     ---------------------------------------- 0.0/500.9 kB ? eta -:--:--
     -- ------------------------------------- 30.7/500.9 kB 435.7 kB/s eta 0:00:02
     -------- ------------------------------- 112.6/500.9 kB 1.3 MB/s eta 0:00:01
     ---------------------------------- --- 460.8/500.9 kB 3.6 MB/s eta 0:00:01
     ---------------------------------- -- 471.0/500.9 kB 2.9 MB/s eta 0:00:01
     ---------------------------------------- 500.9/500.9 kB 2.2 MB/s eta 0:00:00
  Preparing metadata (setup.py): started
  Preparing metadata (setup.py): finished with status 'done'
Requirement already satisfied: tornado>=5.0 in c:\users\gadam\anaconda3\lib\site-packages (from pywebio) (6.4.1)
Collecting user-agents (from pywebio)
  Downloading user_agents-2.2.0-py3-none-any.whl.metadata (7.9 kB)
Collecting ua-parser>=0.10.0 (from user-agents->pywebio)
  Downloading ua_parser-0.18.0-py2.py3-none-any.whl.metadata (1.4 kB)
Downloading user_agents-2.2.0-py3-none-any.whl (9.6 kB)
Downloading ua_parser-0.18.0-py2.py3-none-any.whl (38 kB)
Building wheels for collected packages: pywebio
  Building wheel for pywebio (setup.py): started
  Building wheel for pywebio (setup.py): finished with status 'done'
  Created wheel for pywebio: filename=pywebio-1.8.3-py3-none-any.whl size=509542 sha256=da2ad311114ae32cfb7eaae9
f557611aabfed3f03391053667f18f8a71566ecd
  Stored in directory: c:\users\gadam\appdata\local\pip\cache\wheels\a1\33\ef\5aed4d10777ceb93d7aa9aea2f52db6b0b
3d5d8a4abb741404
Successfully built pywebio
Installing collected packages: ua-parser, user-agents, pywebio
Successfully installed pywebio-1.8.3 ua-parser-0.18.0 user-agents-2.2.0
Note: you may need to restart the kernel to use updated packages.
```

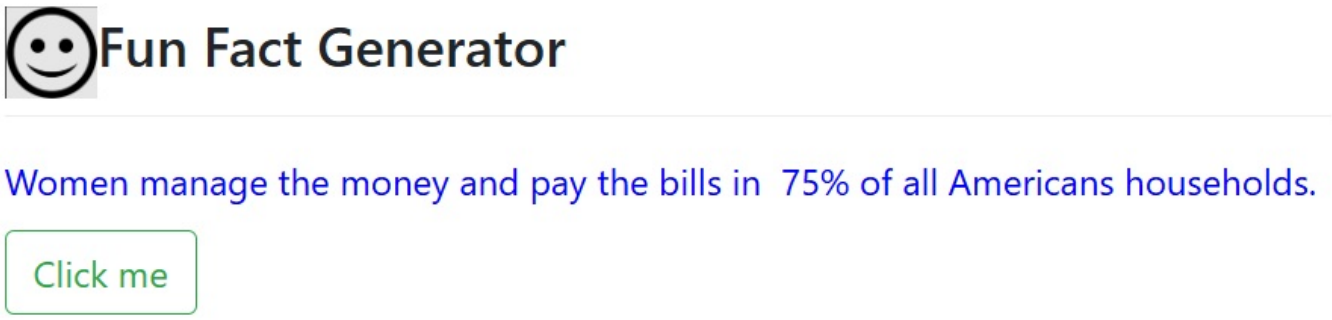
In [ ]:

```python
#Importing the neccessary modules
import json
import requests
from pywebio.input import *
from pywebio.output import *
from pywebio.session import *

def get_fun_fact(_):
    #clear the screen
    clear()
    #Put HTML contents for the fun fact genratorheader
    put_html('<p align="left">'
             '<h2><img src="https://media.geeksforgeeks.org/wp-content/uploads/20210720224119/MessagingHappyico
             '</p>'
            )
    #URL from the where we will fetch the data
    url="https://uselessfacts.jsph.pl/random.json?language=en"
    #use get request
    response=requests.get(url)
    #Load the request in json file
    data=json.loads(response.text)
    #we will need text from the data
    useless_fact=data['text']
    #Put the fact in blue color and increase in font size
    style(put_text(useless_fact),'color:blue;,font-size:30px')
    #Put the "click me button"
    put_buttons(
        [dict(label='Click me',value='outline-success',color='outline-success')],
                onclick=get_fun_fact
               )

if __name__=='__main__':
    #Put a headeing fun fact genrator
    put_html(
        '<p align="left">'
        '<h2><img src="https://media.geeksforgeeks.org/wp-content/uploads/20210720224119/MessagingHappyicon.png
        '</p>'
    )
    #Hold the session for a long time put the click me button
    put_buttons(
        [dict(label='Click me',value='outline-success',color='outline-success')]
        ,onclick=get_fun_fact
    )
    hold()
```

In [3]:

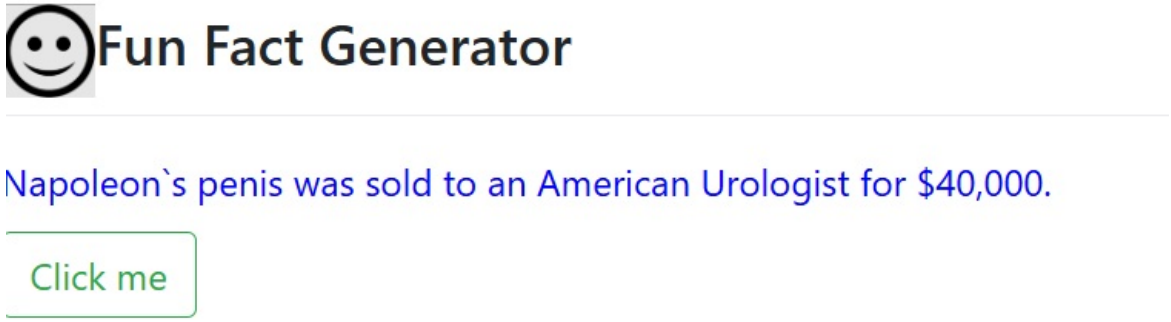
```python
from PIL import Image
Image.open("web1.png")
```

Out[3]:



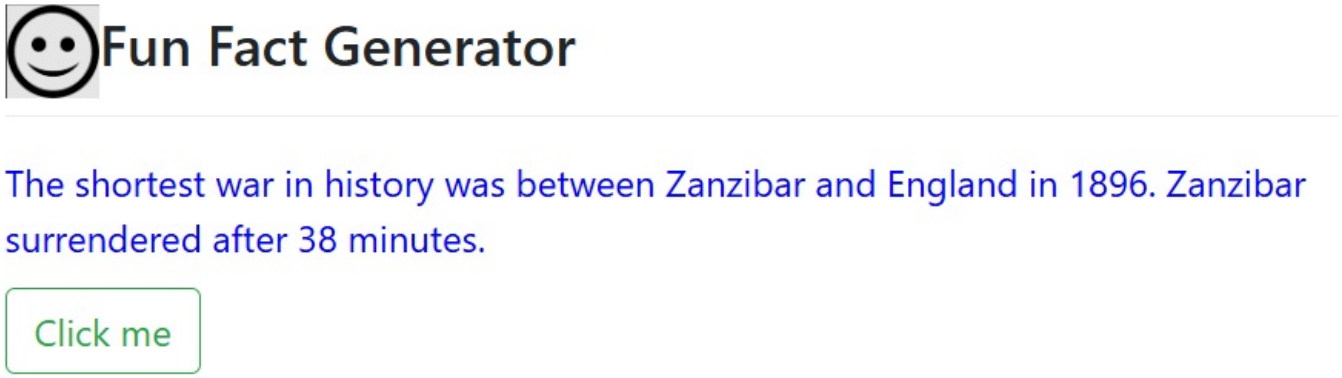
In [9]:
```
Image.open("web2.png")
```

Out[9]:



In [13]:
```
Image.open("web3.png")
```

Out[13]:



"Project 13"

In [ ]:
```
"Check if two PDF documents are identical with Python"
```

Python is an interpreted and general purpose programming language.It is a Object-Oriented and Procedural paradaiagrams language .there are various types of modules imported in Python such as difflib,hashlib

```
Modules used:
difflib : It is a module that contains function that allows to compare set of data.
 SequenceMatcher : It is used to compare pair of input sequences.


"Aproach":
Import module
Declare a function with 2 arguments which is for file.
Declare two objects for hashlib.sha1()
Open files
Read the file by breaking the line into smaller chunks
Now return both file such as h1.hexdigest() which is of 160 bits.
Use hash_file() function to store the hash of a file.
Compare and generate appropriate message
```

In [42]:
```
import hashlib
```

```
from difflib import SequenceMatcher

def hash_file(fileName1,fileName2):
    #Use haslib to store the hash of a file
    h1=hashlib.sha1()
    h2=hashlib.sha1()
    with open(fileName1,"rb") as file:
        #Use file.read() to read the size file and read the file in small chunks because we cannot read the larg

        # Use file.read() to read the size of file
        # and read the file in small chunks
        # because we cannot read the large files.
        chunk = 0
        while chunk != b'':
            chunk = file.read(1024)
            h1.update(chunk)

    with open(fileName2, "rb") as file:

        # Use file.read() to read the size of file a
        # and read the file in small chunks
        # because we cannot read the large files.
        chunk = 0
        while chunk != b'':
            chunk = file.read(1024)
            h2.update(chunk)

        # hexdigest() is of 160 bits
        return h1.hexdigest(), h2.hexdigest()


msg1, msg2 = hash_file("numpy-ufunc.pdf","numpys (1).pdf")
if(msg1 != msg2):
    print("These files are not identical")
else:
    print("These files are identical")
```

```
These files are not identical
```

"Project 14"

In [ ]:
```
"Creating payment receipts using python"
```

Creating payment receipts is a pretty common task,be it an e-commerce website or nay local store for that matter.

Here,we will see How to create out own transaction receipts just by python.we would be using 'reportlab' to generate the PDFs.genarally ,it comes as a built-in-packages but sometimes it might not be present too.If it's present ,then simply type the following in your terminal.

In [46]:
```
pip install reportlab
```

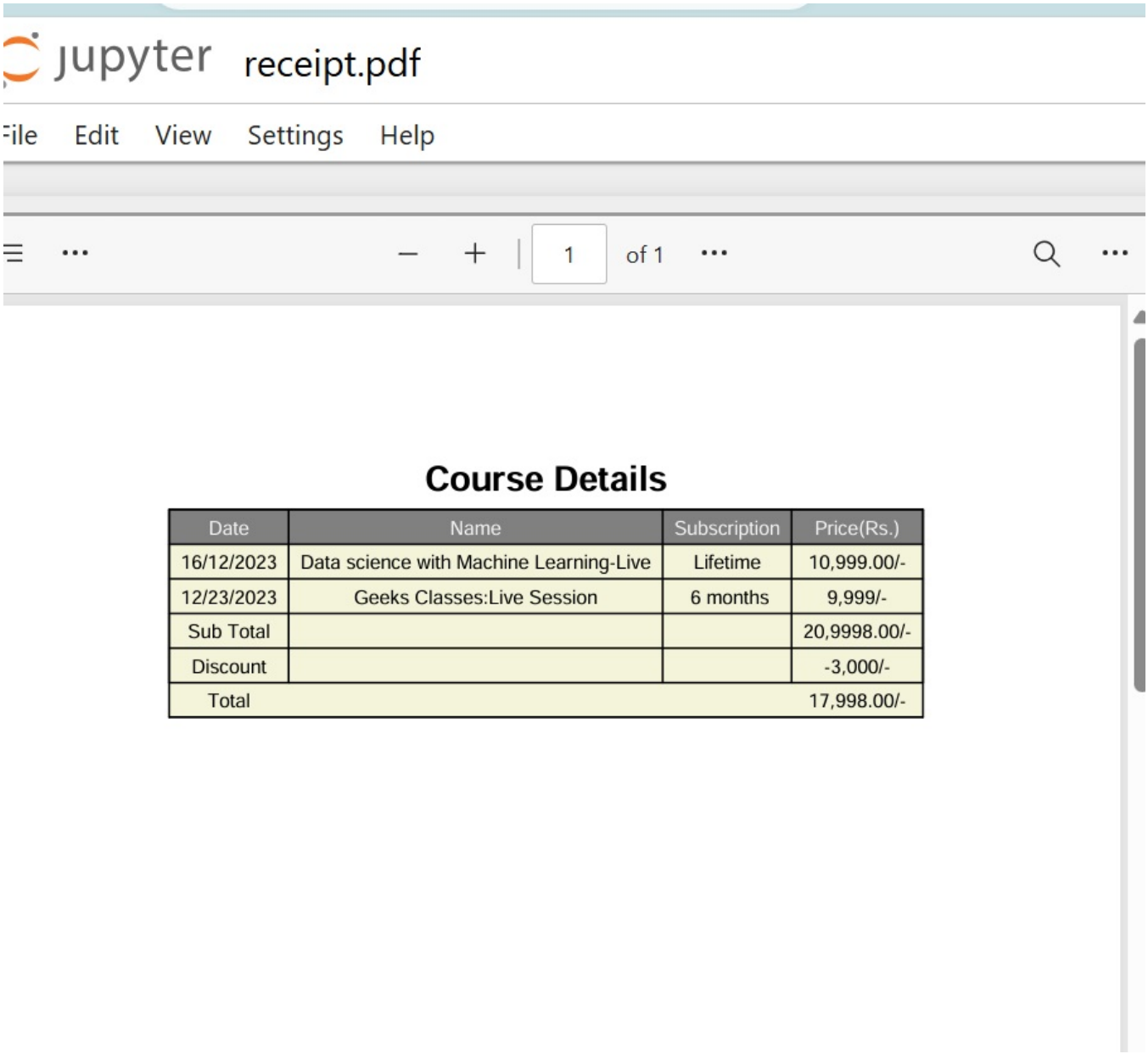
```
Collecting reportlab
  Downloading reportlab-4.2.2-py3-none-any.whl.metadata (1.4 kB)
Requirement already satisfied: pillow>=9.0.0 in c:\users\gadam\anaconda3\lib\site-packages (from reportlab) (10.
3.0)
Requirement already satisfied: chardet in c:\users\gadam\anaconda3\lib\site-packages (from reportlab) (4.0.0)
Downloading reportlab-4.2.2-py3-none-any.whl (1.9 MB)
   ---------------------------------------- 0.0/1.9 MB ? eta -:--:--
   ---------------------------------------- 0.0/1.9 MB ? eta -:--:--
   - -------------------------------------- 0.1/1.9 MB 656.4 kB/s eta 0:00:03
   ----- ---------------------------------- 0.3/1.9 MB 2.3 MB/s eta 0:00:01
   ---------- ----------------------------- 0.5/1.9 MB 3.0 MB/s eta 0:00:01
   ------------------ --------------------- 0.9/1.9 MB 4.1 MB/s eta 0:00:01
   ----------------------- ---------------- 1.2/1.9 MB 4.5 MB/s eta 0:00:01
   ---------------------------- ---------- 1.5/1.9 MB 4.6 MB/s eta 0:00:01
   ---------------------------------------  1.9/1.9 MB 5.4 MB/s eta 0:00:01
   ---------------------------------------- 1.9/1.9 MB 4.8 MB/s eta 0:00:00
Installing collected packages: reportlab
Successfully installed reportlab-4.2.2
Note: you may need to restart the kernel to use updated packages.
```

"Approach:"
1. Import module.
2. We need the data in the form of a list of lists. The 0th index of the outer index is the headers.
3. We create a simple doc template with the specified paper size (here A4)
4. Then get a sample style sheet from the built-in style sheets and add the styling accordingly.
5. After you have created a style object, feed in the data and the style sheet to the pdf object and build it.

In [61]:
```
#import modules
from reportlab.platypus import SimpleDocTemplate,Table,Paragraph,TableStyle
from reportlab.lib import colors
from reportlab.lib.pagesizes import A4
from reportlab.lib.styles import getSampleStyleSheet

#data which we are going to display as tables

DATA=[
    ["Date","Name","Subscription","Price(Rs.)"],
    [
        "16/12/2023",
        "Data science with Machine Learning-Live",
        "Lifetime",
        "10,999.00/-",
    ],
    ["12/23/2023","Geeks Classes:Live Session"," 6 months","9,999/-"],
    ["Sub Total","","","20,9998.00/-"],
    ["Discount","","","-3,000/-"],
    ["Total","","","17,998.00/-"],
]
#creating a Base Document template of page size A4
pdf=SimpleDocTemplate("receipt.pdf",pagesize=A4)

#standard stylesheet defined within reportlab itself
styles=getSampleStyleSheet()

#fetching the style of Top level heading (Headiung1)
title_style=styles["Heading1"]

#0: left, 1 :center,: 3: right
title_style.alignment=1

#ceating the paragraph with the heading text and passing the styles of it
title=Paragraph("Course Details",title_style)

#creates a table Style object and in it,defines the style row wise the tuple which look like coordinates are no
style=TableStyle(
    [
        ("BOX",(0,0),(-1,-1),1,colors.black),
        ("GRID",(0,0),(4,4),1,colors.black),
        ("BACKGROUND",(0,0),(3,0),colors.gray),
        ("TEXTCOLOR",(0,0),(-1,0),colors.whitesmoke),
        ("ALIGN",(0,0),(-1,-1),"CENTER"),
        ("BACKGROUND",(0,1),(-1,-1),colors.beige),
    ]
)
#creates a table ibject and styles passes to it
table=Table(DATA,style=style)
#final step which bhilds the actual pdf putting together all the elements
pdf.build([title,table])
```

In [57]:
```
Image.open("receipt.png")
```

Out[57]:



"Project 15"

In [ ]:
```
"How TO create a Countdown Timer Using Python"
```

In this article we will see the how to create a countdown timer using Python.The code will take input from the user regarding the length of the countdown in seconds .After that,a countdwon will be begin on the screen of the format 'minutes:seconds'.We will use the time module here.

"Approach"

In this project ,we will be using the time module and its sleep() fucnction.Follows the below steps to create a countodwon timer.

```
Step 1: Import the time module.
Step 2: Then ask the user to input the length of the countdown in seconds.
Step 3: This value is sent as a parameter ‘t’ to the user-defined function countdown(). Any
variable read using the input function is a string. So, convert this parameter to ‘int’ as it
is of string type.
Step 4: In this function, a while loop runs until time becomes 0.
Step 5: Use divmod() to calculate the number of minutes and seconds. You can read more about
it here.
Step 6: Now print the minutes and seconds on the screen using the variable timeformat.
Step 7: Using end = ‘\r’ we force the cursor to go back to the start of the screen (carriage
return) so that the next line printed will overwrite the previous one.
Step 8: The time.sleep() is used to make the code wait for one sec.
Step 9: Now decrement time so that the while loop can converge.
Step 10: After the completion of the loop, we will print “Fire in the hole” to signify the end
of the countdown.
```

In [3]:
```
# import the time module
import time
```

```
# define the countdown func.
def countdown(t):

    while t:
        mins, secs = divmod(t, 60)
        timer = '{:02d}:{:02d}'.format(mins, secs)
        print(timer, end="\r")
        time.sleep(1)
        t -= 1

    print('Fire in the hole!!')


# input time in seconds
t = input("Enter the time in seconds: ")

# function call
countdown(int(t))
```

```
Fire in the hole!!
```

"Project 16"

In [ ]:
```
"convert emoji into text in Python"
```

Converting emoticons or emojis into text in Python can be deone using the "demoji module" .It is used to accuratley remove and replace emojis in text strings.To install the demoji module the below command can be used.

In [7]:
```
pip install demoji
```

```
Collecting demoji
  Downloading demoji-1.1.0-py3-none-any.whl.metadata (9.2 kB)
Downloading demoji-1.1.0-py3-none-any.whl (42 kB)
   ---------------------------------------- 0.0/42.9 kB ? eta -:--:--
   ------------------ --------------------- 20.5/42.9 kB 640.0 kB/s eta 0:00:01
   ---------------------------------------- 42.9/42.9 kB 517.6 kB/s eta 0:00:00
Installing collected packages: demoji
Successfully installed demoji-1.1.0
Note: you may need to restart the kernel to use updated packages.
```

The demoji also requires also an initial download of data from the Unicode Consortim "emoji code repository" as the emoji list itself is frequently updated and changed .the below code block should be used for the above mentioned download.

In [11]:
```
import demoji

demoji.download_codes()
```

```
C:\Users\gadam\AppData\Local\Temp\ipykernel_32112\4199042163.py:3: FutureWarning: The demoji.download_codes attr
ibute is deprecated and will be removed from demoji in a future version. It is an unused attribute as emoji code
s are now distributed directly with the demoji package.
  demoji.download_codes()
```

In [19]:
```
import demoji

text="""I will be your secret snata this year   nauseated face
 I willl become a dancer and I am very much do learn data science but present I have
 sneezing face hot face    """

demoji.findall(text)
```

Out[19]:
```
{'': 'exploding head',
 '': 'woozy face',
 '': 'sneezing face',
 '': 'hot face',
 '': 'nauseated face',
 '': 'zany face',
 '': 'cold face',
 '': 'face vomiting'}
```

In [ ]:
```

```

In [23]:
```
text="I am russain  ,I am indian,I am a londan,I am south African I am american "

demoji.findall(text)
```

Out[23]:
```
{'': 'woman: medium-dark skin tone',
 '': 'woman: dark skin tone',
 '': 'woman: medium skin tone',
 '\u200d ': 'woman: blond hair',
 '': 'woman'}
```

"Project 17"

In [ ]:
```
"Create a Voice Recorder using Python"
```

Python can be used to perform a variaty of tasks.One of them is creating a voice recorder .We can use Python's "sounddevice" module to record and play audio.This module along with the "wavio" or the "scipy" module provides a wway to save recorded audio.

In [ ]:
```
Before the we have to install both modules with pip install
```

In [27]:
```
pip install wavio
```

```
Collecting wavio
  Downloading wavio-0.0.9-py3-none-any.whl.metadata (5.7 kB)
Requirement already satisfied: numpy>=1.19.0 in c:\users\gadam\anaconda3\lib\site-packages (from wavio) (1.26.4)
Downloading wavio-0.0.9-py3-none-any.whl (9.5 kB)
Installing collected packages: wavio
Successfully installed wavio-0.0.9
Note: you may need to restart the kernel to use updated packages.
```

In [28]:
```
pip install scipy
```

```
Requirement already satisfied: scipy in c:\users\gadam\anaconda3\lib\site-packages (1.13.1)
Requirement already satisfied: numpy<2.3,>=1.22.4 in c:\users\gadam\anaconda3\lib\site-packages (from scipy) (1.
26.4)
Note: you may need to restart the kernel to use updated packages.
```

In [31]:
```
pip install sounddevice
```

```
Collecting sounddevice
  Downloading sounddevice-0.5.0-py3-none-win_amd64.whl.metadata (1.4 kB)
Requirement already satisfied: CFFI>=1.0 in c:\users\gadam\anaconda3\lib\site-packages (from sounddevice) (1.16.
0)
Requirement already satisfied: pycparser in c:\users\gadam\anaconda3\lib\site-packages (from CFFI>=1.0->sounddev
ice) (2.21)
Downloading sounddevice-0.5.0-py3-none-win_amd64.whl (189 kB)
   ---------------------------------------- 0.0/189.8 kB ? eta -:--:--
   -- ------------------------------------- 10.2/189.8 kB ? eta -:--:--
   -------- ------------------------------- 41.0/189.8 kB 487.6 kB/s eta 0:00:01
   ---------------------------------------- - 184.3/189.8 kB 1.9 MB/s eta 0:00:01
   ---------------------------------------- 189.8/189.8 kB 1.4 MB/s eta 0:00:00
Installing collected packages: sounddevice
Successfully installed sounddevice-0.5.0
Note: you may need to restart the kernel to use updated packages.
```

In [39]:
```
# import required libraries
import sounddevice as sd
from scipy.io.wavfile import write
import wavio as wv

# Sampling frequency
freq = 44100

# Recording duration
duration = 5

# Start recorder with the given values
# of duration and sample frequency
recording = sd.rec(int(duration * freq),
                   samplerate=freq, channels=2)

# Record audio for the given number of seconds
sd.wait()

# This will convert the NumPy array to an audio
# file with the given sampling frequency
write("recording0.wav", freq, recording)

# Convert the NumPy array to audio file
wv.write("recording1.wav", recording, freq, sampwidth=2)
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: