my_ass2

August 24, 2021

[2]:
```python
# importing csv module
import csv
#importing random
import random

class Matrix:
    def __init__(self):
        pass

    def write(self):
        #Loop for creating Matirx files
        for n in range(1,5):
            # field names
            fields = ['Col-1', 'Col-2', 'Col-3']

            #Condition for creating 1X3 matrix
            if(n<3):
                # data rows of csv file
                rows = [ random.sample(range(1, 20), 3) for j in range(3)]
            else: #condition for 3X3 matrix
                rows = [random.sample(range(1, 20), 3)]


            # name of csv file
            name="Matrix_"+str(n)
            filename = name+".csv"

            # writ to csv file
            with open(filename, 'w') as csvfile:
                # creating a csv writer object
                csvwriter = csv.writer(csvfile)

                # writing the fields
                csvwriter.writerow(fields)

                # writing the data rows
                csvwriter.writerows(rows)
```

```
def read(name):
    # csv file name
    filename = name+".csv"

    # initializing the titles and rows list
    fields = []
    rows = []

    # reading csv file
    with open(filename, 'r') as csvfile:
        # creating a csv reader object
        csvreader = csv.reader(csvfile)

        # extracting field names through first row
        fields = next(csvreader)

        # extracting each data row one by one
        for row in csvreader:
            if(len(row)!=0):
                rows.append(row)
    return rows

def transpose(name):
    #Original Matrix
    x = Matrix.read(name)
    result = [[x[j][i] for j in range(len(x))] for i in range(len(x[0]))]
    return result

def multiplication(X,Y):
    result=[]
    # iterate through rows of X
    for i in range(len(X)):
        col=[]
        # iterate through columns of Y
        for j in range(len(Y[0])):

            sum=0
            # iterate through rows of Y
            for k in range(len(Y)):
                sum += int(X[i][k]) * int(Y[k][j])
            col.append(sum)

        result.append(col)

    Matrix.display(result)
    def display(X):
        for r in X:
            print(r)
        print('\n')

    def call(self):
        print("Matrix A\n")
        Matrix.display(Matrix.read("Matrix_1"))
        print("Matrix B\n")
        Matrix.display(Matrix.read("Matrix_2"))
        print("Matrix C\n")
        Matrix.display(Matrix.read("Matrix_3"))
        print("Matrix D\n")
        Matrix.display(Matrix.read("Matrix_4"))

        # CD
        print("CD")
        Matrix.multiplication(Matrix.read("Matrix_3"),Matrix.
 ↪transpose("Matrix_4"))

        # AB
        print('AB')
        Matrix.multiplication(Matrix.read("Matrix_1"),Matrix.read("Matrix_2"))

        # CA
        print('CA')
        Matrix.multiplication(Matrix.read("Matrix_3"),Matrix.read("Matrix_1"))

        # BD
        print('BD')
        Matrix.multiplication(Matrix.read("Matrix_2"),Matrix.
 ↪transpose("Matrix_4"))

Cal=Matrix()
Cal.write()
Cal.call()
```

```
Matrix A

['1', '10', '16']
['12', '3', '9']
['14', '19', '7']


Matrix B

['16', '5', '18']
```

```
['4', '14', '7']
['12', '9', '13']


Matrix C

['16', '6', '13']


Matrix D

['19', '17', '1']


CD
[419]


AB
[248, 289, 296]
[312, 183, 354]
[384, 399, 476]


CA
[270, 425, 401]


BD
[407]
[321]
[394]
```

[3]:
```
#simple algebra of complex numbers
import numpy as np     #to use tan inverse function

class myComplex:
    def __init__(self, a=0.0, b = 0.0):
        self.a = a
        self.b = b

    def display(self):
        print(self.a,"+ i",self.b)

    def modulus(self):
        mod = (self.a**2 + self.b**2)**(0.5)
        return mod

    def phase(self):
        phi = numpy.arctan(self.b/self.a)
        return phi

    def conjugate(self):
        self.b = -self.b


    def addition(self , X):
        A = self.a + X.a
        B = self.b + X.b
        return myComplex(A, B)

    def subtraction(self , X):
        A = self.a - X.a
        B = self.b - X.b
        return myComplex(A, B)

    def multiplication(self, X):
        A = self.a*X.a - self.b*X.b
        B = self.a*X.b + self.b*X.a
        return myComplex(A,B)

    def division(self,X):
        x = self.a
        y = self.b
        z = X.a
        w = X.b
        A = (x*z + y*w)/(z**2 + w**2)
        B = (y*z - x*w)/(z**2 + w**2)
        return myComplex(A,B)

print("Printing complex number")
m = myComplex(4,5)
m.display()
n = myComplex(3,6)
n.display()

print('Conjugate of complex number')
m.conjugate()
m.display()

print('Modulus')
print(m.modulus())

print('Addition')
o = m.addition(n)
o.display()

print('Subtraction')
p = m.subtraction(n)
p.display()

print('Multiplication')
q = p.multiplication(m)
q.display()

print('Division')
r = q.division(n)
r.display()
```

```
Printing complex number
4 + i 5
3 + i 6
Conjugate of complex number
4 + i -5
Modulus
6.4031242374328485
Addition
7 + i 1
Subtraction
1 + i -11
Multiplication
-51 + i -49
Division
-9.933333333333334 + i 3.533333333333333
```

[4]:
```
#finding average distance between two points on a line made of 'N' discrete␣
 ↪points.

#Let 'L' be the length of the line segment.

def av_distbtw2points_line(N,L):
  File "<ipython-input-4-90e7fe344082>", line 6

    ^
SyntaxError: unexpected EOF while parsing
```

```
[ ]: import random
     # generates a random for for the game from a list of countries and capitals
     def generate_word():
     countries=['Argentina', 'Australia', 'Brazil', 'Cameroon', 'Canada', 'Chile',␣
      ↪'China',
     'England',
              'France', 'Germany', 'Italy', 'Jamaica', 'Japan', 'Netherlands', 'New␣
      ↪Zealand',
     'Nigeria',
              'Norway', 'Scotland', 'South Africa', 'South Korea', 'Spain', 'Sweden',␣
      ↪'Thailand',
      'United States']
     capitals=['buenosaires', 'canberra', 'brasilia', 'yaounde', 'ottawa',␣
      ↪'santiago', 'beijing',
      'london',
              'paris', 'berlin', 'rome', 'kingston', 'tokyo', 'amsterdam',␣
      ↪'wellington', 'abuja',
              'oslo', 'edinburgh', 'capetown', 'seoul', 'madrid', 'stockholm',␣
      ↪'bangkok', 'washington']

     val=random.randrange(0,24)
     return countries[val], capitals[val]
```

```
[ ]: import random

     from words import words

     def aword():
     Â   Â   word = random.choice(words)
     Â   Â   return word.upper()

     def game(word):
     Â   Â   WordCompletion = '_'*len(word)
     Â   Â   DoneLetters =[]
     Â   Â   guessed = False
     Â   Â   t = 6
     Â   Â   print(WordCompletion)
     Â   Â   print("\n")
     Â   Â   while not guessed and (t > 0):
```

```
Â  Â  Â  Â  guess = input('your guess: ').upper()

Â  Â  Â  Â  if len(guess) ==1 and guess.isalpha():
Â  Â  Â  Â  Â  Â  if guess in DoneLetters:
Â  Â  Â  Â  Â  Â  Â  Â  print(('you have already guessed it...'))
Â  Â  Â  Â  Â  Â  elif guess not in word:
Â  Â  Â  Â  Â  Â  Â  Â  print(('it is not in the word...'))
Â  Â  Â  Â  Â  Â  Â  Â  t -= 1
Â  Â  Â  Â  Â  Â  Â  Â  DoneLetters.append(guess)
Â  Â  Â  Â  Â  Â  Â  Â  print('you have ' + str(t) + ' tries left')
Â  Â  Â  Â  Â  Â  else:
Â  Â  Â  Â  Â  Â  Â  Â  print(('you got one..'))
Â  Â  Â  Â  Â  Â  Â  Â  wordlist = list(WordCompletion)
Â  Â  Â  Â  Â  Â  Â  Â  indices = [i for i, letter in enumerate(word) if letter␣
↪== guess]
Â  Â  Â  Â  Â  Â  Â  Â  for index in indices:
Â  Â  Â  Â  Â  Â  Â  Â  Â  Â  wordlist[index] = guess
Â  Â  Â  Â  Â  Â  Â  Â  WordCompletion = "".join(wordlist)
Â  Â  Â  Â  Â  Â  Â  Â  if "_" not in WordCompletion:
Â  Â  Â  Â  Â  Â  Â  Â  Â  Â  guessed = True
Â  Â  Â  Â  else:
Â  Â  Â  Â  Â  Â  print('invalid guess')
Â  Â  Â  Â  print((WordCompletion))
Â  Â  if guessed:
Â  Â  Â  Â  print("Congrats, you guessed the word! You win!")
Â  Â  else:
Â  Â  Â  Â  print("Sorry, you ran out of tries. The word was " + word + ". Maybe␣
↪next time!")

def final():
Â  Â  word = aword()
Â  Â  game(word)
Â  Â  while input('want to play? y/n ').upper() == 'Y':
Â  Â  Â  Â  word = aword()
Â  Â  Â  Â  game(word)

final()
```

[7]:
```
#average distance between two points on a straight line made of N discrete pont

def average_distance():
    N = int(input("Enter number of points:"))
    x = int(input("Enter a point from which distance will be measured:"))
    #For calculating distance from the point to the point before it
    a = 0
    for i in range(x):
        a = a + i
    #For calculating distance from the point to the point after it
    b = 0
    for i in range(N-x+1):
        b = b + i
    avg = (a+b)/N
    print(avg)

average_distance()
average_distance()
average_distance()
average_distance()
```

```
Enter number of points:2
Enter a point from which distance will be measured:1
0.5
Enter number of points:4
Enter a point from which distance will be measured:1
1.5
Enter number of points:5
Enter a point from which distance will be measured:1
2.0
Enter number of points:7
Enter a point from which distance will be measured:3
1.8571428571428572
```

[11]:
```
import random


# generates a random for for the game from a list of countries and capitals
def generate_word():
    countries=['Argentina', 'Australia', 'Brazil', 'Cameroon', 'Canada',␣
 ↪'Chile', 'China','England','France', 'Germany', 'Italy', 'Jamaica', 'Japan',␣
 ↪'Netherlands', 'New Zealand','Nigeria','Norway', 'Scotland', 'South Africa',␣
 ↪'South Korea', 'Spain', 'Sweden','Thailand','United States']
    capitals=['buenosaires', 'canberra', 'brasilia', 'yaounde', 'ottawa',␣
 ↪'santiago','beijing','london','paris', 'berlin', 'rome', 'kingston', 'tokyo',␣
 ↪'amsterdam', 'wellington','abuja','oslo', 'edinburgh', 'capetown', 'seoul',␣
 ↪'madrid', 'stockholm', 'bangkok','washington']
    val = random.randrange(0,24)
    return countries[val], capitals[val]
print("Enter guesses in small letterse.\n")
country, capital=generate_word()
print("Guess the capital of the country "+country)
print("\n\n")
word=capital
word2=list(word) # convert into list
```

```python
chance=[]
for i in range(len(word2)):
    chance.append('_')
print(chance)

ch_left=int(len(word2)*0.4)
flag=0 # variable to declare win

i=0
while ch_left>0:
    count=0 # variable to decide number of chances left
    print("\nChances left : "+str(ch_left))
    print()
    guess=input("Enter your guess : ")

    for j in range(len(word2)):
        if chance[j]=='_': # loop runs only for blank spaces left
            if word2[j]==guess:
                chance[j]=guess
                count=1
                flag+=1
            else:
                chance[j]="_"
    print(chance)
    if count==0:
        ch_left-=1

    # checking the losing condition first so that the value of
    # ch_left is not altered by the winning condition ch_left=0
    if ch_left==0: # losing condition
        print("\n Sorry, you lost the game.")
        print("_________") # Hangman picture
        print("|       |")
        print("|      _O_")
        print("|       |")
        print("|.     /\\")
        print("|____________")
        print("|____________|")

    if flag==len(chance): # winning condition
        print("\nCongratulations for winning the game.")
        ch_left=0
    i+=1
```

Enter guesses in small letterse.

```
Guess the capital of the country Spain



['_', '_', '_', '_', '_', '_']

Chances left : 2

Enter your guess : m
['m', '_', '_', '_', '_', '_']

Chances left : 2

Enter your guess : d
['m', '_', 'd', '_', '_', 'd']

Chances left : 2

Enter your guess : r
['m', '_', 'd', 'r', '_', 'd']

Chances left : 2

Enter your guess : i
['m', '_', 'd', 'r', 'i', 'd']

Chances left : 2

Enter your guess : a
['m', 'a', 'd', 'r', 'i', 'd']

Congratulations for winning the game.
```