**DAY4- Programming fundamentals using python**

Assignment 29:

The road transport corporation (RTC) of a city wants to know whether a particular bus-route is running on profit or loss.  
Assume that the following information are given:  
Price per litre of fuel = 70  
Mileage of the bus in km/litre of fuel = 10  
Price(Rs) per ticket = 80  
The bus runs on multiple routes having different distance in kms and number of passengers.  
Write a function to calculate and return the profit earned (Rs) in each route. Return -1 in case of loss

#PF-Assgn-29

def calculate(distance,no\_of\_passengers):

#Remove pass and write your logic here

per\_litre\_fuel=70

mileage=10

price = 80

bus= distance \* per\_litre\_fuel/mileage

pas = no\_of\_passengers \* price

if bus-pas > 0:

return -1

else:

return pas-bus

#Provide different values for distance, no\_of\_passenger and test your program

distance=1000

no\_of\_passengers=50

print(calculate(distance,no\_of\_passengers))

Assignment 30:

Given a string containing uppercase characters (A-Z), compress the string using Run Length encoding. Repetition of character has to be replaced by storing the length of that run.  
Write a python function which performs the run length encoding for a given String and returns the run length encoded String.  
Provide different String values and test your program.

|  |  |
| --- | --- |
| **Sample Input** | **Expected Output** |
| AAAABBBBCCCCCCCC | 4A4B8C |
| AABCCA | 2A1B2C1A |

#PF-Assgn-30

def encode(message):

#Remove pass and write your logic here

encoded\_message = ""

i = 0

while (i <= len(message)-1):

count = 1

ch = message[i]

j = i

while (j < len(message)-1):

if (message[j] == message[j+1]):

count = count+1

j = j+1

else:

break

encoded\_message=encoded\_message+str(count)+ch

i = j+1

return encoded\_message

#Provide different values for message and test your program

encoded\_message=encode("ABBBBCCCCCCCCAB")

print(encoded\_message)

Assignment 31:

Write a function, **check\_palindrome()** to check whether the given string is a palindrome or not. The function should return true if it is a palindrome else it should return false.  
  
**Note**: Initialize the string with various values and test your program. Assume that all the letters in the given string are all of the same case. Example: MAN, civic, WOW etc.

#PF-Assgn-31

def check\_palindrome(word):

#Remove pass and write your logic here

revword=word[::-1]

if revword == word:

return True

else:

return False

status=check\_palindrome("malayalam")

if(status):

print("word is palindrome")

else:

print("word is not palindrome")

Assignment 32:

Care hospital wants to know the medical speciality visited by the maximum number of patients. Assume that the patient id of the patient along with the medical speciality visited by the patient is stored in a list. The details of the medical specialities are stored in a dictionary as follows:  
{  
"P":"Pediatrics",  
"O":"Orthopedics",  
"E":"ENT  
}  
  
Write a function to find the medical speciality visited by the maximum number of patients and return the name of the speciality.  
Also write the pytest test cases to test the program.

#PF-Assgn-32

def max\_visited\_speciality(patient\_medical\_speciality\_list,medical\_speciality):

# write your logic here

speciality=""

pcount= patient\_medical\_speciality\_list.count("P")

ocount= patient\_medical\_speciality\_list.count("O")

evount= patient\_medical\_speciality\_list.count("E")

letter= max(pcount,ocount,evount)

for i in medical\_speciality:

if letter == i:

speciality=medical\_speciality[i]

return speciality

#provide different values in the list and test your program

patient\_medical\_speciality\_list=[301,'P',302, 'P' ,305, 'P' ,401, 'E' ,656, 'E']

medical\_speciality={"P":"Pediatrics","O":"Orthopedics","E":"ENT"}

speciality=max\_visited\_speciality(patient\_medical\_speciality\_list,medical\_speciality)

print(speciality)

Assignment 33:

Write a python program to display all the common characters between two strings. Return -1 if there are no matching characters.

**Note**: Ignore blank spaces if there are any. Perform case sensitive string comparison wherever necessary.

|  |  |
| --- | --- |
| **Sample Input** | **Expected output** |
| "I like Python" "Java is a very popular language" | lieyon |

#PF-Assgn-33

def find\_common\_characters(msg1,msg2):

#Remove pass and write your logic here

msg1=msg1.replace(" ", "")

msg2=msg2.replace(" ", "")

msg1= set(msg1)

msg2 = set(msg2)

word= ''.join(sorted(msg1.intersection(msg2)))

if word=="":

return -1

else:

return word

#Provide different values for msg1,msg2 and test your program

msg1="I like Python"

msg2="Java is a very popular language"

common\_characters=find\_common\_characters(msg1,msg2)

print(common\_characters)