**LAB EXPERIMENT : 1**

**AIM**: Write a program to check whether the given sentence consists of all vowels or not. Here are some of the words: EDUCATION, AUTOMOBILE, EVACUATION, REMUNERATION, REGULATION.

**PROGRAM:**

sentence = input("Enter sentence")

vowel = ['a','e','i','o','u']

cnt=0

for i in str(vowel).lower():

if i in sentence:

cnt+=1

if cnt==5:

print("All vowels exist in given sentence")

else:

print("All vowels do not exist")

OUTPUT:

Enter sentencehello

All vowels do not exist

**LAB EXPERIMENT : 2**

**AIM:** Write a program to find out the list the words that can be formed with the letters 'A', 'K', 'E', 'O','T', 'P', 'N'. from the given list of words. Arun, Varun, Kent, Eat, Pot, net, Peak, Peacock, Zebra, Nato, Toe, Poke, Knife, Peot, Venus, Ant.

**PROGRAM:**

names = ['Arun', 'Varun', 'Kent', 'Eat', 'Pot', 'net', 'Peak', 'Peacock', 'Zebra', 'Nato', 'Toe', 'Poke', 'Knife', 'Peot', 'Venus', 'Ant']

letters = ['A', 'K', 'E', 'O','T', 'P', 'N']

for name in names:

cnt=0

uni = list(set(name.upper()))

for letter in uni:

if letter in letters:

cnt+=1

if cnt==len(name):

print(name)

**OUTPUT**:

Kent

Eat

Pot

net

Peak

Nato

Toe

Poke

Peot

Ant

**EXPERIMENT : 3**

**AIM:** There is a collection of Five Employees data with id, name, basic salary with allowances (TA, DA, HRA, and bonus), number of years experience. Evaluate the following data:

**PROGRAM:**

def computeSalary( basic, grade):

    hra = 0.2 \* basic

    da = 0.5 \* basic

    pf = 0.11 \* basic

    # Condition to compute the

    # allowance for the person

    if grade == 'A':

        allowance = 1700.0

    elif grade == 'B':

        allowance = 1500.0

    else:

        allowance = 1300.0;

gross = round(basic + hra + da +allowance - pf)

    return gross

# Driver code

if \_\_name\_\_ == '\_\_main\_\_':

    basic = 10000

    grade = 'A'

    # Function call

    print(computeSalary(basic, grade));

**OUTPUT:**

**17600**

**EXPERIMENT 4:**

AIM: Write a Python program to sort the names in a dictionary based on a character at particular position of the name. Eg: {1:”Rama”, 2:”Raju”,3:”Ravi”} If position=3rd, output=[“Raju”, “Rama”,” Ravi”}

**PROGRAM:**

names = {}

n = int(input('Enter number of names'))

for i in range(n):

names[i+1]=input('Enter a name')

name = list(names.values())

pos = int(input('Enter position'))

name.sort(key=lambda x:x[pos])

print(name)

**OUTPUT:**

**Enter number of names3**

**Enter a nameWELCOME**

**Enter a nameHELLO**

**Enter a nameHAI**

**Enter position2**

**['HAI', 'WELCOME', 'HELLO']**

**EXPERIMENT 5:**

AIM: Write a program to count the numbers of characters in the string and store them in a dictionary data structure

**PROGRAM:**

s=input('Enter a string')

d={}

if s.isspace():

print('String is empty')

elif s.isdigit():

print('Enter characters only')

else:

for i in s:

d[i]=s.count(i)

print(d)

**OUTPUT:**

**Enter a stringWELCOME**

**{'W': 1, 'E': 2, 'L': 1, 'C': 1, 'O': 1, 'M': 1}**

**EXPERIMENT 6:**

AIM: Demonstrate matrix addition, multiplication of any two matrices using numpy.

**PROGRAM:**

import numpy as np

row1,col1=list(map(int,input('Enter row and col').split()))

A=np.array(list(map(int,input('Enter matrix A').split())))

A.shape=(int(row1),int(col1))

print(A)

row2,col2=list(map(int,input('Enter row and col').split()))

B=np.array(list(map(int,input('Enter matrix B').split())))

B.shape=(int(row2),int(col2))

print(B)

if(row1==row2 and col1==col2):

print("Addition=",np.add(A,B))

print("Element by element multiplication",np.multiply(A,B))

else:

print("Matrices addition and scalar multiplication are not possible")

if(col1==row2):

print("Multiplication=",np.dot(A,B))

else:

print("Multiplication is not possible")

**OUTPUT:**

**Enter row and col2 3**

**Enter matrix A1 2 3 4 5 6**

**[[1 2 3]**

**[4 5 6]]**

**Enter row and col2 2**

**Enter matrix B2 3 4 5**

**[[2 3]**

**[4 5]]**

**Matrices addition and scalar multiplication are not possible**

**Multiplication is not possible**

**EXPERIMENT 7:**

AIM: Write a Python program to compute the eigen values and right eigen vectors of a given square array.

**PROGRAM:**

import numpy as np

row,col=list(map(int,input('Enter row and col').split()))

A=np.array(list(map(int,input('Enter matrix A').split())))

A.shape=(int(row),int(col))

print(A)

if(row==col):

print(np.linalg.eig(A))

print(np.linalg.eigvals(A))

else:

print("Eigen values are calculated only for square matrices”)

**OUTPUT:**

Enter row and col3 3

Enter matrix A1 2 3 8 0 4 7 8 3

[[1 2 3]

[8 0 4]

[7 8 3]]

(array([11.21822293+0.j,-3.60911146+1.39640552j,-3.60911146-1.39640552j]), array([[ 0.33274217+0.j , 0.258239 +0.22097286j, 0.258239 -0.22097286j], [ 0.51821194+0.j , 0.41578024-0.3289411j ,

0.41578024+0.3289411j ], [ 0.78786993+0.j , -0.77679351+0.j , -0.77679351-0.j ]]))

[11.21822293+0.j -3.60911146+1.39640552j -3.60911146-1.39640552j]

**EXPERIMENT 8:**

AIM: Write a Numpy program to swap rows and columns of a given array in reverse order.

**PROGRAM:**

import numpy as np

row,col=list(map(int,input('Enter row and col').split()))

A=np.array(list(map(int,input('Enter matrix A').split())))

A.shape=(int(row),int(col))

print("Given Array=",A)

print("Transpose=",np.transpose(A))

**OUTPUT:**

Enter row and col2 3

Enter matrix A45 67 2 73 23 84

Given Array= [[45 67 2]

[73 23 84]]

Transpose= [[45 73]

[67 23]

[ 2 84]]

**EXPERIMENT 9:**

AIM: Demonstrate the car sales of a show room during the last 6 months with the help of bar chart. (Use matplotlib) demonstrate.

**PROGRAM:**

from matplotlib import pyplot as plt

import numpy as np

x1=['audi','benz','bmw','jaguar','Landrover']

y1=[200,250,280,265,285]

x2=['ikon','figo','figo aspire','eco sport','endeavor']

y2=[190,160,175,165,185]

x3=['Suzuki','Brezza','XL6','Baleno','Scross']

y3=[110,120,135,145,155]

plt.figure(figsize=(15,6))

plt.title('Cars Sales Information')

plt.bar(x1,y1,color='r',label='2019',align='center')

plt.bar(x2,y2,color='m',label='2020',align='center')

plt.bar(x3,y3,color='g',label='2021',align='center')

plt.xlabel('Car Models')

plt.ylabel('Sales in Crores')

plt.legend(loc='upper right')

plt.xticks(rotation=90)

plt.yticks(rotation=90)

plt.show()

**OUTPUT**:

Chart, bar chart

Description automatically generated

**EXPERIMENT 10:**

AIM: Demonstrate the pass percentage of students joined in 2019-20 academic year in I-II semester with the help of bar chart, pie chart, scatter plot.

**PROGRAM:**

import pandas as pd

# Read the file

data = pd.read\_excel("E:/Academics/Python/Advanced Python

Programming/Marks.xlsx",sheet\_name='Sheet1')

import matplotlib.pyplot as plt

fig, (ax1,ax2,ax3,ax4) = plt.subplots(nrows=4,ncols=1,

figsize=(15,15))

plt.title("Student Grades")

ax1.plot(data.iloc[0:10,0],data.iloc[0:10,4],'-.c')

ax2.bar(data.iloc[0:10,0],data.iloc[0:10,4],color='orange')

ax3.pie(data.iloc[0:10,4], labels=data.iloc[0:10,0],

autopct="%1.1f%%", startangle=90)

ax4.scatter(data.iloc[0:10,0],data.iloc[0:10,4], color=‘magenta')

**EXPERIMENT 11:**

AIM: Find out number of lines, words, characters in a given text file.

**PROGRAM:**

filename=input('Enter a file name')

f=open(filename,'r')

lcount=wcount=ccount=0

for line in f:

lcount+=1

wcount+=len(line.split())

for l in line:

if(l!=' ' and l!='\n'):

ccount+=1

print('No. of Lines=',lcount)

print('No. of Words=',wcount)

print('No. of Characters=',ccount)

f.close()

**OUTPUT:**

**Enter a file namesample.txt**

**No. of Lines= 2**

**No. of Words= 15**

**No. of Characters= 68**

**EXPERIMENT 12:**

AIM: Accumulate all words into the respective alphabet buckets from the given text file.

**PROGRAM:**

filename=input('Enter a file name ')

f=open(filename,'r')

alphabet\_buckets={}

for word in f.read().split():

if(word[0].isalpha()):

temp=word.lower()

if(temp[0] not in alphabet\_buckets.keys()):

alphabet\_buckets[temp[0]]=[]

alphabet\_buckets[temp[0]].append(temp)

else:

alphabet\_buckets[temp[0]].append(temp)

print(alphabet\_buckets)

f.close()

**OUTPUT:**

Enter a file name sample.txt

{'h': ['hello'], 'm': ['my', 'my'], 'n': ['name'], 'i': ['is', 'i'], 'b': ['bhargav'], 'v': ['venkat,'], 'a': ['am', 'advanced'], 't': ['testing'], 'p': ['python', 'program'], 'l': ['lab'], 'c': ['cycle']}