EXPERIMENT NO : 1 Program to find the largest of three numbers

print("Enter the three numbers : ")

a = int(input())

b = int(input())

c = int(input())

if a >= b and a >= c:

print(a, "is the largest")

elif b >= c and b >= a :

print(b, "is the largest")

else:

print(c, "is the largest")

ALGORITHM:

1.Start

2. Initialize three numbers a, b, c.

3. If a is higher than b and c then, print a.

4. Else if b is greater than c and a then, print b.

5. Else if c is greater than a and b then, print c.

6.Stop

OUTPUT :

Enter the three numbers :

4

3

5

5 is the largest

EXPERIMENT NO : 2 Program to print the multiplication table of a number

n = int(input("Enter the number to print the table of : "))

for i in range(1, 11) :

print(n, " X ", i, " = ", i \* n)

ALGORITHM:

1.Start

2.Input the number for which the multiplication table is to be generated.

3.set end = 0

4.Repeat from i = 1 to end

5.Display the table values in the given output format (num \* i = num\*i).

6.Stop

OUTPUT :

Enter the number to print the table of : 18

18 X 1 = 18

18 X 2 = 36

18 X 3 = 54

18 X 4 = 72

18 X 5 = 90

18 X 6 = 108

18 X 7 = 126

18 X 8 = 144

18 X 9 = 162

18 X 10 = 180

EXPERIMENT NO : 3 Program to print the surface area and volume of a cylinder

def sa(r, h):

return 2 \* 3.14 \* r \* r + 2 \* 3.14 \* r \* h

def v(r, h):

return 3.14 \* r \* r \* h

r = int(input("Enter the radius of the cylinder : "))

h = int(input("Enter the height of the cylinder ; "))

print("The total surface area of the cylinder is : ", sa(r, h))

print("The volume of the cylinder is : ", v(r, h))

ALGORITHM :

1.Start

2.Input the base radius (r) and the height (h)

3.Set sa = 2 \* 3.14 \* r \* r + 2 \* 3.14 \* r \* h

4.Set v = 3.14 \* r \* r \* h

5.Print the value of sa and v

6.Stop

OUTPUT :

Enter the radius of the cylinder : 4

Enter the height of the cylinder ; 12

The total surface area of the cylinder is : 401.92

The volume of the cylinder is : 602.88

EXPERIMENT NO : 4 Program to replace a word by another in a sentence

sltr = input("Enter the string : ")

rep = input("Enter the string to be replaced : ")

swap = input("Enter the replacement : ")

while sltr.find(rep) != - 1:

index = sltr.find(rep)

sltr = sltr[:index] + swap + sltr[index + len(rep):]

print("The new string is : ", sltr)

ALGORITHM :

1.Start

2.Input the sentence (sltr), the word to replace (rep) and the replacement (swap)

3.If rep is present in sltr repeat steps 4 and 5

4.Set index = beginning index of rep

5.Set sltr = characters of sltr upto index + swap + characters in sltr from (index + length of rep) upto the end of sltr

6.Print the sentence

7.Stop

Output:

Enter the string : hello there

Enter the string to be replaced : hello

Enter the replacement : hey

The new string is : hey there

Experiment No : 5 Program to check for the validity of an email address

import re

regex = r'\b[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b'

s = input("Enter the email id : ")

if(re.fullmatch(regex, s)):

print("The email address is valid")

else:

print("The email address is invalid")

ALGORITHM :

1.Start

2.Import re package and create regex variable for comparison

3.Input the email (s)

4.If email matches the regex variable print that it is valid and go to step 6

5.Print that the email is invalid

6.Stop

Output :

Enter the email id : dashi@gmail.com

The email address is valid

Experiment No : 6 Program to remove all occurrences of a number from a list

a = list(map(int, input("Enter the list elements : ").split()))

for i in a :

count = 0

for k in a :

if k == i :

count = count + 1

for j in range(1, count):

a.remove(i)

print("The modified list is : ", end = " ")

print(a)

ALGORITHM :

1.Start

2.Input the list a

3.For each element (i) in (a) repeat steps 4 to 8

4.Set count = 0

5.For each element (k) in (a) repeat step 6

6.If k == i, set count = count + 1

7.Iterate from j = 1 up to j = count – 1 and repeat step 8

8.Remove (i) from (a)

9.Stop

Output :

Enter the list elements : 4 3 2 1 1 2 2 3 4 5

The modified list is : [1, 2, 3, 4, 5]

Experiment No : 7 Program to add two matrices

print("Enter no of rows and columns of matrix 1 : ")

r1 = int(input())

c1 = int(input())

a = []

b = []

print("Enter the elements of matrix 1 : ")

for i in range(r1) :

temp = []

for j in range(c1):

temp.append(int(input()))

a.append(temp)

print("Enter no of rows and columns of matrix 2 : ")

r2 = int(input())

c2 = int(input())

print("Enter the elements of matrix 2 : ")

for i in range(r2):

temp = []

for j in range(c2):

temp.append(int(input()))

b.append(temp)

c = []

if r1 == r2 and c1 == c2 :

for i in range(r1):

temp = []

for j in range(c1):

x = a[i][j] + b[i][j]

temp.append(x)

c.append(temp)

print("The sum matrix is : ")

for i in c :

print(i)

else:

print("The matrices are incompatible")

ALGORITHM :

1.Start

2.Input matrix 1 and matrix 2.

3.If the number of rows and number of columns of matrix 1 and matrix 2 is equal,

4.for i=1 to rows[matrix 1]

5.for j=1 to columns [matrix 1]

6.Input matrix 1 [i,j]

7.Input matrix 2 [i,j]

8.matrix 3 [i,j]= matrix 1 [i,j]+ matrix 2 [i,j]

9.Display matrix 3 [i,j]

10.Stop

Output :

Enter no of rows and columns of matrix 1 :

2

2

Enter the elements of matrix 1 :

1

2

3

4

Enter no of rows and columns of matrix 2 :

2

2

Enter the elements of matrix 2 :

4

3

2

1

The sum matrix is :

[5, 5]

[5, 5]

Experiment No : 8 Program to print even and odd tuple

a = tuple(map(int, input("Enter the tuple elements : ").split()))

print("Odd positions are : ", a[1::2])

print("Even positions are : ", a[::2])

ALGORITHM :

1.Start

2.Input the tuple elements (a)

3.For each element (i) in (a) repeat step 4

4.If index of (i) % 2 == 0 , print (i) under even positions

5.For each element (i) in (a) repeat step 6

6.If index of (i) % 2 != 0 , print (i) under odd positions

7.Stop

Output :

Enter the tuple elements : 1 2 3 4 5 6

Odd positions are :

[2, 4, 6]

Even positions are :

[1, 3, 5]

Experiment No : 9 Program to implement dictionary in python

def add\_book(books, title, stock):

if title in books :

print("Book already exists")

return

books[title] = stock

def update\_shelf(books):

print("1.Increase stock : ")

print("2.Decrease stock : ")

ch = int(input())

title = input("Enter the title of book to update stock of : ")

if ch == 1 :

print("Enter the amount to increase by : ")

inc = int(input())

books[title] = books[title] + inc

if ch == 2 :

print("Enter the amount to decrease by : ")

dec = int(input())

books[title] = books[title] - dec

def remove\_book(books, title):

test = books.pop(title, "Book not present")

if test == "Book not present":

print("Book not present")

def display(books):

print("Books Stock")

for i in books:

print(i, end = " ")

print(books[i])

books = {}

i = 1

while i == 1 :

print("MENU")

print("1.Add Book")

print("2.Delete Book")

print("3.Update Stock")

print("4.Display Books")

x = int(input())

if x == 1 :

print("Enter the title of the book to add : ")

title = input()

print("Enter the corresponding stock : " )

stock = int(input())

add\_book(books, title, stock)

if x == 2 :

print("Enter the title of the book to remove : ")

title = input()

remove\_book(books, title)

if x == 3 :

update\_shelf(books)

if x == 4 :

display(books)

print("Return to menu : 1 / 0 " )

i = int(input())

ALGORITHM :

1.Start

2.Create dictionary books

3.Print the menu

4.Input the user choice (ch)

5.If ch == 1, perform steps 6 through 8

6.Input the title of the book and its corresponding stock

7.If title already exists, print the same

8.If title is new, then add it to the dictionary

9.if ch == 2, perform steps 10 through 14

10.Input the choice of increasing / decreasing stock (x)

11.Input the title of book to perform the updation on

12.Input the increment / decrement value

13.If x == 1, add the value to the corresponding book

14.If x == 2, decrease the value from the corresponding book

15.If ch == 3, perform steps 16 and 17

16.Input the title of book to remove

17.Pop the book with the corresponding title from the dictionary

18.If ch == 4, print the dictionary contents

19.Input the user choice to return to menu (x)

20.If x == 1, goto step 3

21.Stop

Output :

MENU

1.Add Book

2.Delete Book

3.Update Stock

4.Display Books

1

Enter the title of the book to add :

TalesofKris

Enter the corresponding stock :

21

Return to menu : 1 / 0

1

MENU

1.Add Book

2.Delete Book

3.Update Stock

4.Display Books

1

Enter the title of the book to add :

Chroma

Enter the corresponding stock :

2

Return to menu : 1 / 0

1

MENU

1.Add Book

2.Delete Book

3.Update Stock

4.Display Books

3

1.Increase stock :

2.Decrease stock :

1

Enter the title of book to update stock of : Chroma

Enter the amount to increase by :

2

Return to menu : 1 / 0

1

MENU

1.Add Book

2.Delete Book

3.Update Stock

4.Display Books

4

Books Stock

TalesofKris 21

Chroma 4

Return to menu : 1 / 0

1

MENU

1.Add Book

2.Delete Book

3.Update Stock

4.Display Books

2

Enter the title of the book to remove :

TalesofKris

Return to menu : 1 / 0

1

MENU

1.Add Book

2.Delete Book

3.Update Stock

4.Display Books

4

Books Stock

Chroma 4

Return to menu : 1 / 0

1

MENU

1.Add Book

2.Delete Book

3.Update Stock

4.Display Books

2

Enter the title of the book to remove :

Chroma

Return to menu : 1 / 0

1

MENU

1.Add Book

2.Delete Book

3.Update Stock

4.Display Books

4

Books Stock

Return to menu : 1 / 0

0

Experiment No : 10 Program to read numbers from a file and print the prime numbers

f = open("numbers.txt", "r")

lines = f.readlines()

print("The prime numbers are : ")

for l in lines :

a = map(int,l.split(" "))

for c in a :

if c == 2 or c == 1 :

print(c)

else :

i = 2

flag = 1

for i in range(2, c):

if c % i == 0 :

flag = 0

break

if flag == 1:

print(c)

f.close()

ALGORITHM :

1.Start

2.Open the file in read only mode

3.Convert the file contents onto a list of strings (lines)

4.For each list (l) in list (lines) repeat steps 5 through 7

5.Convert the string onto a list of integers (a)

6.For each element of list (a) repeat step 7

7.If the element if prime, print the element

8.Close the file

9.Stop

Output :

The prime numbers are :

431

1

2

3

5

Experiment No : 11 Program to count letters, words, lines and special characters in a file

f = open("testtext.txt", 'r')

lines = f.readlines()

s, w, ul, ll, ss = 0, 0, 0, 0, 0

spclsym = ['\\','~','`','!','@','#','$','%','^','&','\*','(',')','-','\_','=','+',"|","{","}","[","]",";",":","'",'"',",","<",".",">","/","?"]

for line in lines :

s = s + 1

for c in line :

if c.isupper() :

ul = ul + 1

if c.islower() :

ll = ll + 1

if c in spclsym :

ss = ss + 1

l = list(line.split(" "))

for i in l :

if i != "":

w = w + 1

f.close()

print("Number of words : ", w)

print("Number of lines : ", s)

print("Number of letters : ", ll + ul)

print("Number of lowercase letters : ", ll)

print("Number of uppercase letters : ", ul)

print("Number of special symbols : ", ss)

ALGORITHM :

1.Start

2.Open the file in read only mode

3.Convert the contents of the file into a list of strings (lines)

4.Create variable s, ss, ul, ll and w for storing the various attributes

5.For each list (line) in list (lines) , repeat step 6 through

6.Set s = s + 1

7.For each character in list (line) , repeat steps 8 through 10

8.If character is uppercase, set ul = ul + 1

9.If character is lowercase, set ll = ll + 1

10.If character is a symbol, set ss = ss + 1

11.Split the list (line) into a list of words (a)

12.For each element of a, repeat step 13

13.If the element is not empty, set w = w + 1

14.Print the variables w, ul, ll, ul + ll, ss and s

15.Stop

OUTPUT :

Number of words : 6

Number of lines : 1

Number of letters : 0

Number of lowercase letters : 0

Number of uppercase letters : 0

Number of special symbols : 0

Experiment No : 12 Program to plot the graphs of various mathematical formula

import matplotlib.pyplot as plt

def fact(x):

if x == 0 or x == 1:

return 1

return x \* fact(x - 1)

def sin(x):

j = 1

val = 0

for i in range(1, 20):

term = (x \*\* j) / fact(j)

if i % 2 == 0:

val = val - term

else:

val = val + term

j = j + 2

return val

print("MENU")

print("1.y = x")

print("2.y = x^2")

print("3,y = sin(x)")

print("4,y = 2^x")

print("5.y = x^0.5")

print("6.EXIT")

while 1:

x = int(input("Enter the Option : "))

xc = []

floater = -10

while floater < 10 :

xc.append(floater)

floater = floater + 0.1

if x == 1:

yc = [i for i in xc]

elif x == 2:

yc = [i \* i for i in xc]

elif x == 3:

yc = [sin(i) for i in xc]

elif x == 4:

yc = [2\*\*i for i in xc]

elif x == 5:

xc = range(0, 50)

yc = [i\*\*0.5 for i in xc]

else:

break

plt.plot(xc, yc)

plt.show()

ALGORITHM :

1.Start

2.Print the menu

3.Input the selection variable (ch)

4.Create a list (x) with values in the range (-10, 10)

5.If ch == 1, Create list y as y[i] = x[i]

6.If ch == 2, Create list y as y[i] = x[i] \* x[i]

7.If ch == 3, Create list y as y[i] = sin(x[i])

8.If ch == 4, Create list y as y[i] = 2 ^ x[i]

9.If ch == 5, Create list y as y[i] = x[i] ^ 0.5 , for x[i] >= 0

10.Invoke the plot function of matplotlib with x and y as parameters

12.Invoke the show() function

14.Input the user choice (x) for returning to menu

15.If x == 1, go to step 2

16.Stop

Output :

MENU

1.y = x

2.y = x^2

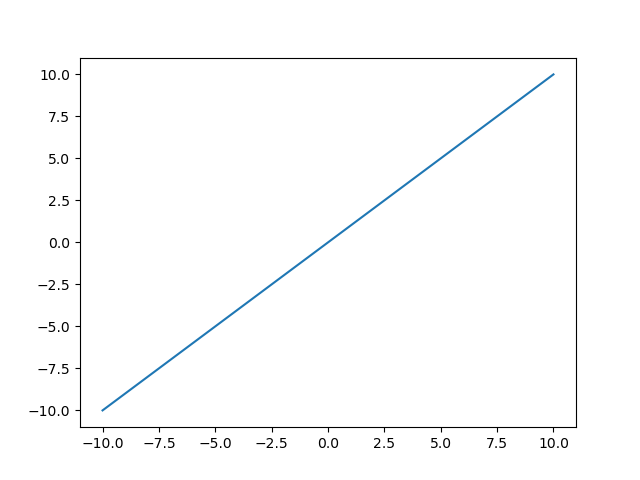
3,y = sin(x)

4,y = 2^x

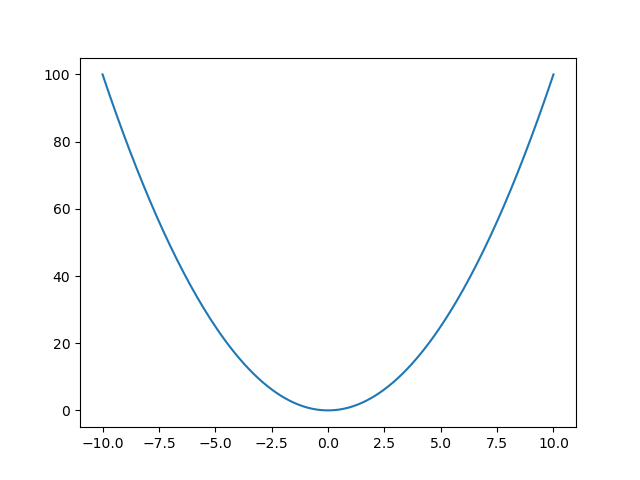
5.y = x^0.5

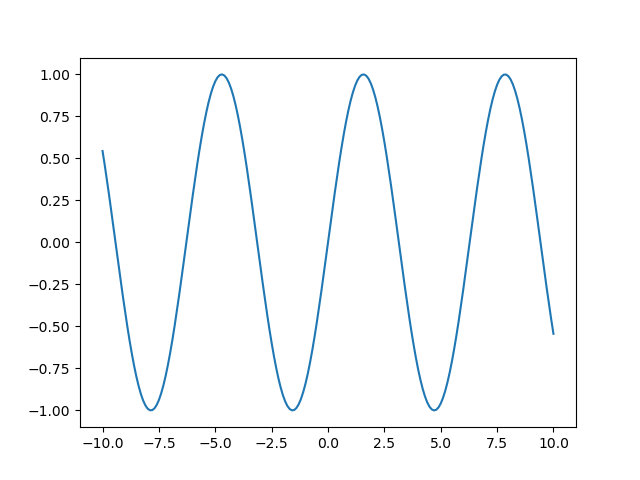
6.EXIT

Enter the Option: 1

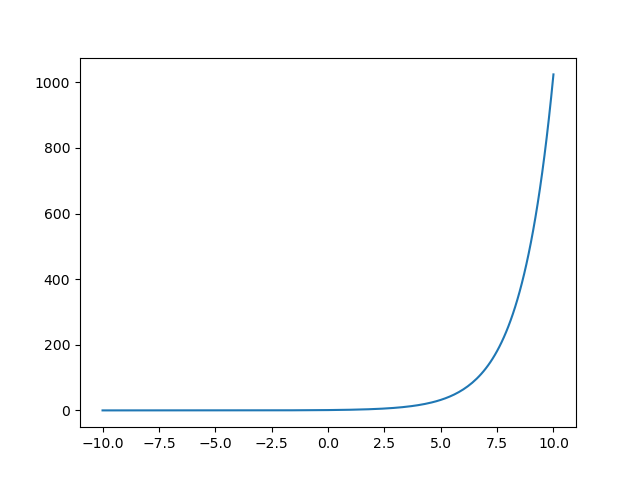


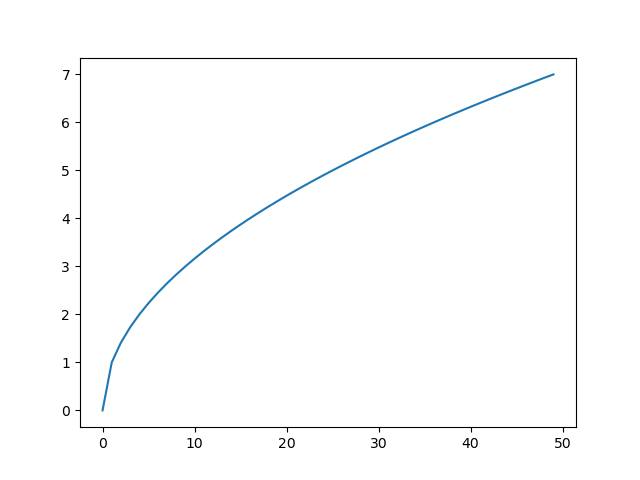
Enter the Option : 2

Enter the Option : 3



Enter the Option : 4



Enter the Option : 5

Enter the Option : 6

Experiment NO : 13 Program to plot bar graph of continental area

import matplotlib.pyplot as plt

areas = {"Africa" : 11.7, "Asia" : 10.4, "Europe" : 1.9, "North America" : 9.4, "Oceania" : 3.3, "South America" : 6.9, "Soviet Union" : 7.9}

plt.title("Continent area bar graph")

cont = list(areas.keys())

val = list(areas.values())

for i in range(0, 7):

plt.text(i, val[i], val[i])

plt.bar(areas.keys(), areas.values())

plt.xlabel("Continents")

plt.ylabel("Area(In million sq kms)")

plt.show()

ALGORITHM :

1.Start

2.Create the dictionary with the name and area as keys and values respectively

3.Set cont = list of dictionary keys

4.Set val = list of dictionary values

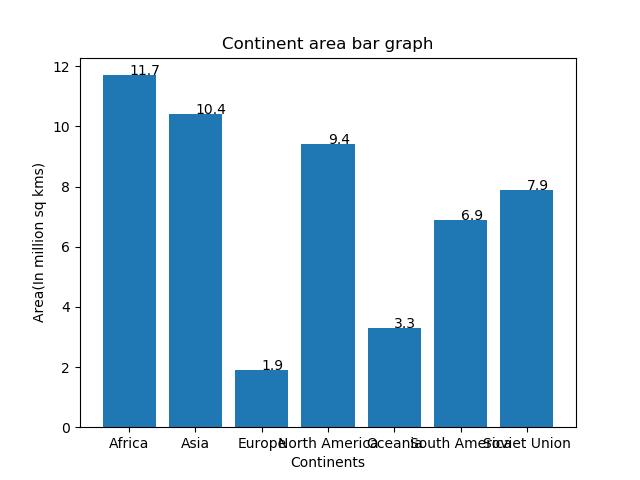
5.For each element in val, invoke the name the corresponding coordinate(i, val[i])

6.Invoke the bar() function to plot the bar graph

7Invoke the show() function

8.Stop

Output :



Experiment No : 14 Program to plot a histogram with the given data

import matplotlib.pyplot as plt

plt.hist([135]\*4 +[140]\*12 +[145]\*16 +[150]\*8,[135,140,145,150,155])

plt.xlabel("heights")

plt.ylabel("No")

plt.show()

ALGORITHM :

1.Start

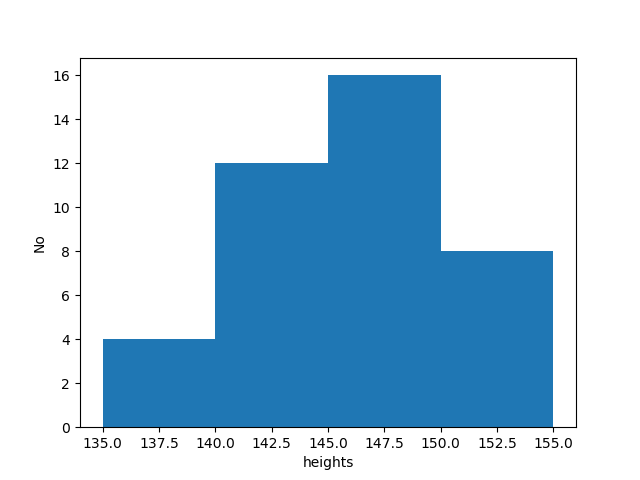
2.Invoke the hist() function of matplotlib with the corresponding height and list of x coordinates

3.Set the value of x axis label and y axis label

4.Invoke the show() function

5.Stop

Output:



Experiment No : 15 Calculation of mean, median and mode

table = [["Alabama", 4779736, 5.7], ["Alaska", 710231, 5.6], ["Arizona", 6392017, 4.7], ["Arkansas", 2915918, 5.6], ["California", 37253956, 4.4], ["Colorado", 5029196, 2.8], ["Conneticut", 3574097, 2.4], ["Delaware", 897934, 5.8]]

print("Input Data")

print("{:<12}{:<14}{:<12}".format("State","Population","Murderrate(per 100000)"))

for i in range(8):

print("{:<12}{:<14}{:<12}".format(table[i][0],table[i][1],table[i][2]))

mean = 0

median = 0

variance = 0

mtable = []

for i in range(8) :

x = table[i][1] / 100000

mtable.append(x \* table[i][2])

mean = mean + mtable[i]

mean = mean / 8

mtable.sort()

median = (mtable[3] + mtable[4]) / 2

#variance evaluation

for i in range(8):

x = (mean - mtable[i]) \*\* 2;

variance = variance + x

variance = variance / 8

print("\nThe mean of the data is : ", round(mean,2))

print("The median of the data is : ", round(median, 2))

print("The variance of the data is : ", round(variance, 2))

ALGORITHM :

1.Start

2.Create a list (table) containing the data , mean = 0, median = 0 and variance = 0

3.Create a new list (mtable)

4.Iterate from i = 0 up to i = 7, repeat steps 5 through

5.Set mtable[i] = (table[i][1] / 100000) \* table[i][2]

6.Set mean = sum of all elements in mtable / 8

7.Sort the list mtable

8.Set median = (mtable[3] + mtable[4] ) / 2

9.For each element (i) of mtable, repeat step 10

10. Set variance = variance + ( i – mean) ^ 2

11.Print the value of mean, median and mode

OUTPUT :

Input Data

State Population Murderrate(per 100000)

Alabama 4779736 5.7

Alaska 710231 5.6

Arizona 6392017 4.7

Arkansas 2915918 5.6

California 37253956 4.4

Colorado 5029196 2.8

Conneticut 3574097 2.4

Delaware 897934 5.8

The mean of the data is : 336.72

The median of the data is : 152.05

The variance of the data is : 250307.45

Experiment No : 16 Calculation of Standard Deviation and Coefficient of variation

t = [[0, 10, 5], [10, 20, 10], [20, 30, 20], [30, 40, 40], [40, 50, 30], [50, 60, 20], [60, 70, 10], [70, 80, 5]]

print("Input Data")

print("Class ",end = " ")

for i in range(8):

print(t[i][0], "-", t[i][1], end = " ")

print("")

print("Frequency ", end = " ")

for i in range(8):

print(t[i][2], end = " ")

mean = 0

fisum = 0

for i in range(8):

x = (t[i][1] + t[i][0]) / 2

mean = mean + (x \* t[i][2])

fisum = fisum + t[i][2]

mean = mean / fisum

variance = 0

for i in range(8):

x = (t[i][1] + t[i][0])/2

x = (x - mean) \*\* 2

variance = variance + (x\* t[i][2])

variance = variance / fisum

sd = variance \*\* 0.5

print("\n\nThe standard deviation of the data is : ", sd)

print("The Coefficient of Variation(CV) of the data is : ", sd/mean)

ALGORITHM:

1.Start

2.Create a list (t) with the classes and their frequency

3.Set mean = 0, fsum = 0

4.Iterate from i = 0 up to i = 7, repeat steps 5 through 7

5.Set x = (t[i][1] + t[i][0]) / 2

6.Set mean = mean + (x \* t[i][2])

7.Set fsum = fsum + t[i][2]

8.Set mean = mean / fsum

9.Set variance = 0

10.Iterate from i = 0 up to i = 7, repeat steps 11 through 13

11.Set x = (t[i][1] + t[i][0]) / 2

12.Set x = (x – mean) ^ 2

13.Set variance = variance + (x \* t[i][2])

14.Set variance = variance / fsum

14.Print the value of mean and CV = (variance ^ 0.5) / mean

15.Stop

OUTPUT :

Input Data

Class 0 - 10 10 - 20 20 - 30 30 - 40 40 - 50 50 - 60 60 - 70 70 - 80

Frequency 5 10 20 40 30 20 10 5

The standard deviation of the data is : 15.91992077715905

The Coefficient of Variation(CV) of the data is : 0.4015835871715796

Experiment No : 17 Probability of random variable in binomial distribution

def fact(n):

if n == 0 or n == 1:

return 1

return n \* fact(n - 1)

def combi(n,r):

return fact(n)/(fact(r)\*fact(n - r))

def bnp(n, p, x):

return combi(n, x)\*(p \*\* x)\*((1 - p) \*\* (n - x))

n = 6

p = 0.25

print("No of trials : ", n)

print("Probability of Success : ", p)

e4 = bnp(n, p, 4)

a1 = 1 - bnp(n, p, 0)

print("Probability of exactly 4 success : ", round(e4,4));

print("Probability of atleast 1 success : ", round(a1,4));

ALGORITHM :

1.Start

2.Set the value of n and p

3.Print the value of exactly 4 success as C(n, 4)\*(p ^ 4)\*((1-p)^(n-x))

4.Print the value of at least 1 success as 1 – C(n, 1)\*(p ^ 1)\*((1-p)^(n-x))

5.Stop

OUTPUT :

No of trials : 6

Probability of Success : 0.25

Probability of exactly 4 success : 0.033

Probability of atleast 1 success : 0.822

Experiment No : 18 Probability of random variable in poisson distribution

def fact(n):

if n == 1 or n == 0 :

return 1

return n \* fact(n - 1)

def p(x, m):

return round(((e \*\* -m)\*(m \*\* x))/fact(x), 3)

mean = 3.4

x = 6

e = 2.7183

print("The mean of the distribution is : ", mean)

print("The required number of trials is : ", x)

print("The probability of P(X = 6) is : ", p(x, mean))

ALGORITHM :

1.Start

2.Set the value of mean and the mathematical constant e

3.Print the value of ((e ^ -mean)\*(mean ^ 6))/6!

4.Stop

OUTPUT :

The mean of the distribution is : 3.4

The required number of trials is : 6

The probability of P(X = 6) is : 0.072

Experiment No : 19 Checking independence using chi-square value

t = [[60, 54, 46, 41, 201],[40, 44, 53, 57, 194], [100, 98, 99, 98, 395]]

print("{:^8}{:^12}{:^12}{:^12}{:^12}{:^12}".format("", "HighSchool", "Bachelors", "Masters", "Phd", "Total"))

for i in range(3):

    if i == 0:

        print("{:^8}{:^12}{:^12}{:^12}{:^12}{:^12}".format("Female",t[i][0], t[i][1], t[i][2], t[i][3], t[i][4]))

    elif i == 1:

        print("{:^8}{:^12}{:^12}{:^12}{:^12}{:^12}".format("Male",t[i][0], t[i][1], t[i][2], t[i][3], t[i][4]))

    else:

        print("{:^8}{:^12}{:^12}{:^12}{:^12}{:^12}".format("",t[i][0], t[i][1], t[i][2], t[i][3], t[i][4]))

calcchi = 0

tablechi = 7.815

for i in range(2):

    for j in range(4):

        ob = t[i][j]

        exp = (t[i][4] \* t[2][j]) / t[2][4]

        t[i][j] = round(exp, 2)

        val = (((ob - exp) \*\* 2)) / exp

        calcchi = calcchi + val

print("\nExpected values : ")

print("{:^8}{:^12}{:^12}{:^12}{:^12}".format("", "HighSchool", "Bachelors", "Masters", "Phd"))

for i in range(2):

    if i == 0:

        print("{:^8}{:^12}{:^12}{:^12}{:^12}".format("Female",t[i][0], t[i][1], t[i][2], t[i][3]))

    elif i == 1:

        print("{:^8}{:^12}{:^12}{:^12}{:^12}".format("Male",t[i][0], t[i][1], t[i][2], t[i][3]))

print("\n\nThe calculated value of chi-square is : ", round(calcchi,3))

print("The table value of chi-square is : ", tablechi)

if tablechi < calcchi:

    print("Therefore relationship is dependent")

else:

    print("Therefore relationship is independent")

ALGORITHM :

1.Start

2.Set calcchi = 0

3.Set tablechi = corresponding mathematical value for given conditions

4.Create a list containing the input data

5.Iterate from I = 0 up to I = 1 and repeat steps 6 through

6.Iterate from j = 0 up to I = 3 and repeat steps 7 through

7.Set ob = t[i][j]

8.Set exp = (t[i][4] \* t[2][j]) / t[2][4]

9.Set calcchi = calcchi + (((ob - exp) ^ 2)) / exp

10.If tablechi < calcchi print that the relationship is dependent, go to step 12

11.Print that the relationship is independent

12.Stop

OUTPUT :

HighSchool Bachelors Masters Phd Total

Female 60 54 46 41 201

Male 40 44 53 57 194

100 98 99 98 395

Expected values :

HighSchool Bachelors Masters Phd

Female 50.89 49.87 50.38 49.87

Male 49.11 48.13 48.62 48.13

The calculated value of chi-square is : 8.006

The table value of chi-square is : 7.815

Therefore relationship is dependent

Experiment No : 20 Program to find the correlation coefficient of the given data

t = [["A", 17, 150, 0, 0, 0, 0, 0], ["B", 15, 154, 0, 0, 0, 0, 0], ["C", 19, 169, 0, 0, 0, 0, 0], ["D", 17, 172, 0, 0, 0, 0, 0], ["E", 21, 175, 0, 0, 0, 0, 0]]

handm = 0

heightm = 0

for i in range(5):

    handm = handm + t[i][1]

    heightm = heightm + t[i][2]

handm = handm / 5

heightm = heightm / 5

t1 = 0

t2 = 0

t3 = 0

print("{:<7}{:<7}{:<7}{:^7}{:^7}{:^7}{:^7}{:^7}".format("Person","Hand", "Height", "e", "f", "e \* f", "e^2", "f^2"))

for i in range(5):

    t[i][3] = t[i][1] - handm

    t[i][4] = t[i][2] - heightm

    t[i][5] = t[i][3] \* t[i][4]

    t[i][6] = t[i][3] \*\* 2

    t[i][7] = t[i][4] \*\* 2

    t1 = t1 + t[i][5]

    t2 = t2 + t[i][6]

    t3 = t3 + t[i][7]

    print("{:<7}{:<7}{:<7}{:<7.1f}{:<7.1f}{:<7.1f}{:<7.1f}{:<7.1f}".format(t[i][0],t[i][1],t[i][2],t[i][3],t[i][4], t[i][5], t[i][6], t[i][7]))

corr = t1 / ((t2 \* t3) \*\* 0.5)

print("\nThe correlation coefficent is : ", round(corr,3))

ALGORITHM :

1.Start

2.Create a list of the required data values

3.Set handmean = 0, heightmean = 0

4.Set handmean = sum of all hand values

5.Set heightmean = sum of all height values

6.Set handmean = handmean / 5

7.Set heightmean = heightmean / 5

8.Set t1 = t2 = t3 = 0

9.Iterate from i = 0 up to i = 4, repeat steps 10 through 17

10.Set t[i][3] = t[i][1] - handmean

11.Set t[i][4] = t[i][2] - heightmean

12.Set t[i][5] = t[i][3] \* t[i][4]

13.Set t[i][6] = t[i][3] \*\* 2

14.Set t[i][7] = t[i][4] \*\* 2

15.t1 = t1 + t[i][5]

16.t2 = t2 + t[i][6]

17.t3 = t3 + t[i][7]

18. Print the correlation coefficient c = t1 / ((t2 \* t3) ^ 0.5)

19.Stop

Output :

The correlation coefficient is : 0.721

Experiment No : 21 T test for checking NULL hypothesis

smean, ssd, size, pmean, sig = 22, 3, 16, 20, 0.05

z = ((size\*\*0.5)\*(smean-pmean))/ssd

t = 1.753

print("The table value of t is :", t, " and the calculated value of z is : ", round(z,3));

if z > t:

    print("Null hypothesis is rejected as Z is in the rejection region, therefore the new model has a higher mileage")

else:

    print("Null hypothesis is not rejected")

ALGORITHM :

1.Start

2. Set the values of samplemean, samplesize, samplestandarddeviation, populationmean and levelofsignificance

3.Set the value of the T test statistic with respect to the levelofsignificance

4.Set z = ((samplesize ^ 0.5) \* (samplemean – populationmean))/ samplestandarddeviation

5.Stop

Output :

The table value of t is : 1.753  and the calculated value of z is :  2.667

Null hypothesis is rejected as Z is in the rejection region, therefore the new model has a higher mileage