#A1 Write a program in python to add two numbers and print the result.

n1=float(input("Enter 1st number:"))

n2=float(input("Enter 2nd number:"))

sum=n1+n2

print(n1,"+",n2,"=",sum)

A screenshot of a computer

Description automatically generated with medium confidence

#A2 Write a program in python to find the area of a triangle.

base=float(input("Enter base length of the triangle:"))

height=float(input("Enter height of the triangle"))

area=(**1**/**2**)\*base\*height

print("area of triangle=",area)

A screenshot of a computer

Description automatically generated with medium confidence

#A3 Write a program in python to find square root of a number.

n=float(input())

sqrt=n\*\*(**1**/**2**)

print("square root=",sqrt)

Graphical user interface, text, application

Description automatically generated

#A4Write a program in python to solve a quadratic equation

a=int(input("coefficient of x^2:="))

b=int(input("coefficient of x="))

c=int(input("constant="))

d=((b\*\***2**)-**4**\*a\*c)\*\*(**1**/**2**)

x1=(-b+d)/**2**

x2=(-b-d)/**2**

print("x=",x1,x2)

A screenshot of a computer

Description automatically generated with medium confidence

#A5Write a program in python to convert Fahrenheit to Celsius.

tempf=float(input("Enter temperature in fahrenheit:"))

tempc=(tempf-**32**)\*(**5**/**9**)

print(tempf,"°F","=",tempc,"°C")

Graphical user interface, text, application

Description automatically generated

#A6Write a program in python to find quotient and reminder after division

dvdt=int(input("Enter the divident:"))

dvsr=int(input("Enter the divisor:"))

qnt=dvdt//dvsr

rem=dvdt%dvsr

print("Quotient=",qnt,"","remainder=",rem)

A screenshot of a computer

Description automatically generated with medium confidence

#A7Write a program in python to swap two numbers using tuple assignment.

n1=int(input("Enter first number:"))

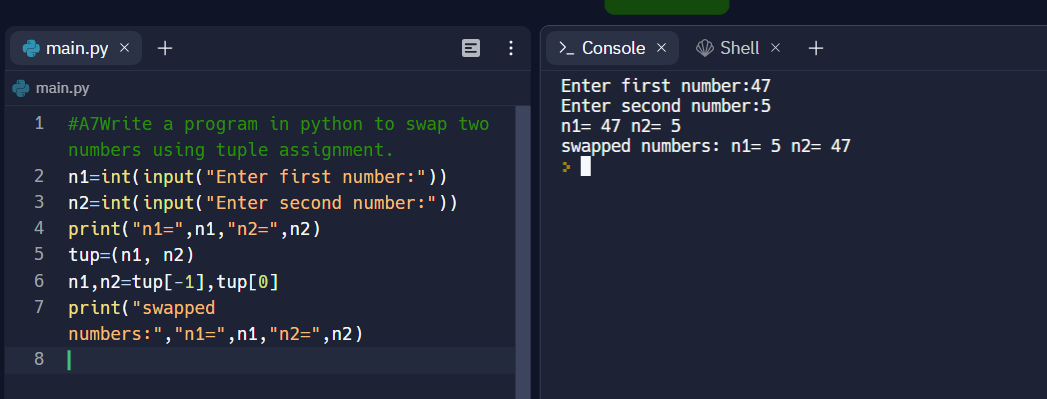
n2=int(input("Enter second number:"))

print("n1=",n1,"n2=",n2)

tup=(n1, n2)

n1,n2=tup[-**1**],tup[**0**]

print("swapped numbers:","n1=",n1,"n2=",n2)



#A8write a program in python to find the average of three marks.

m1=int(input("Enter 1st marks:"))

m2=int(input("Enter 2nd marks:"))

m3=int(input("Enter 3rd marks:"))

avg=(m1+m2+m3)/**3**

print("Average marks=",avg)

A screenshot of a computer

Description automatically generated with medium confidence

#A9Write a program in python to calculate simple interest.

p=int(input("Enter the principle amount:"))

t=int(input("Enter time duration:"))

r=int(input("Enter rate of interest:"))

s=(p\*t\*r)/**100**

print("simple interest=",s)

A screenshot of a computer

Description automatically generated with medium confidence

#A10Write a program in python to calculate the net pay given basic pay, hra, da and deductions.

days=float(input("Enter No Days Present:"))

wages=float(input("Enter wages per Day:"))

basic=wages\*days;

HRA=basic\***0.1**;

DA=basic\***0.05**;

PF=basic\***0.12**;

netsalary=basic+HRA+DA-PF;

print("**\n**Basic:%lf **\n**HRA:%lf **\n**DA:%lf **\n**PF:%lf **\n**Net Salary:%lf" %(basic,HRA,DA,PF,netsalary))

Text, chat or text message

Description automatically generated

#B1Given age determine whether a person is eligible to vote or not. (if else)

age=int(input("Enter your age:"))

**if** age>=**18**:

print("congratulations you are eligible to vote")

**else**:

print("sorry you are not eligible to vote")

A screenshot of a computer

Description automatically generated with medium confidence

#B2Check whether a number is odd or even.

n=int(input("Enter a number:"))

**if** n%**2**==**0**:

print(n,"is an even number")

**else**:

print(n,"is an odd number")

A screenshot of a computer

Description automatically generated with medium confidence

#B3Write a program to find largest of two numbers

n1=float(input("Enter 1st number:"))

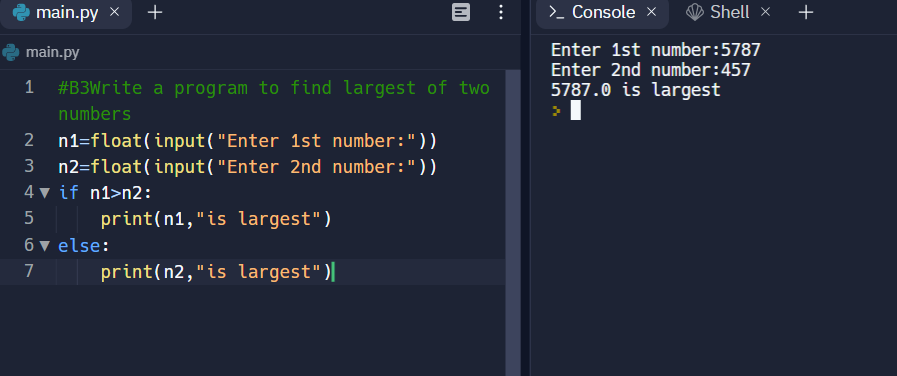
n2=float(input("Enter 2nd number:"))

**if** n1>n2:

print(n1,"is largest")

**else**:

print(n2,"is largest")



#B4Obtain a character convert lower case to uppercase and vice versa.

a=input("Enter a lower case character:")

**if** a.upper()!=a:

print(a.upper())

**else**:

print(a.lower())

Graphical user interface, text, application, chat or text message

Description automatically generated

#B5Find the input year is leap year or not.

y=int(input("Enter a year:"))

**if** y%**4**!=**0**:

print(y,"is not a leap year")

**else**:

**if** y%**100**!=**0**:

print(y,"is a leap year")

**else**:

**if** y%**400**!=**0**:

print(y,"is not a leap year")

**else**:

print(y,"is a leap year")

A screenshot of a computer

Description automatically generated with medium confidence

#B6Read a number, check if it is positive, negative or zero. Increment the number if it is positive, decrement if it is negative.

n=float(input("Enter a number:"))

**if** n<**0**:

print(n,"is negative","decrement=",n-**1**)

**elif** n==**0**:

print(n,"is zero")

**elif** n>**0**:

print(n,"is positive, increment=",n+**1**)

A screenshot of a computer

Description automatically generated with medium confidence

#B7Create a simple calculator

print("1.Addition 2.Subtraction 3.Multiplication 4.Division")

a=int(input("Enter 1,2,3 or 4:"))

n1=float(input("Enter 1st number n1:"))

n2=float(input("Enter 2nd number n2:"))

**if** a==**1**:

sum=n1+n2

print("n1 + n2 =",sum)

**elif** a==**2**:

diff=n1-n2

print("n1 - n2 =",diff)

**elif** a==**3**:

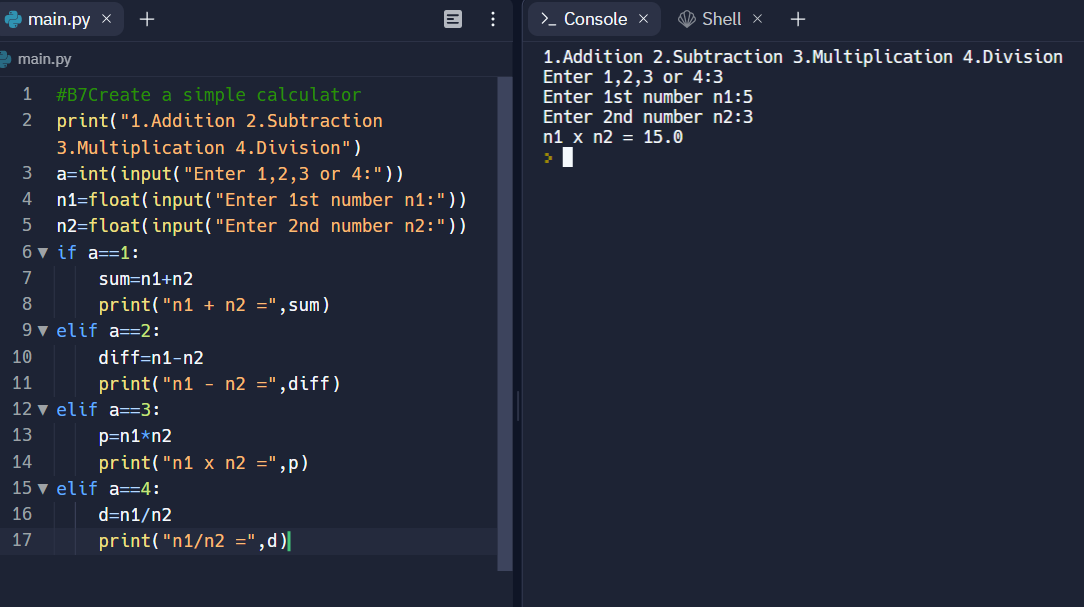
p=n1\*n2

print("n1 x n2 =",p)

**elif** a==**4**:

d=n1/n2

print("n1/n2 =",d)



#B8Estimate the Grade based on the marks obtained by a student.

print("Enter Marks Obtained in 5 Subjects: ")

markOne = int(input())

markTwo = int(input())

markThree = int(input())

markFour = int(input())

markFive = int(input())

tot = markOne+markTwo+markThree+markFour+markFive

avg = tot/**5**

**if** avg>=**91** **and** avg<=**100**:

print("Your Grade is A1")

**elif** avg>=**81** **and** avg<**91**:

print("Your Grade is A2")

**elif** avg>=**71** **and** avg<**81**:

print("Your Grade is B1")

**elif** avg>=**61** **and** avg<**71**:

print("Your Grade is B2")

**elif** avg>=**51** **and** avg<**61**:

print("Your Grade is C1")

**elif** avg>=**41** **and** avg<**51**:

print("Your Grade is C2")

**elif** avg>=**33** **and** avg<**41**:

print("Your Grade is D")

**elif** avg>=**21** **and** avg<**33**:

print("Your Grade is E1")

**elif** avg>=**0** **and** avg<**21**:

print("Your Grade is E2")

**else**:

print("Invalid Input!")

Graphical user interface, text, application

Description automatically generated

#B9Find the largest of 3 numbers.

n1=float(input("Enter 1st number n1:"))

n2=float(input("Enter 2nd number n2:"))

n3=float(input("Enter 3rd number n3:"))

**if** n1>n2 **and** n1>n3:

print(n1,"is largest")

**elif** n2>n1 **and** n2>n3:

print(n2,"is largest")

**elif** n3>n1 **and** n3>n2:

print(n3,"is largest")

Graphical user interface, text

Description automatically generated

#B10 Obtain a character, check if it is lower case, uppercase or digit

n = input("Enter a character: ")

**if** n <=str(**9**) **and** len(n)==**1**:

print(n,"is digit")

**elif** len(n)!=**1**:

print("Error, enter 1 digit character")

**else**:

**if** n==n.upper():

print(n,'is uppercase')

**else**:

print(n,'is lowercase')

Graphical user interface, text, application

Description automatically generated

#C1Add, subtract, multiply and divide two integers by getting inputs from the user.

n1=int(input("Enter 1st number n1:"))

n2=int(input("Enter 2nd number n2:"))

sum=n1+n2

print("n1 + n2 =",sum)

diff=n1-n2

print("n1 - n2 =",diff)

p=n1\*n2

print("n1 x n2 =",p)

d=n1/n2

print("n1/n2 =",d)

Graphical user interface, text

Description automatically generated

#C2Swap the values of two variables using a temporary variable and multiple assignment.

a=int(input("a="))

b=int(input("b="))

c=a

a=b

b=c

print("a =",a,"b =",b)

A screenshot of a computer

Description automatically generated with medium confidence

#C3Read the marks for five subjects and compute the total and average.

n1=int(input("Enter 1st subject marks:"))

n2=int(input("Enter 2nd subject marks:"))

n3=int(input("Enter 3rd subject marks:"))

n4=int(input("Enter 4th subject marks:"))

n5=int(input("Enter 5th subject marks:"))

tot=n1+n2+n3+n4+n5

print("Total marks =",tot)

print("Average marks =",tot/**5**)

Graphical user interface

Description automatically generated

#C4Find the area of rectangle, triangle and circle by reading inputs from the user.

l=float(input("Enter length of rectangle:"))

w=float(input("Enter width of rectangle:"))

b=float(input("Enter base of triangle:"))

h=float(input("Enter height of triangle:"))

r=float(input("Enter radius of circle"))

print("Area of rectangle =",l\*w)

print("Area of triangle =",(**1**/**2**)\*b\*h)

print("Area of circle =",**3.14**\*(r\*\***2**))

Text

Description automatically generated

#C5Compute the square root of a given input number.

n=float(input())

sqrt=n\*\*(**1**/**2**)

print("square root=",sqrt)

print("")

#C6Calculate Simple Interest.

p=int(input("Enter the principle amount:"))

t=int(input("Enter time duration:"))

r=int(input("Enter rate of interest:"))

s=(p\*t\*r)/**100**

print("simple interest=",s)

print("")

#C7 Find the net salary of an employee by getting the basic pay (BP) as input.

bp=float(input("Enter your basic salary:"))

da=(**88**/**100**)\*bp

hra=(**8**/**100**)\*bp

cca=**1000**

insurance=**2000**

pf=(**10**/**100**)\*bp

gpay=bp+da+hra+cca

ded=insurance+pf

netp=gpay=ded

print("net salary =",netp)

A screenshot of a computer

Description automatically generated with medium confidence

#D1Write a program to check whether a number is odd or even

n=int(input("Enter a number:"))

**if** n%**2**==**0**:

print(n,"is an even number")

**else**:

print(n,"is an odd number")

print("")

#D2Write a program in python to find the biggest of two numbers.

n1=float(input("Enter 1st number:"))

n2=float(input("Enter 2nd number:"))

**if** n1>n2:

print(n1,"is largest")

**else**:

print(n2,"is largest")

print("")

#D3Write a program to convert a character from lower case to uppercase and vice versa

a=input("Enter a character:")

**if** a.upper()!=a:

print("uppercase =",a.upper())

**else**:

print("lowercase =",a.lower())

A screenshot of a computer

Description automatically generated with medium confidence

#D4Write a program in python to find whether a number is divisible by both 5 and 7 .

n=int(input("Enter a number:"))

**if** n%**5**==**0** **and** n%**7**==**0**:

print(n,"is divisible by 5 and 7")

**else**:

print(n,"is NOT divisible by 5 and 7")

print("")

#D5 Write a program to find the input year is leap year or not.

y=int(input("Enter a year:"))

**if** y%**4**!=**0**:

print(y,"is not a leap year")

**else**:

**if** y%**100**!=**0**:

print(y,"is a leap year")

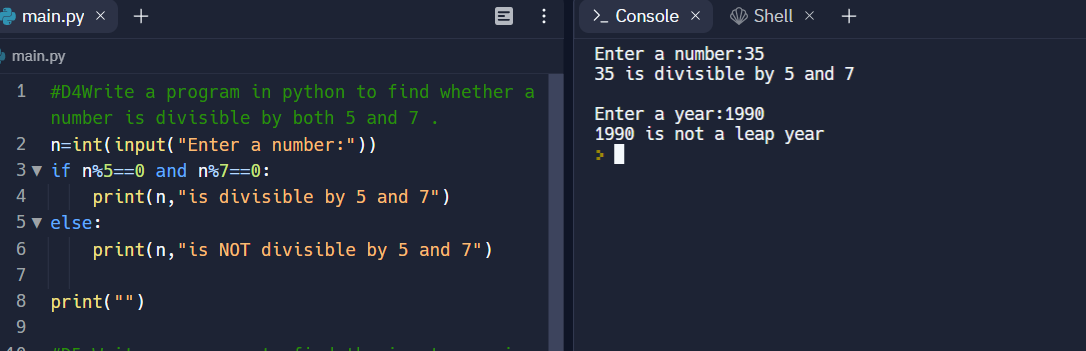
**else**:

**if** y%**400**!=**0**:

print(y,"is not a leap year")

**else**:

print(y,"is a leap year")



#D6Write a program in python to input three sides of a triangle and check whether the triangle is equilateral, isosceles or scalene

a=float(input("side1="))

b=float(input("side2="))

c=float(input("side3="))

**if** a!=b **and** b!=c **and** c!=a:

print("Scalene triangle")

**else**:

**if** a==b **and** b==c **and** c==a:

print("Equilateral triangle")

**else**:

print("Isosceles triangle")

print("")

#D7Write a program in python to input three sides of a triangle and check whether it is right angled one

a=float(input("side1="))

b=float(input("side2="))

c=float(input("side3="))

**if** a\*\***2**==b\*\***2**+c\*\***2** **or** b\*\***2**==a\*\***2**+c\*\***2** **or** c\*\***2**==a\*\***2**+b\*\***2**:

print("Right angled triangle")

**else**:

print("Not a right angled triangle")

Graphical user interface, text

Description automatically generated

#D8 Read a number, check if it is positive, negative or zero. Increment the number if it is positive, decrement if it is negative.

n=float(input("Enter a number:"))

**if** n<**0**:

print(n,"is negative","decrement=",n-**1**)

**elif** n==**0**:

print(n,"is zero")

**elif** n>**0**:

print(n,"is positive, increment=",n+**1**)

print("")

#D9Create a simple calculator.

print("1.Addition 2.Subtraction 3.Multiplication 4.Division")

a=int(input("Enter 1,2,3 or 4:"))

n1=float(input("Enter 1st number n1:"))

n2=float(input("Enter 2nd number n2:"))

**if** a==**1**:

sum=n1+n2

print("n1 + n2 =",sum)

**elif** a==**2**:

diff=n1-n2

print("n1 - n2 =",diff)

**elif** a==**3**:

p=n1\*n2

print("n1 x n2 =",p)

**elif** a==**4**:

d=n1/n2

print("n1/n2 =",d)

Graphical user interface, text, application

Description automatically generated

#D10Estimate the Grade based on the marks obtained by a student.

print("Enter Marks Obtained in 5 Subjects: ")

markOne = int(input())

markTwo = int(input())

markThree = int(input())

markFour = int(input())

markFive = int(input())

tot = markOne+markTwo+markThree+markFour+markFive

avg = tot/**5**

**if** avg>=**91** **and** avg<=**100**:

print("Your Grade is A1")

**elif** avg>=**81** **and** avg<**91**:

print("Your Grade is A2")

**elif** avg>=**71** **and** avg<**81**:

print("Your Grade is B1")

**elif** avg>=**61** **and** avg<**71**:

print("Your Grade is B2")

**elif** avg>=**51** **and** avg<**61**:

print("Your Grade is C1")

**elif** avg>=**41** **and** avg<**51**:

print("Your Grade is C2")

**elif** avg>=**33** **and** avg<**41**:

print("Your Grade is D")

**elif** avg>=**21** **and** avg<**33**:

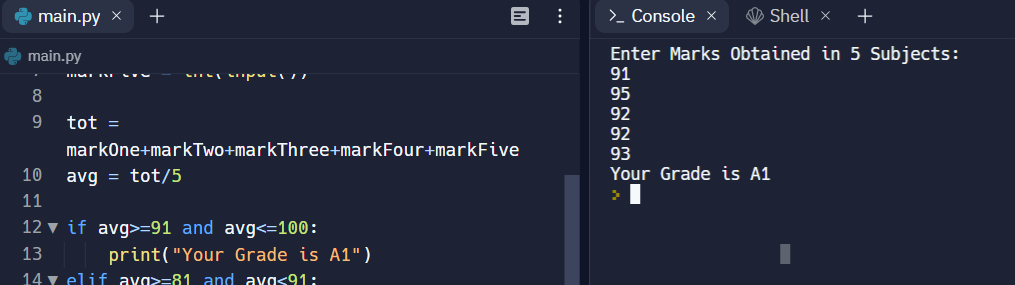
print("Your Grade is E1")

**elif** avg>=**0** **and** avg<**21**:

print("Your Grade is E2")

**else**:

print("Invalid Input!")



#D11 Obtain a character, check if it is lower case, uppercase or digit.

n = input("Enter a character: ")

**if** n <=str(**9**) **and** len(n)==**1**:

print(n,"is digit")

**elif** len(n)!=**1**:

print("Error, enter 1 digit character")

**else**:

**if** n==n.upper():

print(n,'is uppercase')

**else**:

print(n,'is lowercase')

print("")

#D12Find the largest of 3 numbers.

n1=float(input("Enter 1st number n1:"))

n2=float(input("Enter 2nd number n2:"))

n3=float(input("Enter 3rd number n3:"))

**if** n1>n2 **and** n1>n3:

print(n1,"is largest")

**elif** n2>n1 **and** n2>n3:

print(n2,"is largest")

**elif** n3>n1 **and** n3>n2:

print(n3,"is largest")

print("")

#D13Obtain a input from the user and display the corresponding data types primitive and compound data type

a=input("Enter data:")

**if** a==int(a) **or** a==float(a) **or** a==str(a):

print(a,"is a primitive data type")

**else**:

print(a,"is a compound data type")

Text

Description automatically generated

#E1Compute Exponentiation (power of a number) without using \*\* operator.

n=int(input("Enter a number"))

p=int(input("Enter power"))

count=**1**

t=n

**while** count<p:

t=t\*n

count+=**1**

print(n,"^",p,"=",t)

print("")

#E2Write a program in python to print all the two digit numbers which are either divisible by 3 or by 4.

n=**10**

print("All the two digit numbers which are either divisible by 3 or by 4 are:")

**while** n<**100**:

**if** n%**3**==**0** **or** n%**4**==**0**:

print(n)

n+=**1**

print("")

#E3Write a program in python to print the sum of all the digits of a number.

n=int(input("Enter a number:"))

t=n

sum=**0**

**while** t!=**0**:

q=t%**10**

sum+=q

t=t//**10**

print("sum of digits =",sum)

Text

Description automatically generated

#E4Perform the division operation and find the quotient and remainder values (without using /, // % operators)

dvd=int(input("Enter divident:"))

dvsr=int(input("Enter divisor:"))

t=dvd

q=**0**

**while** t-dvsr>=**0**:

t=t-dvsr

q+=**1**

rem=dvd-dvsr\*q

print("quotient =",q,"remainder =",rem)

print("")

#E5Check whether the given number is palindrome or not

num = input('Enter any number : ')

val = int(num)

**while** num == str(num)[::-**1**]:

print('The given number is PALINDROME')

**break**

**else**:

print('The given number is NOT a palindrome')

Graphical user interface, text

Description automatically generated

#E6Check whether the given number is Armstrong number or not

n=int(input("Enter a number"))

t=n

sum=**0**

p=len(str(n))

**while** t!=**0**:

d=t%**10**

sum=sum+d\*\*p

t=t//**10**

**while** n==sum:

print(n,"is an Armstrong number")

**break**

**else**:

print(n,"is NOT an Armstrong number")

print("")

#E7Compute the GCD of two numbers.(Euclidean Method and using common factors)

a=int(input("Enter greater number"))

b=int(input("Enter smaller number"))

t=a

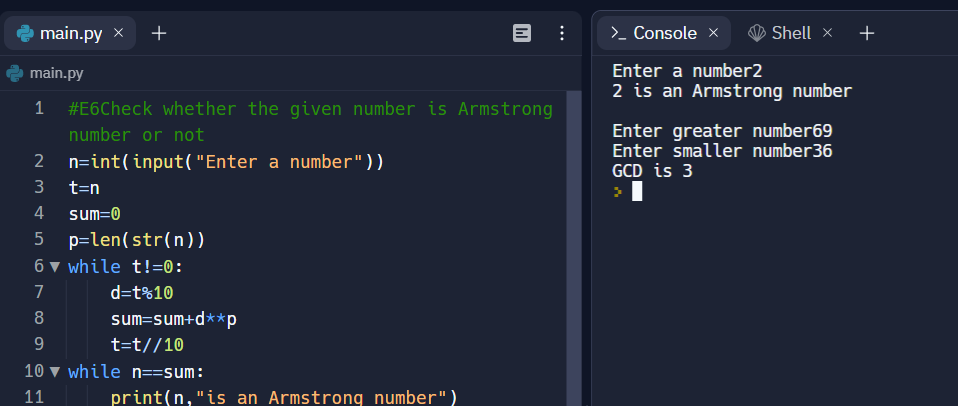
**while** b!=**0**:

a=b

b=t%b

t=a

print("GCD is",a)



#E8Take integer inputs from user until he/she presses q Ask to press q to quit after every integer input Print average and product of all numbers

n=**0**

sum=**0**

p=**1**

count=**0**

**while** n!="q":

n=input("Enter a number or press q to quit")

**if** n!="q":

sum=sum+int(n)

count=count+**1**

p=p\*int(n)

avg=sum/count

print("Average=",avg," Product=",p)

print("")

#E9Find the square root of a number. (Newton’s method)

n=int(input("Enter a number"))

a=n/**2**

b=**0**

**while** b!=a:

b=**0.5**\*(a+n/a)

a=b

print("sqrt",n,"=",a)

Graphical user interface, text

Description automatically generated with medium confidence

#F1Write a Python program to construct the following pattern, using a nested for loop

#a

**for** i **in** range(**0**,**5**):

**for** j **in** range(i):

print("\*",end=" ")

print()

**for** i **in** range(**5**,**0**,-**1**):

**for** j **in** range(i):

print("\*",end=" ")

print()

print()

#b

**for** i **in** range(**1**,**6**):

**for** j **in** range(**1**,i+**1**):

print(j,end=" ")

print()

print()

#c

n=**7**

**for** i **in** range(n+**1**):

**for** j **in** range(n-i):

print(' ', end='')

C = **1**

**for** j **in** range(**1**, i+**1**):

print(C, ' ', sep='', end='')

C = C \* (i - j) // j

print()

A picture containing text

Description automatically generated

#F2 Write a Python program that accepts a word from the user and reverse it.

a='Hellow'

l=[]

**for** i **in** range(len(a)):

l.append(a[i])

**for** i **in** range(len(a)):

l[i]=a[-**1**-i]

b=""

**for** i **in** l:

b+=i

print(b)

print()

#F3 Write a Python program to count the number of even and odd numbers from a series of numbers.

l=[**1**,**2**,**3**,**4**,**5**,**6**,**7**,**8**,**9**,**10**]

evensum=**0**

oddsum=**0**

**for** i **in** l:

**if** i%**2**==**0**:

evensum+=i

**elif** i%**2**==**1**:

oddsum+=i

print('evensum=',evensum)

print("oddsum=",oddsum)

A screenshot of a computer

Description automatically generated with medium confidence

#F4. Write a Python program that prints each item and its corresponding type from the following list.

datalist = [**1452**, **11.23**, **1**+**2**j, **True**, 'w3resource', (**0**, -**1**), [**5**, **12**],

{"class":'V', "section":'A'}]

**for** i **in** datalist:

print(i,type(i))

print()

#F5 Write a Python program that prints all the numbers from 0 to 6 except 3 and 6. Note : Use 'continue' statement.

**for** i **in** range(**0**,**6**):

**if** i==**3** **or** i==**6**:

**continue**

print(i)

Text

Description automatically generated

#F6 Write a Python program which iterates the integers from 1 to 50. For multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".

**for** i **in** range(**1**,**51**):

**if** i%**3**==**0** **and** i%**5**!=**0**:

print("Fizz")

**elif** i%**5**==**0** **and** i%**3**!=**0**:

print("Buzz")

**elif** i%**5**==**0** **and** i%**3**==**0**:

print("FizzBuzz")

**else**:

print(i)

Text

Description automatically generated with medium confidencegf

#F7 Write a Python program to find numbers between 100 and 400 (both included) where each digit of a number is an even number. The numbers obtained should be printed in a comma-separated sequence

Even\_digits\_nos=[]

**for** i **in** range(**0**,**401**):

t=str(i)

**if** len(t)==**1**:

**if** int(t[**0**])%**2**==**0**:

Even\_digits\_nos.append(i)

**if** len(t)==**2**:

**if** int(t[**0**])%**2**==**0** **and** int(t[**1**])%**2**==**0**:

Even\_digits\_nos.append(i)

**if** len(t)==**3**:

**if** int(t[**0**])%**2**==**0** **and** int(t[**1**])%**2**==**0** **and** int(t[**2**])%**2**==**0**:

Even\_digits\_nos.append(i)

print("The numbers from 0 to 400 whose each digit is even are:",Even\_digits\_nos)

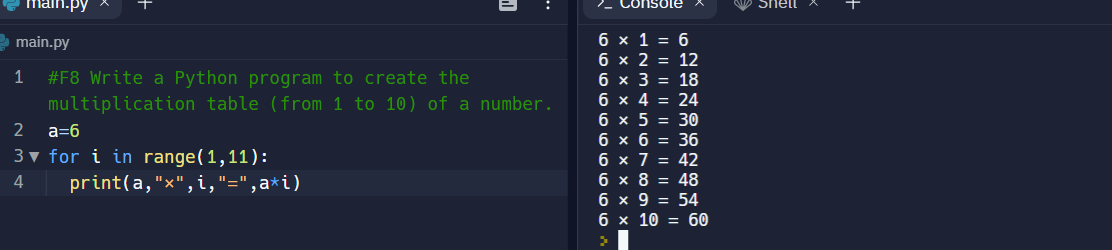
Graphical user interface, application

Description automatically generated#F8 Write a Python program to create the multiplication table (from 1 to 10) of a number.

a=**6**

**for** i **in** range(**1**,**11**):

print(a,"×",i,"=",a\*i)



#F9(a). Find the sum of series:a. 1 + 1/2 + 1/3 + ….. + 1/N

n=**5** #specifing number of terms in series

sum=**0**

**for** i **in** range(**1**,n+**1**):

sum=sum+**1**/int(i)

print(sum)

print()

#F9(b). Find the sum of series 1 + x^2/2 + x^3/3 + … x^n/n

n=**5** #specifing number of terms in series

x=**2** #specifing the value of x

sum=**1**

**for** i **in** range(**2**,n+**1**):

sum+=x\*\*i/i

print(sum)

print()

#F10. Classify the given number is prime or composite number.

n=**11** #test case

**for** i **in** range(**2**,n):

**if** n%i==**0**:

print(n,"is a composite number")

**break**

**else**:

print(n,"is a prime number")

**break**

Graphical user interface, text, application, chat or text message

Description automatically generated

#G1. Write a user-defined function to read the marks of 5 subjects, compute the total marks scored,average, and grade of the student. The function should take the name and ID of the student asarguments and print student name, ID, total, average, and grade. Write a Python program to print the mark details for N students using the function.

**def** **marks\_report**(student):

total\_mks=calculus+chemistry+eee+python+softskills

avg\_mks=total\_mks/**5**

print("Name:",student["name"])

print("ID:",student["id"])

print("Total Marks=",total\_mks)

print("Average Marks=",avg\_mks)

**if** avg\_mks>=**45**:

print("S Grade")

**elif** avg\_mks>=**40** **and** avg\_mks<**45**:

print("A Grade")

**elif** avg\_mks>=**35** **and** avg\_mks<**40**:

print("B Grade")

**elif** avg\_mks>=**30** **and** avg\_mks<**35**:

print("C Grade")

**elif** avg\_mks>=**25** **and** avg\_mks<**30**:

print("D Grade")

**elif** avg\_mks>=**20** **and** avg\_mks<**25**:

print("E Grade")

**elif** avg\_mks<**20**:

print("F Grade")

calculus=**44**

chemistry=**38**

eee=**45**

python=**40**

softskills=**45**

marks\_report({"name":"Hitarth Vyas","id":"22BRS1328"})

Text

Description automatically generated

#G2. Write a function power(X,N) that will allow a floating-point number to be raised to an integerpower and return the result. i.e. Y = X N. Write a Python program to invoke the function

**def** **power**(X,N):

p=X\*\*N

**return** p

print(power(**6.9**,**5**))

print()

#G3.Define a function CheckOddEven(num) that checks if the num is odd or even; the function sets a flag accordingly and returns it. Use this function to find the sum of even and odd numbers separately, from a given input of N numbers.

**def** **CheckOddEven**(num):

**if** num%**2**==**00**:

**global** evensum

evensum+=num

**return** "Even"

**else**:

**global** oddsum

oddsum+=num

**return** "Odd"

evensum=**0**

oddsum=**0**

N=**3** #Number of inputs given

**for** i **in** range(N):

a=int(input())

print(CheckOddEven(a))

print("EvenSum =",evensum,"OddSum =",oddsum)

print()

#G4. Define a function to find the factors of the given number as an argument. If the number is not given, then display the factors of the default argument.

**def** **factors**(n=**5**):

print("factors of",n,"are",end=" ")

**for** i **in** range(**1**,n+**1**):

**if** n%i==**0**:

print(i,end=",")

factors(**10**) #defining an argument

print()

factors() #function taking default argument i.e 5

print("**\n**")

Text

Description automatically generated

#G5. Modify the function in Qn. (1) so that it returns total marks, average and grade of a student.

**def** **marks\_report**(student):

**def** **ttlmks**():

**global** total\_mks

total\_mks=calculus+chemistry+eee+python+softskills

**return** total\_mks

print("Total Marks =",ttlmks())

**def** **avgmks**():

**global** avg\_mks

avg\_mks=total\_mks/**5**

**return** avg\_mks

print("Average Marks =",avgmks())

**def** **grd**():

**global** avg\_mks

**if** avg\_mks>=**45**:

**return** "S Grade"

**elif** avg\_mks>=**40** **and** avg\_mks<**45**:

**return** "A Grade"

**elif** avg\_mks>=**35** **and** avg\_mks<**40**:

**return** "B Grade"

**elif** avg\_mks>=**30** **and** avg\_mks<**35**:

**return** "C Grade"

**elif** avg\_mks>=**25** **and** avg\_mks<**30**:

**return** "D Grade"

**elif** avg\_mks>=**20** **and** avg\_mks<**25**:

**return** "E Grade"

**elif** avg\_mks<**20**:

**return** "F Grade"

print(grd())

calculus=**44**

chemistry=**38**

eee=**45**

python=**40**

softskills=**45**

marks\_report({"name":"Hitarth Vyas","id":"22BRS1328"})

Graphical user interface, text

Description automatically generated

#H1. Calculate factorial of a given number using recursive function. The base case should handle the negative integers by printing an error message and returns none to indicate that something went wrong.

**def** **fact**(n):

**if** n<**0**: #This will diaplay error message and return None for negative numbers

print("Error, factorial is not defined for negative numbers!")

**return** **None** #This line does not have any effect

**if** n==**1**:

**return** **1**

**else**:

factorial=n\*fact(n-**1**)

**return** factorial

print(fact(-**5**)) #test case for negative number

print("factorial is",fact(**5**)) #test case for positive number

print()

#H2. Compute the sum of the digits of a given number using recursion.

**def** **dgts\_sum**(n):

**if** n==**0**:

**return** **0**

**else**:

**return**(n%**10**+dgts\_sum(int(n)//**10**))

n=**222** #test case

print("sum of digits is",dgts\_sum(n))

Text

Description automatically generated

#H3. Check whether a given number is prime or not using recursive function.

**def** **primechk**(n,i=**2**):

**if** n==**2**:

**return** "Prime"

**if** n==**0** **or** n==**1**:

**return** "not prime"

**if** n%i==**0**:

**return** "Not prime"

**else**:

**return** "Prime"

**return** (prime(n,i+**1**))

a=**63** #test case

b=**12** #test case

print(primechk(a))

print(primechk(b))

print()

#H4. Write a recursive function called gcd that takes parameters a and b and returns their greatest common divisor.

**def** **gcd**(a,b):

**if** b==**0**:

**return** a

**return** gcd(b,a%b)

print(gcd(**15**,**18**)) #test case

print()

#H5. The Ackermann function

**def** **A**(m,n):

**if** m==**0**:

**return** n+**1**

**if** m>**0** **and** n==**0**:

**return**(A(m-**1**,**1**))

**if** m>**0** **and** n>**0**:

**return**(A(m-**1**,(A(m,n-**1**))))

print(A(**2**,**2**)) #test case 1

print(A(**0**,**5**)) #test case 2

print(A(**2**,**0**)) #test case 3

Text

Description automatically generated with medium confidence

#I1.Define a function to count the number of occurrences of a substring in a given string and print the starting index of the substring for each occurrence.

**def** **substr**(a):

**global** string

b=len(a)

count=**0**

**for** i **in** range(len(string)):

**if** string[i:(b+i)]==a:

count+=**1**

print(a,"starts at index",i)

**return** count

string="go mango go"

a="go"

print("substring",a,"occurs",substr(a),"times in",string)

print()

#I2.Encrypt a given message by “rotating” each letter by a fixed number of places

**def** **rotate\_word**(word,rot):

alp=["A","B","C","D","E","F","G","H","I","J","K","L","M","N","O","P","Q","R","S","T","U","V","W","X","Y","Z"]

a=""

**for** i **in** word:

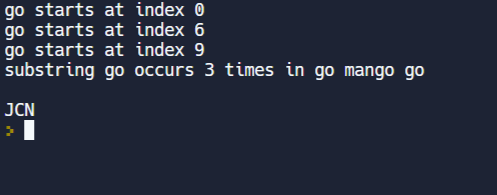
**if** i **in** alp:

i=alp[alp.index(i)+rot]

a+=i

**return** a

print(rotate\_word("HAL",**2**))



#I3.Write a user-defined function to check whether a given text is palindrome or not using string slice method.

**def** **palchk**(a):

b=""

**for** i **in** range(len(a)):

b+=a[-**1**-i]

**if** b==a:

**return** "palindrome"

**else**:

**return** "Not a palindrome"

print(palchk("malayalam"))

print(palchk("tamil"))

A screenshot of a computer

Description automatically generated with medium confidence

#I4.Write a function strip\_characters(str,chars) which removes the characters mentioned in chars from the string str.

**def** **strip\_characters**(str,chars):

charlist=[]

strlist=[]

**for** i **in** chars:

charlist.append(i)

**for** i **in** str:

strlist.append(i)

**for** i **in** charlist:

**for** j **in** strlist:

**if** i==j:

strlist.remove(i)

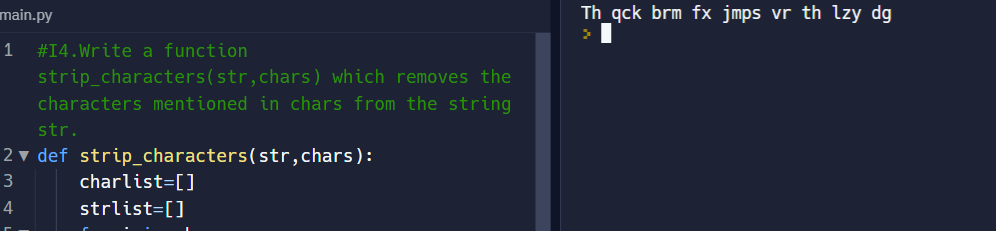
newstr=""

**for** i **in** strlist:

newstr+=i

**return** newstr

print(strip\_characters("The quick brom fox jumps over the lazy dog","aeiou"))



#J1. Read a list of elements from the user and perform the following operations using functions: search(key): to find the given key in the list and display the position of the key if found, otherwise display appropriate message, maximum(Lst) and minimum(Lst) to find the maximum and minimum number respectively from the list.

**def** **search**(key):

**global** lst

**if** key **in** lst:

print("The index of key",key,"is",end=" ")

**return** lst.index(key)

**if** key **not** **in** lst:

**return** "sorry key not found"

lst=[**1**,**2**,**3**,**4**,**5**,**6**]

print(search(**6**))

**def** **maximum**(Lst):

**global** lst

**return** max(lst)

print("Maximum number is",maximum(lst))

**def** **minimum**(Lst):

**global** lst

**return** min(lst)

print("Minimum number is",minimum(lst))

print()

#J2.Two words are anagrams if you can rearrange the letters from one word to spell the other. Write a function called is\_anagram that takes two strings and returns True if they are anagrams.

**def** **is\_anagram**(a,b):

**if** sorted(a.lower())==sorted(b.lower()):

**return** **True**

print(is\_anagram("School master","The"))

print(is\_anagram("classroom Listen","Silent"))

print(is\_anagram("A gentleman","Elegant man"))

Text

Description automatically generated

#J3.Write a function sorted that takes a list as a parameter and sort the elements in lexicographical order. Test the function for a list of names and print the sorted list.

**def** **sorted**(names):

string=""

**for** i **in** names:

string+=i+" "

b=string.split()

b.sort()

print(b)

sorted(["Ram","Shyam","Arjun","Riya","Rahul"])

#J4.A list of students registered for Python course. Perform the following operations (use inbuilt functions) and print the output:

students=["Ram","Shyam","Arjun","Riya","Rahul"]

newreg="Alok"

students.append(newreg) #adding new student

print(students)

print("Number of students registered for python are",len(students))

students[-**2**]="Rajesh" #Modifying a name in the list.

print(students)

students.sort() #sorting name list

print(students)

newstud="Varun"

students.insert(**2**,newstud) #inserting newstudent at index 2

print(students)

**if** "Varun" **in** students: #searching for a student

print("Varun","found!")

students.remove("Riya") #Removing a given name from a list

print(students)

A screen shot of a computer

Description automatically generated with low confidence

#J5.Consider a tuple as T = (1, 3, 2, 4, 6, 5). Write a program to store numbers present at odd index into a new tuple.

T=(**1**,**3**,**2**,**4**,**6**,**5**)

l=[]

**for** i **in** range(len(T)):

**if** i%**2**==**1**:

l.append(T[i])

newT=tuple(l)

print(newT)

print()

#J6.

n=int(input("Enter the number of food items in the menu: "))

food=[]

price=[]

amount=**0**

**for** i **in** range(**1**,n+**1**):

a=input("Enter an item: ")

n=int(input("Enter item's price per k.g: "))

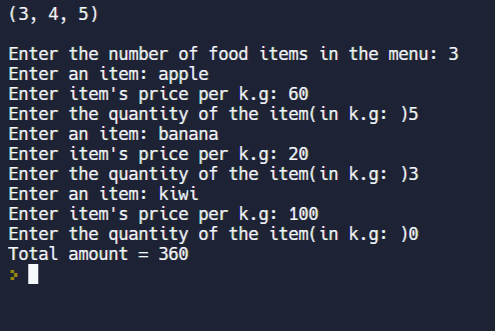
food.append(a)

price.append(n)

q=int(input("Enter the quantity of the item(in k.g: )"))

amount+=q\*n

print("Total amount =",amount)



#K1.Write a function to create and return the r X c matrix with the user input. Write another function to print the sums of each row.

matrix = []

**def** **mat**(R,C):

**global** matrix

**for** i **in** range(R):

a =[]

**for** j **in** range(C):

a.append(int(input()))

matrix.append(a)

**for** i **in** range(R):

**for** j **in** range(C):

print(matrix[i][j], end = " ")

print()

mat(**3**,**3**)

**def** **rowsum**():

**global** matrix

**for** i **in** matrix:

rowsum=**0**

**for** j **in** i:

rowsum+=j

print("rowsum=",rowsum)

rowsum()

print()

#K2.Find the transpose of a given matrix using list comprehension.

m = [[**1**,**2**],[**3**,**4**],[**5**,**6**]]

**for** row **in** m :

print(row)

rez = [[m[j][i] **for** j **in** range(len(m))] **for** i **in** range(len(m[**0**]))]

print("**\n**")

**for** row **in** rez:

print(row)

A picture containing text

Description automatically generated

#K3.For two matrices A and B, compute A+B and A\*B. Show your answer with and without list comprehension

#Add two matrix

X = [[**12**,**7**,**3**],

[**4** ,**5**,**6**],

[**7** ,**8**,**9**]]

Y = [[**5**,**8**,**1**],

[**6**,**7**,**3**],

[**4**,**5**,**9**]]

result = [[**0**,**0**,**0**],

[**0**,**0**,**0**],

[**0**,**0**,**0**]]

**for** i **in** range(len(X)):

**for** j **in** range(len(X[**0**])):

result[i][j] = X[i][j] + Y[i][j]

**for** r **in** result:

print(r)

print()

result = [[X[i][j] + Y[i][j] **for** j **in** range(len(X[**0**]))] **for** i **in** range(len(X))]

**for** r **in** result:

print(r)

print()

#mulitply two matrix

X = [[**12**,**7**,**3**],

[**4** ,**5**,**6**],

[**7** ,**8**,**9**]]

Y = [[**5**,**8**,**1**,**2**],

[**6**,**7**,**3**,**0**],

[**4**,**5**,**9**,**1**]]

result = [[**0**,**0**,**0**,**0**],

[**0**,**0**,**0**,**0**],

[**0**,**0**,**0**,**0**]]

**for** i **in** range(len(X)):

**for** j **in** range(len(Y[**0**])):

**for** k **in** range(len(Y)):

result[i][j] += X[i][k] \* Y[k][j]

**for** r **in** result:

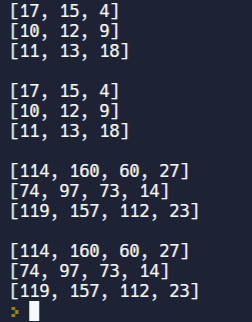
print(r)

print()

result = [[sum(a\*b **for** a,b **in** zip(X\_row,Y\_col)) **for** Y\_col **in** zip(\*Y)] **for** X\_row **in** X]

**for** r **in** result:

print(r)



#L1. Write a user-defined function to print and store squares of numbers for the given range into a dictionary.

d=dict()

**for** x **in** range(**2**,**5**+**1**):

d[x]=x\*\***2**

print(d)

print()

#L2. Write a function named reverseLookup that finds all of the keys in a dictionary that map to a specific value

**def** **reverseLookup**(data,value):

keys=[]

**for** key **in** data:

**if** data[key]==value:

keys.append(key)

**return** keys

**def** **main**():

FrEn={"le":"the", "la":"the", "livre":"book", "pomme":"apple"}

print("The french word for ‘the’ =",reverseLookup(FrEn,'the'))

print("The french word for ‘apple’ =",reverseLookup(FrEn,'apple'))

print("The french word for ‘food’ =",reverseLookup(FrEn,'food'))

main()

print()

#L3.Create a new dictionary by combining two dictionaries whose key-value pairs are given.The new dictionary has to be created by adding values for common keys from the two dictionaries.

A = {'A':**100**, 'B':**200**, 'C':**300**}

B = {'A':**300**, 'B':**500**, 'D':**400**}

c = {x: A.get(x, **0**) + B.get(x, **0**) **for** x **in** set(A).union(B)}

print(c)

Text

Description automatically generated