1. Write a Python function to calculate the factorial of a non-negative integer.

def factorial(n):

if n == 0 or n == 1:

return 1

else:

return n\*factorial(n-1)

num=int(input("enter a number: "))

if num < 0 :

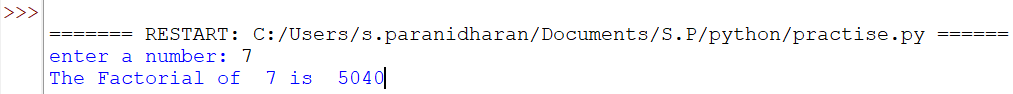
print("enter a non negative number")

else:

print("The Factorial of ", num ,"is ", factorial(num))

A white screen with black text

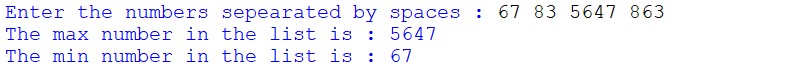
Description automatically generated



1. Given a list of numbers, write a Python function to find the maximum and minimum numbers in the list.

A screen shot of a computer code

Description automatically generated



def find\_max\_and\_min():

user\_input = input("Enter the numbers sepearated by spaces : ")

lst = [int(x) for x in user\_input.split()]

max\_num = max(lst)

min\_num = min(lst)

print("The max number in the list is :",max\_num)

print("The min number in the list is :",min\_num)

find\_max\_and\_min()

1. Write a Python program to check if a string is a palindrome or not.

A white background with green text

Description automatically generated

A close-up of a computer screen

Description automatically generated

strr = input("Enter the string : ")

rstr = strr[::-1]

if strr == rstr:

print("The given string is palindrome")

else:

print("the given string is not palindrome")

1. Write a Python function that takes a list of numbers and returns a new list with only the even numbers from the original list.

A computer code with text

Description automatically generated



def even\_numbers(numbers):

even\_num = []

for x in numbers:

if x % 2 == 0:

even\_num.append(x)

return even\_num

user\_input = input("Enter the list of numbers separated by space: ")

numbers = [int(x) for x in user\_input.split()]

print("The even numbers in the list are:", even\_numbers(numbers))

1. Implement a Python class called Rectangle with methods to calculate the area and perimeter of a rectangle given its length and width.

A screenshot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

def rectangle(b,h):

area = b\*h

perimeter = 2\*b + 2\*h

return area, perimeter

def main():

print("Enter the measurements to find area and perimeter of the rectangle")

b= int(input("Enter the base measurement : "))

h= int(input("Enter the height meaurement : "))

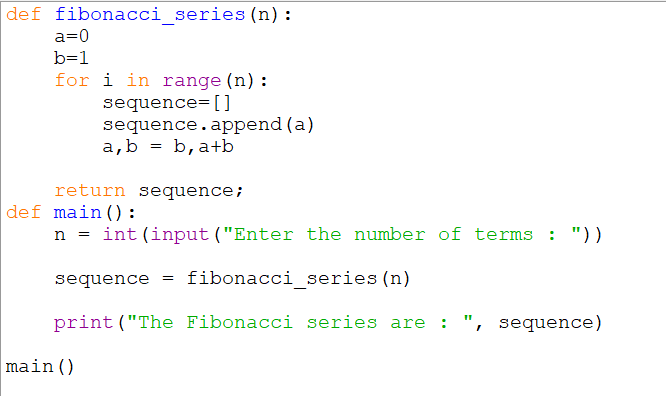
area , perimeter = rectangle(b,h)

print("The area of the rectangle is : ",area)

print("The perimeter of the rectangle is : ", perimeter)

main()

1. Write a Python function to generate the Fibonacci sequence up to a specified number of terms.



A close up of a number

Description automatically generated

def fibonacci\_series(n):

a=0

b=1

for i in range(n):

sequence=[]

sequence.append(a)

a,b = b,a+b

return sequence;

def main():

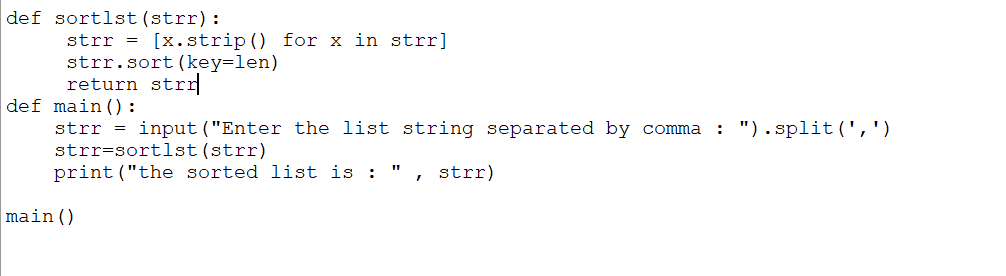
n = int(input("Enter the number of terms : "))

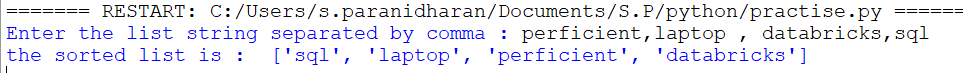
sequence = fibonacci\_series(n)

print("The Fibonacci series are : ", sequence)

main()

1. Given a list of strings, write a Python function to sort the list based on the length of the strings.





def sortlst(strr):

strr = [x.strip() for x in strr]

strr.sort(key=len)

return strr

def main():

strr = input("Enter the list string separated by comma : ").split(',')

strr=sortlst(strr)

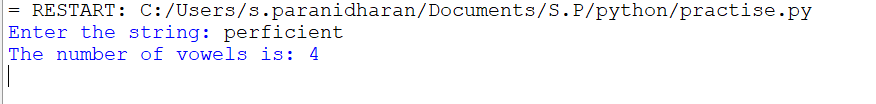
print("the sorted list is : " , strr)

main()

1. Write a Python program to count the number of vowels in a given string.

A white background with colorful text

Description automatically generated with medium confidence



def count\_vowels(strr):

v = []

for x in strr:

if x.lower() == 'a' or x.lower() == 'e' or x.lower() == 'i' or x.lower() == 'o' or x.lower() == 'u':

v.append(x)

return len(v)

def main():

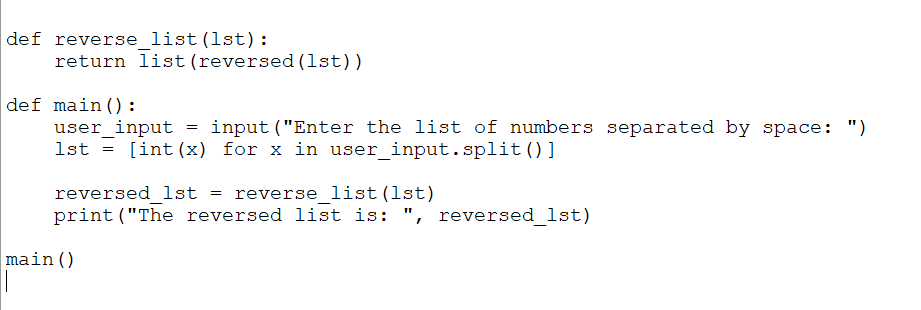
strr = input("Enter the string: ")

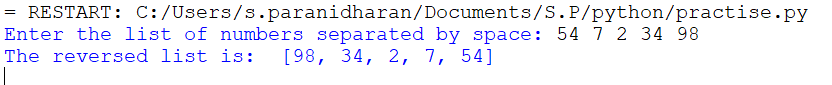
vc = count\_vowels(strr)

print("The number of vowels is:", vc)

main()

1. Implement a Python function that takes a list of integers and returns a new list with the elements reversed.





def reverse\_list(lst):

return list(reversed(lst))

def main():

user\_input = input("Enter the list of numbers separated by space: ")

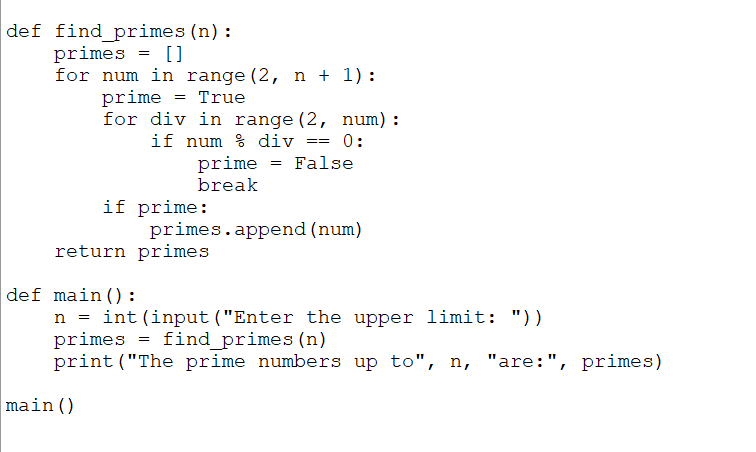
lst = [int(x) for x in user\_input.split()]

reversed\_lst = reverse\_list(lst)

print("The reversed list is: ", reversed\_lst)

main()

1. Write a Python program to find all the prime numbers within a specified range



A close up of a number

Description automatically generated

def find\_primes(n):

primes = []

for num in range(2, n + 1):

prime = True

for div in range(2, num):

if num % div == 0:

prime = False

break

if prime:

primes.append(num)

return primes

def main():

n = int(input("Enter the upper limit: "))

primes = find\_primes(n)

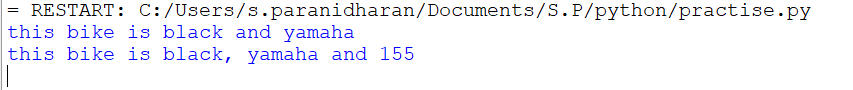
print("The prime numbers up to", n, "are:", primes)

main()

Q3 Python program that demonstrates inheritance to show only the names of a parent and child, along with

A computer code with black text

Description automatically generated with medium confidence



class Bike:

def \_\_init\_\_(self,color,company):

self.color = color

self.company = company

def description(self):

return f"this bike is {self.color} and {self.company}"

class Bike2(Bike):

def \_\_init\_\_(self,color,company,speed):

super().\_\_init\_\_(color,company)

self.speed = speed

def description(self):

return f"this bike is {self.color}, {self.company} and {self.speed}"

bike = Bike("black","yamaha")

print(bike.description())

bike2=Bike2("black","yamaha","155")

print(bike2.description())

Q4 Python program that demonstrates exception handling to check if a salary is sufficient to pay off a loan.

A screenshot of a computer code

Description automatically generated

A close-up of a computer screen

Description automatically generated

class Insufficientsalary(Exception):

pass

def check\_salary(salary,loan):

if loan > salary:

print("Your not eligble to apply loan")

else:

print("Your eligble to apply loan")

try:

salary=float(input("Enter you salary : "))

loan=float(input("Enter you loan amount : "))

check\_salary(salary,loan)

except Insufficientsalary as e:

print(e)

except ValueError:

print("Please enter correct digits")

Q5 Python program that uses linear search to find a word in a single sentence

A computer screen shot of a computer code

Description automatically generated

A close up of a text

Description automatically generated

def Linear\_search(strr,target):

strr=strr.split()

for x in range(len(strr)):

if strr[x] == target:

return x

return -1

strr= input("Enter any sentence : ")

target=input("Enter the target : ")

result=Linear\_search(strr,target)

if result!= -1:

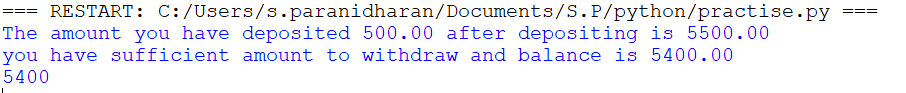
print(f"The Target word {target} is found in the sentence : {result+1}.")

else:

print(f"The Target word {target} is not found")

Q7 python program using construcutors  
A screenshot of a computer program

Description automatically generated



class BankAccount:

def \_\_init\_\_(self,account\_no,balance):

self.account\_no=account\_no

self.balance=balance

def deposit(self,amount):

self.balance += amount

print(f"The amount you have deposited {amount:.2f} after depositing is {self.balance:.2f}")

def withdraw(self,amount):

if self.balance < amount:

print("There is not sufficient amount to withdraw")

else:

self.balance -= amount

print(f"you have sufficient amount to withdraw and balance is {self.balance:.2f}")

def get\_balance(self):

return self.balance

account1= BankAccount("123456",5000)

account1.deposit(500)

account1.withdraw(100)

print(account1.get\_balance())

Q8 alpha pyramid

A screen shot of a computer code

Description automatically generated

A white background with blue letters

Description automatically generated

def print\_alpha\_pyramid(n):

for i in range(n):

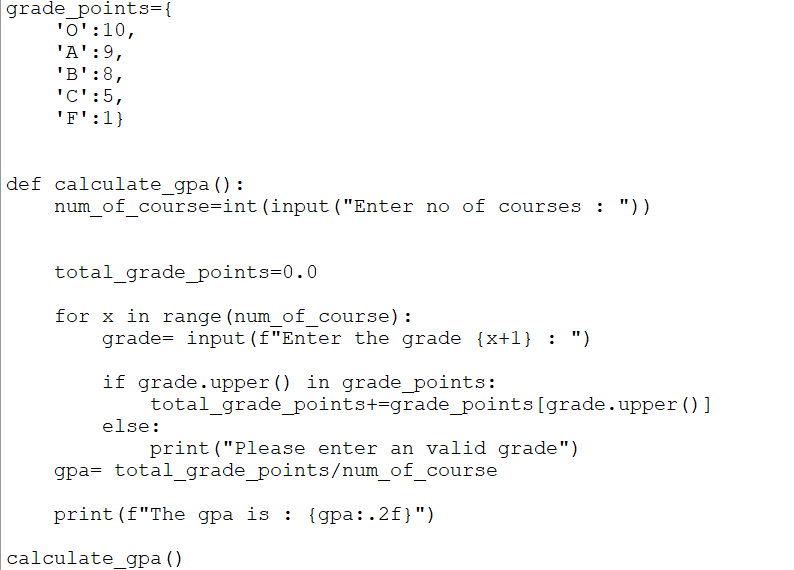
for j in range(i+1):

print(chr(65+j), end=" ")

print()

print\_alpha\_pyramid(5)

9)Python program to calculate GPA



A screenshot of a computer

Description automatically generated

grade\_points={

'O':10,

'A':9,

'B':8,

'C':5,

'F':1}

def calculate\_gpa():

num\_of\_course=int(input("Enter no of courses : "))

total\_grade\_points=0.0

for x in range(num\_of\_course):

grade= input(f"Enter the grade {x+1} : ")

if grade.upper() in grade\_points:

total\_grade\_points+=grade\_points[grade.upper()]

else:

print("Please enter an valid grade")

gpa= total\_grade\_points/num\_of\_course

print(f"The gpa is : {gpa:.2f}")

calculate\_gpa()