#1. Imagine you are working on a financial application that processes transactions represented

by unique numerical IDs. Due to a specific requirement, you need to reverse the digits of

these IDs before storing them in a database. For instance, if the transaction ID is 12345, it

should be stored as 54321.

num=int(input())

new\_num=str(num)

print(new\_num[::-1])

n=int(new\_num)

2. Imagine you are developing a mathematical education application. One of the features is

to check whether a given number is an Armstrong number. Students can input any number,

and the application will determine if it's an Armstrong number.

num=int(input()) #153

sum=0 #initialize the sum value as 0

temp=num #153

#153>0 #15>0 #1>0

while temp>0:

digit=temp%10 #153%10=3 #15%10=5 #1%10=1

sum=sum+digit\*\*3 #0+3\*\*3=27 #27+5\*\*5=152 #152+1\*\*1=153

temp=temp//10 #153//10=15 #15//10=1 #1//10=0.1

if num==sum: #num=153 sum=153

print("armstrong") #armstrong

else:

print("not armstrong")

You are given a positive integer and you need to calculate the sum of its digits using a loop.

For example, if the input number is 6789, the program should output 30 (because 6 + 7 + 8 + 9 = 30).

num = int(input("Enter a positive integer: ")) #123

sum\_digits = 0

while num > 0: #123>0 | 12>0 | 1>0

digit = num % 10 #123%10=3 | 12%10=2 | 1%10

sum\_digits += digit #0+3=3| 3++2=5 | 5+1=6

num //= 10 #123//10=12 | 12//10=1 |1//10=0

print("Sum of digits:", sum\_digits)

You are given a positive integer and you need to calculate the product of its digits using a

loop. For example, if the input number is 234, the program should output 24 (because 2 \* 3 \* 4 = 24).

num = int(input("Enter a positive integer: ")) #543

product\_digits = 1

while num > 0: #543>0 | 54>0 | 5>0

digit = num % 10 #543%10=3 | 54%10 =4 | 5%10=5

product\_digits \*= digit #1\*3=3 | 3\*4 =12 | 12\*5=60

num //= 10 #543//10=54 | 54//10=5 | 5//10=0

print("Product of digits:", product\_digits)

You are given a positive integer, and you need to print its decremented values down to 1

using a for loop and the range function. For example, if the input number is 10, the program

should output 10, 9, 8, ..., 2, 1.

num = int(input("Enter a positive integer: "))

for i in range(num, 0, -1):

print(i, end=" ")

A positive integer that is multiplied by itself and the resultant product is known as a perfect square.Write the Python program to check if the number is a perfect square or not.

import math

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

root = math.sqrt(num)

value=math.trunc(root)

if root==value:

print(f"{num} is a Perfect Square")

else:

print(f"{num} is not a Perfect Square")

Write a program to print below pattern.

n=int(input())

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

for j in range(i):

print(i,end="\*")

print()

for i in range(n,0,-1):

for j in range(i):

print(i,end="\*")

print()

Write a program to print the pattern such that the elements of first row, first column, last row and last column should be one and the remaining elements

should be zero

rows= int(input())

cols=int(input())

for i in range(rows):

for j in range(cols):

if i == 0 or i == rows-1 or j==0 or j==cols-1:

print(1, end=' ')

else:

print(0, end=' ')

print()

Suppose you are developing a small piece of code to validate the age of a

person before allowing them to register for an event. The event requires

participants to be at least 18 years old but not older than 60 years.

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

try:

assert age >= 18, "Registration error: Participant must be at least 18 years old"

assert age <= 60, "Registration error: Participant must be no older than 60 years old"

print("Participant is eligible for registration.")

except AssertionError as e:

print(e)

Suppose you are developing a simple calculator that performs addition. The

calculator expects both inputs to be numbers. If a user provides inputs that are

not numbers, a TypeError should be raised and handled appropriatel

try:

n1=int(input())

n2=int(input())

n=n1+n2

print(n)

except ValueError as ve:

print(ve)

Imagine you are developing a banking application. The task is to withdraw

money from a user's account. The withdrawal process should raise an exception

if the withdrawal amount is greater than the available balance or if the withdrawal amount is negative.

class NegativeWithdrawalError(Exception):

pass

class InsufficientFundError(Exception):

pass

balance = 1000

try:

amount = int(input("Enter the amount to withdraw: "))

if amount < 0:

raise NegativeWithdrawalError("Error: Withdrawal amount cannot be negative.")

elif amount > balance:

raise InsufficientFundError("Error: Insufficient funds in your account.")

else:

balance -= amount

print(f"Withdrawn successfully. Your new balance is {balance}.")

except NegativeWithdrawalError as e:

print(e)

except InsufficientFundError as e:

print(e)

You are a manager at a small company and you want to keep track of the monthly sales figures for each of your 12 employees over the past year.

sales=[]

sales= [1200, 1500, 1100, 1800, 1300, 1900, 1750, 1400]

print(sales[5])

sales.extend([1600,1700])

print(sales)

sales.extend([1600,1700])

print(sales)

sales.reverse()

print(sales)

Imagine you are a teacher who has a class of 10 students. You want to keep track of their scores in a recent coding test

scores = [35, 92, 78, 22, 88, 66, 95, 59, 84, 40]

print(scores)

scores[3]=95

print(scores)

highest\_score=max(scores)

print(highest\_score)

lowest\_score=min(scores)

print(lowest\_score)

Find the sample prices of items in a store: 10.99, 5.49, 20.00, 7.95, 12.75.

prices = [10.99, 5.49, 20.00, 7.95, 12.75]

print(prices)

Total\_cost=sum(prices)

print(Total\_cost)

prices.pop(2)

print(prices)

prices.append(8.98)

print(prices)

Minimum Element

num\_elements = int(input("Enter the number of elements: "))

elements = []

for i in range(num\_elements):

elements.append(int(input("Enter element {}: ".format(i+1))))

print(elements)

min\_element = elements[0]

for element in elements:

if element < min\_element:

min\_element = element

print("Minimum element =", min\_element)

Palindrome Numbers

number=int(input()) #121

num=number #121

rev=0

while num>0: #121>0 | 12>0 |1>0

digit=num%10 #121%10=1 | 12%10=2 | 1%10=1 |0.1>0-->failed

rev=(rev\*10)+digit #(0\*10)+1=1 | (1\*10)+2=12 |(12\*10)+1=121

num//=10 #121//10=12 | 12//10=1 |1//10=0.1

if number==rev: #121==1 | 121==12 |121==121

print(number,"is palindrome")

else:

print(number,"Not a palindrome")

Sum of series Factoiral

import math

x = float(input("Enter the value of x: "))

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

sum = 0

for i in range(n+1):

sum += (x\*\*i) / math.factorial(i)

print("Sum of the series:", sum)

sample input: Enter the value of x: 2

Enter the number of terms: 5

Sum of the series: 7.37704822

calculates the product of the digits of a given integer.

num=int(input())

product=1

for digit in str(num):

product\*=int(digit)

print(product)

sample Input: 123

Output: 6 (because 1\*2\*3 = 6)

Input: 456

Output: 120 (because 4\*5\*6 = 120)

Calculate The Factorial Of Given Number

import math

n=int(input())

print(math.factorial(n))

Prime OR Not Prime Number

n=int(input())

if n>2:

for i in range(2,n):

if n%i==0:

print(n,"Not prime")

break

else:

print(n,"prime")

break

elif n==0 or 1:

print("neither")

else:

print()

Range Function

for i in range(5,9):

print(i) #5,6,7,8

code loops through each character in the string

for i in "python":

print(i)

code loops through each character in the string and followed by character $

for i in "python":

print(i,"?$")

code calculates and prints the square of x as long as x is less than or equal to 20, incrementing x by 3 after each iteration.

x=8

while x<=20:

print(x\*\*2)

x+=3

code uses nested loops to print a triangle of numbers.

for i in range(1,5):

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

print(j,end=" ")

print()

output

1

1 2

1 2 3

1 2 3 4

code prints all odd numbers between 100 and 500 (exclusive) that are multiples of 11.

for i in range(100,500):

if i%11==0 and i%2==1:

print(i)

# Create a quiz using python dictionary and list. Store the question and answer in a dictionary and add score for each correct option chosen. Display the score for the quiz.

print("Welcome To Quiz Game")

player=input("Do You Want To Play? (yes or no):")

if player.lower()!="yes":

quit()

else:

print("Let's Play And Enjoy")

score=0

question=input("Who developed Pyhton Programming Language?")

if question.lower()=="guido van rossum":

print("Correct Answer")

score+=1

else:

print("Incorrect Answer")

question=input("Is Pyhton is Case Sensitive When dealing with identifiers?")

if question.lower()=="yes":

print("Correct Answer")

score+=1

else:

print("Incorrect Answer")

question=input("Which is used to define Block Of code in Pyhton Language?")

if question.lower()=="identation":

print("Correct Answer")

score+=1

else:

print("Incorrect Answer")

question=input("Python is interpreted or complier?")

if question.lower()=="interpreted":

print("Correct Answer")

score+=1

else:

print("Incorrect Answer")

print(f"Your Score is :{score}")

Create a simple Python program for a Student Management System using a list or dictionary. The student details are Id, Name, Grade and Major The program must allow the user to add, update, remove, search for, and display students. Create function for add/update/remove/search/display/exit.

student\_details = {}

def add\_student():

id = input("Enter student ID: ")

name = input("Enter student name: ")

grade = input("Enter student grade: ")

major = input("Enter student major: ")

student\_details[id] = {"Name": name, "Grade": grade, "Major": major}

print("Student added successfully!")

def view\_students():

for id, details in student\_details.items():

print(f"ID: {id}, Name: {details['Name']}, Grade: {details['Grade']}, Major: {details['Major']}")

def search\_student():

id = input("Enter student ID to search: ")

if id in student\_details:

print(f"ID: {id}, Name: {student\_details[id]['Name']}, Grade: {student\_details[id]['Grade']}, Major: {student\_details[id]['Major']}")

else:

print("Student ID not found!")

def update\_student():

id = input("Enter student ID to update: ")

if id in student\_details:

student\_details[id]["Name"] = input("Enter new name: ")

student\_details[id]["Grade"] = input("Enter new grade: ")

student\_details[id]["Major"] = input("Enter new major: ")

print("Student updated successfully!")

else:

print("Student ID not found!")

def delete\_student():

id = input("Enter student ID to delete: ")

if id in student\_details:

del student\_details[id]

print("Student deleted successfully!")

else:

print("Student ID not found!")

def End\_Details():

print("Students Details are closed")

while True:

print("\nStudent Management System")

print("1. Add Student")

print("2. View Students")

print("3. Search Student")

print("4. Update Student")

print("5. Delete Student")

print("6. Exit")

choice = input("Enter your choice: ")

case "1":

add\_student()

elif choice == "2":

view\_students()

elif choice == "3":

search\_student()

elif choice == "4":

update\_student()

elif choice == "5":

delete\_student()

elif choice == "6":

End\_Details()

break

else:

print("Invalid choice. Please try again.")

print("Note:Enter The Number within 1 To 6")

Given a 2D array, print it in a wave-like pattern. Start from the first column, go down, then move to

the next column and go up, and so on.

Explanation : Iterate over the columns of the matrix. If the column index is even, print the column

from top to bottom. If the column index is odd, print the column from bottom to top.

rows\_columns = int(input())

arr= []

for i in range(rows\_columns):

l = input().split()

arr.append(l)

for col in range(len(arr[0])):

if col % 2 == 0: #if the column is even it print same

for row in range(rows\_columns):

print(arr[row][col], end=' ')

else:

for row in range(rows\_columns-1, -1, -1): #if the coulmn is odd its prints reverse order

print(arr[row][col], end=' ')

You are tasked with managing student grades for a course. The grades are stored in a dictionary where the keys are student names and the values are lists of their grades.

#a) How would you add a new grade for a student in the dictionary? If the

student doesn’t exist, how would you add the student with their grade?

grades={"Rickshi":[9,8,7], "Preety":[6,7,8]}

grades["Sudar"]=[4,5,6]

print(grades)

b) How would you calculate the average grade for a given student? What would you do if the student doesn’t exist?

student\_name=input()

if student\_name in grades:

stu\_grade=grades[student\_name]

avg\_grade=sum(stu\_grade)/len(grades)

print(avg\_grade)

else:

print("Srudent Not Found")

c) How would you remove a student and all their grades from the dictionary? What should happen if the student doesn’t exist?

remove\_grade=input()

if remove\_grade in grades:

del grades[remove\_grade]

print(grades)

else:

print("Student Not Foound")

2 Tuples

a) Create a tuple with single element

name=("Anitha",)

print(name)

b) How can you multiply a tuple by an integer

multiply=int(input())

print(name\*multiply)

c) Get the index of an element in a tuple.

name=("Anitha","Ananadakumar","Mirthula")

index\_element=input()

if index\_element in name:

print(name.index(index\_element))

else:

print("Name Not found")

d) Modify an element in a tuple.

l=list(name)

l[0]="Rickshi"

t=tuple(l)

print(t)

e) Convert a tuple to a string

string=str(name)

print(string)

f) Find the maximum and minimum values in a tuple.

age=(37,42,20)

print("Maximum Age:",max(age))

print("Minimum Age:",min(age))

g) Count the occurrences of an element in a tuple

age=(38,42,20,18,18)

occurance=int(input())

print(age.count(occurance))

h) create a nested tuple

number=(1,2,(3,4,),5,6,(7,8,),)

print(number)

i) Access elements in a nested tuple.

access=int(input())

print(number[access])

j) Delete an element from a tuple

l=list(number)

removing\_element=int(input())

l.remove(removing\_element)

t=tuple(l)

print(t)

Imagine you have a base class Employee, and you want to create a child class Manager that inherits from Employee. -->Parent class (Employee): Attributes →name and salary and a method→ display\_details() to show employee details. Child class (Manager): This class inherits from the Employee class and adds an additional attribute department and method to display the details for a manager.

class Employee:

def \_\_init\_\_(self,name,salary):

self.name=name

self.salary=salary

def display(self):

print(f"Name={self.name}\nSalary={self.salary}")

class Manager(Employee):

def \_\_init\_\_(self,name,salary,department):

super().\_\_init\_\_(name,salary)

self.department=department

def printDetails(self):

self.display()

print(f"Department={self.department}")

s=Manager('Rickshi',12000,'AI')

s.printDetails()

Write a python program for single inheritance. -->Parent class->LibraryItem(Title,author ,publication year and method displayInfo()) -->Child class->Book(inherit the attributes of parent class and include additional attribute → genre) Display all the details together.

class Libraryitem:

def \_\_init\_\_(self,title,author,year):

self.title=title

self.author=author

self.year=year

def display(self):

print(f"Book Title={self.title}\nAuthor={self.author}\nPublication Year={self.year}")

class Book(Libraryitem):

def \_\_init\_\_(self,title,author,year,genre):

super().\_\_init\_\_(title,author,year)

self.genre=genre

def printDetails(self):

self.display()

print(f"Genre={self.genre}")

s=Book('The Great Gatsby','F.Scott Fitzgerald',1925,'Modernist literature')

s.printDetails()

Create a BankAccount class that has methods deposit money, withdraw money, and check the balance. Create an object. Access all the methods using the object.

class BankAccount:

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

self.balance=0

def deposit(self,amount):

if amount>0:

self.balance+=amount

print(f"Your Deposited Amount : {amount:.2f}")

else:

print("Amount must be in positive")

def withdraw(self,amount):

if amount>self.balance:

print("Insufficient Balance, Please Try Again")

else:

self.balance-=amount

print(f"Your Withdraw Amount: {amount:.2f}")

def check\_balance(self):

print(f"Your Current Balance: {self.balance:.2f}")

s=BankAccount()

s.deposit(10000)

s.check\_balance()

s.withdraw(500)

s.check\_balance()

Imagine you are a teacher preparing a seating chart for your classroom. You want to visually represent the arrangement of seats using a simple symbol (like ^) for each seat. Each row in the classroom has a fixed number of seats, and there are several rows of seats. You decide to use a Python program to print out the seating arrangement using any of the given symbols( \* & ^ $ # @), which you can then display or share with your students.

rows=int(input())

columns=int(input())

for i in range(rows):

for j in range(columns):

print("@",end=" ")

print()

Imagine a simple classroom exercise where you want to teach students about multiples of four. You decide to write a Python program that prints the first 8 multiples of 4. This will help students understand the concept of multiplication and pattern recognition.

Sample Output:0 4 8 12 16 20 24 28

for z in range(0,29,4):

print(z, end=" ")

Imagine you are a teacher processing student scores from a recent exam. You have a list of scores, and you want to print all the scores, but you want to stop processing if you encounter a score of -1, which indicates missing data. Additionally, you want to skip any negative scores (other than -1) as they are considered invalid.Hint: Use break and continue to handle this scenario in the loop

scores = [85, 90, -5, 76, 92, -1, 88, 79, 55]

for mark in scores:

if mark == -1:

print("Encountered missing data. Stopping processing.")

break

elif mark < 0 and mark != -1:

print(f"Invalid score {mark} encountered. Skipping.")

continue

print(f"Score: {mark}")

Rahul wants to create a bank application in which he wants help to perform basic bank operations such as view balance, withdraw and deposit using python program.Help Rahul to write a program that shows the following menu’s:

1. Deposit 2.Withdraw 3. View Balance 4.Exit . Each option should perform the following task: Deposit-(Increase the balance when deposited)Withdraw -> Reduce the balance amount after withdrawal. 3-> Show balance available after operations. Exit -> quit the application.

balance = 0

while True:

print("1. Deposit")

print("2. Withdraw")

print("3. View Balance")

print("4. Exit")

choice = int(input("Choose an option: "))

match choice:

case 1:

amount = float(input("Enter amount to deposit: "))

balance += amount #balance=balance+amount

print(f"Deposited: {amount}")

case 2:

amount = float(input("Enter amount to withdraw: "))

if amount> balance :

print("Insufficient balance")

else:

balance -= amount #balance=balance-amount

print(f"Withdrawn: {amount}")

print(f"Current Balance: {balance}")

case 3:

print(f"Available balance: {balance}")

case 4:

print("Exiting")

break

case \_:

print("Invalid option")

2. Choose any restaurant of your choice. Make a hotel Menu using conditional statements.

menu = {

"Biryani": ["Mutton Biryani....290","Chicken Biryani...180","Fish Biryani.. 200",

"Veg Biryani...150"],

"Beverages": ["Diet Coke...98","Mineral water ...80","Fanta...76"],

"Fried Rice": ["Veg Fried Rice...219","Egg Fried Rice...139"],

"Starters": ["Samosas...120","Spring Rolls...100","Paneer Tikka..150"]

}

print("Welcome to Tasty Bites!")

print("Today Menu Are Biryani,Beverages,Fried rice,Starters")

favourite=input("Enter your Favourite:")

if favourite in menu:

print(menu [favourite])

else:

print("invalid menu")

Create a dictionary containing cosmetic brands. Each key should contain list of products.

cosmetic\_brands = {

'Lakme': ['foundation', 'Matte finish lipstick', 'Eyeliner', 'face wash'],

'Maybelline': ['Pot kajal', 'Liner', 'Lipliner'],

'Nivea': ['Soft Moisturizing Creme', 'Hydrating Face Mask', 'Lip Balm'],

'Loreal': ['Colour Riche Moisturising Lipliner', 'Volume Million Lashes Mascara','True Match Super Blendable Foundation'],

'Revlon': ['Ultra HD Matte Lip Color','ColorStay Foundation','Precision Lash Adhesive']

}

brand\_name = input("Enter the brand name: ")

if brand\_name in cosmetic\_brands:

print(cosmetic\_brands[brand\_name])

else:

print("Brand not found")

Use match case to display a calculator (Add, mul, div ,sub, exponent

num1=int(input())

num2=int(input())

print(": 1. Add 2. Sub 3. Mul 4. Div 5.Exp")

option=int(input())

match option:

case 1:

num=num1+num2

print("Sum is :",num)

case 2:

num=num1-num2

print("Sub is :",num)

case 3:

num=num1\*num2

print("Mul is",num)

case 4:

num=num1/num2

print("Div is",num)

case 5:

num=num1\*\*num2

print("Exp is",num)

case \_:

print("invalid option")

You need to find the maximum value in a 2D array

n=int(input())

arr=[]

for i in range(n):

l=list(map(int,input().split()))

arr.append(l)

print(arr)

arr=max(arr)

print(arr[i] or arr[j])

Check if a given square matrix is symmetric. A matrix is symmetric if A[i][j] == A[j][i] for all valid i and j

n=int(input())

arr=[]

for i in range(n):

l=list(map(int,input().split()))

arr.append(l)

for i in range(n):

for j in range(n):

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

print()

ans=True

for i in range(n):

if arr[i][j]!=arr[j][i]:

ans=False

if ans==False:

print('Not Symmetric matrix')

break

if ans==True:

print('Symmetric matrix')

Rotate a given N x N matrix by 90 degrees clockwise.

n=int(input())

arr=[]

for i in range(n):

L=list(map(int,input().split()))

arr.append(L)

transpose=[[arr[j][i] for j in range(n)] for i in range(n)]

for i in transpose:

i.reverse()

for i in range(n):

for j in range(n):

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

print()

You need to develop a program that takes two matrices as input and outputs their sum. The matrices must have the same dimensions for the addition to bevalid

n1=int(input("row 1:"))

n2=int(input("rows 2:"))

c1=int(input("column 1:"))

c2=int(input("column 2:"))

if c1!=c2:

print("columns must be same")

else:

matrix1=[]

matrix2=[]

for i in range (n1):

x=list(map(int,input().split()))

matrix1.append(x)

for j in range (n2):

y=list(map(int,input().split()))

matrix2.append(y)

if n1!=n2:

print("rows must be same")

else:

matrix=[]

for i in range(n1):

row=[]

for j in range(c2):

row.append(matrix1[i][j]+matrix2[i][j])

matrix.append(row)

for row in matrix:

print(" ".join(map(str,row)))

You have a 2D array representing a matrix, and you want to create a new matrix that is the transpose the original one (i.e., rows become columns and columns become rows).

row\_col=int(input())

arr=[]

for i in range(row\_col):

L=list(map(int,input().split()))

arr.append(L)

print(arr)

for i in range(row\_col):

for j in range(row\_col):

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

print()

print("Transpose matrix is:")

for i in range(row\_col):

for j in range(row\_col):

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

print()

Suppose you have a 2D array representing a game board, and you want to check if a player has achieved a certain score on the board. If the score is present in the array, print (“found at position”) else print “Not found”.

rows = int(input("Enter the number of rows: "))

cols = int(input("Enter the number of columns: "))

score\_board = []

for i in range(rows):

l= list(map(int, input().split()))

score\_board.append(l)

target\_score = int(input("Enter the element to be found: "))# Get the element to be found from the user

found = False

for i in range(rows):

for j in range(cols):

if score\_board[i][j] == target\_score:

print(f"Found at position ({i},{j})")

found = True

break

if not found:

print("Not found")

Imagine you have a 2D array representing the sales figures for a store over a week, where each row corresponds to a different product, and each column corresponds to a different day. You want to calculate the total sales for the week.

n= int(input("Enter the number of rows and columns: "))

arr = []

total=0

for i in range(n):

l= list(map(int, input().split()))

arr.append(l)

print(arr)

for i in range(n):

for j in range(n):

total+=arr[i][j]

print(total)

Imagine you have a theater with a seating arrangement where each seat is represented by a number. You want to display this seating chart to show which seats are available and which are taken

n = int(input("Enter the number of rows and column: "))

seat\_arraning = []

for i in range(n):

row = list(map(int, input().split()))

seat\_arraning.append(row)

for i in range(n):

for j in range(n):

if seat\_arraning[i][j] == 0:

seat\_arraning[i][j] = '0'

elif seat\_arraning[i][j] == 1:

seat\_arraning[i][j] = 'X'

for row in seat\_arraning:

print(' '.join(row))

Write a program using python function that accepts different values as parameters and returns a list.

def get\_list(\*data): #\* unpacks a list or tuple into separate data

return list(data)

string\_data= get\_list("duke", "ktm", "ns")

print("Bikes",string\_data)

numeric\_data= get\_list(91, 92, 93, 94, 95)

print("Numbers:", numeric\_data)

mixed\_data = get\_list(2020, "duke200", 2024, True, ["duke250", 300])

print("Bike\_year:", mixed\_data)

Write a program using python function to find the factorial of a numbe

def factorial(n):

if n==0:

return 1

else:

return n\*factorial(n-1)

num=int(input("Enter a Number to factorial:"))

print(factorial(num))

Write a program using python function that accepts two numbers as arguments and returns the product

def product(a,b):

multiple=a\*b

print("Product of two values is:",multiple)

a=int(input())

b=int(input())

product(a,b)

Write a function that displays the last name and first name passed as arguments

def name(last\_name,first\_name):

print("{} {}".format(last\_name,first\_name))

first\_name=input()

second\_name=input()

third\_name=input()

fourth\_name=input()

last\_name=input()

name(last\_name,first\_name)

Write a function to find the square of a number entered by the user.

def squareroot(a):

squarenumber=a\*a

print("The Square root of {a} is:",squarenumber)

a=int(input("Enter a Number:"))

squareroot(a)

Write a program to check whether a number is odd or even.

n=int(input())

if(n%2==0):

print("Even")

else:

print("Odd")

Get a number as integer input. Check whether the entered number is positive or negative or 0

n=int(input())

if n>0:

print("Postive")

elif n<0:

print("Negative")

else:

print("0")

Ram plans to enroll in college soon. To be eligible for a free course enrollment, he must receive >90% in math, physics, and chemistry. Assist Ram with adding up his three subjects' grades and calculating the resultant % so that you can determine whether or not he meets the requirements to enroll.

maths=int(input())

physics=int(input())

chemistry=int(input())

total=maths+physics+chemistry

print(total)

percentage=total/3

print(f"{percentage:.2f}")

if percentage>90:

print("Eligible")

else:

print("Not eligible")

Reena wishes to use just uppercase or lowercase letters when registering names on her site. Determine if a string input is uppercase, lowercase, or a combination of the two.

s=str(input())

if s.islower():

print("Lower")

elif s.isupper():

print("Upper")

else:

print("Combination of Both")

You are building a system to manage a library's collection of books. You need to create functions to manage book lists, including displaying available books,

n=int(input("enter the number of books"))

l=[]

for i in range(n):

books=input()

l.append(books)

print("1)display\_books 2)add\_new\_books 3)remove\_book")

while True:

option=int(input("enter the option"))

match option:

case 1:

def display\_books():

for i in l:

print(i)

display\_books()

case 2:

def add\_new\_books():

add=input("Enter the new book you wnant to be added")

l.append(add)

print("After adding:",l)

add\_new\_books()

case 3:

def remove\_books():

remove=input("enter the book you want to be removed")

l.remove(remove)

print("After Removing:",l)

remove\_books()

case \_:

quit()

You are building an inventory management system for a store. You need to handle a list of products,including viewing the inventory, adding new products, and removing discontinued products.

d={"product 1":8,"product2":9,"product 3":10}

print("1)display 2)add 3)remove")

while True:

choice=int(input("enter the choice:"))

match choice:

case 1:

def display():

for i in d.items():

print(i)

display()

case 2:

def add\_new\_product():

product=input("enter the new product name")

quantity=int(input("enter the new product's quantity:"))

d[product]=quantity

print("after adding the products and quantity:",d)

add\_new\_product()

case 3:

def remove():

remove=input("Enter the discontinued product:")

if remove in d:

del d[remove]

print("after removing the product:",d)

remove()

case \_:

quit()

Write a python program to find the minimum of two numbers using ternary operator.

a=int(input())

b=int(input())

print(a if a<b else b)

Given two numbers, write a python program to swap two numbers.

a=int(input())

b=int(input())

a,b=b,a

print(a,b)

Create a list L. Add 5 items to the list. Check whether an item is present in the list L using membership operator.

L=["abc",1,3,"def",8]

print("abc" in L)

Given a string S. Check for a particular substring not present in string S using membership operator.

S="I am Rickshitha"

print( True if "Rickshi" not in S else False)

Given the principal amount, rate and time. Find the compound interest in python.

import math

principal=float(input("principal:")) #principal amount 1000

rate=float(input("rate:")) #rate 5.5%

rate\_int=rate/100 # convert peprcentage to decimal rate\_int=5.5/100=0.055

time=float(input("time:")) #time 2

annual\_amount=principal\*(1+rate\_int)\*\*time #1000\*(1+0.055)\*\*2= 1113.02

compound=annual\_amount-principal # 1113.02-1000= 113.02

print(math.trunc(compound))

Check whether a number is prime or not.

n=int(input())

c=0

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

if (n%i==0):

c+=1

if c==2:

print("It is Prime Number")

else:

print("It is Not A Prime Number")

Find the factorial of a number.

import math

n=int(input())

num=math.factorial(n)

print(num)

Find the fibanocci series upto N number.

num=int(input())

n1=0

n2=1

c=0

if num <=0:

print("Enter A Positive Numbers")

elif num==1:

print(n1)

else:

while c<num:

print(n1)

n3=n1+n2

n1=n2

n2=n3

c+=1

Check whether a number is divisible by 11 or not.

n=int(input())

if (n%11==0):

print("It is Divisible By 11")

else:

print("Not Divisible By 11")

Find the factors of a number.

def factors\_num(n):

factors=[]

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

if n%i==0:

factors.append(i)

return factors

num=int(input())

print(factors\_num(num))

Create a set and check if a set is a subset of another set.

natural={1,2,3,4,5,6,7,8,9,10}

even={2,4,6,8,10}

if even.issubset(natural):

print( "even is a subset of natural")

else:

print( "even is not a subset of natural")

Create a tuple and slice the tuple. Example: 8 elements in tuple means slice from 2 to 5.

odd=(1,3,5,7,9,11,13,15)

odd=odd[2:6]

print(odd)

Create a tuple and unpack a tuple into variables.

number=(-1,2,3,4,0,8)

negative,even\_prime,odd,positive,neutral,even=number

print("Negative=",negative)

print("Even Prime=",even\_prime)

print("Odd=",odd)

print("Positive=",positive)

print("Netural=",neutral)

print("Even=",even)

Given list of tuples, remove all the tuples with length K.

lst = [(2, 5), (9, ), (8, 7, 6), (4, ), (12, 4, 16, 7)]

new=[]

for i in lst:

if len(i)<3 or len(i)>3:

new.append(i)

print(new)

Convert a tuple of strings to a single concatenated string?

string=("I","Am","Rickshi")

concatenated=" ".join(string)

print(concatenated)

Write a program to convert a list of tuples into a dictionary

friends=("Rickshi","Hafeez","Harini")

age=(18,17,17)

my\_dict=dict(zip(friends,age))

print(my\_dict)

Write a program to print the following pattern.

n=int(input())

for i in range(n,0,-1):

print(" "\*(n-i)+"\* "\*i)

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

print(" "\*(n-i)+"\* "\*i)

Bulidin functions

txt="Hafeesha and Rickshi are friends"

print("upper=",txt.upper())#conveerts a string into upper case

print("lower=",txt.lower())#conveerts a string into lowewr case

print("capitalize=",txt.capitalize ())#first letter converets into caps

print("center=",txt.center(40))

print("count=",txt.count("are"))#it checks how many times the word occurs

print("encode=",txt.encode())#it converts into binary

print("endswith=",txt.endswith("." ))#returns true if the string ends with the specified value

print("startswith=",txt.startswith ("R"))#returns true if the string starts with the specified value

print("casefold=",txt.casefold())#it converts all letters to small

print("expandtabs=", txt.expandtabs (2))

print("index=",txt.index("a"))#it returns the index value of given value

print("alnum=",txt.isalnum())#it checks whther it is alphabets and number or not

print("alpha=",txt.isalpha())#it checks whther it is alphabets or not

print("ascii=",txt.isascii())#it checks whther it is dascii value or not

print("decimal=",txt.isdecimal())#it checks whther it is decimal or not

print("digit=",txt.isdigit())#it checks whther it is digit or not

mytrans=str.maketrans("h", "H")

print(txt.translate(mytrans))#returns a translate string

print("partition=",txt.partition ("Rickshi"))#partition after given word

print("replace=",txt.replace ("Rickshi", "Rickshitha"))#replace the string which the word given only

print("strip=",txt.strip())#trimmied version spaces should be removed

print("title=", txt.title())# change first letter caps

print("swapcase=",txt.swapcase())#convert small to big and big to small

print("zfill=", txt.zfill(35))#fill string with specified no of 0 values at begining

print("rjust=",txt.rjust(40,"#"))#fill with # for 40 times

print("identifier=", txt .isidentifier())#check whether is alpha numeric or underscore is present

print("printable=",txt.isprintable())

print("Join=",txt.join("and"))#join the given letter ahafesha....nhafeesha...dhafeesha

print("Rfind=",txt.rfind("are"))#find the last occurance of the given word

AGE ELIGIBLE CHECK

Age=int(input())

print(f"{'Eligible' if Age>=24 else 'Not Eligible'}")

ACCOUNT D ETAILS

a\_name=input()

a\_num=int(input())

balance=float(input())

print(f"Account Name: {a\_name}")

print(f"Account Number: {a\_num}")

print(f"Balance: {balance:.2f}")

INSERT USING INDEX

l1=int(input())

l2=int(input())

l3=int(input())

l4=int(input())

l5=int(input())

list=[l1,l2,l3,l4,l5]

print(list)

list[2]=int(input())

print(list)

FINDING AREA

radius=float(input("Enter A Radius:"))

Area=3.14\*radius\*radius

print(Area)

CHOICES

print("1.Sunday\t2.Monday\t3.Tuesday\t4.Wednesday\t5.Thursday\t6.Friday\t7.Saturday")

option=int(input("Enter a Option:"))

if option==1:

print("sunday")

elif option==2:

print("Monday")

elif option==3:

print("tuesday")

elif option==4:

print("wednesday")

elif option==5:

print("thursday")

elif option==6:

print("friay")

elif option==7:

print("saturday")

else:

print("Invalid")

MULTIPLICATION

for i in range(1,11):

for j in range(1,11):

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

print("\t")

REMOVING USING LIST

List=[12,34,56,67]

List.append(90)

List.remove(56)

print(List)

COMPOUND INTEREST

import math

principal=float(input())

rate=float(input())

time=float(input())

amount=math.pow(principal\*(1+rate/100),time)

print(amount)

compound=amount-principal

print(compound)

CHECK PRIME OR NOT

n=int(input())

count=0

for i in range(1,n):

if(n% i==0):

count+=1

if(count==2):

print("yes it is a prime number")

else:

print("no its not a prime number")

ADDING FIRST AND LAST USING DEF FUNCTION

def display():

firstname="rickshi"

lastname="anandakumar"

print(firstname+lastname)

print("program compeleted")

display()

print ("execution completed")

CHECK WHETHER THE SUBSTRING IS FOUND OR NOT

string=input()

substring=input()

try:

pos=string.index(substring)

print(f"{substring} is found")

except:

print(f"{substring} is not found")

ZeroDivisionError

grades=[10,20,30,40,50]

try:

average=sum(grades)/len(grades)

print(average)

except ZeroDivisionError as zde:

print(zde)

IndexError

l=["mobile","laptop","television","personal computer"]

index=int(input())

try:

print(l[index])

except IndexError:

print("Error: Index out of range")

fibonnaci

x=int(input())

a,b=1,1

even=0

result=None

fibonaci=[a,b]

while even<x:

a,b=b,a+b

fibonaci.append(b)

if b%2==0:

even+=1

if even==x:

result=b

print(result)

print(fibonaci)

append and sum

num = int(input("Enter the number of students: "))

books\_read=[]

for i in range(num):

data=int(input("Enter books read by student:"))

books\_read.append(data)

total\_books = sum(books\_read)

print(f"The total number of books read by all the students is: {total\_books}")

row and column using \*

row=int(input())

for i in range(1,row+1,):

for j in range(i):

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

print()

harshad number

num=int(input()) #153

sum\_digit=0

number=num

while num>0: #153>0 | 15>0 | 1>0 | 0.1>0-->failed exit

last=num%10 #153%10=3 |15%10=5 |1%10=1

sum\_digit+=last #0+3=3 | 3+5=8 | 8+1=9

num=num//10 #153//10=15 | 15//10=1 | 1//10=0.1

if number%sum\_digit==0: #153%9==0

print(f"{number} is a Harshad Number")

else:

print(f"{number} is not a Harshad Number")

question

n = int(input()) ## enter the number of students

grades = []# Initialize an empty list to store grades

i = 0

while i < n: #row-->represent a student

l=list(map(int, input().split()))

grades.append(l)

i += 1

i = 0 # Calculate average grade for each student

student\_averages = []

while i < n: #0<3,1<3,2<3,3<3--->failed

total = sum(grades[i]) #sum of row 1,sum of row 2, sum of row 3

avg = total / len(grades[i]) #sum of row/rows length=average

student\_averages.append(avg)

i += 1 # 0+1=1 row 1 | 1+1=2 row 2 | 2+1=3

subject\_high = [] # Find highest grade in each subject

for col in zip(\*grades): # column--> represent a subject

max\_val = max(col)

subject\_high.append(max\_val)

total\_grades = 0 # Calculate overall class average

for row in grades:

total\_grades += sum(row)

class\_average = total\_grades / (n \* len(grades)) #toal\_grades/no.of students\*len(grades)

print('Average grades for each student:') # Print results

i = 0

while i < n:

print(f'Student {i+1}: {student\_averages[i]:.2f}')

i += 1

print('Highest grade in each subject:')

print('Math:', subject\_high[0])

print('Science:', subject\_high[1])

print('English:', subject\_high[2])

print('Overall class average:', class\_average)

question

n=int(input()) #enter the number of products

inventory = []

i = 0

while i < n: # Assuming n products 0<3 | 1<3 |2<3 |3<3-->failed

row = list(map(int, input().split()))

inventory.append(row)

i += 1

i = 0

product\_totals = [] #calulates the total quantity

while i < n:

total = sum(inventory[i])

product\_totals.append(total)

i += 1

print("Total quantities of each product:")# Print total quantities of each product

i = 0

while i < n:

print(f"Product {i+1}: {product\_totals[i]}")

i += 1

product\_3\_quantities = inventory[2] # Product 3 is at index 2

max\_quantity = max(product3\_quantities)

section\_index = product\_3\_quantities.index(max\_quantity)

sections = ['A', 'B', 'C'] # Assuming 3 sections

print(f"Section with the highest quantity for Product 3: Section {sections[section\_index]}")

lowest\_total = min(product\_totals) # product with lowest total quantity

product\_index = product\_totals.index(lowest\_total)

print(f"Product with the lowest total quantity: Product {product\_index + 1}")

question

start=int(input())

end=int(input())

print("Odd Numbers:",end=" ")

for i in range(start,end):

if i%2==1:

print(i,end=" ")

print("\nEven Numbers:",end=" ")

for i in range(start,end):

if i%2==0:

print(i,end=" ")

reverse the string and convert to upper the string

string=input()

upper=string.upper()

print(upper[::-1])

count of negative and positive number

s=[150,-20,300,-50,200,-10,400,-30]

suc=0

los=0

for i in s:

if i>0:

suc+=1

elif i<0:

los+=1

print("Number of successful sales:",suc)

print("Number of returns or losses:",los)

class using employeedetails

class Employee:

def \_\_init\_\_(self,emp\_id,emp\_name,emp\_salary,emp\_dept):

self.emp\_id=emp\_id

self.emp\_name=emp\_name

self.emp\_salary=emp\_salary

self.emp\_dept=emp\_dept

def assign\_department(self,new\_dept):

self.emp\_dept=new\_dept

def print\_employee\_details(self):

print(f"The id is :{self.emp\_id}")

print(f"the name is :{self.emp\_name}")

print(f"the salary is :{self.emp\_salary}")

print(f"the department is: {self.emp\_dept}")

def calculate\_salary(self,hoursworked):

if hoursworked>50:

over\_time=hoursworked-50

overtime\_amount=over\_time\*(self.emp\_salary/50)

total\_salary=self.emp\_salary+overtime\_amount

else:

total\_salary=self.emp\_salary

return total\_salary

employees=[

Employee(101,"Rickshi",14000,"Developer"),

Employee(102,"Preety",16500,"Tester"),

Employee(103,"Sudar",50000,"Analyst"),

Employee(104,"Sandhiya",68000,"Manager")

]

employees[1].assign\_department("HR")

employees[1].print\_employee\_details()

salary\_with\_overtime=employees[0].calculate\_salary(55)

print(f"Total salary:{salary\_with\_overtime}")

check the divisible

n=int(input())

if n%3==0 and n%5==0:

print("Divisible By Both")

elif n%3==0:

print("Divisible by 3")

elif n%5==0:

print("Divisible by 5")

else:

print("Invalid Number")

question

import math

I1 = float(input("Enter cost of first item: "))

I2 = float(input("Enter cost of second item: "))

I3 = float(input("Enter cost of third item: "))

total\_bill = I1 + I2 + I3 #200+70+150=420

if total\_bill > 400:

tax =math.trunc(total\_bill \* 0.0675) #420\*0.0675=28.35

totalbill\_with\_tax = math.trunc(total\_bill + tax) #420+28.35=448.35

tip = math.trunc(totalbill\_with\_tax \* 0.10) #448.35\*0.10=44.83

final\_total = math.trunc(totalbill\_with\_tax + tip) #448.35+44.83=493.18

else:

tax = 0

tip = 0

final\_total = total\_bill

print(int(total\_bill))

print(int(tax))

print(int(tip))

print(int(final\_total))

checking obesity

weight=float(input("weight:"))

height=float(input("height:"))

BMI=weight/height\*height

if BMI<16:

print("Severe Thinness")

elif BMI>16 and BMI<17:

print("Moderate Thinness")

elif BMI>17 and BMI<18.5:

print("Mild Thinness")

elif BMI>18.5 and BMI<25:

print("Normal")

elif BMI>25 and BMI<30:

print("Overweight")

elif BMI>30 and BMI<35:

print("Obese Class I")

elif BMI>35 and BMI<40:

print("Obese Class II")

elif BMI>40:

print("Obese Class III")

else:

print("Obese")

question

n,\*arr, x = map(int, input("Enter the number ele, array ele, and the majority element: ").split())

target=int(input("target element:"))

count = arr.count(x)

if count > n/ 2:

print(f"{target} is a majority element.")

else:

print(f"{target} is not a majority element.")

finding second largest

arr = list(map(int, input().split()))

arr = sorted(set(arr))

print(arr)

if len(arr) < 2:

print("No second-largest element")

else:

print("Second-largest element:", arr[-2])

or

n=int(input("Enter the number of elemnts in the array"))

array=[]#5,3,2,1

for i in range(n):

elements=int(input())

array.append(elements)

largest=min(array)#1#5

for i in array:

if i>largest:

second\_largest=largest#1

largest=i#5

elif i>second\_largest and i<largest:

second\_largest=i#3

print(second\_largest)

replacing the numbers

number=9984043892

number=str(number)

print("Replace Number:", number.replace('0','7'))

question

login\_credentials = {

"E24AIOU01": "E24AIOU01@123",

"E24AIOU37": "E24AIOU37@123",

"E24AIOU06": "E24AIOU06@123"}

username = input("Enter your username: ")

password = input("Enter your password: ")

if username in login\_credentials:

if login\_credentials[username] == password:

print("Successfully logged in")

else:

print("Incorrect password")

else:

print("Invalid username")

add negative values then positive value

size = int(input("Enter the size of the array: "))

arr = list(map(int, input("Enter the elements of the array: ").split()))

negatives = [x for x in arr if x < 0]

positives= [x for x in arr if x >= 0]

result = negatives + positives

print(result)

#or

n=int(input()) #4

a=[] #list based for i in range(n):

a.append(int(input())) #2-87-7 i=0

for i in range(0, n): # if (a[i] < (0): # a[0]<0? | a[1] <0? | a[2]<0? | a[3]<0?--stop

t=a[i] #t=-7 a[i]=a[j] #a[3]-2

a[j]=t #a[1]=-7

j=j+1#j=2

print(a)#-8-772

question

for i in range(65,70):

for j in range(65,i+1):

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

print()

Write the output for the following:

for x in range(10,20):

if x%2==0:

continue

print(x) #11 13 15 17 19

question

for x in range(3, 9, 2):

print(x \* 10)

question

import array as arr

n=int(input())

array=[]

list\_elements=[]

for i in range(n):

user\_id=int(input())

array.append(user\_id)

for i in array:

if i not in list\_elements:

list\_elements.append(i)

else:

print(i)

question

l=[10,45,1,67,56]

print("List=",l)

l.sort()

print("Ascending Order=",l)

l.reverse()

print("Descending Order=",l)

question

s={1,"a",2,'b',3,'c'}

s1={2,"c",4,"e",5}

s2=(s,s1)

print("Value:",s2)

s3=s.intersection(s1)

print("Intersection=",s3)

s4=s.symmetric\_difference(s1)

print("Symmetric=",s4)

s.remove(1)

print("Remove=",s)

question

t=("a","b","c","d")

print("Tuple=",t)

l=list(t)

l.append("e")

l.append("f")

t=tuple(l)

print("Append tuple=",t)

l=list(t)

l.remove("b")

t=tuple(l)

print("Remove=",t)

question

s=" Tomorrow's Homework"

print("Str=",s)

Last=s.rindex("o")

print("Last Index:",Last)

Replace=s.replace("H","h")

print("Replace:",Replace)

Remove=s.lstrip()

print("Remove:",Remove)