# A) Session 1 & 2

# Input / Output

# 1. Accept Empid,EmpName,Monthly\_Salary,Tot\_Deductions, Tot\_Allowances

# and Display Employee Name and Salary in hand

empId = int(input("Enter empId: "))

empName = input("Enter name: ")

salary = int(input("Enter salary: "))

allowances = int(input("Enter allownces: "))

print(f"EmpName: {empName}, Salary: {salary}")

# if Conditions :

# 1. Accept 3 integers from the User and Display Maximum

number1 = int(input("Enter number 1 :"))

number2 = int(input("Enter number 2 :"))

number3 = int(input("Enter number 3 :"))

if(number1>number2 and number1>number3):

    print(f"{number1} is maximum")

elif(number2>number1 and number2>number3):

    print(f"{number2} is greatest")

else:

    print(f"{number3} is greatest")

# 2. Accept 3 integers from USer and display Minimum

number1 = int(input("Enter number 1 :"))

number2 = int(input("Enter number 2 :"))

number3 = int(input("Enter number 3 :"))

if(number1<number2 and number1<number3):

    print(f"{number1} is smallest")

elif(number2<number1 and number2<number3):

    print(f"{number2} is smallest")

else:

    print(f"{number3} is smallest")

# loops (Solve without Using Functions if any)

# 1. Accept Integers from User till Users Choice and do the Following:

# 1. Sum of all Integers

# 2. Average of all Integers

# 3. Maximum Integer from all

# 4. Minimum Integer from all

list =[]

while(True):

    num = input("Enter a number, enter end to exit: ")

    if(num.upper() == 'END'):

        break

    list.append(int(num))

if(len(list)>0):

    sum = 0

    avg = 0

    max = list[0]

    min = list[0]

    for nums in list:

        sum+= nums

        if(nums>max):

            max = nums

        if(nums<min):

            min = nums

    avg = sum/len(list)

    print(f"Sum : {sum}\nAverage : {avg}\nMaximum : {max}\nMinimum : {min}")

# 2. Accept a String from User an do the following :

# 1. Find the Length

stringInput = input("Enter string: ")

lengthOfString = len(stringInput)

print(f"Length of string : {lengthOfString}")

# 2. Display String in reverse

reverseStr = ""

for i in range(len(stringInput)-1,-1,-1):

    reverseStr+= stringInput[i]

print(f"Reverse of string : {reverseStr}")

# 2. Display every alternate Character in Upper Case

for i in range(0,len(stringInput),2):

    print(f"Alternating char: {stringInput[i].upper()}")

# 3. Find out No of Vowels in the String

numberOfVowels = 0

for i in range(0,len(stringInput)):

    if(stringInput[i].lower() in ['a','e','i','o','u']):

        numberOfVowels+= 1

# 4. Accept Username and Date of Birth (dd-mon-yy) from User

userName = input("Enter userName: ")

dob = input("Enter DOB in DD-MMM-YY format: ")

print(f"UserName: {userName}")

print(f"DOB: {dob}")

# Create a Password String which will be combination of

# 1st 4 letters of username and last 2digits of Date of Birth

# followed by $ sign

password = ""

if(len(userName)<4):

    password+= userName

else:

    password+= userName[0:4]

password+= dob[-2:]

print(f"Password: {password}")

# 5. Encrypt the String and return Encrypted String

# (Assume your Algorithm)

encryptedPassword = ""

for i in range(int(len(password)/2),len(password)):

    encryptedPassword+= password[i]

for i in range(0,int(len(password)/2)):

    encryptedPassword+= password[i]

print(f"Encrypted password : {password} to {encryptedPassword}")

# 3. Write Python Program to do the following :

# 1. Display Area of

# Circle

radius = int(input("Enter radius: "))

areaOfCircle = 3.14\*radius\*radius

print(f"Area of circle with radius {radius} is : {areaOfCircle}")

# Parallelogram

base = int(input("Enter the base of parrallelogram: "))

height = int(input("Enter the height of parrallelogram: "))

areaOfParrellolagram = base\*height

print(f"Area of parrellolagram is {areaOfParrellolagram}")

# 4. Accept Integer and find Square root of Integer

toSquareNum = int(input("Enter a number to find sqaure of it"))

squared = toSquareNum\*toSquareNum

print(f"Square of {toSquareNum} is {squared}")

# B) Session 3 / 4

# List / Tuples / Dictionary / Sets

# 1. Create a List for the following :

# a. Accept Fruits Name and their price(per kg)

# b. Fruits Name should be at odd index position in the List.

# Price at even index position

fruitsList = []

while(True):

    fruitName = input("Enter a fruit name, enter end to Exit: ")

    if(fruitName.lower() == 'end'):

        break

    fruitPrice = int(input(f"Enter Price for {fruitName}: "))

    fruitsList.append(fruitName)

    fruitsList.append(fruitPrice)

print(f"FruitList : {fruitsList}")

# 2. Customer will buy fruits from you (Show him the Fruits Menu)

# Write a Program to

# a. Calculate Total Price of Fruits Bought .

# (Assume price for 1 kg )

# b. Add New Fruits in the List

# c. Show Total Fruits in the List

totalFruitPrice = 0

for i in range(1,len(fruitsList),2):

    totalFruitPrice += fruitsList[i]

print(f"TotalCost: {totalFruitPrice}")

while(True):

    fruitName = input("Enter a fruit name, enter end to Exit: ")

    if(fruitName.lower() == 'end'):

        break

    fruitPrice = int(input(f"Enter Price for {fruitName}: "))

    fruitsList.append(fruitName)

    fruitsList.append(fruitPrice)

print(f"FruitList : {fruitsList}")

# 3. Create Foll. Information in the Tuple (atleast 5 Employees)

# 1. EmpId - Phone Numbers (One Employee can have Multiple Numbers )

# 2. Accept Empid from User.

# Display his Numbers only if he exists in the Database(Tuple)

# Display App. Message if not present

employeeTuple = ([101,"+91 8838902352","+91 8848902352"],

                 [102,"+91 9999999890","+91 8888888890"],

                 [103,"+91 7777777789","+91 6666666679"],

                 [104,"+91 5555555567","+91 4444444467"],

                 [105,"+91 2222222224","+91 3333333331"])

empIdInput = int(input("Enter EmpId: "))

isEmployeeFound = False

for i in range(0,len(employeeTuple)):

    if(employeeTuple[i][0] == empIdInput):

        isEmployeeFound = True

        print(f"Mobile numbers of {empIdInput}: {employeeTuple[i][1:]}")

if(isEmployeeFound == False):

    print("Employee not found")

# 3. Update Employee phone Number

# Accept Empid from User

# Check whether he / she Exists

# Accept New Phone Number

# Update

# Display Appropriate Message for any task

empIdInput = int(input("Enter EmpId: "))

isEmployeeFound = False

tempList = list(employeeTuple)

for i in range(0,len(tempList)):

    if(tempList[i][0] == empIdInput):

        isEmployeeFound = True

        newMobileNumber = input("Enter new number: ")

        updatedEmpList = [empIdInput]

        updatedEmpList.append(newMobileNumber)

        tempList[i] = updatedEmpList

        employeeTuple = tuple(tempList)

        print("Updated ",employeeTuple)

if(isEmployeeFound == False):

    print("Employee not found")

# 4. Store the Following info in Dictionary

# Department Name and their Employee Names

# Note : One Department can have multiple Employees

departmentDict = {"CSE": ["Jay","Sam"]}

# Perform the Following Operations :

# 1. Add a New Department Name and Employees in that Department

# If a New Department Name doesnot Exists

departmentNameInput = input("Enter a new department name: ")

if(departmentNameInput not in departmentDict):

    employeeList = []

    while(True):

        employeesName = input("Enter employee names, enter 'end' to Exit: ")

        if(employeesName.lower() == 'end'):

            break

        employeeList.append(employeesName)

    departmentDict[departmentNameInput] = employeeList

    print(f"Updated dict : {departmentDict}")

# 2. Accept Dept Name from User and List all Employees

# If Dept Name Exists in the Database

departmentNameInput = input("Enter an existing department name to display: ")

if(departmentNameInput in departmentDict):

    print(f"Department exisits, {departmentNameInput} employees:  {departmentDict[departmentNameInput]}")

# 3. Add a New Employee in Existing Department

departmentNameInput = input("Enter an existing department name to add new employee: ")

if(departmentNameInput in departmentDict):

    newEmployee = input("Enter a new employee to add: ")

    departmentDict[departmentNameInput].append(newEmployee)

    print(f"Department updated, {departmentNameInput} employees:  {departmentDict[departmentNameInput]}")

# 4. Delete Existing Employee From Department

departmentNameInput = input("Enter an existing department name to delete a employee: ")

if(departmentNameInput in departmentDict):

    removeEmployee = input("Enter an existing employee to delete: ")

    departmentDict[departmentNameInput].remove(removeEmployee)

    print(f"Department updated, {departmentNameInput} employees:  {departmentDict[departmentNameInput]}")

# 5. Create Following two Sets

# 1. Fruit\_Salesman1

# 2. Fruit\_Salesman2

fruit\_salesMan1 = {"Orange","Papaya"}

fruit\_salesMan2 = {"Apple","Orange"}

# Create Fruits for both Salesmans

fruit\_salesMan1.add("Pineapple")

fruit\_salesMan1.add("Lichi")

fruit\_salesMan2.add("DragonFruit")

fruit\_salesMan2.add("Watermelon")

# Perform the Following Operations

# 1. Find out Common Fruits with both Salesman

commonFruits = fruit\_salesMan1.intersection(fruit\_salesMan2)

print(f"Common fruits in both: {commonFruits}")

# 2. List Extra Fruits with Both Salesman

print(f"Extra fruits of salesman1 from salesman2: {fruit\_salesMan1.difference(fruit\_salesMan2)}")

print(f"Extra fruits of salesman2 from salesman1: {fruit\_salesMan2.difference(fruit\_salesMan1)}")

# 3. List Total Fruits with both Salesman

allFruits = fruit\_salesMan1.union(fruit\_salesMan2)

print(f"Total fruits of both salesman: {allFruits}")