**Session 1/2**

'''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: ')

Monthly\_Salary = int(input('Enter Salary: '))

Tot\_Deductions = int(input('Enter Deductions: '))

Tot\_Allowances =  int(input('Enter Total allowances: '))

print('Employee name is {}'.format(EmpName))

print('Employee salary is {}'.format(Monthly\_Salary - Tot\_Deductions + Tot\_Allowances))

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

num1 = int(input('Enter a number: '))

num2 = int(input('Enter a number: '))

num3 = int(input('Enter a number: '))

if num1 > num2 and num1 > num3:

  print(num1, 'is the Maximum')

elif num2 > num1 and num2 > num3:

  print(num2, 'is the Maximum')

else:

  print(num3, 'is the Maximum')

'''2. Accept 3 integers from the User and Display Minimum'''

num1 = int(input('Enter a number: '))

num2 = int(input('Enter a number: '))

num3 = int(input('Enter a number: '))

if num1 < num2 and num1 < num3:

  print(num1, 'is the Minimum')

elif num2 < num1 and num2 < num3:

  print(num2, 'is the Minimum')

else:

  print(num3, 'is the Minimum')

'''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'''

itr = int(input('Enter the number of integers: '))

arr = []

for i in range(itr):

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

  arr.append(num)

print('Sum of all Integers={}'.format(sum(arr)))

print('Average of all Integers={}'.format(sum(arr)/itr))

print('Maximum Integer from all={}'.format(max(arr)))

print('Minimum Integer from all={}'.format(min(arr)))

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

1. Find the Length

2. Display String in reverse

2. Display every alternate Character in Upper Case

3. Find out No of Vowels in the String

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

Create a Password String which will be combination of

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

followed by $ sign

5. Encrypt the String and return Encrypted String

(Assume your Algorithm)'''

user\_string = input("Enter a string: ")

string\_length = len(user\_string)

print("Length of the string: {}".format(string\_length))

reversed\_string = user\_string[::-1]

print("Reversed string: {}".format(reversed\_string))

alternate\_uppercase = ""

for i in range(len(user\_string)):

    if i % 2 == 0:

        alternate\_uppercase += user\_string[i].upper()

    else:

        alternate\_uppercase += user\_string[i]

print("Alternate characters in uppercase: {}".format(alternate\_uppercase))

vowels = "aeiouAEIOU"

vowel\_count = 0

for char in user\_string:

    if char in vowels:

        vowel\_count += 1

print("Number of vowels in the string: {}".format(vowel\_count))

username = input("Enter your username: ")

dob = input("Enter your Date of Birth (dd-mon-yy): ")

password = username[:4] + dob[-2:] + "$"

print("Generated password: {}".format(password))

encrypted\_password = ""

for char in password:

    encrypted\_password += chr(ord(char) + 3)  # Shift each character by 3 in ASCII

print("Encrypted password: {}".format(encrypted\_password))

'''3. Write Python Program to do the following :

1. Display Area of

Circle

Parallelogram'''

radius = float(input("Enter the radius of the circle: "))

area\_circle = 3.14 \* radius \*\* 2

print(f"Area of the circle: {area\_circle:.2f}")

base = float(input("Enter the base of the parallelogram: "))

height = float(input("Enter the height of the parallelogram: "))

area\_parallelogram = base \* height

print(f"Area of the parallelogram: {area\_parallelogram:.2f}")

'''4. Accept Integer and find Square root of Integer'''

num = int(input("Enter an integer: "))

square\_root = num \*\* 0.5

print("The square root of {} is {}".format(num,square\_root))

**Session 3/4**

'''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'''

num = int(input("Enter the no. of fruits: "))

List1 = []

for i in range(num):

  fruits = input('Enter the name of the fruit: ')

  price\_per\_kg = int(input('Enter price peer kg: '))

  List1.append(fruits)

  List1.append(price\_per\_kg)

print(List1)

'''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'''

fruit\_menu = ['Apple', 150, 'Banana', 50, 'Orange', 70, 'Grapes', 120, 'Mango', 100]

# Display the fruit menu

print("Fruits Menu:")

for i in range(0, len(fruit\_menu), 2):

    print("{}: Rs.{} per kg".format(fruit\_menu[i], fruit\_menu[i + 1]))

# Calculate total price of fruits bought

total\_price = 0

while True:

    fruit\_choice = input("Enter the fruit you want to buy (or type 'done' to finish): ").capitalize()

    if fruit\_choice.lower() == 'done':

        break

    # Check if the fruit is in the menu

    found = False

    for i in range(0, len(fruit\_menu), 2):

        if fruit\_choice == fruit\_menu[i]:

            quantity = float(input("Enter the quantity in kg for {}: ".format(fruit\_choice)))

            total\_price += fruit\_menu[i + 1] \* quantity

            found = True

            break

    if not found:

        print("Fruit not available. Please choose from the menu.")

# Display total price of fruits bought

print("\nTotal Price of Fruits Bought: ₹{:.2f}".format(total\_price))

# Add new fruits to the list

while True:

    add\_fruit = input("Do you want to add a new fruit to the list? (yes/no): ").lower()

    if add\_fruit == 'yes':

        new\_fruit\_name = input("Enter the name of the new fruit: ").capitalize()

        new\_fruit\_price = float(input("Enter the price for 1 kg of {}: ".format(new\_fruit\_name)))

        fruit\_menu.append(new\_fruit\_name)

        fruit\_menu.append(new\_fruit\_price)

        print("{} has been added to the menu.".format(new\_fruit\_name))

    elif add\_fruit == 'no':

        break

    else:

        print("Please answer with 'yes' or 'no'.")

# Show total fruits in the list

print("\nUpdated Fruits Menu:")

for i in range(0, len(fruit\_menu), 2):

    print("{}: ₹{} per kg".format(fruit\_menu[i], fruit\_menu[i + 1]))

print("\nTotal number of fruits in the list: {}".format(len(fruit\_menu) // 2))

'''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

'''

employee\_data = (101, '1234567890', '9876543210', 102, '9856555555', 103, '9912223333', '8745556666', 104, '9998887777', 105, '6667778888', '3334445555'

)

i = 0

while i < len(employee\_data):

    emp\_id = employee\_data[i]

    print(f"Employee ID: {emp\_id}, Mobile Numbers:", end=" ")

    i += 1

    while i < len(employee\_data) and isinstance(employee\_data[i], str):

        print(employee\_data[i], end=" ")

        i += 1

    print()

# Accept Employee ID from the user

emp\_id\_input = int(input("Enter Employee ID: "))

i = 0

found = False

# Search through the tuple

while i < len(employee\_data):

    emp\_id = employee\_data[i]

    if emp\_id == emp\_id\_input:

        found = True

        print(f"Employee ID: {emp\_id}, Mobile Numbers:", end=" ")

        i += 1

        while i < len(employee\_data) and isinstance(employee\_data[i], str):

            print(employee\_data[i], end=" ")

            i += 1

        print()

        break

    i += 1

    while i < len(employee\_data) and isinstance(employee\_data[i], str):

        i += 1

if not found:

    print(f"Employee ID {emp\_id\_input} not found in the database.")

'''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'''

employee\_data = (101, '1234567890', '9876543210', 102, '9856555555', 103, '9912223333', '8745556666', 104, '9998887777', 105, '6667778888', '3334445555'

)

employee\_data\_list = list(employee\_data)

emp\_id\_input = int(input("Enter Employee ID: "))

i = 0

found = False

while i < len(employee\_data\_list):

    emp\_id = employee\_data\_list[i]

    if emp\_id == emp\_id\_input:

        found = True

        print(f"Employee ID: {emp\_id}, Mobile Numbers:", end=" ")

        i += 1

        while i < len(employee\_data\_list) and isinstance(employee\_data\_list[i], str):

            print(employee\_data\_list[i], end=" ")

            i += 1

        print()

        break

    i += 1

    while i < len(employee\_data\_list) and isinstance(employee\_data\_list[i], str):

        i += 1

if not found:

    print(f"Employee ID {emp\_id\_input} not found in the database.")

else:

    new\_phone\_number = input("Enter new phone number to add: ")

    index\_to\_insert = i

    employee\_data\_list.insert(index\_to\_insert, new\_phone\_number)

    print(f"Phone numbers updated for Employee ID {emp\_id\_input}. New phone numbers:")

    i -= 1

    i += 1

    while i < len(employee\_data\_list) and isinstance(employee\_data\_list[i], str):

        print(employee\_data\_list[i], end=" ")

        i += 1

    print()

employee\_data = tuple(employee\_data\_list)

'''4. Store the Following info in Dictionary

Department Name and their Employee Names

Note : One Department can have multiple Employees

Perform the Following Operations :

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

If a New Department Name doesnot Exists

2. Accept Dept Name from User and List all Employees

If Dept Name Exists in the Database

3. Add a New Employee in Existing Department

4. Delete Existing Employee From Department'''

# Initialize the dictionary with department names as keys and employee names as values (list)

departments = {

    'HR': ['John', 'Emily'],

    'Finance': ['Anna', 'Mike'],

    'IT': ['James', 'Sophia'],

    'Marketing': ['David', 'Linda']

}

# 1. Add a new department and employees if it doesn't exist

dept\_name = input("Enter the new department name to add: ")

if dept\_name not in departments:

    employees = input(f"Enter the employee names for the {dept\_name} department (comma-separated): ").split(', ')

    departments[dept\_name] = employees  # Add new department with its employees

    print(f"Department '{dept\_name}' has been added with employees: {employees}")

else:

    print(f"Department '{dept\_name}' already exists.")

# 2. Accept department name from user and list all employees if department exists

dept\_name = input("\nEnter the department name to list employees: ")

if dept\_name in departments:

    print(f"Employees in the {dept\_name} department: {', '.join(departments[dept\_name])}")

else:

    print(f"Department '{dept\_name}' does not exist.")

# 3. Add a new employee to an existing department

dept\_name = input("\nEnter the department name to add a new employee: ")

if dept\_name in departments:

    new\_employee = input(f"Enter the name of the new employee to add to the {dept\_name} department: ")

    departments[dept\_name].append(new\_employee)

    print(f"Employee '{new\_employee}' has been added to the {dept\_name} department.")

else:

    print(f"Department '{dept\_name}' does not exist.")

# 4. Delete an existing employee from a department

dept\_name = input("\nEnter the department name to delete an employee from: ")

if dept\_name in departments:

    employee\_to\_remove = input(f"Enter the name of the employee to remove from the {dept\_name} department: ")

    if employee\_to\_remove in departments[dept\_name]:

        departments[dept\_name].remove(employee\_to\_remove)

        print(f"Employee '{employee\_to\_remove}' has been removed from the {dept\_name} department.")

    else:

        print(f"Employee '{employee\_to\_remove}' does not exist in the {dept\_name} department.")

else:

    print(f"Department '{dept\_name}' does not exist.")

# Display the final state of the departments and their employees

print("\nUpdated Departments and Employees:")

for dept, employees in departments.items():

    print(f"{dept}: {', '.join(employees)}")

'''Create Following two Sets

1. Fruit\_Salesman1

2. Fruit\_Salesman2

Create Fruits for both Salesmans

Perform the Following Operations

1. Find out Common Fruits with both Salesman

2. List Extra Fruits with Both Salesman

3. List Total Fruits with both Salesman'''

# Creating sets for both salesmen

Fruit\_Salesman1 = {'Apple', 'Banana', 'Mango', 'Orange'}

Fruit\_Salesman2 = {'Banana', 'Orange', 'Grapes', 'Pineapple'}

# 1. Find out common fruits with both Salesmen

common\_fruits = Fruit\_Salesman1.intersection(Fruit\_Salesman2)

print("Common Fruits between Salesman1 and Salesman2:", common\_fruits)

# 2. List extra fruits with both Salesmen

extra\_fruits\_salesman1 = Fruit\_Salesman1.difference(Fruit\_Salesman2)

extra\_fruits\_salesman2 = Fruit\_Salesman2.difference(Fruit\_Salesman1)

print("Extra Fruits with Salesman1:", extra\_fruits\_salesman1)

print("Extra Fruits with Salesman2:", extra\_fruits\_salesman2)

# 3. List total unique fruits with both Salesmen

total\_fruits = Fruit\_Salesman1.union(Fruit\_Salesman2)

print("Total Fruits with both Salesmen:", total\_fruits)