Task 4 - Use various data type list, Tuples and Dictionary in python. 22/8/25

a) Students Marks manager.

Aim: write a python program to:

- Store marks of student in a list.
- Find the highest and lowest marks.
- Calculate the average mark.
- Sort the marks in ascending order.

Algorithm:

1. Start
2. Input number of students n.
3. Initialize an empty list marks = [ ]
4. Repeat for i = 1 to n.
5. Find highest marks = max (marks) and Lowest mark = min (marks)
6. Calculate Average mark = sum (marks) / n
7. Sort the marks in Ascending order → marks.sort()
8. Display: All student marks
   Highest mark, Lowest mark, Average mark
9. Stop.

program:

```
# students marks manager
marks = [85, 92, 76, 58, 90, 71]
highest = max (marks)
lowest = min (marks)
average = Sum (marks)/len (marks)
Sorted _ marks = Sorted (marks)
Print ("Student marks :", marks)
print ("Highest marks :", highest)
print ("Lowest marks :", lowest)
print ("Average marks :", Average)
Print ("Marks in Ascending. order:", Sorted_marks)
```

Output:

```
Students marks : [85, 92, 76, 58, 90, 71]
highest marks : 92
lowest marks : 58
Average marks : 78.6666
marks in Ascending order : [58, 71, 76, 85, 90, 92]
marks in Descendinig order : [58, 71, 76, 85, 90, 92]
```

b> Employee Records :

22/8/25

Aim : write a python program to :

- Store records of 5 employees.
- Display the details of highest paid employee.
- Extract only the employee name into a list
- count how many employees earn more than 50,000.

Algorithm :-

1. Start
2. Input details (ID, name, salary) of 5 employees and Store them as tuples in a list.
3. Find and display highest paid employees.
4. Extract employee name into a seperate list.
5. Display Results

program :-

```
# Employee Records
employee = [(101, "Ravi", 45000),
            (102, "Abhi", 55000),
            (103, "Kiran", 60000),
            (104, "Sita", 48000),
            (105, "Arjun", 75000)]

# Display all records
print ("Employee Records:", employees)
# Highest paid employee
highest_paid = max (employees, key= lambda x : x[2])
print ("Highest paid employee :", highest_paid)
# Extract employee names
names = [emp[1] for emp in employees]
print ("Employee names :", names)
# count employees earning more than 50,000
count = sum (1 for emp in employees if emp [2] >
                                         50000)
print ("Employees earning more than 50,000 :", count)
```

Output :

```
Employee Records : [(101, 'Ravi', 45000), (102; 'Abhi', 55000),
          (103, 'Kiran', 60000), (104, 'Sita', 48000),
           (105, 'Arjun', 75000)]

 Highest paid Employee : (105, 'Arjun', 75000)
 Employee Names : ['Ravi', 'Abhi', 'Kiran', 'Sita', 'Arjun'
 Employees earning more than 50,000 : 3
```

C. Shopping cart

22/8/25

Aim: To write a python program that manages a shopping cart using a dictionary.

Algorithm:

1. Create a dictionary with product Name as key and price as value.
2. Display all products with prices
3. Find total bill using sum (values())
4. update the price of one product.
5. Remove a product from the dictionary
6. Display final cart

program:

```
# Shopping cart using dictionary
 cart = {
        "Shirt" : 1200,
        "Jeans" : 2000,
        "Shoes" : 3000,
        "watch" : 2500}

 # Display all products
   print ("product in cart :", cart)

# Total bill
 total = sum (cart. values ())
 print ("Total Bill Amount :", total)

 # update price
   cart ["Jeans"] = 2200
   print ["update cart (Jeans price changed) :", cart)

# Remove product
     cart. pop ("watch")
   print ("Cart after removing watch, cart)
```

VELTECH	
EX No.	4
PERFORMANCE (5)	5
RESULT AND ANALYSIS (3)	5
VIVA VOCE (3)	5
RECORD (4)	
TOTAL (15)	15
SIGN WITH DATE	

17/09/25

Result:

Thus, the program successfully manages an online shopping cart using a dictionary.

output :-

```
products in cart : { 'Shirt' : 1200, 'Jeans' : 2000,
                     'Shoes' : 3000, 'watch' : 2500}
Total Bill Amount : 8700
updated cart (Jeans price changed) : {'Shirt' : 1200,
                     'Jeans' : 2200, 'Shoes' : 3000, 'watch:25
cart after remaining watch : { 'Shirt' :1000,
                     'Jeans' : 2200, 'Shoes' : 30000}
```