13/08/25

Task 4: Use various data types, list, tuples and dictionary in python programming

(a) shopping cart price calculator (list)

Aim:

To write a python program the stores the prices of items purchased in a list and calculate:

- The total bill amount
- The highest-priced item
- the lowest-priced item

Algorithm:

```
1. Start the program
2. Create a list to store price of item
3. use sum() to calculate the total bill amount
4. use max() to find the highest-priced item
5. use min() to find the lowest-priced item
6. Display the results
7. End the program
```

Program:

```
prices = [120,85,300,40,150]
total = sum(prices)
highest = max(prices)
lowest = min(prices)
Print ("Total Bill Amount:", total)
print ("Highest-Priced item:", highest)
print ("Lowest-pricesd item:", lowest)
```

Result:

The program successfully calculated the total, highest and lowest-priced items using a list.

output

```
Total Bill Amount : ~~695~~ 695
Highest - priced item : 300
Lowest - priced item : 40
```

13/08/25

b) student Exam Result (tuple)

Aim:

To write a python program that stores students' name and marks in tuples, finds the student with the highest marks, and display all students who scored above 400.

Algorithm:

1. Start the program
2. Create a list of tuples with (student_name, total_marks).
3. Find the student with the highest marks using max()
4. Loop through the list and display students scoring above 400
5. End the program

Program:

```
students = [
    ("Rahul", 456),
    ("Priya", 398),
    ("Amit", 420),
    ("Neha", 389),
    ("Kiran", 470)]
topper = max(students, key = lambda x: x[1])
Print ("Topper:", topper[0], "with", topper[1], "marks")
Print ("Students scoring above 400:")
for name, marks in students:
    if marks > 400:
        Print (name, "-", marks)
```

Result:

The program successfully displayed the student with the highest marks and listed all students who scored above 400 using tuple output:

```
Topper: kiran with 470 marks
students scoring above 400:
Rahul - 456
Amit - 420
kiran - 470
```

13/08/25

c) Country - Capital Finder (Dictionary)

Aim:

To create a python program that manages a dictionary of countries and capitals

Algorithm:

1. Start the program
2. Create a dictionary with country - capital pairs
3. Add a new country - capital entered by the user
4. Search for the capital of a given country.
5. Display all country - capital pairs sorted alphabetically by country.
6. End the program

Program:

```
countries = {"India": "New Delhi", "France": "Paris",
             "Japan": "Tokyo"}
new_country = input("Enter country:")
new_capital = input("Enter capital:")
countries[new_country] = new_capital
search_country = input("Enter country to search:")
if search_country in countries:
     print("Capital of", search_country, "is",
               countries[search_country])
else:
     print("Country not found")
```

Output:

```
Enter Country: Germany
Enter Capital: Berlin
Enter Country to Search: India
Capital of India is New Delhi

All Country-Capital pairs:
France : Paris
Germany: Berlin
India : New delhi
Japan: Tokyo
```

```
Print ("\n All country - Capital pairs:")
for country in sorted(countries):
    print (country, ":", countries[country])
```

VEL TECH - CSE	
EX NO.	4
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	—
TOTAL (20)	15
SIGN WITH DATE	[illegible] 20/8/25

[illegible] 20/8/25

Result: The program successfully allowed adding new country - capital pairs, searching for a Capital, and displaying all pairs in alphabetic order using a dictionary