

4.

13/08/25

Use the various data types, list, tuples and dictionary in python programming.

15

13/08/25

13/8/25

Task-4: Use various data types, List, Tuples

And Dictionary In Python

(a) Shopping Cart Price Calculator (List)

Aim:-

To create a python program that uses a list to store the prices and calculates the total bill amount, the highest priced item, and the lowest priced item.

Algorithm:-

1. Initialize an empty list to store item prices
2. Prompt the user to enter item prices and add them to the list. Continue until the user indicates they are done.
3. Calculate the sum of all prices in the list to get the total bill.
4. Find the maximum value in the list to get the highest priced item.
5. Find the minimum value in the list to get the lowest priced item.
6. Display the total bill, highest priced item, and lowest priced item.

Program:-

prices [ ]

while True:

price - input = input("Enter item price (or 'done' to finish): ")

if price - input . lower ( ) == 'done':

break

try:

price = float (price - input)

prices . append (price)

except value error:

print("Invalid input. please enter a number or 'done'.")

if prices:

total - bill = sum(prices)

highest - price = max (prices)

lowest - price = min (prices)

Program  
 Cost Price Calculator (C++)  
 Author: [Name]  
 Date: [Date]

The program is designed to calculate the cost price of an item based on the selling price and the profit margin. It takes the selling price and the profit margin as input and outputs the cost price.

The program is written in C++ and uses the following variables:

Selling Price (SP), Profit Margin (PM), Cost Price (CP)

### Output:-

Total Bill Amount : 715  
 Highest Priced Item : 300  
 Lowest priced Item : 60

(b) The program is designed to calculate the cost price of an item based on the selling price and the profit margin. It takes the selling price and the profit margin as input and outputs the cost price.

```
print(f "\n Total Bill Amount: $ {total_bill:.2f}")  
print(f " Highest priced item: $ {highest_price:.2f}")  
print(f " Lowest priced item: $ {lowest_price:.2f}")
```

else:

```
print("No prices entered.")
```

Result :-

The program successfully calculates the total bill, highest, and lowest priced items from a list of user-entered prices.

Aim:-

To create a python program that stores student names and marks in tuples, displays the student with the highest marks, and lists all students who scored above 400 marks.

Algorithm:-

1. Start the program
2. Initialize an empty list to store student tuples (name, marks).
3. Loop 5 times to get input for 5 students:
  - a. prompt for student name and marks.
  - b. Create a tuple (name, marks).
  - c. Add the tuple to the list.
4. Find the student with the highest marks by iterating through the list and comparing marks.
5. Iterate through the list again to identify and display students who scored above 400 marks.
6. Display the student with the highest marks and the list of students scoring above 400.
7. Stop the program.

Program:-

```
students = [ ]
```

```
for i in range(5):
```

```
    name = input (f"Enter name of student {i+1} : ")
```

```
    while True :
```

```
        try :
```

```
            marks = int (input (f"Enter marks for {name} : "))
```

```
            if marks < 0 :
```

```
                print ("Marks cannot be negative . Please  
enter a valid number.")
```

```
            else :
```

```
                break
```

```
            except value error :
```

```
                print ("Invalid input. please enter a number for  
marks.")
```

```
    students.append ((name, marks))
```

15000  
3600  
18500

3500

6990  
2000

Output:-

Student with Highest

Marks: Anita - 470

Student Scoring above

400:

Rahul - 456

Karan - 420

Anita - 470

highest\_scorer = students[0]

for student in students:

if student[1] > highest\_scorer[1]:

highest\_scorer = student

print(f"Student with highest marks: {highest\_scorer[0]}")

print(f"Students who scored above 400 marks:")

found\_above\_400 = False

for student in students:

if student[1] > 400:

print(f"- {student[0]} ({student[1]} marks)")

found\_above\_400 = True

if not found\_above\_400:

print("No students scored above 400 marks")

else:

print("No student data entered")

### Result :-

The program successfully identifies the student with the highest marks and lists students scoring above 400 from a collection of student-mark tuples.

### Sample output:-

Enter a country: Germany

Enter a its Capital : Berlin

Enter country to search capital: India

Capital of India is New Delhi

All country- capital pairs (sorted):

France: Paris

Germany: Berlin

India: New Delhi

Japan: Tokyo

Aim:- To store countries and capitals in dictionary and perform operations add new entry, search for capital, display all pairs alphabetically.

Algorithm:-

1. Start the program.
2. Create a dictionary with country - capital pairs.
3. Add a new country - capital pair entered by user.
4. Search for a country's capital using dictionary loop up.
5. Display all country - capital pairs sorted by country name.
6. End the program.

Program:-

python

```
#country - capital finder (Dictionary)
countries = {"India": "New Delhi", "France": "Paris", "Japan": "Tokyo"}
```

#Add new entry

```
country = input("Enter a country: ")
```

```
capital = input("Enter its capital: ")
```

```
countries[country] = capital
```

# Search for a capital

```
search-country = input("Enter country to search capital: ")
```

```
if search-country in countries:
```

```
    print("capital of ", search-country, "is", countries[search-country])
```

else:

```
    print("country not found :")
```

```
print("In All country - Capital pairs (sorted):")
```

```
for c in sorted(countries.keys()):
```

```
    print(c, ":", countries[c])
```

Sample

Result:-

The program to add, search and display country - capital pairs using dictionary was successfully executed.

VEL TECH - CSE	
EX NO.	4
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
AVA VOCE (5)	5
TECH (5)	5
TOTAL	15