Task 4 : Use various data types, list, Tuples and dictionary in Python programming key terms covered : Data types, list, Tuple, Set, Dict.

4.1. list - cafeteria sales

In your college cafeteria, the sales
Aim: Record a cafeteria's snack sales for 7 days using a list, compute total and average sales, find the best/worst day and count how many days crossed a target.

Algorithm:

1. Start
2. Create an Empty list sales = [].
3. For 7 days, append entries sales to list using append()
4. Find max_val = max ; min_val = min
5. Stop.

program :

```
# list scenario.
days = 7
sales = []
Target = 500 # target sales for the day
for s in range(s):
    sample_entries = int(input("Enter the seven days sales count "))
    sales.append(sample_entries) # list.append()

total = sum(sales)
avg = total/days
max_val = max
min_val = min(sales)
best_day = sales.index(max_val) + 1 # list.index()
worst_day = sales.index(min_val) + 1
print("sales (Mon..Sun):", sales)
Print("Total:", total)
Print("Avage", round(avg, 2))
print("Best Day.", best_day, "with", max_val)
print("Worst Day.", worst_day, "with", min_val)
```

output :-

Participant ID : A1
participant ID : A2
Participant ID : A4
" ID : A5

Robotics challenge : 4

" ID = A1
" = A2
" = A3
" AD
"

A1 Hackathon : {'A1', A4, A5, A2}
Robotics challenge {'A2', A1, A3}

Both Events : {'A2'}

only A1 : {'A4', A3, A5, A1}

only Robotics {'A1', A3}

unique participants :

4.2 Tuple - Lab Timetable

Aim: To manage & query an immutable daily lab slot schedule using a tuple, demonstrating membership check, count(), index() & slicing

Algorithm:

1. Start
2. Define slots as a fixed tuple of integers
3. Read query hour.
4. Slice into morning & afternoon
5. Stop.

Program:

```
# TUPLE scenario
slots = (9, 11, 14, 16, 14)  # immutable daily schedule

query = 14
exists = (query in slots)
freq = slots.count(query)  # tuple.count()
first_pos = slots.index(query)+1 if exists else "N/A"

morning = slots[:2]
afternoon = slots[2:]
print("All lab slots:", slots)
print(f"{query}:00 present?", exists)
print("Morning slots:", morning)
print("Afternoon slots:", afternoon)
```

4.3 Dictionary - Bookstore Billing

Aim: bookstore's price list is stored in a dictionary. where keys are item names and values are prices.

Algorithm:

1. Start
2. Create an empty dictionary prices.
3. Ask the user for the number of items in price list (n)
4. Repeat for each item:
5. Get the item name
6. Get the items price.

7. Add the item & price to prices.
8. Ask the user to enter an item, get the new price
9. Stop.

Program:

```
prices = {}
n1 = int(input("Enter number of items in price list:"))
for _ in range(n1):
    item = input("Enter item name:")
    price = float(input(f"Enter price of {item}:"))
    prices[item] = price

# optional price revision.
rev_item = input("Enter the item to update price (or press
Enter to skip):")
if rev_item in prices:
    new_price = float(input(f"Enter new price for {rev_item}:"))
prices.update({rev_item: new_price}) # dict.update()
# Find costliest item
costliest_item = None
  max_price = 0
for item, price in prices.items():
  if price > max_price:
    max_price = price
    costliest_item = item
# remove out-of-stock item
remove_item = input("Enter an item to remove from
price list (or press Enter to skip):")
# display results
print("Available items:", list(prices.keys())) # dict.keys()
print("price:", list(prices.values())) # dict.values()
if costliest_item:
    print("costliest item:", costliest_item, "at", max_price)
```

44. Set - Tech Fest Participation.

Aim: Two events, AI Hackathon & Robotics challenge, have participants' IDs stored in two sets.

Get AI Hackathon participants

```
ai_hackathon = set()
n1 = int(input("Enter number of participants in AI Hackathon"))
for _ in range(n1):
    pid = input("Enter participant ID: ")
    ai_hackathon.add(pid)
```

Get Robotics Challenge participants

```
robotics_challenge = set()
n2 = int(input("Enter number of participants in Robotics challenge: "))
for _ in range(n2):
    pid = input("Enter participant ID: ")
    robotics_challenge.add(pid)
```

add a late registrant

```
late_id = input("Enter withdrawn participant ID from Robotics challenge: ")
if remove_id:
```

set operations

```
both = ai_hackathon.intersection
only_ai = ai_hackathon.difference
only_robotics = robotics_challenge
unique_all = ai_hackathon.union
```

VEL TECH	
EX NO.	4
PERFORMANCE (5)	5
RESULT AND ANALYSIS (5)	5
VIVA VOCE (5)	5
RECORD (5)	
TOTAL (20)	
SIGN WITH DATE	15

Result: Thus, implementing & use various datatypes, list, tuples & dictionary in Python programming has executed successfully.