Part-A Questions

May 28, 2025

[6]:
```
'''
    1. Write a python program to create a list with all the subject names of␣
 ↪the 4th semester and perform the following operations.
    • Display the list using for loop.
    • Display 2nd and 5th element of the list.
    • Display first 4 elements of the list using the range of indexes.
    • Display last 4 elements of the list using the range of negative indexes.
    • Display if "Python Programming Lab" is available in the List or not.
    • Demonstrate the working of append() and insert() function.
    • Demonstrate the working of remove() and pop() function.
    • Demonstrate the working of extend() function by adding subject codes.
    • Display the list in ascending and descending order.
'''

# Step 1: Create a list of subject names for the 4th semester
subjects = [
    "OS",
    "DBMS",
    "Python Lab",
    "Web Lab",
    "DCN",
    "DBMS Lab",
    "OS Lab"
]

# Step 2: Display the list using a for loop
print("Subject List:")
for subject in subjects:
    print(subject)

# Step 3: Display 2nd and 5th element of the list
print("\n2nd Element:", subjects[1])  # DBMS
print("5th Element:", subjects[4])    # DCN

# Step 4: Display first 4 elements of the list using the range of indexes
print("\nFirst 4 Elements:", subjects[0:4])  # OS, DBMS, Python Lab, Web Lab
```

```python
# Step 5: Display last 4 elements of the list using the range of negative␣
 ↪indexes
print("Last 4 Elements:", subjects[-4:])  # Python Lab, Web Lab, DCN, DBMS Lab

# Step 6: Check if "Python Lab" is available in the list
is_python_lab_available = "Python Lab" in subjects
print("\nIs 'Python Lab' available in the list?", is_python_lab_available)

# Step 7: Demonstrate the working of append() and insert() function
subjects.append("Machine Learning")  # Append a new subject
print("\nList after appending 'Machine Learning':", subjects)

subjects.insert(2, "Cloud Computing")  # Insert a new subject at index 2
print("List after inserting 'Cloud Computing' at index 2:", subjects)

# Step 8: Demonstrate the working of remove() and pop() function
subjects.remove("Web Lab")  # Remove a subject
print("\nList after removing 'Web Lab':", subjects)

popped_subject = subjects.pop()  # Pop the last subject
print("Popped Subject:", popped_subject)
print("List after popping the last subject:", subjects)

# Step 9: Demonstrate the working of extend() function by adding subject codes
subject_codes = ["CS101", "CS102", "CS103", "CS104"]
subjects.extend(subject_codes)  # Extend the list with subject codes
print("\nList after extending with subject codes:", subjects)

# Step 10: Display the list in ascending and descending order
print("\nList in Ascending Order:", sorted(subjects))
print("List in Descending Order:", sorted(subjects, reverse=True))
```

```
Subject List:
OS
DBMS
Python Lab
Web Lab
DCN
DBMS Lab
OS Lab

2nd Element: DBMS
5th Element: DCN

First 4 Elements: ['OS', 'DBMS', 'Python Lab', 'Web Lab']
Last 4 Elements: ['Web Lab', 'DCN', 'DBMS Lab', 'OS Lab']

Is 'Python Lab' available in the list? True
```

```
List after appending 'Machine Learning': ['OS', 'DBMS', 'Python Lab', 'Web Lab',
'DCN', 'DBMS Lab', 'OS Lab', 'Machine Learning']
List after inserting 'Cloud Computing' at index 2: ['OS', 'DBMS', 'Cloud
Computing', 'Python Lab', 'Web Lab', 'DCN', 'DBMS Lab', 'OS Lab', 'Machine
Learning']

List after removing 'Web Lab': ['OS', 'DBMS', 'Cloud Computing', 'Python Lab',
'DCN', 'DBMS Lab', 'OS Lab', 'Machine Learning']
Popped Subject: Machine Learning
List after popping the last subject: ['OS', 'DBMS', 'Cloud Computing', 'Python
Lab', 'DCN', 'DBMS Lab', 'OS Lab']

List after extending with subject codes: ['OS', 'DBMS', 'Cloud Computing',
'Python Lab', 'DCN', 'DBMS Lab', 'OS Lab', 'CS101', 'CS102', 'CS103', 'CS104']

List in Ascending Order: ['CS101', 'CS102', 'CS103', 'CS104', 'Cloud Computing',
'DBMS', 'DBMS Lab', 'DCN', 'OS', 'OS Lab', 'Python Lab']
List in Descending Order: ['Python Lab', 'OS Lab', 'OS', 'DCN', 'DBMS Lab',
'DBMS', 'Cloud Computing', 'CS104', 'CS103', 'CS102', 'CS101']
```

[2]:

```
'''
2. Write a python program to initialize a dictionary of usernames and passwords␣
 ↪associated with it. passwd={'rahul': 'genius', 'kumar': 'smart', 'ankita':␣
 ↪'intelligent'} Define the following functions:
a) To print all the items in the dictionary.
b) To print all the keys in the dictionary.
c) To print all the values in the dictionary.
d) To get the passwords of users. For example, passwd['rahul']= genius
e) Change the password of a particular user. For example,␣
 ↪passwd['ankita']='brilliant'
'''

passwd = {'rahul': 'genius', 'kumar': 'smart', 'ankita': 'intelligent'}

def print_items():
    print("All items:", passwd.items())

def print_keys():
    print("All keys:", passwd.keys())

def print_values():
    print("All values:", passwd.values())

def get_password(username):
    if username in passwd:
        print(f"passwd['{username}'] = {passwd[username]}")
```

```
    else:
        print(f"User '{username}' not found.")

def change_password(username, new_password):
    if username in passwd:
        passwd[username] = new_password
        print(f"Password for '{username}' changed to '{new_password}'")
    else:
        print(f"User '{username}' not found.")

print_items()
print_keys()
print_values()
get_password('rahul')
change_password('ankita', 'brilliant')
print_items()
```

```
All items: dict_items([('rahul', 'genius'), ('kumar', 'smart'), ('ankita',
'intelligent')])
All keys: dict_keys(['rahul', 'kumar', 'ankita'])
All values: dict_values(['genius', 'smart', 'intelligent'])
passwd['rahul'] = genius
Password for 'ankita' changed to 'brilliant'
All items: dict_items([('rahul', 'genius'), ('kumar', 'smart'), ('ankita',
'brilliant')])
```

[9]:
```
'''
3. Develop a python program to match all the occurrences of a word which has an␣
 ↪"A/a" followed by
i. Zero or more b's
ii. Zero or one b
iii. One and more b's
iv. Four consecutive b's followed by a "c".
'''

import re

text = "A ab abb abbb abbbb abbbbb abc abbbc Ab Abbb a abbbb abc"

pattern1 = r"[Aa]b*"
pattern2 = r"[Aa]b?"
pattern3 = r"[Aa]b+"
pattern4 = r"[Aa]bbbbc"

matches1 = re.findall(pattern1, text)
matches2 = re.findall(pattern2, text)
matches3 = re.findall(pattern3, text)
```

```
matches4 = re.findall(pattern4, text)

print("i. A/a followed by zero or more b's:", matches1)
print("ii. A/a followed by zero or one b:", matches2)
print("iii. A/a followed by one or more b's:", matches3)
print("iv. A/a followed by four b's and a c:", matches4)
```

```
i. A/a followed by zero or more b's: ['A', 'ab', 'abb', 'abbb', 'abbbb',
'abbbbb', 'ab', 'abbb', 'Ab', 'Abbb', 'a', 'abbbb', 'ab']
ii. A/a followed by zero or one b: ['A', 'ab', 'ab', 'ab', 'ab', 'ab', 'ab',
'ab', 'Ab', 'Ab', 'a', 'ab', 'ab']
iii. A/a followed by one or more b's: ['ab', 'abb', 'abbb', 'abbbb', 'abbbbb',
'ab', 'abbb', 'Ab', 'Abbb', 'abbbb', 'ab']
iv. A/a followed by four b's and a c: []
```

[1]:
```
'''
4. Develop a python program to create a text file and ask the user to enter 5-6␣
 ↪lines of text.
Count the number of upper case, lower case and digits in the file. Display the␣
 ↪details of the file.
'''
filename = "user_text.txt"

with open(filename, "w") as file:
    print("Enter 5-6 lines of text:")
    for _ in range(6):
        line = input()
        file.write(line + "\n")

with open(filename, "r") as file:
    content = file.read()

upper_count = sum(1 for c in content if c.isupper())
lower_count = sum(1 for c in content if c.islower())
digit_count = sum(1 for c in content if c.isdigit())

print("\nFile Details:")
print("Uppercase letters:", upper_count)
print("Lowercase letters:", lower_count)
print("Digits:", digit_count)
```

```
Enter 5-6 lines of text:
Hello
Testing
Testing
123
234
yaeh
```

```
File Details:
Uppercase letters: 3
Lowercase letters: 20
Digits: 6
```

[3]:
```
'''
5. Given daily temperatures for 2 weeks (14 days), write a Python program using␣
 ↪NumPy to:
    • Convert the 1D temperature list into a 2D array (2 weeks × 7 days),
    • Slice and display weekdays (Mon-Fri) of Week 2,
    • Sort the full data and each week's temperatures,
    • Find max and min temperatures: Per week, Per day (across weeks), Overall.
'''

import numpy as np

temps = [30, 32, 31, 29, 28, 27, 26, 33, 35, 34, 32, 30, 31, 29]

temp_array = np.array(temps).reshape(2, 7)

print("2D Temperature Array (2 weeks × 7 days):\n", temp_array)

week2_weekdays = temp_array[1, 0:5]
print("\nWeek 2 Weekdays (Mon-Fri):", week2_weekdays)

sorted_all = np.sort(temp_array, axis=None)
print("\nSorted Full Data:", sorted_all)

sorted_per_week = np.sort(temp_array, axis=1)
print("\nSorted Temperatures Per Week:\n", sorted_per_week)

max_per_week = np.max(temp_array, axis=1)
min_per_week = np.min(temp_array, axis=1)
print("\nMax per Week:", max_per_week)
print("Min per Week:", min_per_week)

max_per_day = np.max(temp_array, axis=0)
min_per_day = np.min(temp_array, axis=0)
print("\nMax per Day (across weeks):", max_per_day)
print("Min per Day (across weeks):", min_per_day)

overall_max = np.max(temp_array)
overall_min = np.min(temp_array)
print("\nOverall Max Temperature:", overall_max)
print("Overall Min Temperature:", overall_min)
```

```
2D Temperature Array (2 weeks × 7 days):
```

```
 [[30 32 31 29 28 27 26]
 [33 35 34 32 30 31 29]]

Week 2 Weekdays (Mon-Fri): [33 35 34 32 30]

Sorted Full Data: [26 27 28 29 29 30 30 31 31 32 32 33 34 35]

Sorted Temperatures Per Week:
 [[26 27 28 29 30 31 32]
 [29 30 31 32 33 34 35]]

Max per Week: [32 35]
Min per Week: [26 29]

Max per Day (across weeks): [33 35 34 32 30 31 29]
Min per Day (across weeks): [30 32 31 29 28 27 26]

Overall Max Temperature: 35
Overall Min Temperature: 26
```

[5]:

```
'''
6. Develop a Python program using pandas to perform data analysis on a CSV file␣
 ↪containing student records. The program should:
    • Load the data
    • Display statistics
    • Filter based on marks
    • Group and average by department
    • Sort and rank students by GPA
    • Handle missing data
    • Add a derived "Grade" column
    • Visualize average marks by department
'''

import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv("students.csv")

print("\nBasic Statistics:\n", df.describe())

print("\nStudents with Marks > 75:\n", df[df["Marks"] > 75])

grouped = df.groupby("Department")["Marks"].mean()
print("\nAverage Marks by Department:\n", grouped)

sorted_df = df.sort_values(by="GPA", ascending=False)
sorted_df["Rank"] = sorted_df["GPA"].rank(ascending=False)
```

```
print("\nSorted & Ranked by GPA:\n", sorted_df[["Name", "GPA", "Rank"]])

df.fillna({"Marks": df["Marks"].mean(), "GPA": df["GPA"].mean()}, inplace=True)

def get_grade(marks):
    if marks >= 90:
        return 'A'
    elif marks >= 75:
        return 'B'
    elif marks >= 60:
        return 'C'
    elif marks >= 50:
        return 'D'
    else:
        return 'F'

df["Grade"] = df["Marks"].apply(get_grade)
print("\nWith Grade Column:\n", df[["Name", "Marks", "Grade"]])

grouped.plot(kind="bar", title="Average Marks by Department")
plt.ylabel("Average Marks")
plt.xlabel("Department")
plt.tight_layout()
plt.show()


'''
students.csv file contents:
StudentID,Name,Department,GPA,Marks
101,Alice,CSE,8.5,88
102,Bob,ECE,7.8,76
103,Charlie,ME,6.2,65
104,Diana,CSE,9.1,92
105,Evan,ECE,7.0,
106,Fay,ME,5.4,55
107,George,CSE,,83
108,Helen,ECE,6.5,70
'''
```

```
Basic Statistics:
        StudentID       GPA      Marks
count    8.00000  7.000000   7.000000
mean   104.50000  7.214286  75.571429
std      2.44949  1.318368  13.201731
min    101.00000  5.400000  55.000000
25%    102.75000  6.350000  67.500000
50%    104.50000  7.000000  76.000000
```

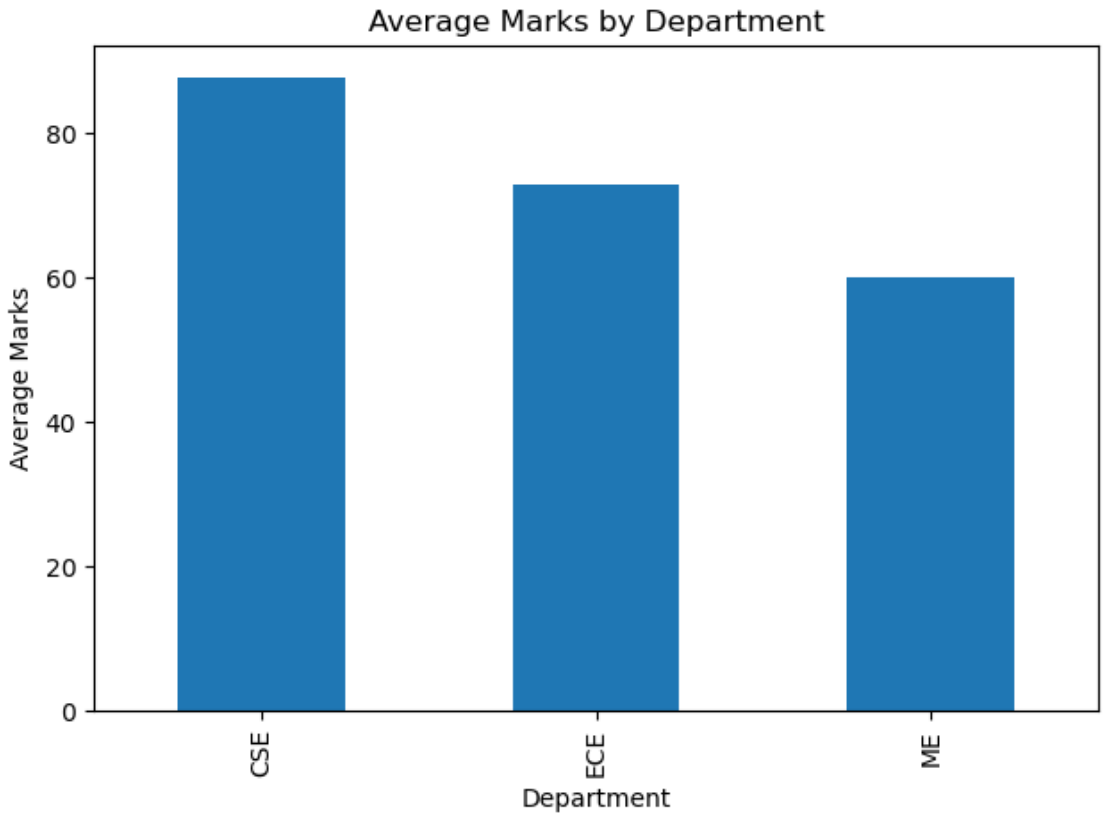
```
75%    106.25000  8.150000  85.500000
max    108.00000  9.100000  92.000000

Students with Marks > 75:
    StudentID    Name Department  GPA  Marks
0         101   Alice        CSE  8.5   88.0
1         102     Bob        ECE  7.8   76.0
3         104   Diana        CSE  9.1   92.0
6         107  George        CSE  NaN   83.0

Average Marks by Department:
 Department
CSE    87.666667
ECE    73.000000
ME     60.000000
Name: Marks, dtype: float64

Sorted & Ranked by GPA:
       Name  GPA  Rank
3    Diana  9.1   1.0
0    Alice  8.5   2.0
1      Bob  7.8   3.0
4     Evan  7.0   4.0
7    Helen  6.5   5.0
2  Charlie  6.2   6.0
5      Fay  5.4   7.0
6   George  NaN   NaN

With Grade Column:
       Name      Marks Grade
0    Alice  88.000000     B
1      Bob  76.000000     B
2  Charlie  65.000000     C
3    Diana  92.000000     A
4     Evan  75.571429     B
5      Fay  55.000000     D
6   George  83.000000     B
7    Helen  70.000000     C
```



[5]: '\nstudents.csv file contents:\nStudentID,Name,Department,GPA,Marks\n101,Alice,CSE,8.5,88\n102,Bob,ECE,7.8,76\n103,Charlie,ME,6.2,65\n104,Diana,CSE,9.1,92\n105,Evan,ECE,7.0,\n106,Fay,ME,5.4,55\n107,George,CSE,,83\n108,Helen,ECE,6.5,70\n'