PES UNIVERSITY, BENGALURU

Perseverance | Excellence | Service

UE25CS151A – PYTHON FOR COMPUTATIONAL PROBLEM SOLVING LAB MANUAL

WEEK 6

TOPICS:

Programs on Sets,Dictionaries and Strings

OBEJCTIVE:

Solve practical problems using set, dictionary and string to manage collections of data.

Problem Statement 1: (set)

In a school event, two groups of students participate in different activities.

- List1 represents students (by ID numbers) in the **music club**, and
- List2 represents students (by ID numbers) in the **sports club.**
- Some students participate in **both**.

Write a Python program to convert these lists to sets and perform the following operations:

- Find all students participating in at least one club.
- Find students participating in both clubs.
- Find students only in the music club.
- Find students who are in exactly one club.

Display the results clearly, showing the original lists and the results of each operation.

Solution:

```
list1 = [1, 3, 5, 4, 7, 2, 9]
list2 = [3, 4, 5, 6, 7]
set1 = set(list1)
set2 = set(list2)
union_set = set1 | set2
intersection_set = set1 & set2
difference_set = set1 - set2
symmetric_diff = set1 ^ set2
```

```
print("Music club student IDs:", list1)
print("Sports club student IDs:", list2)
print("All students in at least one club:", union_set)
print("Students in both clubs:", intersection_set)
print("Students only in music club:", difference_set)
print("Students in exactly one club:", symmetric_diff)
```

Sample Output:

Music club student IDs: [1, 3, 5, 4, 7, 2, 9]
Sports club student IDs: [3, 4, 5, 6, 7]
All students in at least one club: {1, 2, 3, 4, 5, 6, 7, 9}
Students in both clubs: {3, 4, 5, 7}
Students only in music club: {1, 2, 9}
Students in exactly one club: {1, 2, 6, 9}

Problem Statement 2: (set)

Write a Python program to count how many stones are jewels. You are given a string 'jewels' representing the types of stones that are jewels, and a string ‘stones’ representing the stones you have. Each character in 'stones' is a type of stone, and you need to count how many of these are present in 'jewels'. Take the strings as user input.

Sample output:

Jewels: aAb
Stones: aAAbcccadddaA
Number of stones that are jewels: 7

Solution:

```
jewels = "aAb"
stones = "aAAbcccadddaA"
jewel_set = set(jewels)
count = 0
for stone in stones:
    if stone in jewel_set:
        count += 1
print("Jewels:", jewels)
print("Stones:", stones)
print("Number of stones that are jewels:", count)
```

Problem Statement 3: (Dictionary)

Write a Python program to store student names and their grades in a dictionary. Compute the average grade and identify students with grades above the average.

Sample output:

Student grades: {'Ram': 85, 'Sita': 90, 'Akbar': 78, 'Antony': 92}
Average grade: 86.25
Students above average: {'Sita': 90, 'Antony': 92}

Solution:

```
students = {"Ram": 85, "Sita": 90, "Akbar": 78, "Antony": 92}
total = 0
for grade in students.values():
    total += grade
average = total / len(students)
above_average = {}
for name, grade in students.items():
    if grade > average:
        above_average[name] = grade
print("Student grades:", students)
print("Average grade:", average)
print("Students above average:", above_average)
```

Problem Statement 4: (Dictionary)

Python Program for Detecting Duplicates with Counts and Non-Duplicates

In a voting system, each number in a list represents a candidate ID voted by a student.

- Write a Python program to analyse the votes in the given list.
- Determine if there are any duplicate votes (same candidate ID appearing more than once), create a dictionary showing the candidate IDs that received multiple votes along with their vote counts, and
- Identify the candidate IDs that received exactly one vote, storing them in a set.
- Display the original list, whether duplicates exist, the dictionary of duplicated candidate IDs with their counts, and the set of non-duplicated candidate IDs.

Sample output:

Vote list (candidate IDs): [1, 2, 3, 1, 1, 2, 3, 4, 5, 6, 7]
Contains duplicate votes? True
Duplicated candidate IDs with vote counts: {1: 3, 2: 2, 3: 2}
Non-duplicated candidate IDs: {4, 5, 6, 7}

Solution:

```
nums = [1, 2, 3, 1, 1, 2, 3, 4, 5, 6, 7]
num_dict = {}
has_duplicate = False
for num in nums:
    if num in num_dict:
        num_dict[num] += 1
        has_duplicate = True
    else:
        num_dict[num] = 1
duplicate_dict = {}
non_duplicates = set()
for num, count in num_dict.items():
    if count > 1:
        duplicate_dict[num] = count
    else:
        non_duplicates.add(num)
print("Vote list (candidate IDs):", nums)
print("Contains duplicate votes?", has_duplicate)
print("Duplicated candidate IDs with vote counts:", duplicate_dict)
print("Non-duplicated candidate IDs:", non_duplicates)
```

Problem Statement 5: (Strings)

Write a python program to count the number of vowels and consonants in the string.

Sample output:

String: Python Programming
Vowel count: 4
Consonant count: 13

Solution:

```
input_str = "Python Programming"
vowels = "aeiouAEIOU"
vowel_count = 0
consonant_count = 0
for char in input_str:
   if char.isalpha():
      if char in vowels:
         vowel_count += 1
      else:
         consonant_count += 1
print("String:", input_str)
print("Vowel count:", vowel_count)
print("Consonant count:", consonant_count)
```

Problem Statement 6: (Strings)

Write a Python program that takes an input string, replaces every occurrence of the letter 'o' with the digit '0', and then converts the entire modified string to uppercase.

Sample Output:
Original string: Hello world, how are you today?
Output: HELL0 W0RLD, H0W ARE Y0U T0DAY?

Solution:

```
# An empty string to build the result
modified_str = ""

# Iterate through each character of the input string
for char in input_str:
  # If the character is 'o', append '0' to the new string
  if char.lower() == 'o':
    modified_str += '0'
  # Otherwise, append the original character
  else:
    modified_str += char

# Capitalize the entire modified string
capitalized = modified_str.upper()
```

```
# Print the results
print("Original string:", input_str)
print("After replacing 'o' with '0' and capitalizing:", capitalized)
```

If the implementation is hard to explain, it's a bad idea – Make it simple