Tuple

Q1: Join Tuples if similar initial element

While working with Python tuples, we can have a problem in which we need to perform concatenation of records from the similarity of initial element. This problem can have applications in data domains such as Data Science.

For eg.

```
Input  : test_list = [(5, 6), (5, 7), (5, 8), (6, 10), (7, 13)]
Output : [(5, 6, 7, 8), (6, 10), (7, 13)]
```

In [ ]:

```
# write your code here
```

Q2: Multiply Adjacent elements (both side) and take sum of right and lest side multiplication result.

For eg.

```
The original tuple : (1, 5, 7, 8, 10)
Resultant tuple after multiplication :

(1*5, 1*5+5*7, 7*5 + 7*8, 8*7 + 8*10, 10*8) -> (5, 40, 91, 136, 80)

output-(5, 40, 91, 136, 80)
```

In [ ]:

```
# write your code here
```

Q3: Check is tuples are same or not?

Two tuples would be same if both tuples have same element at same index

```
t1 = (1,2,3,0)
t2 = (0,1,2,3)

t1 and t2 are not same
```

In [ ]:

```
# write your code here
```

Q4: Count no of tuples, list and set from a list

```
list1 = [{'hi', 'bye'},{'Geeks', 'forGeeks'},('a', 'b'),['hi', 'bye'],['a', 'b']]
```

Output:

```
List-2
Set-2
Tuples-1
```

```
In [ ]:
```

```
# write your code here
```

Q5 : Shortlist Students for a Job role

Ask user to input students record and store in tuples for each record. Then Ask user to input three things he wants in the candidate- Primary Skill, Higher Education, Year of Graduation.

Show every students record in form of tuples if matches all required criteria.

It is assumed that there will be only one primry skill.

If no such candidate found, print `No such candidate`

```
Input:
```

```
   Enter No of records- 2
   Enter Details of student-1
   Enter Student name- Manohar
   Enter Higher Education- B.Tech
   Enter Primary Skill- Python
   Enter Year of Graduation- 2022
   Enter Details of student-2
   Enter Student name- Ponian
   Enter Higher Education- B.Sc.
   Enter Primary Skill- C++
   Enter Year of Graduation- 2020

   Enter Job Role Requirement
   Enter Skill- Python
   Enter Higher Education- B.Tech
   Enter Year of Graduation- 2022
```

```
Output
```

```
   ('Manohar', 'B.tech', 'Python', '2022')
```

```
In [ ]:
```

```
# write your code here
```

Set

Q1 : Write a program to find set of common elements in three lists using sets.

```
   Input : ar1 = [1, 5, 10, 20, 40, 80]
           ar2 = [6, 7, 20, 80, 100]
           ar3 = [3, 4, 15, 20, 30, 70, 80, 120]

   Output : [80, 20]
```

```
In [ ]:
```

```
# write your code here
```

Q2 : Write a program to count unique number of vowels using sets in a given string. Lowercase and upercase vowels will be taken as different.

```
Input:
```

```
Str1 = "hands-on data science mentorship progrAm with live classes at affordable fe
e only on CampusX"
```

Output:

```
No of unique vowels-6
```

In [ ]:

```
# write your code here
```

Q3: Write a program to Check if a given string is binary string of or not.

A string is said to be binary if it's consists of only two unique characters.

Take string input from user.

```
Input: str = "01010101010"
Output: Yes

Input: str = "1222211"
Output: Yes

Input: str = "Campusx"
Output: No
```

In [ ]:

```
# write your code here
```

Q4: find union of n arrays.

Example 1:

Input:

```
[[1, 2, 2, 4, 3, 6],
 [5, 1, 3, 4],
 [9, 5, 7, 1],
 [2, 4, 1, 3]]
```

Output:

```
[1, 2, 3, 4, 5, 6, 7, 9]
```

In [ ]:

```
# write your code here
```

Q5: Intersection of two lists. Intersection of two list means we need to take all those elements which are common to both of the initial lists and store them into another list. Only use using list-comprehension.

Example 1:

Input:

```
lst1 = {15, 9, 10, 56, 23, 78, 5, 4, 9}
lst2 = {9, 4, 5, 36, 47, 26, 10, 45, 87}
```

Output:
```

```
[9, 10, 4, 5]
```

Example 2:

Input:

```
lst1 = {4, 9, 1, 17, 11, 26, 28, 54, 69}
lst2 = {9, 9, 74, 21, 45, 11, 63, 28, 26}
```

Output:

```
[9, 11, 26, 28]
```

In [ ]:

```
# write your code here
```

Dictionary

Q1 : Key with maximum unique values

Given a dictionary with values list, extract key whose value has most unique values.

Example 1:

Input:

```
test_dict = {"CampusX" : [5, 7, 9, 4, 0], "is" : [6, 7, 4, 3, 3], "Best" : [9, 9,
6, 5, 5]}
```

Output:

```
CampusX
```

Example 2:

Input:

```
test_dict = {"CampusX" : [5, 7, 7, 7, 7], "is" : [6, 7, 7, 7], "Best" : [9, 9, 6,
5, 5]}
```

Output:

```
Best
```

In [ ]:

```
# write your code here
```

Q2 : Replace words from Dictionary. Given String, replace it's words from lookup dictionary.

Example 1:

Input:

```
test_str = 'CampusX best for DS students.'
repl_dict = {"best" : "is the best channel", "DS" : "Data-Science"}
```

Output:

```
CampusX is the best channel for Data-Science students.
```

Example 2:

Input:

```
test_str = 'CampusX best for DS students.'
repl_dict = {"good" : "is the best channel", "ds" : "Data-Science"}
```

Output:

```
CampusX best for DS students.
```

In [ ]:

```
# write your code here
```

Q3 : Convert List to List of dictionaries. Given list values and keys list, convert these values to key value pairs in form of list of dictionaries.

Example 1:

Input:

```
test_list = ["DataScience", 3, "is", 8]
key_list = ["name", "id"]
```

Output:

```
[{'name': 'DataScience', 'id': 3}, {'name': 'is', 'id': 8}]
```

Example 2:

Input:

```
test_list = ["CampusX", 10]
key_list = ["name", "id"]
```

Output:

```
[{'name': 'CampusX', 'id': 10}]
```

In [ ]:

```
# write your code here
```

Q4 : Convert a list of Tuples into Dictionary.

Example 1:

Input:

```
[("akash", 10), ("gaurav", 12), ("anand", 14), ("suraj", 20), ("akhil", 25), ("as
hish", 30)]
```

Output:

```
{'akash': [10], 'gaurav': [12], 'anand': [14], 'suraj': [20], 'akhil': [25], 'ash
ish': [30]}
```

Example 2:

Input:

```
[('A', 1), ('B', 2), ('C', 3)]
```

Output:

```
{'A': [1], 'B': [2], 'C': [3]}
```

```
{'A': [1], 'B': [2], 'C': [3]}
```

In [ ]:

```
# write your code here
```

Q5 : Sort Dictionary key and values List.

Example 1:

Input:

```
{'c': [3], 'b': [12, 10], 'a': [19, 4]}
```

Output:

```
{'a': [4, 19], 'b': [10, 12], 'c': [3]}
```

Example 2:

Input:

```
{'c': [10, 34, 3]}
```

Output:

```
{'c': [3, 10, 34]}
```

In [ ]:

```
# write your code here
```