CS 171

Lab Assignment 10

This lab assignment uses many elements provided in the main bibliographic reference for these lectures:

Programming in Python 3

A Complete Introduction to the Python Language,

2nd Edition,

Mark Summerfield

**Exercises**

**Exercise 1 Pattern Matching**

Given a string “str” and another string “sub\_str”. We are allowed to delete “sub\_str” from “str” any number of times. It is also given that the “sub\_str” appears only once at a time. The task is to find if “str” can become empty by removing “sub\_str” again and again.

Examples:

Input: str = "GEEGEEKSKS", sub\_str = "GEEKS"

Output: Yes

*Explanation: In the string GEEGEEKSKS, we can first delete the substring GEEKS from position The new string now becomes GEEKS. We can again delete sub-string GEEKS from position 1. Now the string becomes empty.*

Input: str = "GEEGEEKSSGEK", sub\_str = "GEEKS"

Output: No

*Explanation: In the string it is not possible to make the string empty in any possible manner.*

**Exercise 2 Sort List**

Given two lists, sort the values of one list using the second list.

Examples:

Input: list1 = ["a", "b", "c", "d", "e", "f", "g", "h", "i"]

list2 = [ 0, 1, 1, 0, 1, 2, 2, 0, 1]

Output: ['a', 'd', 'h', 'b', 'c', 'e', 'i', 'f', 'g']

Input: list1 = ["g", "e", "e", "k", "s", "f", "o", "r", "g", "e", "e", "k", "s"]

list2 = [ 0, 1, 1, 0, 1, 2, 2, 0, 1]

Output: ['g', 'k', 'r', 'e', 'e', 'g', 's', 'f', 'o']

**Exercise 3. Generate words**

Given a dictionary and a character array, print all valid words that are possible using characters from the array.

Note: Repetitions of characters is not allowed.

Examples:

Input: dict = ["go", "bat", "me", "eat", "goal", "boy", "run"]

arr = ['e', 'o', 'b', 'a', 'm', 'g', 'l']

Output: [‘go’, ‘me’, ‘goal’]

**Exercise 4. Reform Dictionary**

Sometimes, while working with Python dictionaries, we can have problem in which we need to reform the dictionary, in the form in which all the values point to the keys that they belong to. This kind of problem can occur in many domains including web development and data domains. Lets discuss certain way in which this task can be performed.

Examples:

Input: test\_dict = {‘abc’ : [10, 30], ‘bcd’ : [30, 40, 10]}

Output: {10 : [‘abc’, ‘bcd’], 30 : [‘abc’, ‘bcd’], 40 : [‘bcd’]}

Input : test\_dict = {‘gfg’ : [1, 2, 3], ‘is’ : [1, 4], ‘best’ : [4, 2]}

Output: {1 : [‘is’, ‘gfg’], 2 : [‘gfg’, ‘best’], 3 : [‘gfg’], 4 : [‘is’, ‘best’]}

**Exercise 5 Modular Equations**

Given A and B, the task is to find the number of possible values that X can take such that the given modular equation (A mod X) = B holds good. Here, X is also called a solution of the modular equation.

Examples:

Input : A = 26, B = 2

Output : 6

Explanation

X can be equal to any of {3, 4, 6, 8, 12, 24} as A modulus any of these values

equals 2 i. e., (26 mod 3) = (26 mod 4) = (26 mod 6) = (26 mod 8) =(26 mod 12) = (26 mod 24)=2

Input : 21 5

Output : 2

Explanation

*X can be equal to any of {8, 16} as A modulus any of these values equals 5 i.e. (21 mod 8) = (21 mod 16) = 5*