



Inspiring Excellence

**Course Title: Programming Language II**

**Course Code: CSE 111**

**Lab Assignment no: 1**

## List

1. Write a python program which prints the frequency of the numbers that were given as input by the user. Stop taking input when you find the string “STOP”. **Do not print the frequency of numbers that were not given as input.**

### **Sample Input**

10  
20  
20  
30  
10  
50  
90  
STOP

### **Sample Output**

10 - 2 times  
20 - 2 times  
30 - 1 times  
50 - 1 times  
90 - 1 times

2. Write a python program that calculates the sum of N given lists and prints the highest sum and its respective list. Input starts with N and followed by N lists.

### **Sample Input**

4  
1 2 3  
4 5 6  
10 11 12  
7 8 9

### **Sample Output**

33  
[10, 11, 12]

3. Write a python program that prints a list which contains cross multiplication between two given lists.

**Sample Input**

2 3 6  
3 4 5

**Sample Output**

[6, 8, 10, 9, 12, 15, 18, 24, 30]

4. Let there are N numbers in a list and that list is said to be a UB Jumper if the absolute values of the difference between the successive elements take on all the values 1 through N – 1. For example, 2 1 4 6 10 is a UB Jumper because the absolute differences between them are 1 3 2 4 which is all numbers from 1 to (5 - 1) or 4. **Write a python program that takes a number sequence as input and prints whether it is a UB Jumper or Not UB Jumper. Input will stop after getting “STOP” as input.** (Number order or absolute difference order doesn't follow any sequence.)

**Sample Input**

1 4 2 3  
2 1 4 6 10  
1 4 2 -1 6  
STOP

**Sample Output**

UB Jumper  
UB Jumper  
Not UB Jumper

5. You are given a string that contains alphanumeric characters only. Your task is to sort the string in the following manner:
- All sorted lowercase letters are ahead of uppercase letters.
  - All sorted uppercase letters are ahead of digits.
  - All sorted odd digits are ahead of sorted even digits.

**Sample Input**

Bracu1234

**Sample Output**

acruB1324

6. BRACU has  $n$  students who are regular competitive programmers. According to the ACM ICPC rules, each person can participate in the regional championship at most 5 times.

The head of the BRACU ACM Chapter is recently gathering teams to participate in this championship. Each team must consist of exactly three people, at that, any person cannot be a member of two or more teams. What maximum number of teams can the head make if he wants each team to participate in the world championship with the same members at least  $k$  times?

The first line of input contains two integers,  $n$  and  $k$ . The next line contains  $n$  integers:  $y_1, y_2, \dots, y_n$  ( $0 \leq y_i \leq 5$ ), where  $y_i$  shows the number of times the  $i$ th person participated in the ACM ICPC Regional.

**Write a python program that prints how many teams can be formed according to the above problem statement.**

Sample Input 1

```
5 2
0 4 5 1 0
```

Sample Input 2

```
6 4
0 1 2 3 4 5
```

Sample Input 3

```
6 5
0 0 0 0 0 0
```

Sample Output 1

1

Sample Output 2

0

Sample Output 3

2

## Dictionary & Tuple

1. Write a Python program to combine two dictionaries into one by adding values for common keys. Input contains two comma separated dictionaries. Print the new dictionary and create a **tuple** which contains unique values in sorted order.

### **Sample Input**

a: 100, b: 100, c: 200, d: 300

a: 300, b: 200, d: 400, e: 200

### **Sample Output**

{'a': 400, 'b': 300, 'c': 200, 'd': 700, 'e': 200}

Values: (200, 300, 400, 700)

2. Write a python program which prints the frequency of the numbers that were given as input by the user. Stop taking input when you find the string “STOP”. Do not print the frequency of numbers that were not given as input. **Use a dictionary to solve the problem**

### **Sample Input**

10

20

20

30

10

50

90

STOP

### **Sample Output**

10 - 2 times

20 - 2 times

30 - 1 times

50 - 1 times

90 - 1 times

3. Write python code to invert a dictionary. It should print a dictionary where the keys are values from the input dictionary and the values are lists of keys from the input dictionary having the same value. **Make sure the program handles multiple same values.**

**Sample Input**

key1 : value1, key2 : value2, key3 : value1

**Sample Output**

```
{ "value1" : ["key1", "key3"], "value2" : ["key2"] }
```

4. Write a Python program that takes a String as an input from the user and counts the frequency of each character using a dictionary. For solving this problem, you may use each character as a key and its frequency as values. [You are not allowed to use the count() function]. You may ignore the case for simplicity (i.e. may consider P and p to be the same).

**Sample Input:**

Python programming is fun

**Sample Output:**

```
{'p': 2, 'y': 1, 't': 1, 'h': 1, 'o': 2, 'n': 3, 'r': 2, 'g': 2, 'a': 1, 'm': 2, 'i': 2, 's': 1, 'f': 1, 'u': 1 }
```

5. Two words are anagrams if they contain all of the same letters, but in a different order. For example, “evil” and “live” are anagrams because each contains one “e”, one “i”, one “l”, and one “v”.

Write a program that reads two strings from the user and determines whether or not they are anagrams. **Use a dictionary to solve the problem.**

**Sample Input**

evil  
live

**Sample Output**

Those strings are anagrams.

6. On some basic cell phones, text messages can be sent using the numeric keypad. Because each key has multiple letters associated with it, multiple key presses are needed for most letters. Pressing the number once generates the first character listed for that key. Pressing the number 2, 3, 4 or 5 times generates the second, third, fourth or fifth character.

Key	Symbols
1	.,?!:
2	ABC
3	DEF
4	GHI
5	JKL
6	MNO
7	PQRS
8	TUV
9	WXYZ
0	Space

Write a program that displays the key presses needed for a message entered by the user. Construct a dictionary that maps from each letter or symbol to the key presses needed to generate it. Then use the dictionary to create and display the presses needed for the user's message.

**Sample Input**  
Hello, World!

**Sample Output**  
4433555555666110966677755531111