PRACTICE SHEET

What you will learn:

Tuples and Unpacking, Dictionaries, Pattern Matching (Bijective mapping), Greedy Algorithm, Set Operations, Statistical Calculations, Combinatorics & Subset Generation, Vectors, Bubble Sort

[EASY] Q1) What would be the output of the following code if ntpl = ("Hello", "Nita", "How's", "life?")**?**

```
(a, b, c, d) = ntpl

print("a is:", a)

print("b is:", b)

print("c is:", c)

print("d is:", d)

ntpl = (a, b, c, d)

print(ntpl[0][0] + ntpl[1][1], ntpl[1])
```

[EASY] Q2) What will be the output of the following code?

```
tuple_a = 'a', 'b'

tuple_b = ('a', 'b')

print(tuple_a == tuple_b)
```

Options:
(a) 0
(b) 1
(c) False
(d) True

[EASY] Q3) What will be the output of the following code snippet?

```
my_dict = {}
```

```
my_dict[(1, 2, 4)] = 8

my_dict[(4, 2, 1)] = 10

my_dict[(1, 2)] = 12

sum = 0

for k in my_dict:

   sum += my_dict[k]

print(sum)

print(my_dict)
```

[MEDIUM] Q4) Given the values x, y, p, q of a simple chemical equation of the type:
$b_1 A_x + b_2 B_y \rightarrow b_3 A_pB_q$
The task is to find the values of constants b_1, b_2, b_3 such that the equation is balanced on both sides and it must be the reduced form.
Examples:

Input: x = 2, y = 3, p = 4, q = 5
Output: b_1 = 6, b_2 = 5, b_3 = 3
Input: x = 1, y = 2, p = 3, q = 1
Output: b_1 = 6, b_2 = 1, b_3 = 2

[MEDIUM] Q5) Check if the string follows a given pattern **(bijective mapping)**.

Input: pattern = "abba", s = "dog cat cat dog"

Output: True

[EASY] Q6) Given a list with elements, construct a dictionary with frequency of factors.

Input : test_list = [2, 4, 6, 8]

Output : {1: 4, 2: 4, 3: 1, 4: 2, 5: 0, 6: 1, 7: 0, 8: 1}

Explanation : All factors count mapped, e.g 2 is divisible by all 4 values, hence mapped with 4. 3 is only a factor of 6, hence mapped with 1.

[EASY] Q7) American currency has coins in the denominations of **1 (pennies), 5 (nickels), 10 (dimes), and 25 cents (quarters)**. Imagine that we were programming a cash register to dispense correct change. In this exercise, we would need to calculate the number of each coin for a given amount of change.

Write a makeChange() function with an amount parameter. The amount parameter contains an integer of the number of cents to make change for. For example, 30 would represent 30 cents and 125 would represent $1.25. This function should return a dictionary with keys 'quarters', 'dimes', 'nickels', and 'pennies', where the value for a key is an integer of the number of this type of coin.

The value for a coin's key should never be 0. Instead, the key should not be present in the dictionary. For example, makeChange(5) should return {'nickels': 1} and not {'quarters': 0, 'dimes': 0, 'nickels': 1, 'pennies': 0}.

For example, makeChange(30) would returns the dictionary {'quarters': 1, 'nickels': 5} to represent the coins used for 30 cents change. **The function should use the minimal number of coins.** For example, makeChange(10) should return {'dimes': 1} and not {'nickels': 2}, even though they both add up to 10 cents.

makeChange(30) == {'quarters': 1, 'nickels': 1}

makeChange(10) == {'dimes': 1}

makeChange(57) == {'quarters': 2, 'nickels': 1, 'pennies': 2}

makeChange(100) == {'quarters': 4}

makeChange(125) == {'quarters': 5}

[EASY] Q8) You're given a list of words. Write a program to print how many unique vowels appear in **all** the words.

words = ['hello', 'world', 'python']

Expected Output: Number of vowels common in all words = 1 (only 'o')

[MEDIUM] Q9) Write a mode() function that has a numbers parameter. This function returns the mode, or most frequently appearing number, of the list of integer and floating-point numbers passed to the function. This exercise tests your ability to use a dictionary to keep a count of the numbers in a list to find the most frequent number.

Example :

mode([1, 1, 2, 3, 4]) → 1

mode([1, 2, 3, 4, 4]) → 4

[EASY] Q10) Write a Python function switcher(*args, **kwargs) that:

1. Counts the number of positional arguments (*args).
2. Reverses the order of keyword argument names (**kwargs).
3. Returns a formatted string in the form:
   "Positional arguments: X, Reversed kwargs keys: Y"
   where X is the count of positional arguments and Y is a comma-separated string of reversed keyword argument names.

Example:

print(switcher(1, 2, a=3, b=4, c=5))

Expected Output:

Positional arguments: 2, Reversed kwargs keys: c, b, a

Assumptions:

- Works in Python 3.7+ (where dictionary insertion order is preserved).
- Handles any number of positional/keyword arguments.

[EASY] Q11) A **pangram** is a sentence that contains every letter of the alphabet at least once.

Given a string, check if the given string is a pangram or not **using sets.**

Input : The quick brown fox jumps over the lazy dog

Output : The string is a pangram

Input : Sky is beautiful

Output : The string is not pangram

[HARD] Q12) Given a set, write a Python program to generate all possible subset of size n of given set within a list.

Examples:

Input : {1, 2, 3}, n = 2

Output : [{1, 2}, {1, 3}, {2, 3}]

Input : {1, 2, 3, 4}, n = 3

Output : [{1, 2, 3}, {1, 2, 4}, {1, 3, 4}, {2, 3, 4}]

[HARD] Q13) The p-norm of a vector v = (v_1, v_2 ,..., v_n) in n-dimensional space is defined as

$$||v|| = (v_1^p + v_2^{\ p} + \cdots + v_n^{\ p})^{\ 1/p}.$$

For the special case of p = 2, this results in the traditional Euclidean norm, which represents the length of the vector. For example, the Euclidean norm of a two-dimensional vector with coordinates (4,3) has a Euclidean norm of $(4^2+3^2)^{1/2} = 25^{1/2} = 5$. Give an implementation of a function named norm such that norm(v, p) returns the p-norm value of v and norm(v) returns the Euclidean norm of v. You may assume that v is a list of numbers.

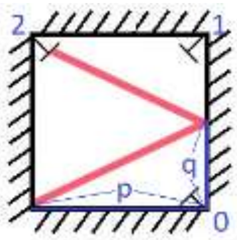
[MEDIUM] Q14) Write a program to sort a dictionary's keys using Bubble Sort and return the sorted keys as a list.

[HARD] Q15) There is a special square room with mirrors on each of the four walls. Except for the southwest corner, there are receptors on each of the remaining corners, numbered 0, 1, and 2.

The square room has walls of length p and a laser ray from the southwest corner first meets the east wall at a distance q from the 0th receptor.

Given the two integers p and q, return the number of the receptor that the ray meets first.

The test cases are guaranteed so that the ray will meet a receptor eventually.



Input: p = 2, q = 1

Output: 2

Explanation: The ray meets receptor 2 the first time it gets reflected back to the left wall.