Tutorials on list structures in Python

Q 1> Given two numbers r1 and r2 (which defines the range), write a Python program to create a list with the given range (inclusive).

Input:

r1=1 and r2=8

Output:

```
[1, 2, 3, 4, 5, 6, 7, 8]
```

Q 2> Write a program to find sum of the smallest and largest numbers in the list.

Input:

10 23 14 25

OUTPUT:

35

Q 3> Initialize a 2D list of 3*3 matrix. E.g.-

1	2	3
4	5	6
7	8	9

Check if the matrix is symmetric or not.

Q 4> Alex works at a clothing store. There is a large pile of socks that must be paired by color for sale. Given a list of integers representing the color of each sock, determine how many pairs of socks with matching colors there are.

For example, there are n=7 socks with List=[1,2,1,2,1,3,2] colors . There is one pair of color 1 and one of color 2 . There are three odd socks left, one of each color. The number of pairs is 2 .

Write a python program than print an integer representing the number of matching pairs of socks that are available.

Input Format

The first line contains an integer n , the number of socks represented in List.

The second line contains n space-separated integers describing the colors List[i] of the socks in the pile.

Constraints

$1 \leq n \leq 100$

$1 \leq List[i] \leq 100$ where $0 \leq i < 100$

Output Format

Return the total number of *matching pairs* of socks that Alex can sell.

Sample Input

9
10 20 20 10 10 30 50 10 20

Sample Output

3

Q 5> What will be the output

```
data = [x for x in range(5)]

temp = [x for x in range(7) if x in data and x%2==0]

print(temp)
```

Q 6 > Sorting refers to arranging data in a particular format. Sort a list of integers in ascending order (without using built-in sorted function). One of the algorithm is selection sort. Use below explanation of selection sort to do this.

INITIAL LIST :

2 3 1 45 15

First iteration : Compare every element after first element with first element and if it is larger then swap. In first iteration, 2 is larger than 1. So, swap it.

1 3 2 45 15

Second iteration : Compare every element after second element with second element and if it is larger then swap. In second iteration, 3 is larger than 2. So, swap it.

1 2 3 45 15

Third iteration : Nothing will swap as 3 is smaller than every element after it.

1 2 3 45 15

Fourth iteration : Compare every element after fourth element with fourth element and if it is larger then swap. In fourth iteration, 45 is larger than 15. So, swap it.

1 2 3 15 45

Q 7> Given three integers A, B and C, the task is to find the length of a race track if 3 racers are competing in a race where the first racer beats the second racer by A metres, the first racer beats the third racer by B meters and the second racer beats the third by C meters.

Case 1: By the time when the First racer finished the race, distances covered by all the 3 racers are:
First = X, Second = X – A, Third = X – B
Let the time taken by the First racer to finish the race be **T_1.**
Case 2: By the time when the Second racer finished the race, distances covered by the remaining 2 racers are:
Second = X, Third = X – C
Let the time taken by the Second racer to finish the race be **T_2.**

Sample Input & Output:
Input: *A = 11, B = 90, C = 80*
Output: *880*
Input: *A = 10, B = 20, C = 12*
Output: *60*

Q 8> Using range(1,101), make two list, one containing all even numbers and other containing all odd numbers

Q 9> Shallow coping of a list by copy()

```
round1 = ['chuck norris', 'bruce lee', 'sonny chiba']

round2 = round1.copy()

round1.remove('sonny chiba')

print(round1)

print(round2)
```

Q 10> Predict the output

```
lst = ['python', 'is', 'cool', 'language']

for i in range(len(lst)+1):

    print(lst[i])
```

Q 11> An avid hiker keeps meticulous records of their hikes. During the last hike that took exactly ***steps*** steps, for every step it was noted if it was an *uphill*, ***U*** , or a *downhill*, ***D*** step. Hikes always start and end at sea level, and each step up or down represents a 1 unit change in altitude. We define the following terms:

- A *mountain* is a sequence of consecutive steps *above* sea level, starting with a step *up* from sea level and ending with a step *down* to sea level.
- A *valley* is a sequence of consecutive steps *below* sea level, starting with a step *down* from sea level and ending with a step *up* to sea level.

Given the sequence of *up* and *down* steps during a hike, Write a python program to find and print the number of *valleys* walked through.

Example

***Steps* = 8 paths = [DDUUUUDD]**

The hiker first enters a valley 2 units deep. Then they climb out and up onto a mountain 2 units High. Finally, the hiker returns to sea level and ends the hike.

Returns

int: the number of valleys traversed

Input Format

The first line contains an integer ***steps***, the number of steps in the hike.

The second line contains a single string ***path***, of ***steps*** characters that describe the path.

Constraints

*2 ≤ **steps** ≤ **1000000***

Path[i] ∈ {U, D}

Sample Input

```
8
UDDDUDUU
```

Sample Output

```
1
```