Linear Arrays [CO5]

[Each method carries 5 marks]

Instructions for students:

- Complete the following methods on Arrays
- You may use any language to complete the tasks.
- All your methods must be written in one single .java or .py or .pynb file. DO NOT CREATE separate files for each task.
- If you are using JAVA, you must include the main method as well which should test your other methods and print the outputs according to the tasks.
- If you are using PYTHON, then follow the coding templates shared in this folder.

NOTE:

- **YOU CANNOT USE ANY BUILT-IN FUNCTION EXCEPT** len **IN PYTHON. [negative indexing, append is prohibited]**
- **YOU HAVE TO MENTION SIZE OF ARRAY WHILE INITIALIZATION**
- **YOUR CODE SHOULD WORK FOR ANY VALID INPUTS. [Make changes to the Sample Inputs and check whether your program works correctly]**

Array Tasks Part I:

1. **Play Right!:**

   Suppose you are having a party with your friends. Your friends are playing a game similar to pillow passing. In this game they stand in a line and with each beat of the music they change their position. When the beat is high, they move one position right and when the beat is low they remain still. As for the person standing at the end of the line, with a high change of beat he/she runs to the beginning. Now you are not allowed to participate in the game because you didn't bring enough cola for everyone! Rather, they decided to play a game on you. They will give you their standing sequence in an array where each element is their ID and another array where low beat is denoted as 0 and high beat is denoted as 1. With this information you have to tell their final standing sequence within seconds! Now as a CS student, implement the scenario using a proper method where two arrays are taken as parameters and return the resulting array.

   Sample Input / Driver Code:
   sequence=[10,20,30,40,50,60]
   beats = [1,0,0,1,0,1]

   #Function Call:
   print(playRight(sequence,beats))

   Sample Output:
   [40, 50, 60, 10, 20, 30]

2. Discard Cards:

This time you guys are playing a card game. You pick n numbers of UNO cards from the deck. The rule of the card game is- one person says a number and you have to discard the cards with the same number from your hand without changing the relative position of other cards. Implement the scenario using a proper method which takes the cards in your hands as an array and a number as parameters; and returns the resulting array. (Modularize as much as possible)

Sample Input / Driver Code:

```
cards=[10,2,30,2,50,2,2,0,0]

#Function Call:
print(discardCards(cards,2))
```

Sample Output:

```
[10,30,50,0,0,0,0,0,0]
```

[Hint: First write a helper function that discards an element from a specific index and use the function to discard all occurrences.]

3. **Merge Lineup:**

Now you guys are playing a 2v2 fierce pokemon battle. At one point your and your teammate's pokemons have very low hp. So you created a new rule to merge the hp of your and your teammate's pokemons and create a new pokemon lineup. But the opposite team placed a special condition. You and your teammate have to add the hp of your team's pokemons from opposite directions. That is, your first pokemon's hp will be added to your teammate's last pokemon's hp; your second pokemon's hp will be added to your teammate's second last pokemon's hp and so on and so forth. The None elements denote 0 hp. You and your teammate have the same number of pokemons. Implement the scenario using a proper method which takes your and your teammate's pokemon hp as two arrays (as parameters); and returns the resulting arr.

Sample Input / Driver Code:
pokemon_1: [4, 5, -1, None, None]
pokemon_2: [2, 27, 7, 12, None]

#Function Call: print(mergeLineup(pokemon_1, pokemon_2))

Sample Output:
[4,17,6,27,2]

Explanation:
4+None(0) = 4 {pokemon_1[0]+pokemon_2[4]},
5+12 = 17 {pokemon_1[1]+pokemon_2[3]},
-1+7 = 6 {pokemon_1[2]+pokemon_2[2]},
None(0)+27 = 27 {pokemon_1[3]+pokemon_2[1]},
None(0)+2 = 2 {pokemon_1[4]+pokemon_2[0]}

Sample Input / Driver Code:
pokemon_1: [12, 3, 25, 1, None]
pokemon_2: [5, -9, 3, None, None]

#Function Call: print(mergeLineup(pokemon_1, pokemon_2))

Sample Output:
[12, 3, 28, -8, 5]

4. Balance Your Salami:

Suppose the elements of an array ‘salami’ containing positive integers, denote the amount of Salami you got from each elder in Eid. Everyone gave you a single note of taka. Now your mother orders you to distribute the Salami between your younger sibling and you. Being a lazy person, you do not want to split any note into changes. So, you want to distribute the notes in such a way that for some index 0 < i < A.length - 1, you get all the notes starting from A[0], A[1], up to A[i - 1]. And all the notes starting from A[i] up to A[A.length - 1] will be given to your sibling. Thus, both of you get equal amounts. If such a distribution is possible, return true. Else, return false.

Sample Input / Driver Code:
salami: [1, 1, 1, 2, 1]
#Function Call: print(balanceSalami(salami))

Sample Output: True

Explanation:
summation of [1, 1, 1] = summation of [2,1]. So you get three 1 taka notes and your sibling gets one 2 taka & one 1 taka note

Sample Input / Driver Code:
salami: [2, 1, 1, 2, 1]
#Function Call: print(balanceSalami(salami))

Sample Output: False

Sample Input / Driver Code:
salami: [10, 3, 1, 2, 10]
#Function Call: print(balanceSalami(salami))

Sample Output: True

Explanation:
summation of [10, 3] = summation of [1,2,10].

5. **Protect Salami:**

On the night of Eid day, your mother gently tells you and your sibling, "You will lose your Salami money. Give them to me, I will keep them safe." However, both of you now know your mother's trick and are quite adamant to keep some money for yourselves this time. So, after brainstorming, you get an idea. You told your mother "Both of us got some repeated taka notes, i.e. these taka notes have appeared more than once. You can only take those repeated notes if there are at least two notes with the same number of repetition." Implement the scenario using a proper method which takes the taka notes you both got as Salami as an array; and returns True if mother can take money. Otherwise returns False.

Sample Input / Driver Code:
salami: [4,5,6,6,4,3,6,4]
#Function Call: print(protectSalami(salami))

Sample Output: True

Explanation:
Two notes repeat in this array: 4 and 6. 4 has a repetition of 3, 6 has a repetition of 3. Since two notes have the same repetition, mother can take money and the output is True.

Sample Input / Driver Code:
salami: [3,4,6,3,4,7,4,6,8,6,6]
#Function Call: print(protectSalami(salami))

Sample Output: False

Explanation:
Three notes repeat in this array:3,4 and 6 .3 has a repetition of 2, 4 has a repetition of 3, 6 has a repetition of 4. Since no two notes have the same repetition output is False and mother cannot take money.

6. Odd Even Wave:

Write a method that takes an array as a parameter, and creates a new array that stores an odd even wave of the array. An odd even wave is when the adjacent elements of an odd number are even and the adjacent elements of the even numbers are odd. It is guaranteed that the given array will have no extra odd or even numbers left.

Sample Input / Driver Code:
arr: [2,12,3,8,1,5]

#Function Call:
print(waveYourFlag(arr))

Sample Output:
[2,3,12,1,8,5]

Sample Input / Driver Code:
arr: [45,23,78,84,41]

#Function Call:
print(waveYourFlag(arr))

Sample Output:
[45,78,23,84,41]