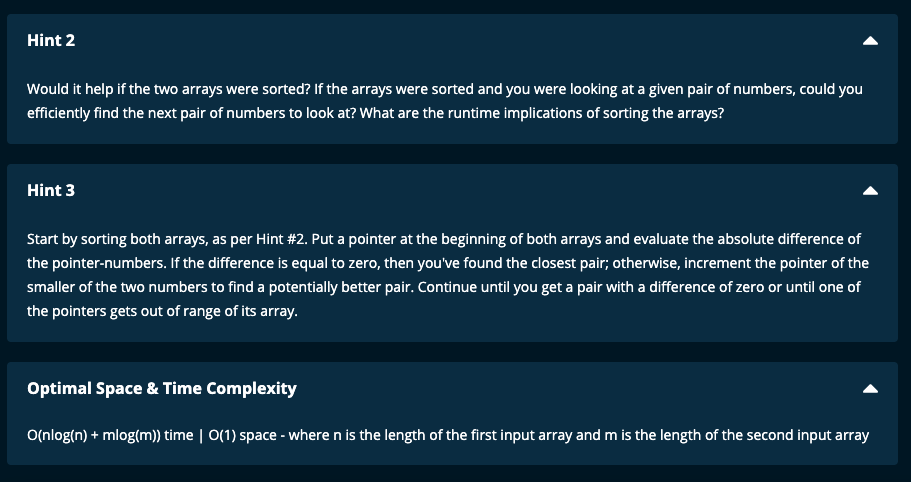
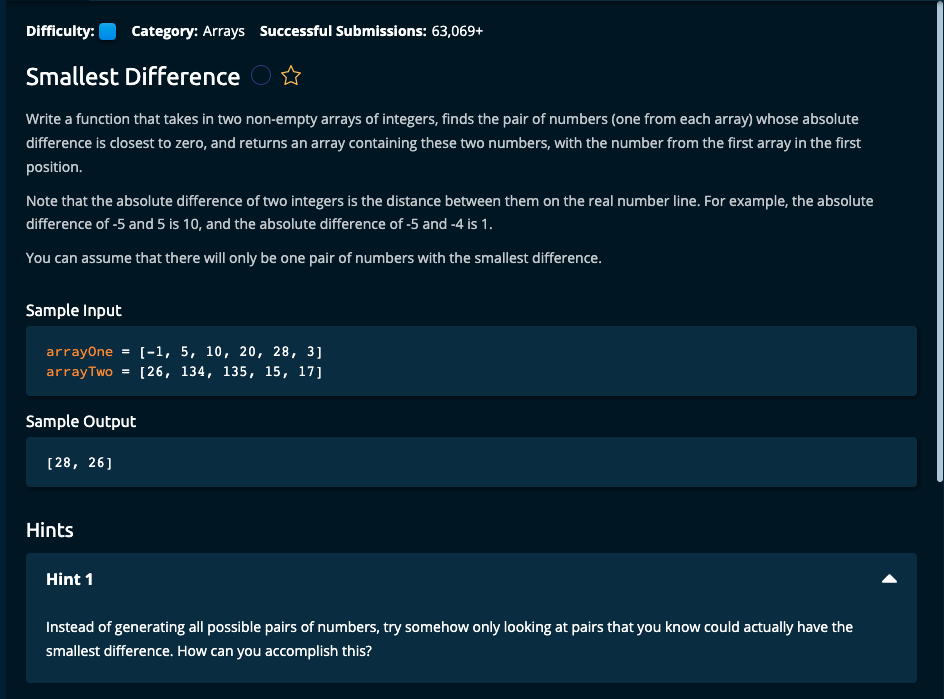
Smallest Difference (Medium)

<https://www.algoexpert.io/questions/smallest-difference>



My Solution :

Solution 1: Uses absolute value

def smallestDifference(arrayOne, arrayTwo):

arrayOne.sort()

arrayTwo.sort()

i = 0

j = 0

minAbsDiff = abs(arrayOne[0] - arrayTwo[0])

firstNum = arrayOne[0]

secondNum = arrayTwo[0]

n = len(arrayOne)

m = len(arrayTwo)

while (i < n) and (j < m):

absDiff = abs(arrayOne[i] - arrayTwo[j])

if absDiff < minAbsDiff:

minAbsDiff = absDiff

firstNum = arrayOne[i]

secondNum = arrayTwo[j]

if arrayOne[i] == arrayTwo[j]:

return [arrayOne[i], arrayTwo[j]]

elif arrayOne[i] < arrayTwo[j]:

i += 1

else:

j += 1

return [firstNum, secondNum]

Solution 2:

def smallestDifference(arrayOne, arrayTwo):

arrayOne.sort()

arrayTwo.sort()

i = 0 # index for arrayOne

j = 0 # index for arrayTwo

smallest = float("inf") # minimum absolute difference

current = float("inf") # current absolute difference

smallestPair = []

while i < len(arrayOne) and j < len(arrayTwo):

firstNum = arrayOne[i]

secondNum = arrayTwo[j]

if firstNum < secondNum:

current = secondNum - firstNum

i += 1

elif firstNum > secondNum:

current = firstNum - secondNum

j += 1

else:

return [firstNum, secondNum]

if current < smallest:

smallest = current

smallestPair = [firstNum, secondNum]

return smallestPair

Algoexpert Solution:

def smallestDifference(arrayOne, arrayTwo):

arrayOne.sort()

arrayTwo.sort()

i = 0 # index for arrayOne

j = 0 # index for arrayTwo

smallest = float("inf") # minimum absolute difference

current = float("inf") # current absolute difference

smallestPair = []

while i < len(arrayOne) and j < len(arrayTwo):

firstNum = arrayOne[i]

secondNum = arrayTwo[j]

if firstNum < secondNum:

current = secondNum - firstNum

i += 1

elif secondNum < firstNum:

current = firstNum - secondNum

j += 1

else:

return [firstNum, secondNum]

if smallest > current:

smallest = current

smallestPair = [firstNum, secondNum]

return smallestPair

---------------------------------------------------

Notes:

1. Sort arrayOne and arrayTwo in place.
2. Initialize the indexes of the two arrays to 0 with I and j.
3. Initialize the minimum absolute difference as smallest to infinity
4. Initialize the current absolute difference to infinity.
5. Initialize smallestPair to an empty list.
6. With a while loop, loop until one of the arrays index is exceeded and do the following:

If the element of arrayOne is less than that of arrayTwo, then current will be the difference between the two elements. Then increment the index of arrayOne.

If the element of arrayOne is greater than that of arrayTwo, then current will be the difference between the two elements. Then increment the index of arrayTwo.

If the elements of arrayOne and arrayTwo are equal, then return the two elements as a list.

Check if the current absolute difference is less than smallest. If so, then update smallest to the value of current and the smallest pair with the elements from arrayOne and arrayTwo

1. Finally, return the smallest pair.

Time Complexity: O(nlog(n) + mlog(m)) where n is the length of arrayOne and m the length of arrayTwo.

Though we traverse through each array once and it is linear, we had first sorted each array and this done in nlog(n) for arrayOne and mlog(m) time for arrayTwo.

Space Complexity: O(1)

Since the arrays are sorted in place, no additional space is required. We just need to keep track of smallest absolute difference, current absolute difference and smallest pair, and this can be done in constant time.