**題目：**How to implement merge-sort such that the extra space used is about n/2 where n is the number of input elements?

想法:原本在最後一次merge 需要create 一個長度為n的陣列來存放merge結果，但如果是要在n/2的空間下就完成merge就必須把結果存在現有的陣列中來節省空間，所以想法是先排好一半並且把結果存入左邊陣列，剩下的元素用append的方式依順序由小到大加入左邊陣列合成一個長度為n的陣列，所消耗的空間就只會是append時內部copy的空間也就是n/2 (若append為deepcopy)

L 為 merge sort 左邊陣列中要比較的元素index

R 為 merge sort 右邊陣列中要比較的元素index

P 為 下一個最小的值要存放的位置(都會在左陣列)

**步驟：**

Step1:比較 L 與 R 哪個大

Step2:將小的與P做交換

Step3:直到L陣列被排列完成後繼續比較剩餘的L與R elements

Step4:將小的append到L陣列

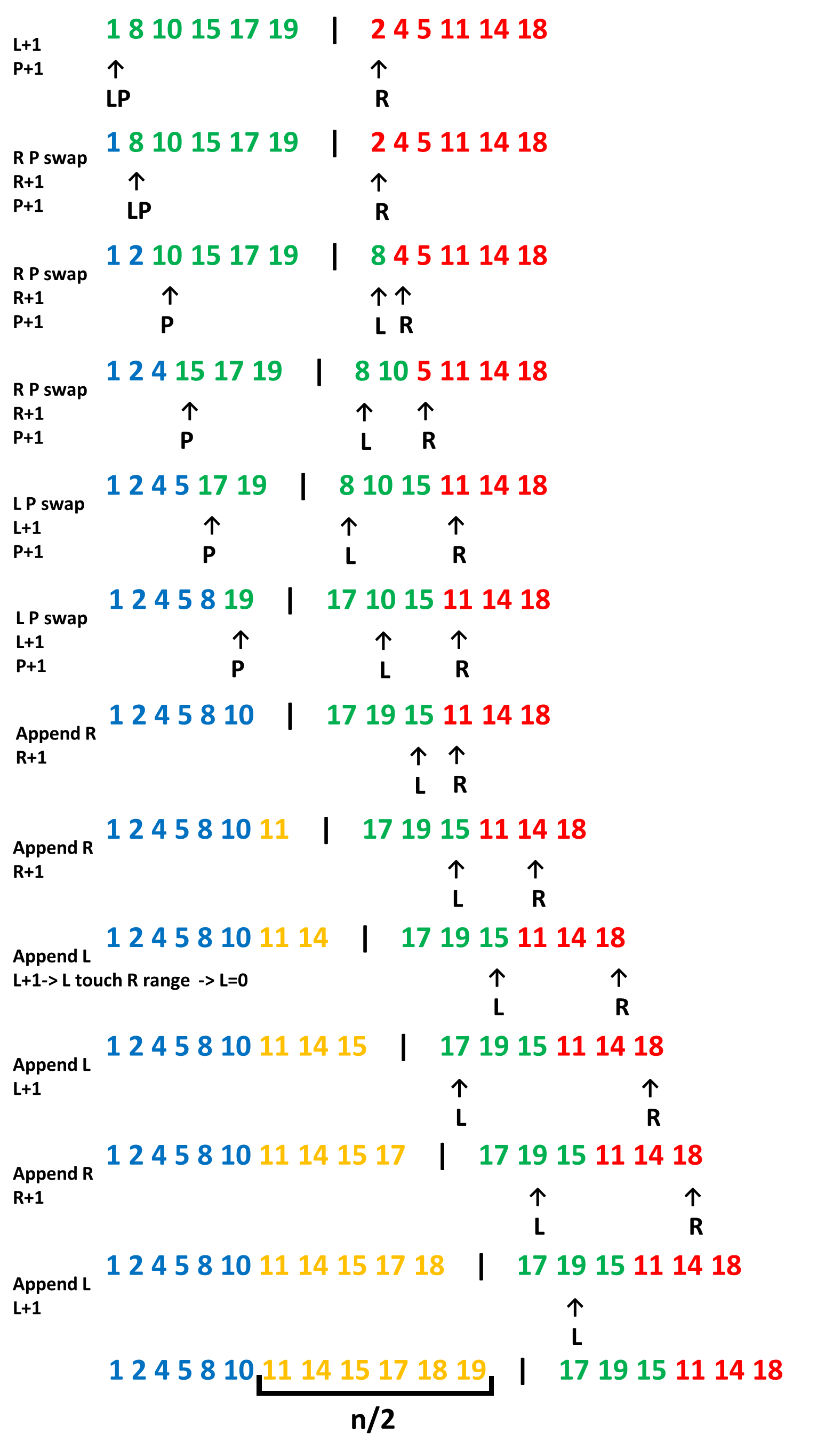
**圖例說明：**

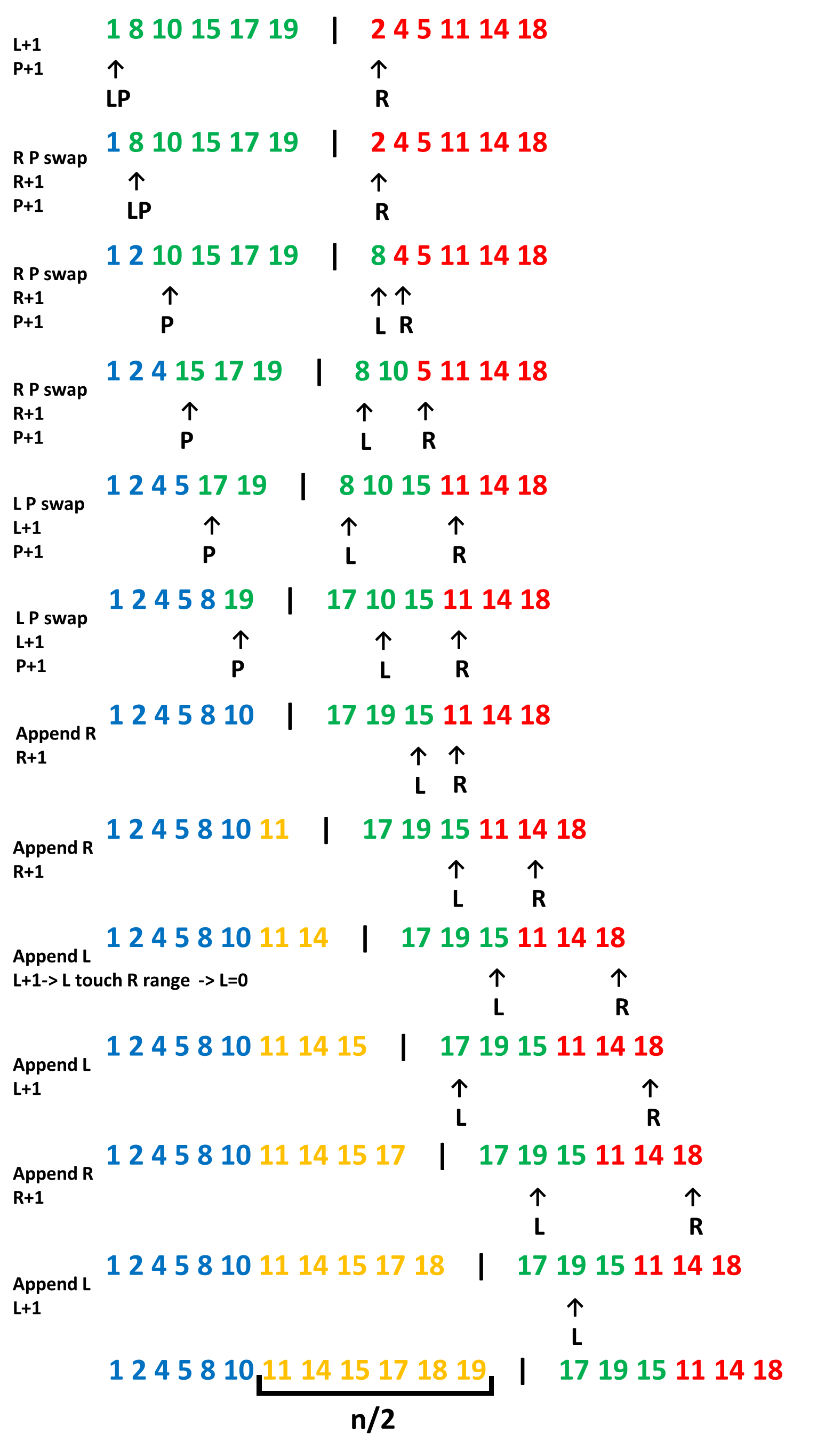
綠：左陣列

紅：右陣列

藍：排好元素

黃：多餘空間/排好元素





**Pseudocode**:

function merge(array1, array2):

# Sort left list

for position in range(len(array1)):

if left\_index == -1:

if array1 [position] > array2[right\_index]:

array1 [postion], array2[right\_index] = swap(array1[position], array2[right\_index])

left\_index = right\_index

right\_index += 1

else:

if array2[left\_index] > array2[right\_index]:

array1[postion], array2[right\_index] = swap(array1[postion], array2[right\_index])

right\_index += 1

else:

array1[postion], array2[left\_index] = swap(array1[postion], array2[left\_index])

left\_index += 1

# when list is 3 2 5 3 78 32 6 2 234 6 34 354 56 and do not have below code then program will be error at the last merge

if left\_index == right\_index:

left\_index = 0

left\_index2 = 0

right\_index2 = right\_index

if left\_index == -1:

array1.extend(array2)

return array1

# append right list element into left list

while left\_index2 < right\_index and right\_index2 < right\_max:

if array2[left\_index2] < array2[right\_index2]:

array1.append(array2[left\_index2])

left\_index2 += 1

else:

array1.append(array2[right\_index2])

right\_index2 += 1

for i in range(right\_index2, right\_max):

array1.append(array2[i])

for i in range(left\_index2, right\_index):

array1.append(array2[i])

return array1