Clarusway

JS-CC-004 : Merge Arrays

Purpose of the this coding challenge is to write a code that given two sorted arrays, returns merged array of these inputs.

Learning Outcomes

At the end of the this coding challenge, students will be able to;

- Analyze a problem, identify and apply programming knowledge for appropriate solution.
- Demonstrate their knowledge of algorithmic design principles by using JavaScript effectively.

Problem Statement

- Write a function that takes two arrays of sorted numbers and combines them into one array as result.
- Please note that, empty arrays to be checked to avoid exception!
- Please take a look at the empty function and test code below:

JavaScript Solution

```
function mergeArrays(ArrayA, ArrayB) {
  // Set up our ResultArray
  const ResultArr = [];

  let ArrB_curInd = 0;
  let ArrA_curInd = 0;
  let ResArr_curInd = 0;

  while (ResArr_curInd < ArrayA.length + ArrayB.length) {
    const isArrA_finished = ArrA_curInd >= ArrayA.length;
    const isArrB_finished = ArrB_curInd >= ArrayB.length;

    // Case: next comes from my ArrayA
    // ArrayA must not be finished, and EITHER:
    //  - ArrayB is finished, or
    //  - The current element in ArrayA is less
    //    than the current element in ArrayB
    if (
      !isArrA_finished &&
      (isArrB_finished || ArrayA[ArrA_curInd] < ArrayB[ArrB_curInd])
    ) {
      ResultArr[ResArr_curInd] = ArrayA[ArrA_curInd];
      ArrA_curInd++;
      // Case: next comes from ArrayB
    } else {
      ResultArr[ResArr_curInd] = ArrayB[ArrB_curInd];
      ArrB_curInd++;
    }
    ResArr_curInd++;
  }
```

```
  return ResultArr;
}

/* *** Tests *** */
let desc = "with empty arrays";
let actual = mergeArrays([], []);
let expected = [];
assertDeepEqual(actual, expected, desc);

desc = "first array empty";
actual = mergeArrays([], [7, 8, 9]);
expected = [7, 8, 9];
assertDeepEqual(actual, expected, desc);

desc = "second array empty";
actual = mergeArrays([10, 11, 12], []);
expected = [10, 11, 12];
assertDeepEqual(actual, expected, desc);

desc = "arrays with same lengths";
actual = mergeArrays([12, 14, 16], [11, 13, 17]);
expected = [11, 12, 13, 14, 16, 17];
assertDeepEqual(actual, expected, desc);

desc = "arrays with different lengths";
actual = mergeArrays([22, 24, 26, 28], [21, 27]);
expected = [21, 22, 24, 26, 27, 28];
assertDeepEqual(actual, expected, desc);

function assertDeepEqual(a, b, desc) {
  const aStr = JSON.stringify(a);
  const bStr = JSON.stringify(b);
  if (aStr !== bStr) {
    console.log(`${desc} ... FAIL: ${aStr} !== ${bStr}`);
  } else {
    console.log(`${desc} ... PASS`);
  }
}
```

Python Solution

```
import unittest

def merge_lists(listA, listB):

    # Combine the sorted lists into result_list
    # Set up our result_list
    result_list_size = len(listA) + len(listB)
    result_list = [None] * result_list_size

    cur_ind_listB = 0
    cur_ind_listA = 0
    cur_ind_result_list = 0
    while cur_ind_result_list < result_list_size:
        is_listA_finished = cur_ind_listA >= len(listA)
        is_listB_finished = cur_ind_listB >= len(listB)
```

```
        if not is_listA_finished and (
            is_listB_finished or listA[cur_ind_listA] < listB[cur_ind_listB]
        ):
            # Case: next comes from listA
            # listA must not be finished, and EITHER:
            #  - listB is finished, or
            #  - the current element in listA is less
            #    than the current element in listB
            result_list[cur_ind_result_list] = listA[cur_ind_listA]
            cur_ind_listA += 1
        else:
            # Case: next comes from listB
            result_list[cur_ind_result_list] = listB[cur_ind_listB]
            cur_ind_listB += 1

        cur_ind_result_list += 1

    return result_list


# **  Tests  **

class Test(unittest.TestCase):
    def test_with_empty_lists(self):
        actual = merge_lists([], [])
        expected = []
        self.assertEqual(actual, expected)

    def test_first_list_empty(self):
        actual = merge_lists([], [7, 8, 9])
        expected = [7, 8, 9]
        self.assertEqual(actual, expected)

    def test_second_list_empty(self):
        actual = merge_lists([10, 11, 12], [])
        expected = [10, 11, 12]
        self.assertEqual(actual, expected)

    def test_lists_with_same_lengths(self):
        actual = merge_lists([12, 14, 16], [11, 13, 17])
        expected = [11, 12, 13, 14, 16, 17]
        self.assertEqual(actual, expected)

    def test_lists_with_different_lengths(self):
        actual = merge_lists([22, 24, 26, 28], [21, 27])
        expected = [21, 22, 24, 26, 27, 28]
        self.assertEqual(actual, expected)

unittest.main(verbosity=3)
```

⌛ Happy Coding ✍