GeeksforGeeks

A computer science portal for geeks

Practice IDE Q&A GeeksQuiz

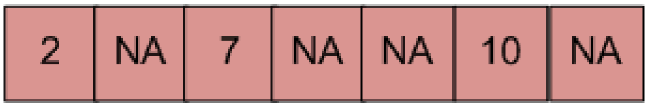
Merge an array of size n into another array of size m+n

Asked by Binod

Question:

There are two sorted arrays. First one is of size m+n containing only m elements. Another one is of size n and contains n elements. Merge these two arrays into the first array of size m+n such that the output is sorted.

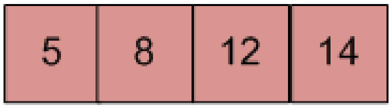
Input: array with m+n elements (mPlusN[]).



2	NA	7	NA	NA	10	NA

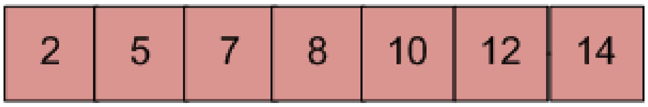
NA => Value is not filled/available in array mPlusN[]. There should be n such array blocks.

Input: array with n elements (N[]).



5	8	12	14

Output: N[] merged into mPlusN[] (Modified mPlusN[])



2	5	7	8	10	12	14

Algorithm:

```
Let first array be mPlusN[] and other array be N[]
1) Move m elements of mPlusN[] to end.
2) Start from nth element of mPlusN[] and 0th element of N[] and merge them
    into mPlusN[].
```

Implementation:

```
#include <stdio.h>
```

```
/* Assuming -1 is filled for the places where element
   is not available */
#define NA -1

/* Function to move m elements at the end of array mPlusN[] */
void moveToEnd(int mPlusN[], int size)
{
  int i = 0, j = size - 1;
  for (i = size-1; i >= 0; i--)
    if (mPlusN[i] != NA)
    {
      mPlusN[j] = mPlusN[i];
      j--;
    }
}

/* Merges array N[] of size n into array mPlusN[]
   of size m+n*/
int merge(int mPlusN[], int N[], int m, int n)
{
  int i = n;  /* Current index of i/p part of mPlusN[]*/
  int j = 0; /* Current index of N[]*/
  int k = 0; /* Current index of of output mPlusN[]*/
  while (k < (m+n))
  {
    /* Take an element from mPlusN[] if
       a) value of the picked element is smaller and we have
          not reached end of it
       b) We have reached end of N[] */
    if ((i < (m+n) && mPlusN[i] <= N[j]) || (j == n))
    {
      mPlusN[k] = mPlusN[i];
      k++;
      i++;
    }
    else  // Otherwise take emenet from N[]
    {
      mPlusN[k] = N[j];
      k++;
      j++;
    }
  }
}

/* Utility that prints out an array on a line */
void printArray(int arr[], int size)
{
  int i;
  for (i=0; i < size; i++)
```

```
    printf("%d ", arr[i]);

  printf("\n");
}

/* Driver function to test above functions */
int main()
{
  /* Initialize arrays */
  int mPlusN[] = {2, 8, NA, NA, NA, 13, NA, 15, 20};
  int N[] = {5, 7, 9, 25};
  int n = sizeof(N)/sizeof(N[0]);
  int m = sizeof(mPlusN)/sizeof(mPlusN[0]) - n;

  /*Move the m elements at the end of mPlusN*/
  moveToEnd(mPlusN, m+n);

  /*Merge N[] into mPlusN[] */
  merge(mPlusN, N, m, n);

  /* Print the resultant mPlusN */
  printArray(mPlusN, m+n);

  return 0;
}
```

Output:

```
2 5 7 8 9 13 15 20 25
```

Time Complexity: O(m+n)

Please write comment if you find any bug in the above program or a better way to solve the same problem.

82 Comments Category: Arrays Tags: array

Related Posts:

- Longest Span with same Sum in two Binary arrays
- Count Inversions of size three in a give array
- Find the subarray with least average
- Count triplets with sum smaller than a given value
- Find zeroes to be flipped so that number of consecutive 1’s is maximized
- Reorder an array according to given indexes
- Find maximum value of Sum(i*arr[i]) with only rotations on given array allowed
- Find maximum average subarray of k length

(Login to Rate and Mark)

2.3 Average Difficulty : **2.3/5.0**
Based on **16** vote(s)

□ Add to TODO List

□ Mark as DONE

Writing code in comment? Please use code.geeksforgeeks.org, generate link and share the link here.