Merge Sort

locked

Problem | Submissions | Leaderboard | Discussions

Submitted a few seconds ago • Score: 100.00 Status: Accepted

✔ Test Case #0	✔ Test Case #1	✔ Test Case #2
✔ Test Case #3		

Submitted Code

Language: C++ Open in editor

```
1 //program on merge sort
2 //using array
3 #include<stdio.h>
4 #include<stdlib.h> //for allocating the memory dynamically
5 #include<time.h>
6
```

```
//function to merge the sub array in ascending order
void merge(int array[], int lower, int mid, int upper){
//array to store sub sorted array
int a[mid-lower+1], b[upper-mid];

//copy the data in the temporary array
int i=0;
for(; i<mid-lower+1; i++){
    a[i]=array[lower+i];
}
i=0;
for(;i<upper-mid;i++){
    b[i]=array[mid+1+i];
}

//pointers to point the array a and b's current element
int p=0,q=0;
//pointer to point the current element of the actual array
i=lower;

while(p<mid-lower+1 && q<upper-mid){
    if(a[p]<b[q]){
        array[i]=a[p];
        //increment the value of p and i
        p++; i++;
    }
    else{
        array[i]=b[q];
        //increment the value of q and i
        q++; i++;
    }
}
//add the remaining element
while(p<mid-lower+1){
    array[i]=a[p];
    p++; i++;
}
while(q<upper-mid){
```

```
    array[i]=b[q];
    q++; i++;
}
}
//function to sort the array
void merge_sort(int array[], int lower, int upper)
{
    if(lower<upper)
    {
        int mid=(lower+upper)/2;  //divide
        merge_sort(array,lower,mid ); //conquer
        merge_sort(array,mid+1,upper); //conquer

        merge(array,lower,mid,upper); //combine

    }
}


int main(void)
{
    int size=0;
    scanf("%d",&size);

    //allocating the memory for the array
    int* array=(int*)calloc(size,sizeof(int)); //calloc initializes all the element with 0

    //input the elements
    int i=0;
    for(int i=0; i<size; i++){
        scanf("%d",&array[i]);
    }

    //calling the function merge_sort
    merge_sort(array,0,size-1);

    printf("[");
    i=0;
```

```
    //print the sorted array
    for(;i<size;i++){
        if(i!=size-1){
            printf("%d,",array[i]);
        }
        else{
            printf("%d]",array[i]);
        }
    }

    //free the array pointer
    free(array);
    return 0;

}

```