Sort

1. Bubble Sort

func BubbleSort(arr []int){

for i:=0;i<len(arr)-1;i++{

for j:=0;j<len(arr)-i-1;j++{

if arr[j] > arr[j+1]{

arr[j],arr[j+1]=arr[j+1],arr[j]

}

}

}

}

1. Selection Sort

func SelectionSort(arr []int){

for i:=0;i<len(arr);i++{

minIndex:=i

for j:=i;j<len(arr);j++{

if arr[minIndex]>arr[j]{

minIndex=j

}

}

if minIndex!=i{

arr[i],arr[minIndex]=arr[minIndex],arr[i]

}

}

}

1. Insertion Sort

func InsertionSort(arr []int){

for i:=1;i<len(arr);i++{

temp:=arr[i]

j:=i-1

for j>=0 && temp < arr[j]{

arr[j+1],arr[j]=arr[j],temp

j--

}

}

}

1. Merge Sort

func merge(arr1,arr2 []int)[]int{

combined:=make([]int,len(arr1)+len(arr2))

index,i,j:=0,0,0

for i<len(arr1) && j < len(arr2){

if arr1[i]<arr2[j]{

combined[index]=arr1[i]

index++

i++

}else{

combined[index]=arr2[j]

index++

j++

}

}

for i<len(arr1){

combined[index]=arr1[i]

index++

i++

}

for j<len(arr2){

combined[index]=arr2[j]

index++

j++

}

return combined

}

func MergeSort(arr []int) []int{

if len(arr)==1{

return arr

}

mid:=len(arr)/2

left:=MergeSort(arr[:mid])

right:=MergeSort(arr[mid:])

return merge(left,right)

}

1. Quick Sort

func Pivot(arr []int, pivotIndex,endIndex int) int {

swapIndex:=pivotIndex

for i:=pivotIndex+1;i<=endIndex;i++{

if arr[i] < arr[pivotIndex]{

swapIndex++

arr[swapIndex],arr[i]=arr[i],arr[swapIndex]

}

}

arr[pivotIndex],arr[swapIndex]=arr[swapIndex],arr[pivotIndex]

return swapIndex

}

func QuickSortHelper(arr []int ,left,right int){

if left < right{

pivotIndex:=Pivot(arr,left,right)

QuickSortHelper(arr,left,pivotIndex-1)

QuickSortHelper(arr,pivotIndex+1,right)

}

}

func QuickSort(arr []int){

QuickSortHelper(arr,0,len(arr)-1)

}