

# **Algoritmos de Orientamiento Avanzados**

---

Ing. Luis Humberto González

# Merge Sort

- Divide el arreglo en 2 subarreglos.
- Se ordenan ambos subarreglos.
- Se forma el arreglo ordenado, considerando que se tienen 2 subarreglos ya ordenados.



## Merge Sort

# Algoritmo: Merge Sort

```
void MergeSort (inicio, fin)
if (inicio < fin) {
    mitad = (inicio+fin) /2;
    MergeSort(inicio, mitad);
    MergeSort(mitad+1, fin);
    Une(inicio, mitad, fin);
}
```

# Algoritmo: Une (Merge)

```
void Une (inicio, mitad, fin) {  
    i = inicio; j = mitad+1; k = inicio;  
    while (i<=mitad) and (j<=fin) {  
        if (arreglo[i] < arreglo[j]) then  
            aux[k] = arreglo[i]; i = i+1;  
        else  
            aux[k] = arreglo[j]; j = j+1;  
        k = k +1;  
    }  
    if (i>mitad)  
        Mover elementos j a fin del arreglo al arreglo aux de k a fin;  
    else  
        Mover elementos i a mitad del arreglo al arreglo aux de k a fin;  
    Copiar aux a arreglo;  
}
```

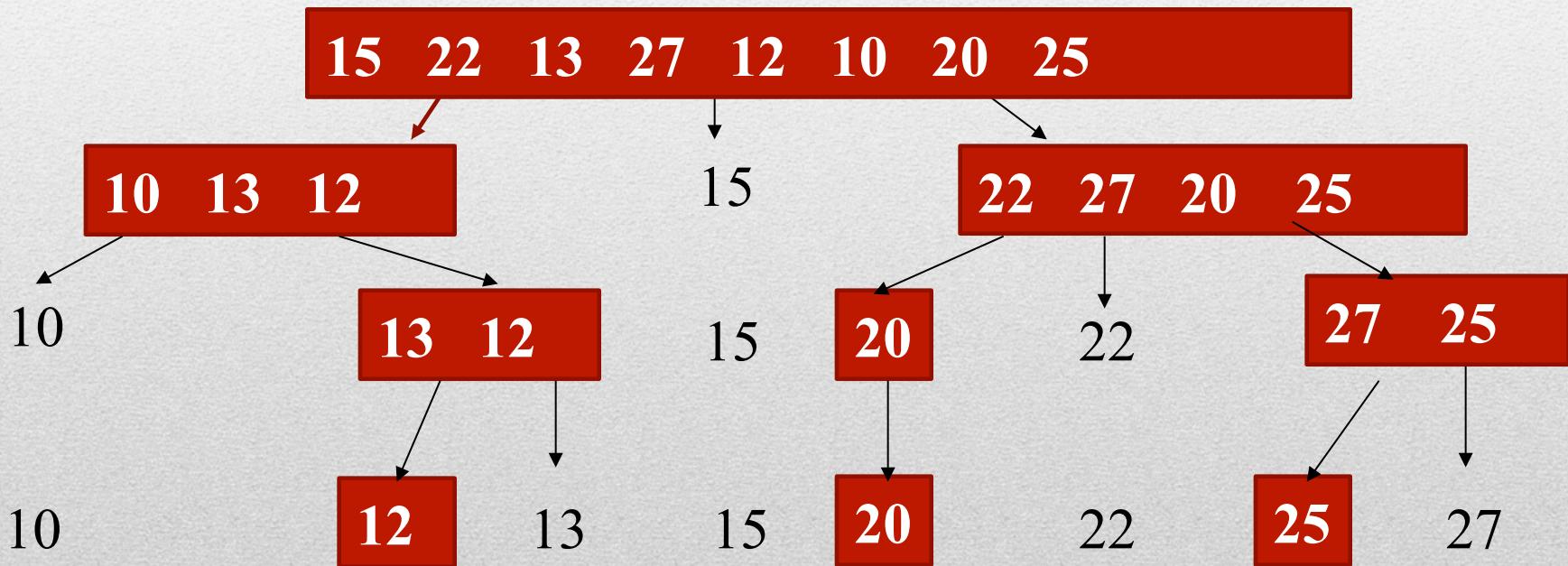
- ✓ Divide el arreglo en 2 particiones, una que contiene a los elementos menores a un elemento pivote, y otra que contiene a los elementos mayores al pivote.
- ✓ Se ordenan ambas particiones, y automáticamente se tiene todo el arreglo ordenado.
- ✓ La elección del elemento pivote es libre (por facilidad, se toma el primer elemento del arreglo).

# Quick Sort

---

# Ejemplo: Quick Sort

Considerando que el primer elemento del arreglo es el pivote:



# Algoritmo: Quick Sort

```
void QuickSort (inicio, fin){  
if (inicio < fin) {  
    Partición(inicio, fin, pivote)  
    QuickSort(inicio, pivot-1);  
    QuickSort(pivot+1, fin);  
}  
}
```

# Algoritmo: Partición

```
void Partición (inicio, fin, pivotе){  
    elempivote = arreglo[inicio]; j = inicio;  
    for (int i=inicio+1; i<=fin; i++) {  
        if (arreglo[i] < elempivote) {  
            j = j+1;  
            Intercambia arreglo[i] con arreglo[j]  
        }  
    }  
    pivotе = j;  
    Intercambia arreglo[inicio] con arreglo[pivotе]  
}
```

EJEMPLO: 15 22 13 27 12 10 20 25

# Complejidad de los algoritmos

## ✓ MERGE SORT:

- Peor caso:  $O(n \log n)$
- Caso promedio:  $O(n \log n)$

## ✓ QUICK SORT:

- Peor caso:  $O(n^2)$
- Caso promedio:  $O(n \log n)$