Ordenamiento Recursivo MergeSort y QuickSort

Curso de Estructuras de Datos

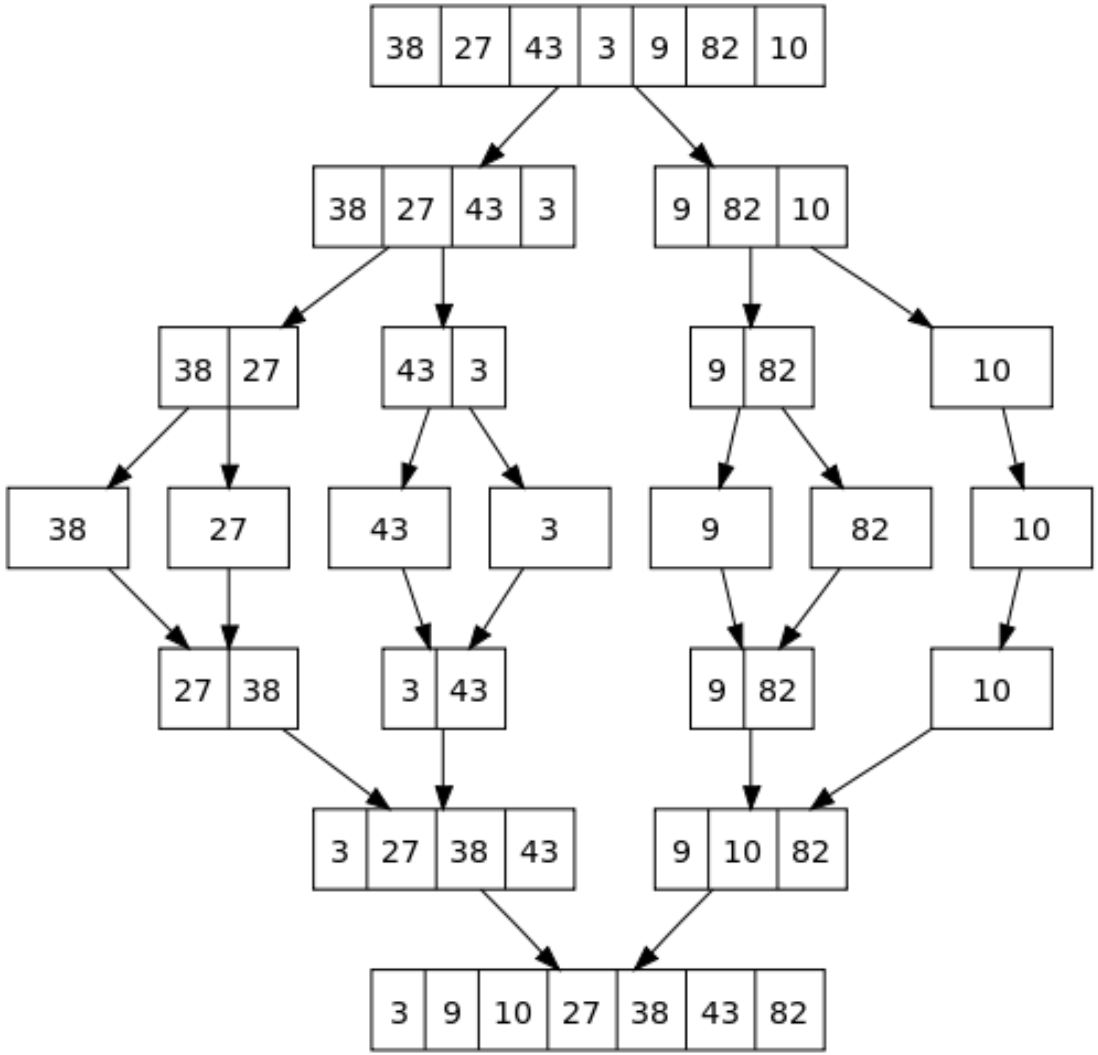
Prof. Luis E. Garreta U.

lgarreta@uao.edu.co

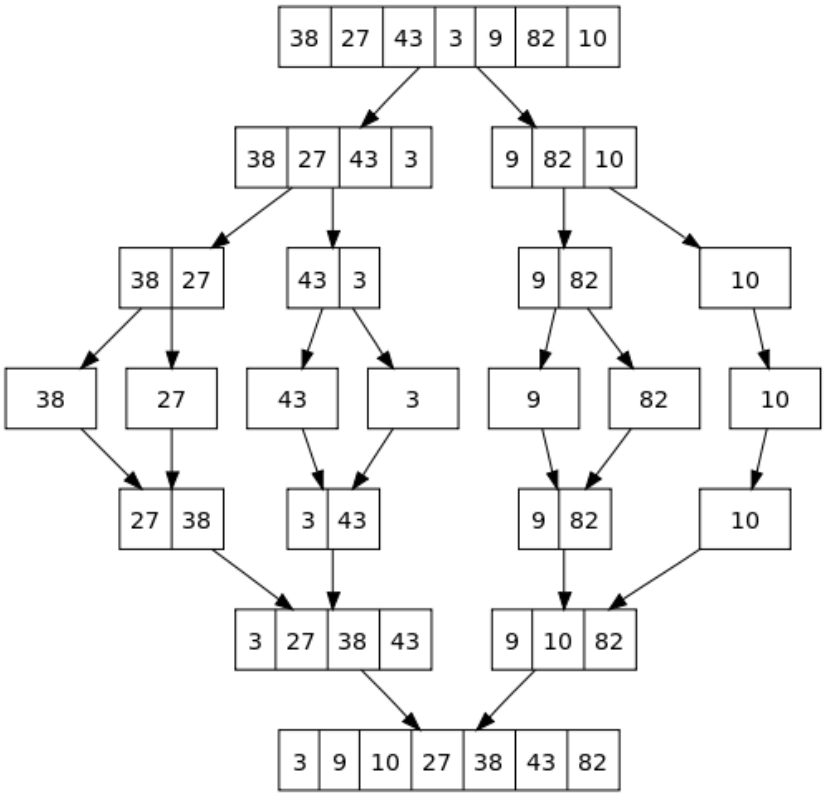
Universidad Autonoma de Occidente – Cali
Depto. Operaciones y Sistemas
Facultad de Ingeniería

24 de marzo de 2018

Ordenamiento Merge Sort



Mergesort Dividir y Vencerás



DIVIDIR: Dividir el arreglo a ordenar de n elementos en dos arreglos de tamaño n/2

CONQUISTAR: Ordenar recursivamente los dos subarreglos usiando Mergesort

COMBINAR: Mezclar los dos subarreglos ordenados de tal manera que se obtiene otro arreglo ordenado

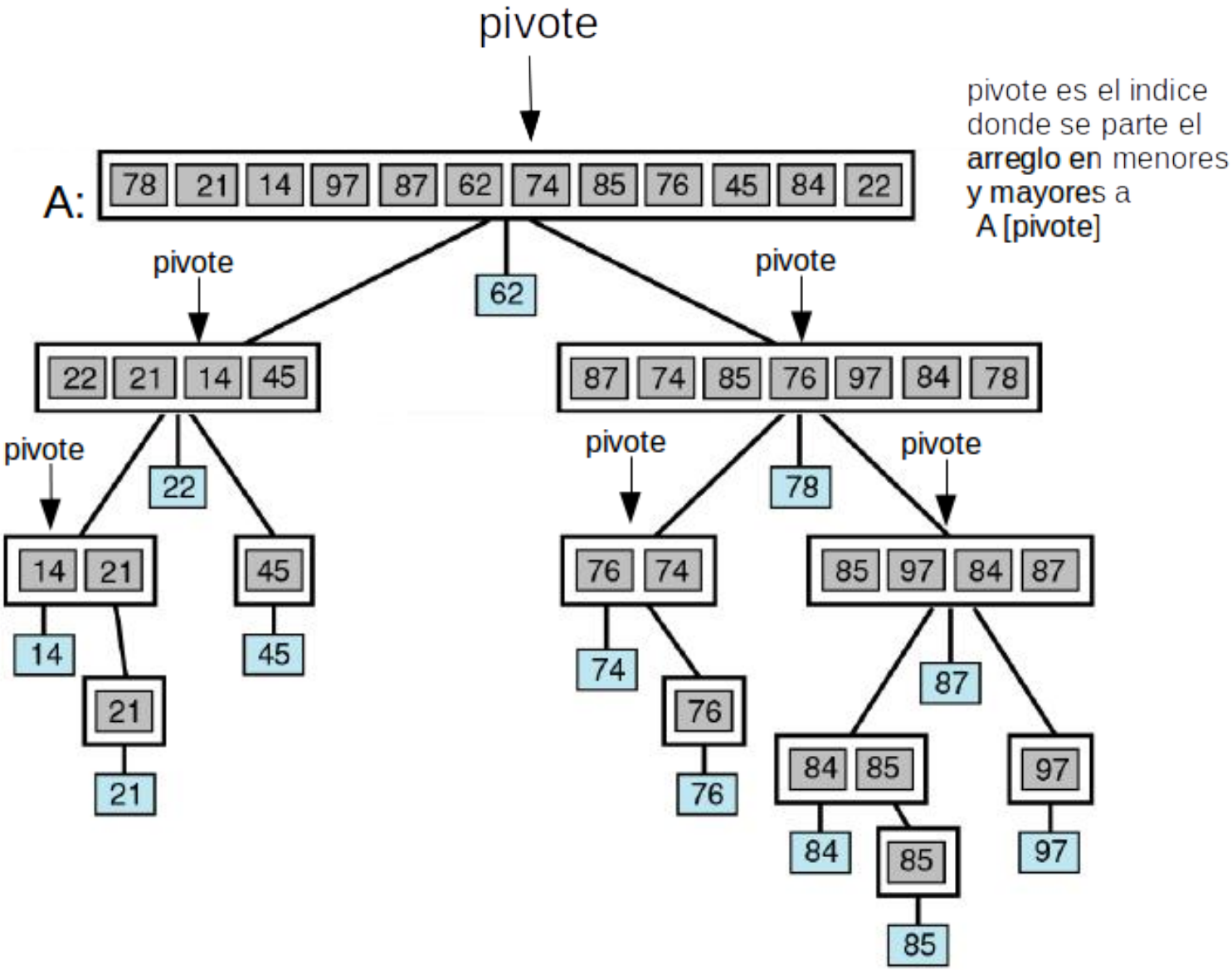
Implementación MergeSort

```
static void mergeSort(int[] datos, int ini, int fin) {
    if (ini < fin) {
        int mitad = (ini + fin) / 2;
        mergeSort(datos, ini, mitad);
        mergeSort(datos, mitad + 1, fin);
        mezclar (datos, ini, mitad, fin);
    }
}
```

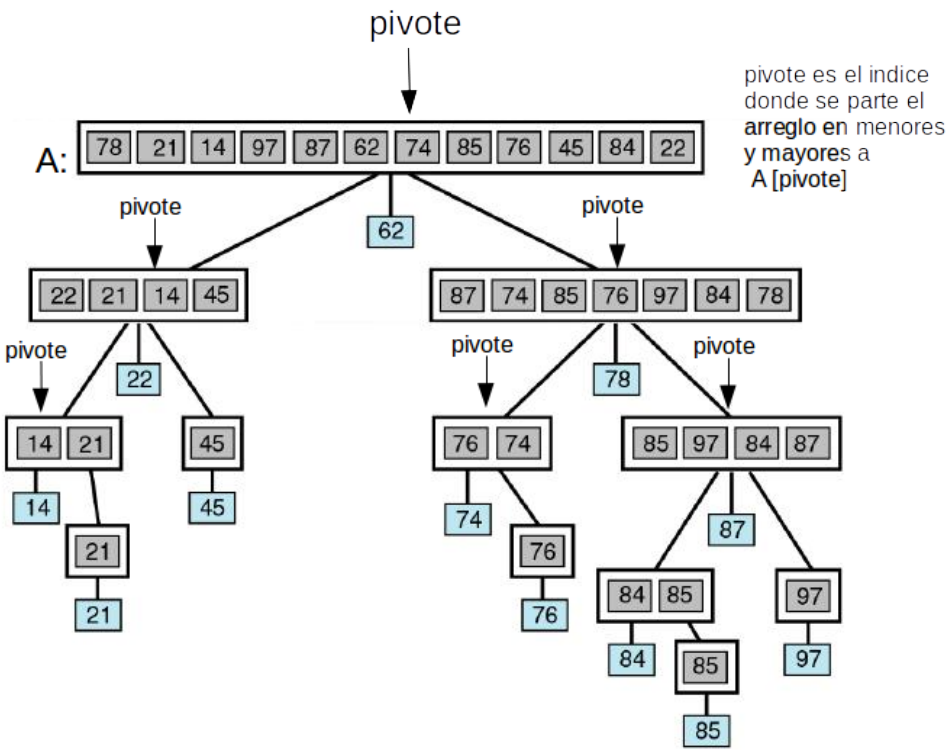
Implementación Función Mezclar

```
static void mezclar(int[] datos, int ini, int mitad, int fin) {
    int[] temporal = new int[datos.length];
    for (int i = ini; i <= fin; i++) {
        temporal[i] = datos[i];
    }

    int temporalIzquierda = ini;
    int temporalDerecha = mitad + 1;
    int actual = ini;

    while (temporalIzquierda <= mitad && temporalDerecha <= fin) {
        if (temporal[temporalIzquierda] <= temporal[temporalDerecha]) {
            datos[actual] = temporal[temporalIzquierda];
            temporalIzquierda++;

        } else {
            datos[actual] = temporal[temporalDerecha];
            temporalDerecha++;
        }
        actual++;
    }

    int resto = mitad - temporalIzquierda;
    for (int i = 0; i <= resto; i++) {
        datos[actual + i] = temporal[temporalIzquierda + i];
    }
}
```

Ordenamiento QuickSort



Quicksort Dividir y Vencerás



DIVIDIR y COMBINAR: organizar el arreglo alrededor de una posición *i (pivote)* con menores a la izquierda y mayores a a la derecha del pivote **CONQUISTAR:** Ordenar recursivamente los dos subarreglos: lado izquierdo y lado derecho usando Quicksort

Implementación Quicksort

```
static void quickSort(int datos[], int ini, int fin)
{
    if (ini < fin)  {
        /* pivote es el indice donde se parte el arreglo
        en menores y mayores a pivote */
        int pivote = particionar(datos, ini, fin);
        quickSort(datos, ini, pivote-1);
        quickSort(datos, pivote+1, fin);
    }
}
```

Implementacion función particionar

```
static int particionar (int arr[], int ini, int fin)     {
    int pivote = arr[fin];
    int i = (ini-1); // index of smaller element
    for (int j=ini; j<fin; j++)
    {
        // If current element is smaller than or
        // equal to pivote
        if (arr[j] <= pivote)
        {
            i++;

            // swap arr[i] and arr[j]
            int temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
        }
    }
```