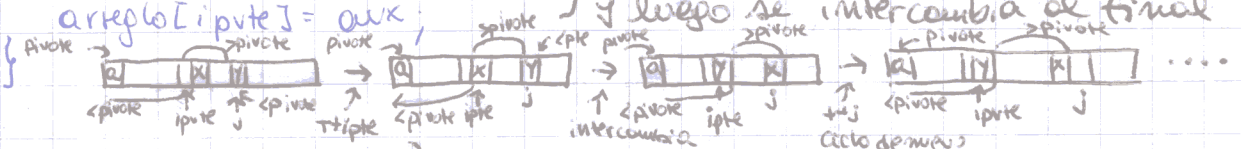
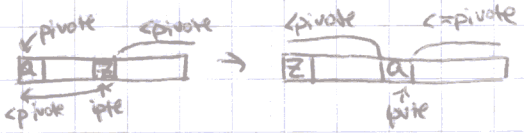
```
void QuickSort(String[] arreglo, int ip, int iu){
    if(ip >= iu) ← si tiene 1 elemento, ya está ordenado.
        return;
    int i = particionar(arreglo, ip, iu); ← coloca los > pivote a la derecha, los
    QuickSort(arreglo, ip, i-1);            < pivote a la izquierda y retorna el
    QuickSort(arreglo, i+1, iu);            índice del pivote
}   ↑ repetir c/elementos > pivote.   ↑ repetir con los elementos < pivote
int particionar(String[] arreglo, int ip, int iu){
    String pivote = arreglo[ip]; ← toma como pivote el primero.
    int ipvte = ip; ← índice inicial del pivote.
    for(int j = ip+1; j <= iu; ++j){ ← (ip+1) pq' comienza c/el de la der del pvte
        if(arreglo[j].compareTo(pivote) < 0){ ← si es menor que el pivote
            ++ipvte;                    ipte es el índice que indica que antes
            String aux = arreglo[j];    de él se colocarán todos los < pivote
            arreglo[j] = arreglo[ipvte]; y después de él todos los ≥ pivote
            arreglo[ipvte] = aux;       y luego se intercambia al final
        }
    }
    String aux2 = arreglo[ip];
    arreglo[ip] = arreglo[ipvte];
    arreglo[ipvte] = aux2;
    return ipvte;
}
```





Intercambio de 2 objetos de un arreglo sin variable auxiliar

- Para int y double

```
arreglo[i] += arreglo[j];
arreglo[j] = arreglo[i] - arreglo[j];
arreglo[i] -= arreglo[j];
```

- Para Strings

```
arreglo[i] = arreglo[i].concat(arreglo[j]);
arreglo[j] = arreglo[i].substring(0, arreglo[i].indexOf(arreglo[j]));
arreglo[i] = arreglo[i].substring(arreglo[j].length());
```

Solo sirve si una palabra no está contenida en otra

Algoritmo