Ordenación rápida (QuickSort)

1. Se toma un elemento arbitrario del vector, al que denominaremos pivote (p).

2. Se divide el vector de tal forma que todos los elementos a la izquierda del pivote sean menores que él, mientras que los que quedan a la derecha son mayores que él.

3. Ordenamos, por separado, las dos zonas delimitadas por el pivote.

```
void quicksort (double v[], int izda, int dcha)
{
  int pivote;  // Posición del pivote

  if (izda<dcha) {

     pivote = partir (v, izda, dcha);

     quicksort (v, izda, pivote-1);

     quicksort (v, pivote+1, dcha);
  }
}
```

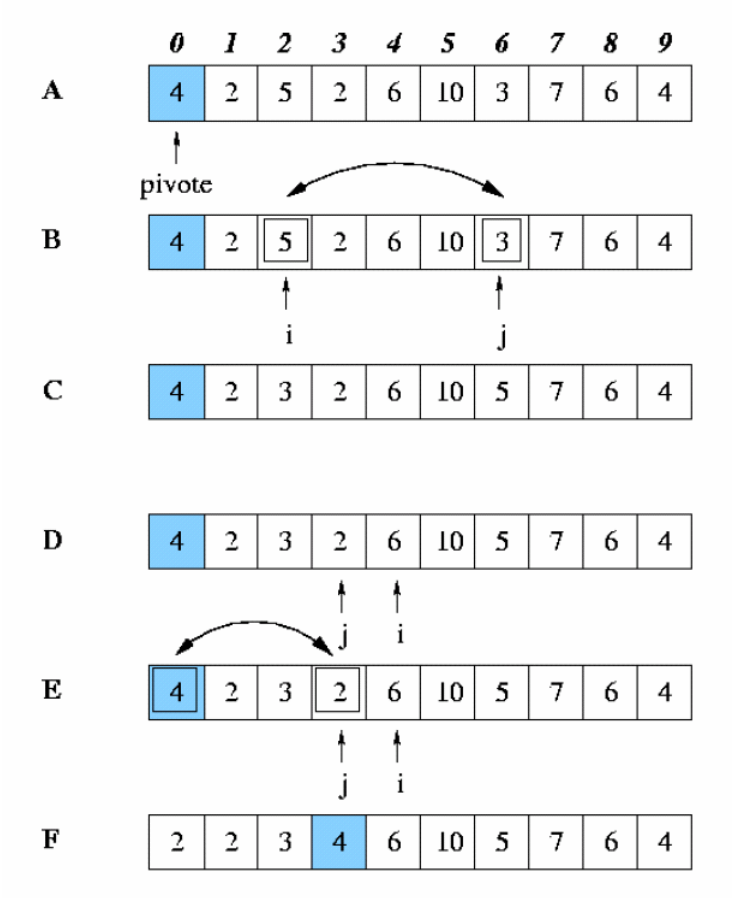
Obtención del pivote

Mientras queden elementos mal colocados respecto al pivote:

a. Se recorre el vector, de izquierda a derecha, hasta encontrar un elemento situado en una posición i tal que v[i] > p.

b. Se recorre el vector, de derecha a izquierda, hasta encontrar otro elemento situado en una posición j tal que v[j] < p.

c. Se intercambian los elementos situados en las casillas i y j (de modo que, ahora, v[i] < p < v[j]).



```
// Intercambio de dos valores

void swap (double *a, double *b)
{
  double tmp;

  tmp = *a;
  *a = *b;
  *b = tmp;
}



// División el vector en dos partes
// - Devuelve la posición del pivote

int partir (double v[], int primero, int ultimo)
{
  double pivote = v[primero]; // Valor del pivote
  int izda = primero+1;
  int dcha = ultimo;

  do {  // Pivotear…

    while ((izda<=dcha) && (v[izda]<=pivote))
          izda++;

    while ((izda<=dcha) && (v[dcha]>pivote))
          dcha--;

    if (izda < dcha) {
       swap ( &(v[izda]), &(v[dcha]) );
       dcha--;
       izda++;
    }

  } while (izda <= dcha);

  // Colocar el pivote en su sitio
  swap (&(v[primero]), &(v[dcha]) );

  return dcha; // Posición del pivote
}
```