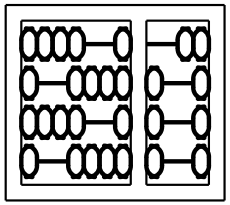


Universidade Estadual de Campinas - UNICAMP
Instituto de Computação - IC

Ordenação recursiva

Tópicos

1. Ordenação por *Quicksort*

- O algoritmo de ordenação Quicksort:

– Notação:

1. $v[h..j] \leq x \longrightarrow v[i] \leq x$ para todo i no conjunto de índices $h..j$
2. $v[h..j] \leq v[k..m] \longrightarrow v[i] \leq v[l]$ para todo i no conjunto $h..j$ e todo l no conjunto $k..m$.

– A base do Quicksort é representada pelo *algoritmo da separação* que, informalmente, consiste em rearranjar um vetor $v[p..r]$ de modo que, em relação a um certo *pivô*, os elementos menores fiquem todos do lado esquerdo e os maiores fiquem todos do lado direito.

Assim, para um pivô na posição j:
$v[p..j-1] \leq v[j] < v[j+1..r]$

- O algoritmo de ordenação Quicksort:

– Notação:

1. $v[h..j] \leq x \longrightarrow v[i] \leq x$ para todo i no conjunto de índices $h..j$
2. $v[h..j] \leq v[k..m] \longrightarrow v[i] \leq v[l]$ para todo i no conjunto $h..j$ e todo l no conjunto $k..m$.

– A base do Quicksort é representada pelo *algoritmo da separação* que, informalmente, consiste em rearranjar um vetor $v[p..r]$ de modo que, em relação a um certo *pivô*, os elementos menores fiquem todos do lado esquerdo e os maiores fiquem todos do lado direito.

Assim, para um pivô na posição j:
$v[p..j-1] \leq v[j] < v[j+1..r]$

- O algoritmo de ordenação Quicksort:

– Notação:

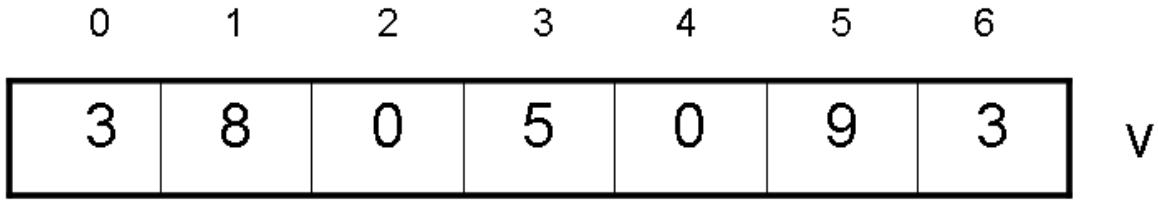
1. $v[h..j] \leq x \longrightarrow v[i] \leq x$ para todo i no conjunto de índices $h..j$
2. $v[h..j] \leq v[k..m] \longrightarrow v[i] \leq v[l]$ para todo i no conjunto $h..j$ e todo l no conjunto $k..m$.

– A base do Quicksort é representada pelo *algoritmo da separação* que, informalmente, consiste em rearranjar um vetor $v[p..r]$ de modo que, em relação a um certo *pivô*, os elementos menores fiquem todos do lado esquerdo e os maiores fiquem todos do lado direito.

Assim, para um pivô na posição j:
$v[p..j-1] \leq v[j] < v[j+1..r]$

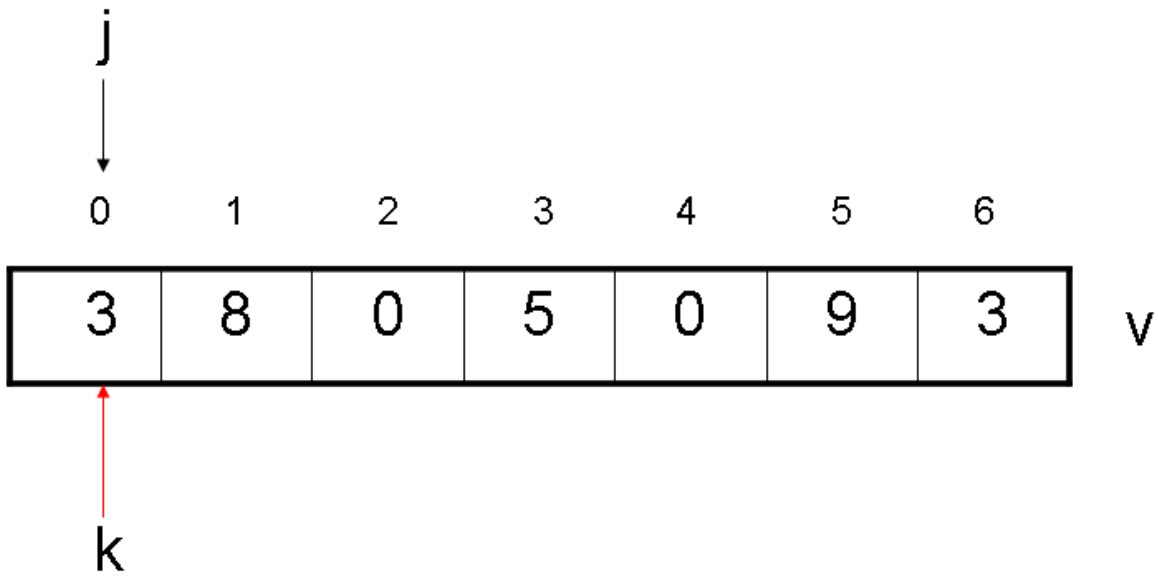
- Exemplo: algoritmo da separação

Para k $<$ 6:



- Exemplo: algoritmo da separação

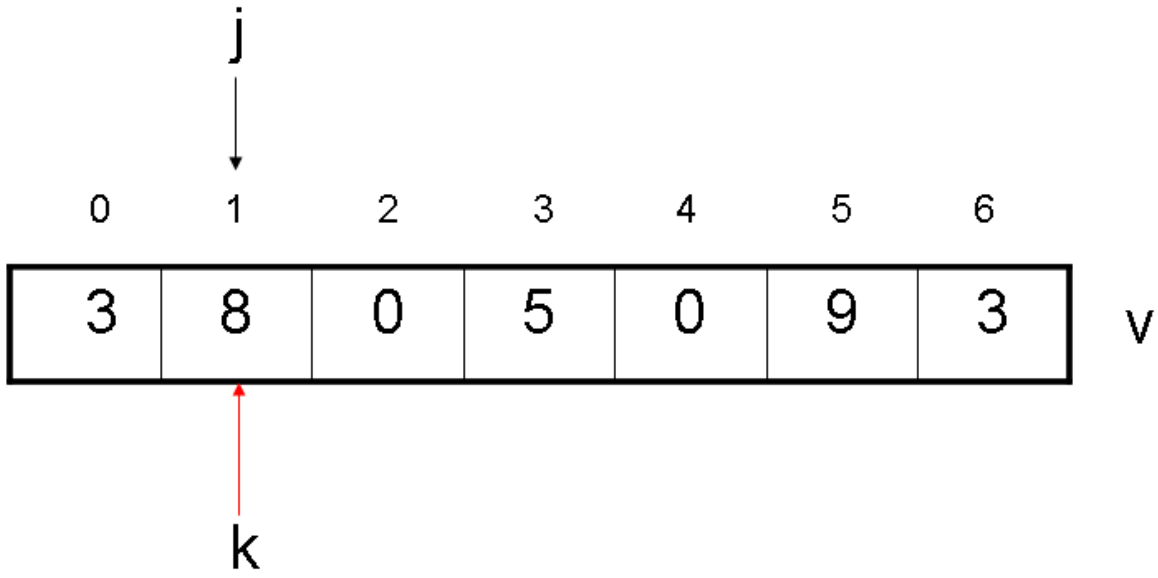
Para k $<$ 6:



-- troca v[j] com v[k] se v[k] <= v[6]
-- neste caso, avança k e j
-- senão, avança k

- Exemplo: algoritmo da separação

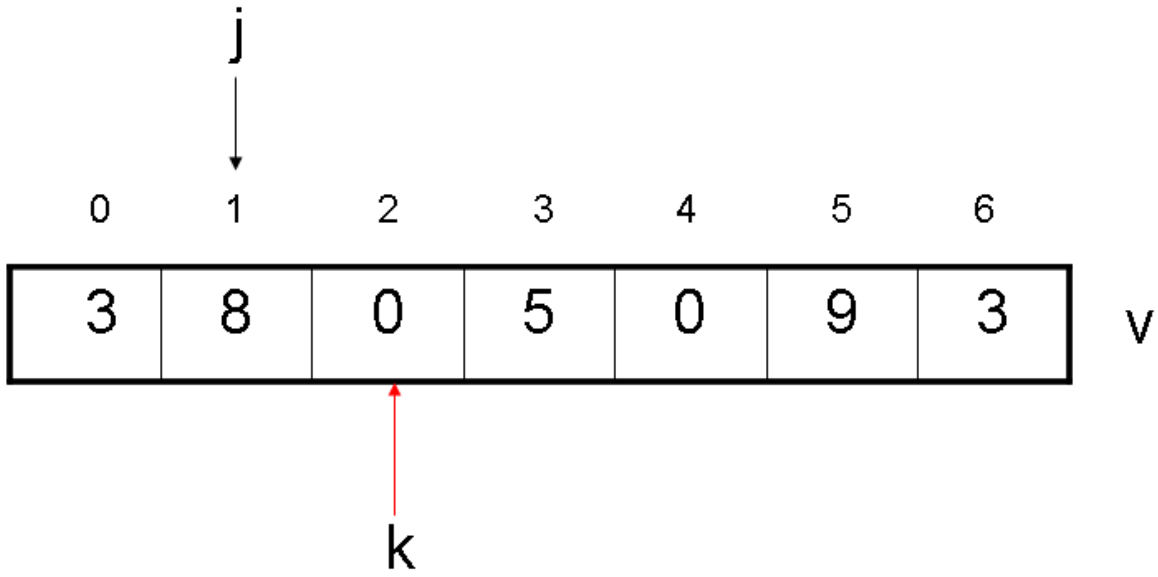
Para k $<$ 6:



-- troca v[j] com v[k] se v[k] <= v[6]
-- neste caso, avança k e j
-- senão, avança k

- Exemplo: algoritmo da separação

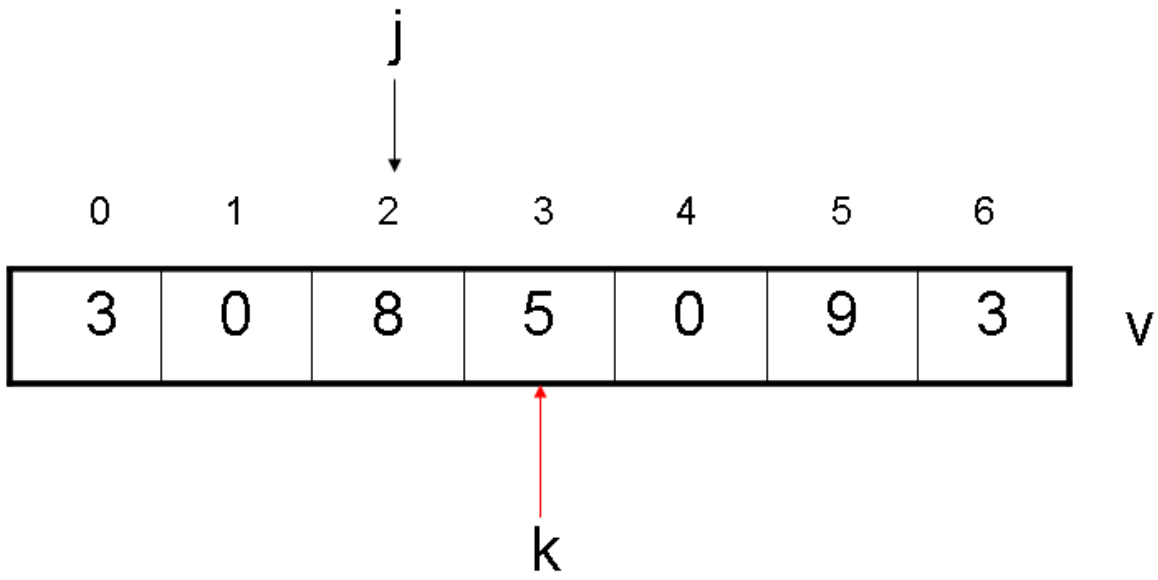
Para k $<$ 6:



-- troca v[j] com v[k] se v[k] <= v[6]
-- neste caso, avança k e j
-- senão, avança k

- Exemplo: algoritmo da separação

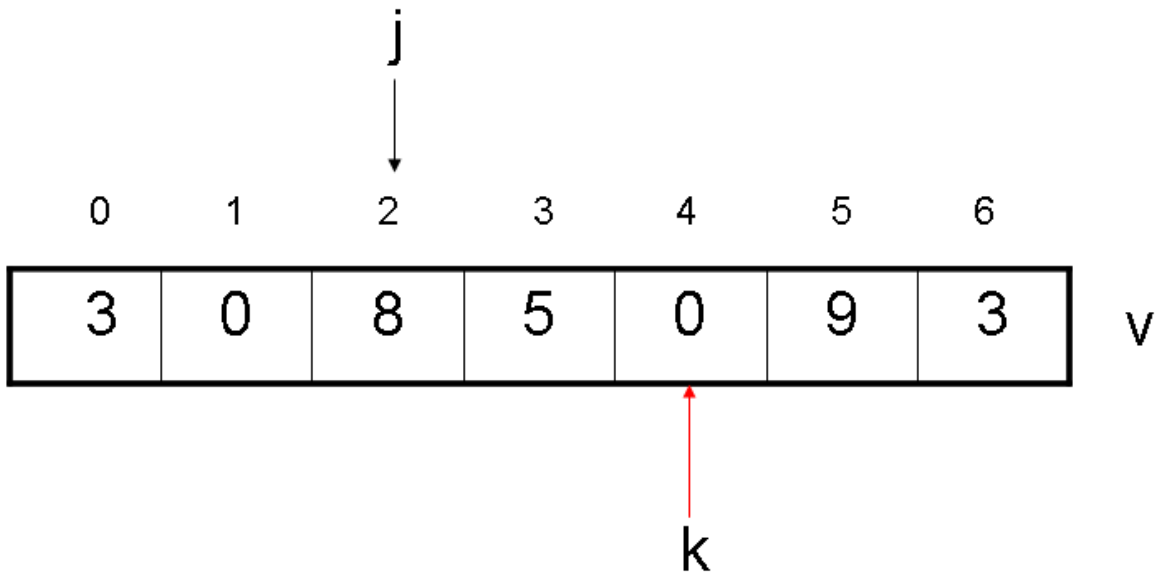
Para k $<$ 6:



-- troca v[j] com v[k] se v[k] <= v[6]
-- neste caso, avança k e j
-- senão, avança k

- Exemplo: algoritmo da separação

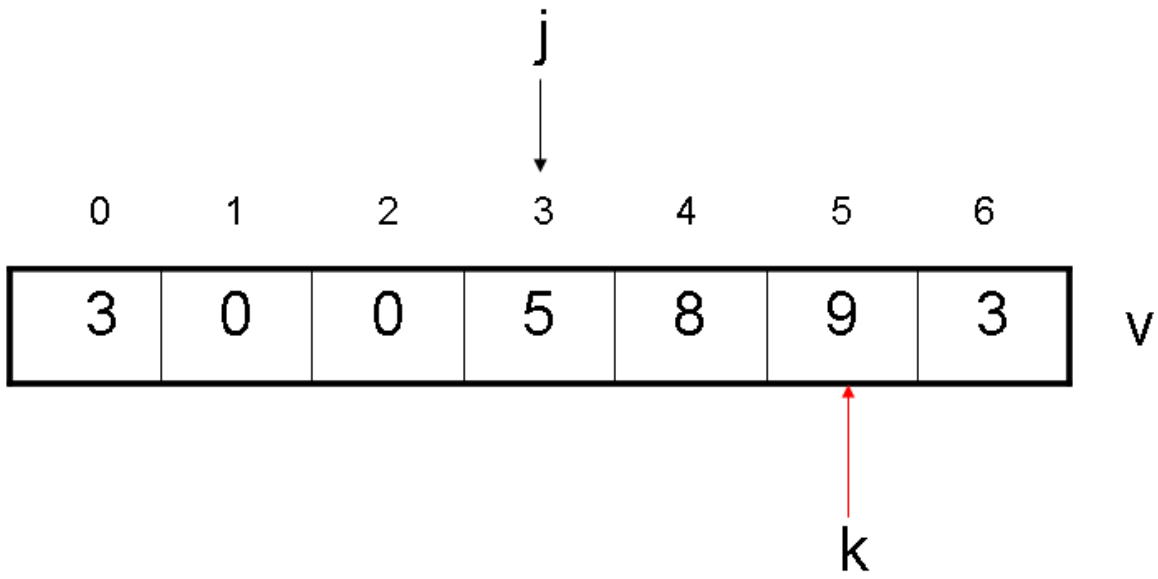
Para k $<$ 6:



-- troca v[j] com v[k] se v[k] <= v[6]
-- neste caso, avança k e j
-- senão, avança k

- Exemplo: algoritmo da separação

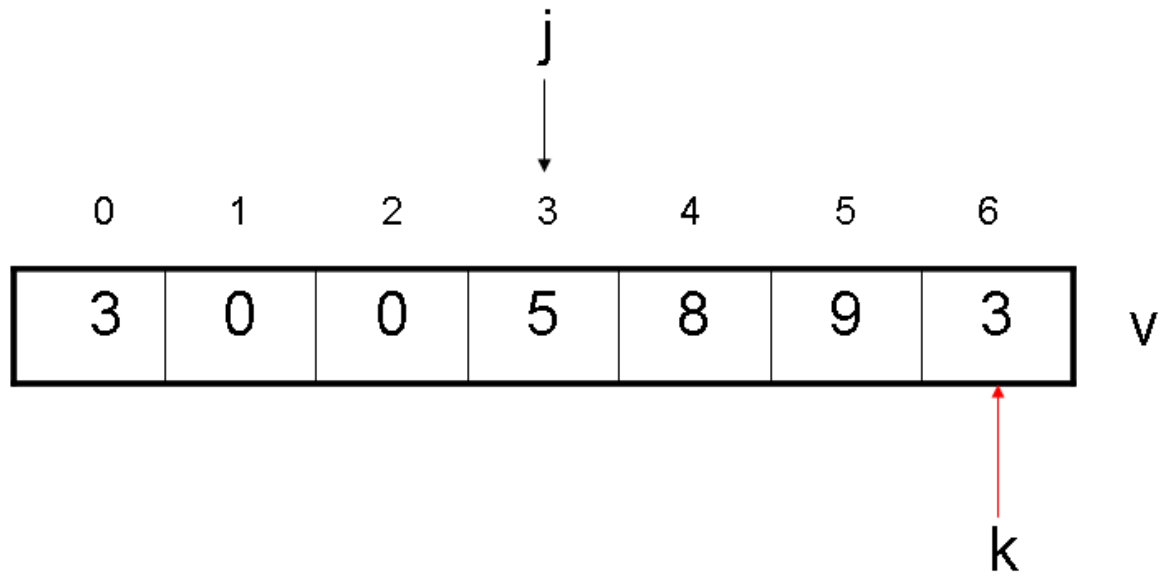
Para k $<$ 6:



-- troca v[j] com v[k] se v[k] <= v[6]
-- neste caso, avança k e j
-- senão, avança k

- Exemplo: algoritmo da separação

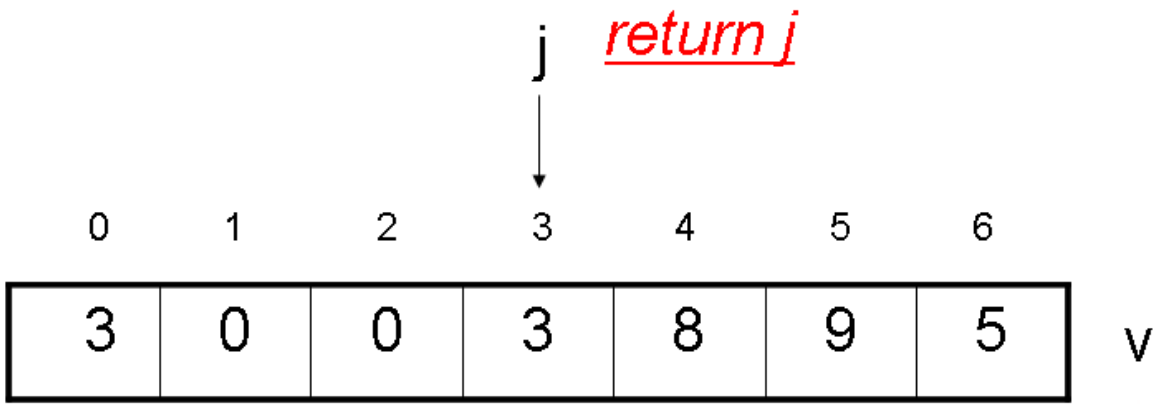
Para k == 6:



-- troca v[j] com v[k]

- Exemplo: algoritmo da separação

Para k == 6:



- Programa:

```
/* Recebe um vetor v[p..r] com p<=r. Rearranja os
elementos do vetor e devolve o pivo em algum j tal
que v[p..j-1] <= v[j] < v[j+1..r]. */

int Separa(int p, int r, int v[]) {
  int c, j, k, t ;
  c = v[r]; j = p ;
  for (k = p; k < r; k++)
    if (v[k] <= c) {
      t = v[j]; v[j] = v[k]; v[k]=t ;   /* troca */
      j++ ;
    }
  v[r] = v[j] ; v[j] = c ;
  return j ;       /* retorna posicao do pivo */
}
```

- Algoritmo do Quicksort:

```
/* rearranja o vetor v[p..r] em ordem crescente */

void Quicksort(int p, int r, int v[]) {
  int j ;
  if(p < r) {
    j = Separa (p, r, v);  /* v[j] esta na posicao correta */
    Quicksort (p, j-1, v); /* ordena lado esquerdo */
    Quicksort (j+1, r, v); /* ordena lado direito */
  }
}
```

Referências:

– Paulo Feofiloff. Algoritmos em linguagem C, Elsevier 2009.