ALGORITMOS AVANÇADOS

UNIDADE IV – Algoritmo de Ordenação Rápida (*Quicksort*)

Luiz Leão – luizleao@gmail.com
http://www.luizleao.com

Conteúdo Programático

- 4.1 - Definição
- 4.2 - Ordenação rápida
- 4.3 - O algoritmo de ordenação rápida *Quicksort*
- 4.4 - Análise da complexidade do algoritmo *Quicksort*

Definição

- Inventado por Charles Antony Richard Hoare em 1960;
- Utiliza a estratégia de divisão e conquista, onde um grande problema é dividido em problemas menores que são resolvidos com recursividade;
- Consiste na ordenação de sub-listas divididas por um pivô (elementos menores que o pivô são colocados à sua esquerda e elementos maiores a sua direita);
- É o mais rápido algoritmo de ordenação não paralelo (a ling. c oferece uma função – qsort() – na bib. stdlib.h

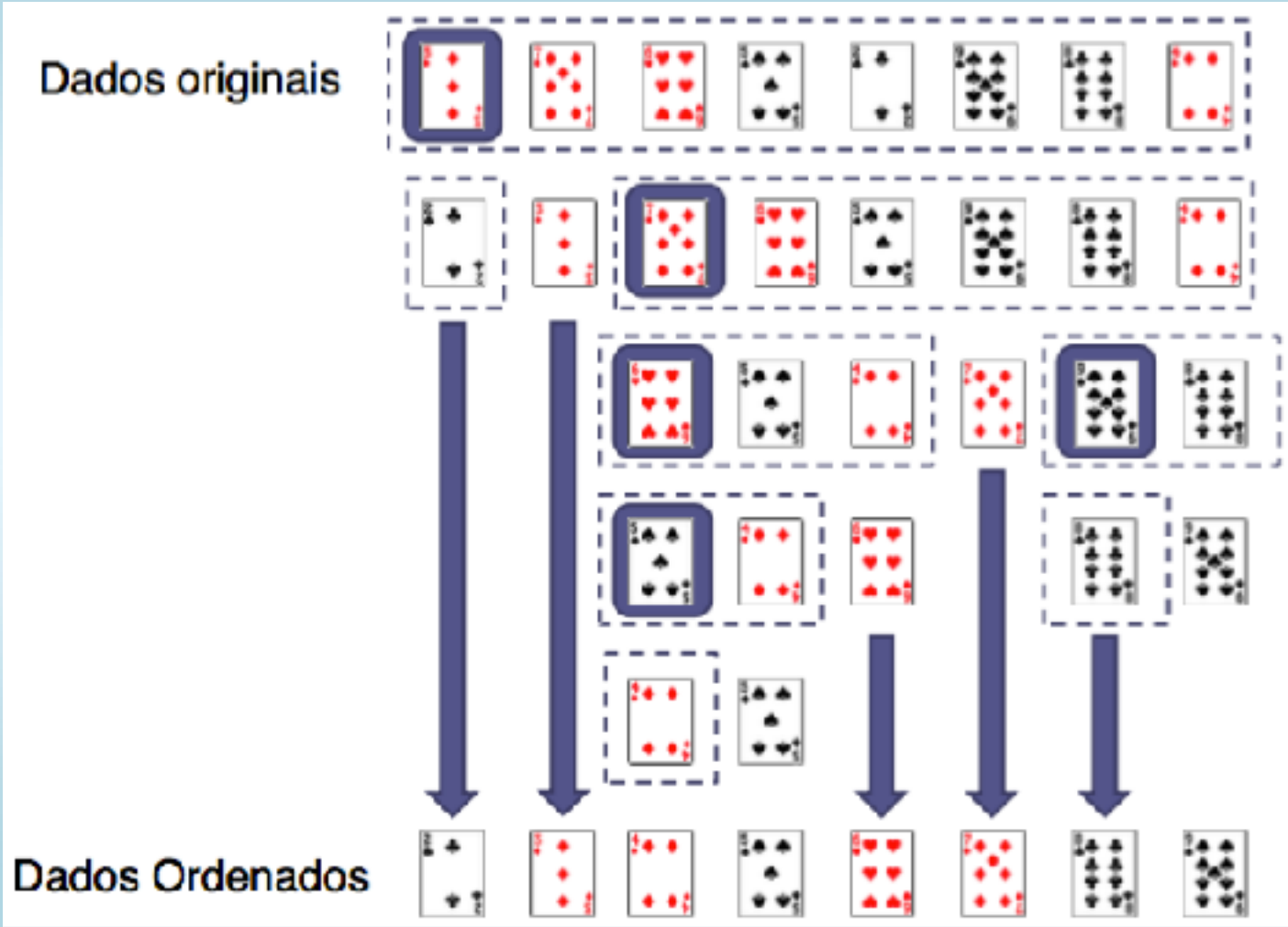
Algoritmo

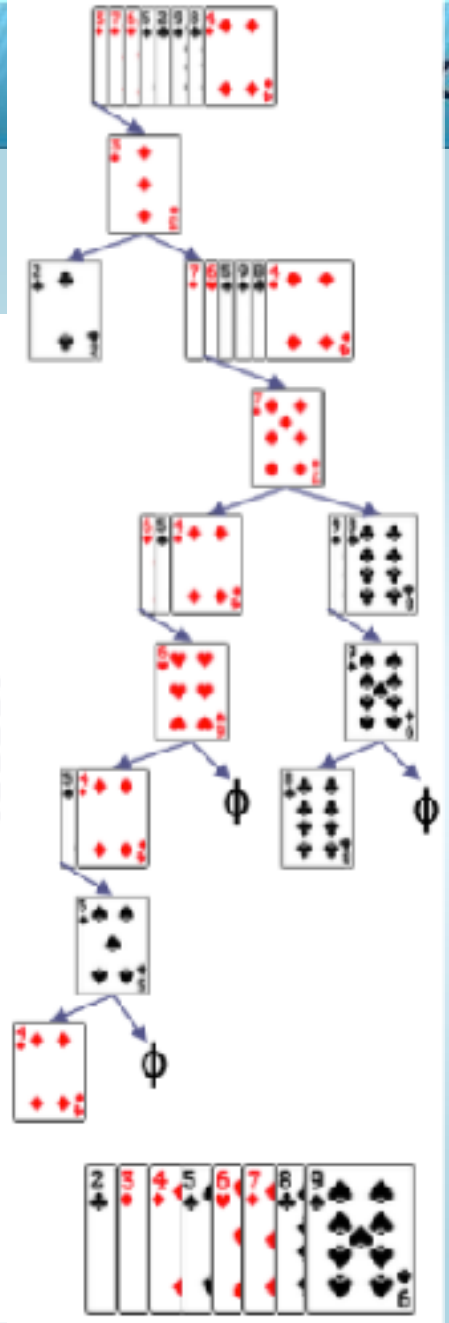
- Se o número de elementos a ordenar for zero ou um então termine (não há o que ordenar);

- Escolha um elemento do conjunto para ser o pivô (geralmente o primeiro elemento);
  - **Organize os elementos de acordo com o pivô (os elementos menores ficam a esquerda e os maiores ficam a direita do pivô)**

- Chame recursivamente o algoritmo para ordenar o grupo de elementos a direita e a esquerda do pivô.

Quick Sort - Simulação





Algoritmo - *Quick Sort*

```
void quickSort(int vet[], int esquerda, int direita){
  int pivo, j;
  if (direita > esquerda) {
     pivo = esquerda;
     for (j = esquerda + 1; j <= direita; j++) {
       if (vet[j] < vet[esquerda]) {
          pivo++;
          troca(&vet[pivo], &vet[j]);
       }
     }
     troca(&vet[esquerda], &vet[pivo]);
     quickSort(vet, esquerda, pivo - 1);
     quickSort(vet, pivo + 1, direita);
  }
}
```

Análise da complexidade do algoritmo *Quicksort*

- Pior Caso: $O(n^2)$
- Médio Caso: Θ (nlogn)
- Melhor Caso: Ω (nlogn)

Exercício 01

- Implemente uma subrotina para ordenar uma lista de palavras armazenadas em um vetor utilizando o algoritmo *QuickSort.* Após a implementação do algoritmo, implementar as seguintes funcionalidades:
  a) Desenvolva a geração aleatória dos elementos do vetor, apenas passando como parâmetro a quantidade de elementos.
  b) Efetue a contagem do tempo de processamento da atividade de ordenação
  c) Implemente a busca de elementos no vetor, retornando uma msg informando que o elemento foi encontrado e a sua posição, ou, que o elemento não foi encontrado
  d) Implemente a eliminação dos elementos repetidos do vetor