Estrutura de Dados

Métodos básicos de ordenação

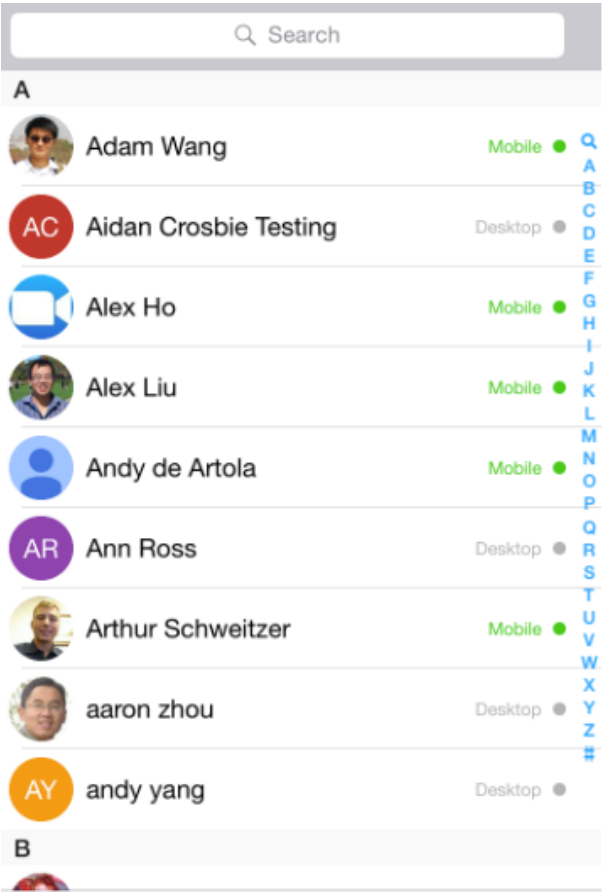
Prof. Dr. Daniel Vecchiato

Agenda

- Introdução
- Bubble sort
- Selection sort
- Insertion sort
- Exercícios

Introdução

- Ordenação
  - Rearranjar um conjunto de objetos em uma ordem ascendente ou descendente
    - Já imaginou tentar achar o telefone de uma pessoa em uma lista que não esteja ordenada?
      - A ordenação visa facilitar a recuperação posterior de itens do conjunto ordenado



Introdução

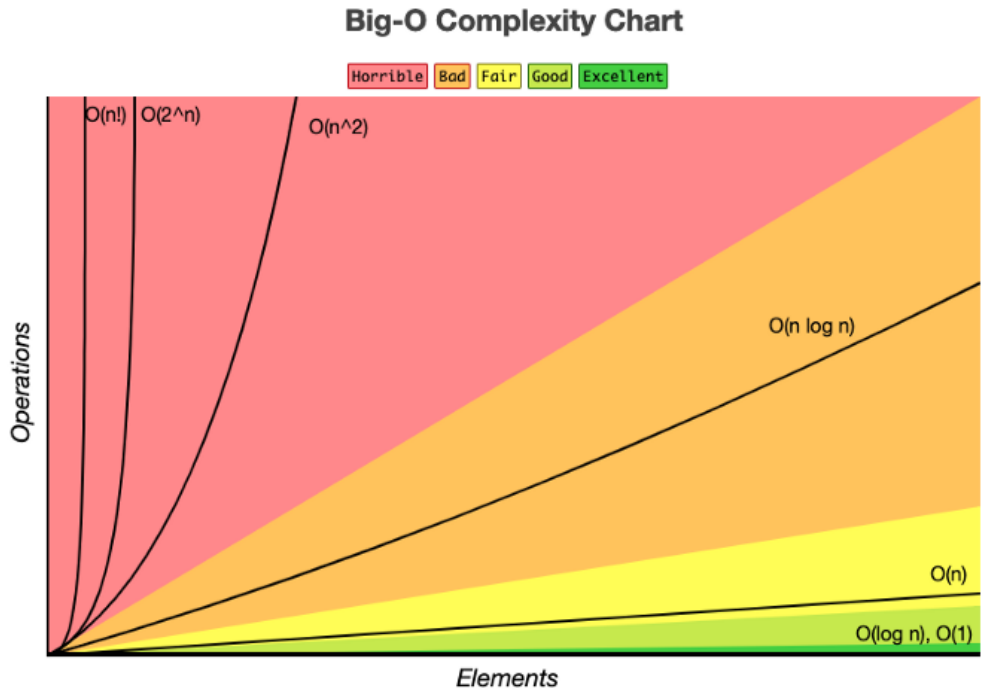
- Em programação, um método de ordenação coloca os elementos de uma dada sequência em uma certa ordem
- As ordens mais usadas são
  - Numérica
  - Lexicográfica (ordem alfabética)
- Qualquer tipo de chave que possui uma regra de ordenação pode ser utilizado
- A ordenação de registros ocorrerá baseado em um campo de chave específica

Introdução

- Um método de ordenação é estável se a ordem relativa dos itens com chaves iguais não se altera durante a ordenação
- Os métodos podem ser classificados como
  - Ordenação interna: os dados cabem todos na memória principal
  - Ordenação externa: é necessário acesso à memória secundária

Introdução

- Análise de algoritmos de ordenação
  - Medidas relevantes
    - número de comparações: C(n)
    - número de movimentações (trocas): M(n)
- Métodos simples
  - Requerem $O(n^2)$ comparações
- Métodos eficientes
  - Requerem O(n log n) comparações



Introdução

- O algoritmo de ordenação ideal deve possuir essas propriedades:
  - Estável
  - Custo de comparação O(n log n) no pior caso
  - Custo de movimentação O(n) no pior caso
  - O(n) quando os dados estão praticamente ordenados ou quando há poucas chaves únicas
- Não há nenhum algoritmo com todas essas propriedades.
  - Deve ser analisado o problema e aplicado o melhor método para o caso específico

Bubble sort

- Também chamado de Método da bolha ou flutuação
- A cada iteração do método um elemento se aproxima até a sua posição correta
- Os elementos são trocados frequentemente, o que gera um alto custo

Bubble sort

- Comportamento em diferentes situações
  - https://www.toptal.com/developers/sorting-algorithms/bubble-sort
- Custo
  - Comparações e trocas: $O(n^2)$
  - O(n) quando os dados estão quase ordenados
- Simples e estável
- Um conjunto já ordenado ainda necessitará de várias comparações

https://commons.wikimedia.org/wiki/File:Bubble-sort-example-300px.gif

Bubble sort

```
typedef struct {
	int chave;
} TItem;

void bubbleSort (TItem *v, int n) {
	int i, j;
	TItem aux;
	for (i = 0; i < n-1; i++) {
		for (j = 1; j < n-i; j++) {
			if (v[j].chave < v[j-1].chave) {
				aux = v[j];
				v[j] = v[j-1];
				v[j-1] = aux;
			}
		}
	}
}
```

Bubble sort

- Melhoria

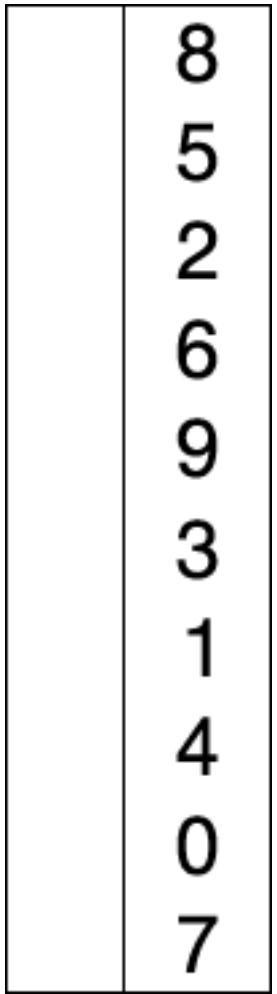
```
void bubbleSort2 (TItem *v, int n) {
        int i, j, troca;
        TItem aux;

        for (i = 0; i < n-1; i++) {
                troca = 0;
                for (j = 1; j < n-i; j++) {
                        if (v[j].chave < v[j-1].chave) {
                                aux = v[j];
                                v[j] = v[j-1];
                                v[j-1] = aux;
                                troca++;
                        }
                }

                if (troca == 0)
                        break;
        }
}
```

Selection sort

- Também chamado de Método de seleção
- Em qual posição o maior elemento do vetor deve ficar?
  - Na última posição
- E o menor valor?
  - Na primeira posição
- Funcionamento
  - Seleciona o n-ésimo menor elemento da lista
  - Troca do n-ésimo menor com a n-ésima posição da lista

Selection sort

- Comportamento em diferentes situações
  - https://www.toptal.com/developers/sorting-algorithms/selection-sort
- Custo
  - Comparações: $O(n^2)$
  - Trocas: O(n)
- Não é estável
- Um dos mais rápidos quando a lista é pequena



https://commons.wikimedia.org/wiki/File:Selection-Sort-Animation.gif

Selection sort

```
void selectionSort (TItem *v, int n) {
	int i, j, min;
	TItem aux;

	for (i = 0; i < n-1; i++) {
		min = i;
		for (j = i+1; j < n; j++) {
			if (v[j].chave < v[min].chave) {
				min = j;
			}
		}
		if (i != min) {
			aux = v[min];
			v[min] = v[i];
			v[i] = aux;
		}
	}
}
```

Insertion sort

- Também chamado de Método de inserção
- O seu funcionamento pode ser comparado com a ordenação de cartas na mão de um computador
  - As cartas são ordenadas da esquerda para a direta, uma de cada vez
  - É selecionado a 2ª carta verificando se ela deve ficar antes ou na posição que está
  - Depois a 3ª carta é classificada, deslocando-se até sua posição correta
  - Repetir esses procedimentos até a última carta da direta

Insertion sort

- Comportamento em diferentes situações
  - https://www.toptal.com/developers/sorting-algorithms/insertion-sort
- Custo
  - Comparações e trocas: $O(n^2)$
  - O(n) quando os dados estão quase ordenados
- Estável
- Aconselhado quando:
  - Dados já podem estar ordenados
  - O conjunto de dados é pequeno

6 5 3 1 8 7 2 4

https://commons.wikimedia.org/wiki/File:Insertion-sort-example.gif

Insertion sort

```
void insertionSort (TItem *v, int n) {
	int i, j;
	TItem aux;

	for (i = 1; i < n; i++) {
		aux = v[i];
		j = i - 1;

		while (j >= 0 && aux.chave < v[j].chave) {
			v[j+1] = v[j];
			j--;
		}

		v[j+1] = aux;
	}
}
```

Comparação

Algoritmo	Comparações			Movimentações		
	Melhor	**Médio**	**Pior**	**Melhor**	**Médio**	**Pior**
Bubble	$O(n^2)$			$O(n^2)$		
Selection	$O(n^2)$			$O(n)$		
Insertion	$O(n)$	$O(n^2)$		$O(n)$	$O(n^2)$	

Exercícios

- Implemente os três métodos de ordenação
- Imprima quantas comparações e movimentações foram realizadas em cada método nas seguintes situações
  - Dados aleatórios
  - Dados já ordenados
  - Dados ordenados decrescentemente

Estrutura de Dados

Material elaborado por:
Thiago Meirelles Ventura

Baseado em:

- Ascencio, A. F. G; Araújo, G. S. Estruturas de Dados. Pearson, 2011.
- Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; Stein, C. Algoritmos: teoria e prática. Elsevier, 2002.
- Aulas do Prof. Reinaldo Silva Fortes (http://www.decom.ufop.br/reinaldo/)