

TECNOLOGIA EM INFORMÁTICA

Estrutura de Dados

Prof. Glauco da Silva
contato: glaucoslv@gmail.com

Roteiro

- *Objetivos*
- *Arranjos ordenados*
- *Ordenamento em arranjos*
- *Arranjos bidimensionais (matrizes)*
- *Exercícios*

Roteiro

- ***Objetivos***
- *Arranjos ordenados*
- *Ordenamento em arranjos*
- *Arranjos bidimensionais (matrizes)*
- *Exercícios*

Objetivos

- *Estudar e trabalhar com arranjos ordenados, explicando as vantagens de sua utilização*
- *Trabalhar com ordenação de vetores*
- *Apresentar arranjos bidimensionais, mostrando sua aplicação e suas funcionalidades*

Roteiro

- *Objetivos*
- ***Arranjos ordenados***
- *Ordenamento em arranjos*
- *Arranjos bidimensionais (matrizes)*
- *Exercícios*

Arranjos ordenados

- *Armazenar dados de forma ordenada em um vetor*
Ex: Resultados de jogos, Pessoas mais velhas, Quantidade de anos trabalhados, etc.
- *Geralmente, os dados contém mais de uma informação, por esse motivo é necessário armazenar um objeto dentro do vetor.*

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*
 - ➔ *É necessário a criação de duas classes:*
 - *Uma para armazenar os dados do jogador e dos pontos (EntradaJogo);*
 - *Uma para armazenar o conjunto de objetos EntradaJogo (Pontuacao).*

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*
 - *Classe EntradaJogo*

EntradaJogo
nome: String pontos: int
EntradaJogo() toString

Arranjos ordenados

```
public class EntradaJogo {
    protected String nome;
    protected int pontos;

    public EntradaJogo(String nome, int pontos){
        this.nome = nome;
        this.pontos = pontos;
    }

    public String getNome() { return nome; }

    public void setNome(String nome) { this.nome = nome; }

    public int getPontos() { return pontos; }

    public void setPontos(int pontos) { this.pontos = pontos;
}

    public String toString(){
        return "(" + nome + ", " + pontos + ")";
    }
}
```

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*
 - *Classe Pontuacao*

Pontuacao
maxEntradas: final int numEntradas: int registros: EntradaJogo[]
Pontuacao() toString() Inserir() Remover()

Arranjos ordenados

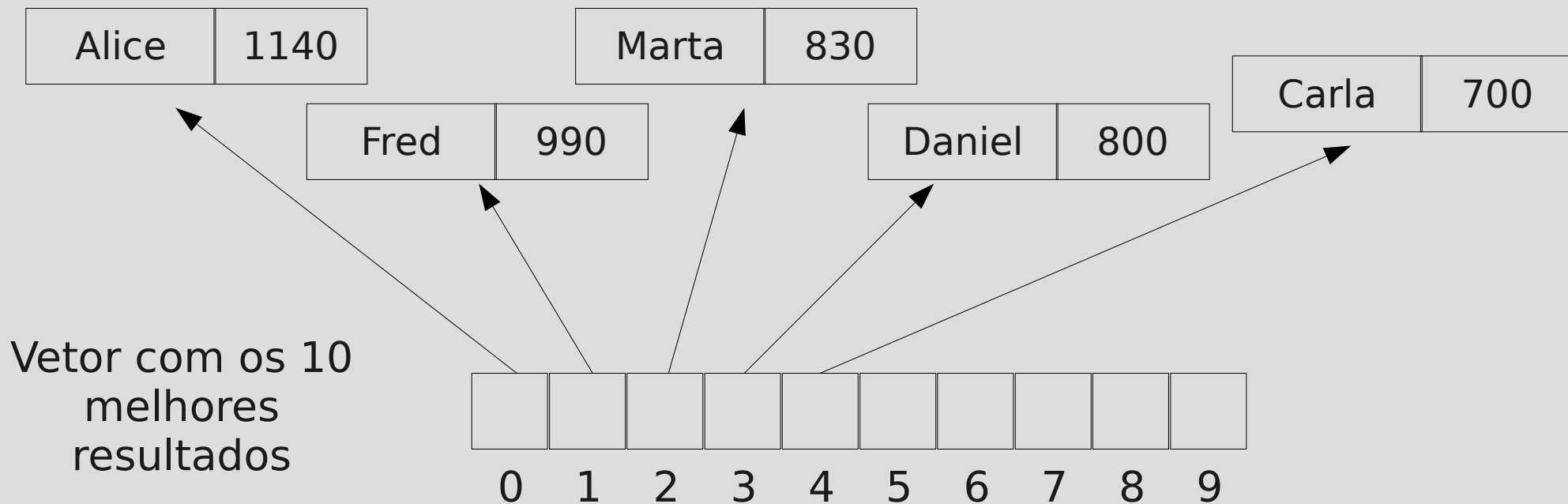
```
public class Pontuacao {
    public static final int maxEntradas = 10;
    protected int numEntradas;
    protected EntradaJogo[] registros;

    public Pontuacao(){
        registros = new EntradaJogo[maxEntradas];
        numEntradas = 0;
    }

    public String toString(){
        String s = "[";
        for (int i=0; i<numEntradas; i++){
            if (i>0)
                s += ", ";
            s += registros[i];
        }
        return s + "]";
    }
}
```

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*
 - *Classe Pontuacao*



Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*
 - *Inserção de valores no vetor*
 - *É necessário verificar os valores já existentes para que o vetor permaneça ordenado após a inserção*
 - *É necessário realizar deslocamentos no vetor, caso o valor inserido seja maior que os valores presentes no vetor*

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*

→ *Inserção de valores no vetor*

```
public void Inserir(EntradaJogo e){
    int novaPontuacao = e.getPontos();
    if (numEntradas == maxEntradas){
        if (novaPontuacao <= registros[numEntradas-1].getPontos())
            return;
    }
    else
        numEntradas++;
    int i = numEntradas-1;
    for (;(i>=1)&&(novaPontuacao>registros[i-1].getPontos());i--)
        registros[i] = registros[i-1];
    registros[i] = e;
}
```

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*
 - *Remoção de valores no vetor*
 - *É necessário conhecer o índice do valor a ser excluído*
 - *É necessário realizar deslocamentos no vetor, caso o valor removido esteja entre outros elementos do vetor*

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*
 - *Remoção de valores no vetor*

```
public EntradaJogo Remove (int i) throws
IndexOutOfBoundsException{
    if ((i<0) || (i>=numEntradas))
        throw new IndexOutOfBoundsException ("Indice
invalido: " + i);
    EntradaJogo temp = registros[i];
    for (int j=i; j<numEntradas-1; j++)
        registros[j] = registros[j+1];
    registros[numEntradas-1] = null;
    numEntradas--;
    return temp;
}
```

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*
 - *Inserção dos seguintes valores no vetor:*
 - *Marta – 830 pontos*
 - *Fred – 990 pontos*
 - *Daniel – 800 pontos*
 - *Carla – 700 pontos*
 - *Alice – 1140 pontos*

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*

Vetor inicial

null	null	null	null	null	null	null	null	null	null
------	------	------	------	------	------	------	------	------	------

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*

null	null	null	null	null	null	null	null	null	null
------	------	------	------	------	------	------	------	------	------

Inserir (Marta, 830)

Marta 830	null	null	null	null	null	null	null	null	null
--------------	------	------	------	------	------	------	------	------	------

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*

Marta 830	null	null	null	null	null	null	null	null	null
--------------	------	------	------	------	------	------	------	------	------

Inserir (Fred, 990)

Desloca (Marta, 830) e
Insere (Fred, 990)

Fred 990	Marta 830	null	null	null	null	null	null	null	null
-------------	--------------	------	------	------	------	------	------	------	------

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*

Fred 990	Marta 830	null	null	null	null	null	null	null	null
-------------	--------------	------	------	------	------	------	------	------	------

Inserir (Carla, 700)

Fred 990	Marta 830	Carla 700	null	null	null	null	null	null	null
-------------	--------------	--------------	------	------	------	------	------	------	------

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*

Fred 990	Marta 830	Carla 700	null	null	null	null	null	null	null
-------------	--------------	--------------	------	------	------	------	------	------	------

Inserir (Daniel, 800)

Desloca (Carla, 700) e
Insere (Daniel, 800)

Fred 990	Marta 830	Daniel 800	Carla 700	null	null	null	null	null	null
-------------	--------------	---------------	--------------	------	------	------	------	------	------

Arranjos ordenados

- *Exemplo: Maiores resultados em um jogo*

Fred 990	Marta 830	Daniel 800	Carla 700	null	null	null	null	null	null
-------------	--------------	---------------	--------------	------	------	------	------	------	------

Inserir (Alice, 1140)

Desloca todos os elementos
existentes no vetor e Insere
(Alice, 1140)

Alice 1140	Fred 990	Marta 830	Daniel 800	Carla 700	null	null	null	null	null
---------------	-------------	--------------	---------------	--------------	------	------	------	------	------

Arranjos ordenados

- *Vantagem e desvantagem dos vetores ordenados*
 - *Tempo de pesquisa (+)*
 - *Inserção demorada, pois todos os elementos maiores que o valor inserido devem ser deslocados (-)*
 - *Remoção demorada, pois todos os elementos menores que o valor removido devem ser deslocados*

Arranjos ordenados

- *Quando utilizar?*

→ *Em sistemas onde o número de pesquisas é grande, mas as inserções e remoções não são*

Roteiro

- *Objetivos*
- *Arranjos ordenados*
- ***Ordenamento em arranjos***
- *Arranjos bidimensionais (matrizes)*
- *Exercícios*

Ordenamento em Arranjos

- *Problema de ordenação*
- *O arranjo inicial não é ordenado, é necessário que os elementos sejam ordenados*
- *OBS: Métodos de ordenação serão estudados com mais detalhes no decorrer do curso*

Ordenamento em Arranjos

- *Algoritmo InsertionSort*

- *Possui como entrada um arranjo desordenado*

- *Trabalha com um elemento de cada vez, colocando-o na posição correta dentro do vetor*

- *O primeiro elemento a ser verificado é o do índice 0, que por si só já está ordenado*

- *Os outros elementos são verificados e deslocados caso seja necessário, até que todo o arranjo esteja ordenado*

Ordenamento em Arranjos

Algoritmo InsertionSort

Entrada: Arranjo A desordenado com n elementos

Saída: O arranjo A com os elementos ordenados em ordem crescente

Para $i=1$ até $n-1$ **faça**

 corrente = $A[i]$

$j = i-1$

Enquanto $j \geq 0$ e $A[j] > \text{corrente}$ **faça**

$A[j+1] = A[j]$

$j = j-1$

$A[j+1] = \text{corrente}$

Ordenamento em Arranjos

- *Algoritmo InsertionSort*

→ *Exemplo: Ordenar o seguinte vetor*

25	57	48	37	12	92	86	33
----	----	----	----	----	----	----	----

Ordenamento em Arranjos

- *Algoritmo InsertionSort*

25	57	48	37	12	92	86	33
----	----	----	----	----	----	----	----

25	57	48	37	12	92	86	33
----	----	----	----	----	----	----	----

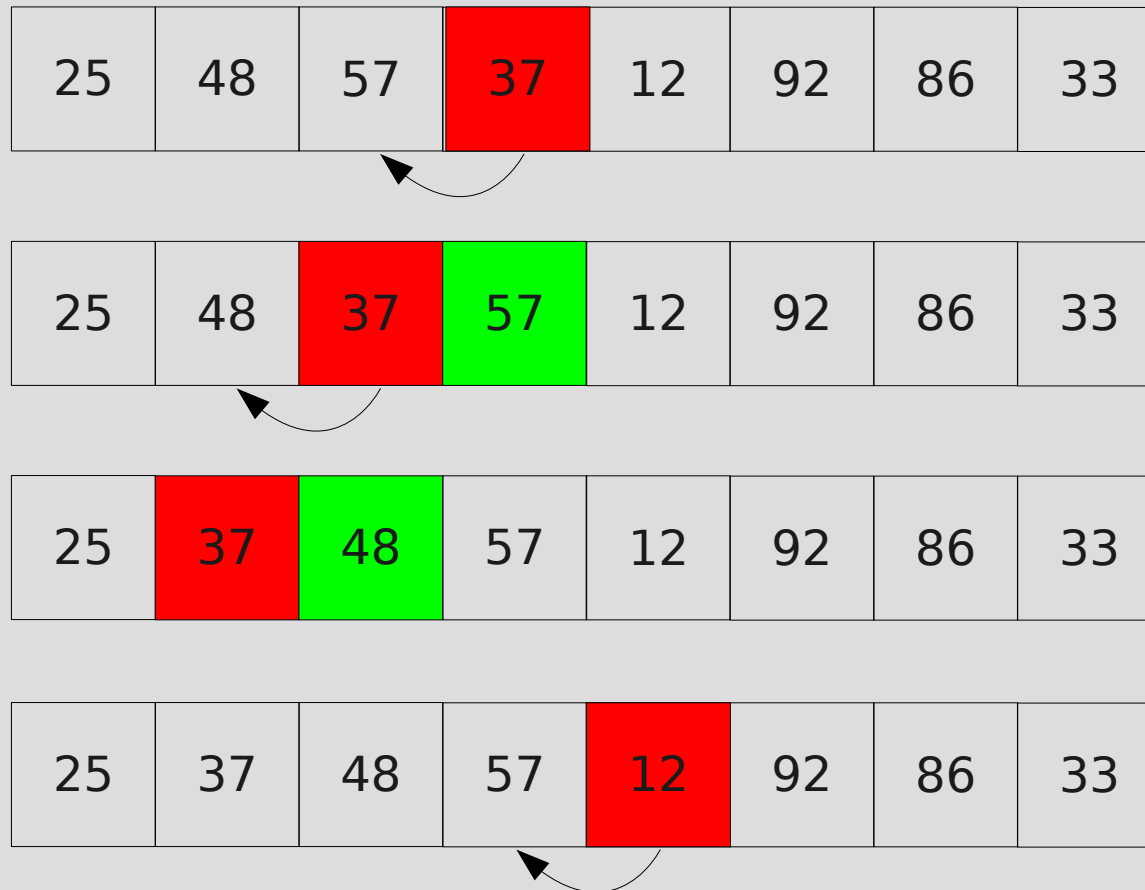
25	57	48	37	12	92	86	33
----	----	----	----	----	----	----	----



25	48	57	37	12	92	86	33
----	----	----	----	----	----	----	----

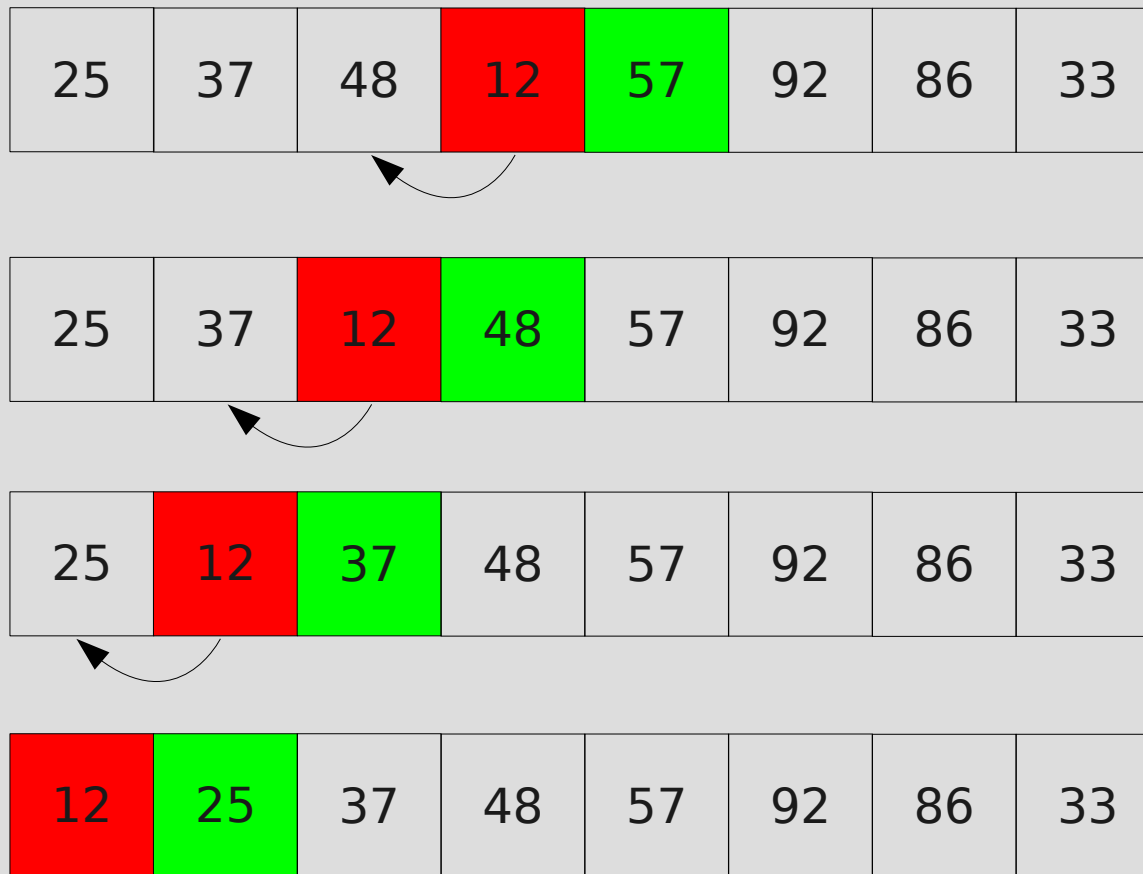
Ordenamento em Arranjos

- *Algoritmo InsertionSort*



Ordenamento em Arranjos

- *Algoritmo InsertionSort*



Ordenamento em Arranjos

- *Algoritmo InsertionSort*

12	25	37	48	57	92	86	33
----	----	----	----	----	----	----	----

12	25	37	48	57	92	86	33
----	----	----	----	----	----	----	----



12	25	37	48	57	86	92	33
----	----	----	----	----	----	----	----

12	25	37	48	57	86	92	33
----	----	----	----	----	----	----	----



Ordenamento em Arranjos

- *Algoritmo InsertionSort*



Ordenamento em Arranjos

- *Algoritmo InsertionSort*

- *Vetor final ordenado*

12	25	33	37	48	57	86	92
----	----	----	----	----	----	----	----

- *Considerações*

- *Grande número de trocas*

- *Pior caso: Vetor em ordem decrescente*

- *Melhor caso: Vetor em ordem crescente*

Ordenamento em Arranjos

- *Algoritmo InsertionSort*

```
public static void InsertionSort(int[] vetor){
    int tam = vetor.length;
    for (int i=1; i<tam; i++){
        int corrente = vetor[i];
        int j=i-1;
        while ((j>=0) && (vetor[j]>corrente))
            vetor[j+1] = vetor[j--];
        vetor[j+1] = corrente;
    }
    System.out.println("O vetor foi ordenado com sucesso.");
}
```

Roteiro

- *Objetivos*
- *Arranjos ordenados*
- *Ordenamento em arranjos*
- ***Arranjos bidimensionais (matrizes)***
- *Exercícios*

Arranjos bidimensionais

- *Também conhecidos por matrizes*
- *São arranjos de arranjos*
- *Utilizados em jogos de posição, que precisam representar objetos em um espaço bidimensional*
- *São acessados por dois índices, um referente a linha e outro referente a coluna*

Arranjos bidimensionais

- int[][] matriz = new int[10][10];*

	0	1	2	3	4	5	6	7	8	9
0										
1										
2										
3										
4										
5										
6										
7										
8										
9										

Arranjos bidimensionais

- *Exemplos de aplicação*
 - *Jogo da velha;*
 - *Xadrez;*
 - *Batalha naval;*
 - *Multiplicação de matrizes.*

Roteiro

- *Objetivos*
- *Arranjos ordenados*
- *Ordenamento em arranjos*
- *Arranjos bidimensionais (matrizes)*
- ***Exercícios***

Exercícios

- *Escreva um programa que realize a multiplicação de duas matrizes e mostre o resultado na tela.*
- *Ordene o vetor a seguir com o método InsertionSort e verifique o número de trocas necessárias*

G	P	B	C	F	A	L
---	---	---	---	---	---	---

- *Simule o programa de pontuação utilizando um vetor de 5 posições. Em seguida, adicione os valores apresentados em aula para preencher o vetor. Adicione um novo registro na 2ª. posição, e em seguida, remova o valor da 3ª. posição.*