Instituto de computação uff

Variáveis Compostas

Leonardo Gresta Paulino Murta

leomurta@ic.uff.br

Aula de hoje

- Veremos os diferentes tipos de variáveis compostas (*arrays*)
  - Com uma dimensão (vetores)
  - Com duas ou mais dimensões (matrizes)

Exemplo Motivacional

- Programa para auxiliar a escrever “Parabéns!” nas melhores provas de uma disciplina com 3 alunos
  - Ler os nomes e as notas de 3 alunos
  - Calcular a média da turma
  - Listar os alunos tiveram nota acima da média

Exemplo Motivacional

```
import java.util.Scanner;

public class Notas {

  public static void main(String[] args) {
    Scanner teclado = new Scanner(System.in);
    String nome1, nome2, nome3;
    float nota1, nota2, nota3, media;

    System.out.print("Informe o nome do aluno 1: ");
    nome1 = teclado.nextLine();
    System.out.print("Informe o nome do aluno 2: ");
    nome2 = teclado.nextLine();
    System.out.print("Informe o nome do aluno 3: ");
    nome3 = teclado.nextLine();
```

Exemplo Motivacional

```
    System.out.print("Informe a nota de " + nome1 + ": ");
    nota1 = teclado.nextFloat();
    System.out.print("Informe a nota de " + nome2 + ": ");
    nota2 = teclado.nextFloat();
    System.out.print("Informe a nota de " + nome3 + ": ");
    nota3 = teclado.nextFloat();
    media = (nota1 + nota2 + nota3)/3;

    if (nota1 > media)
      System.out.println("Parabéns " + nome1);
    if (nota2 > media)
      System.out.println("Parabéns " + nome2);
    if (nota3 > media)
      System.out.println("Parabéns " + nome3);
  }
}
```

E se fossem 40 alunos?

- É possível definir variáveis que guardam mais de um valor de um mesmo tipo
- Essas variáveis são conhecidas como variáveis compostas, variáveis subscritas, variáveis indexáveis ou arranjos (*array*)
- Existem dois tipos principais de variáveis compostas:
  - Vetores
  - Matrizes

Vetores

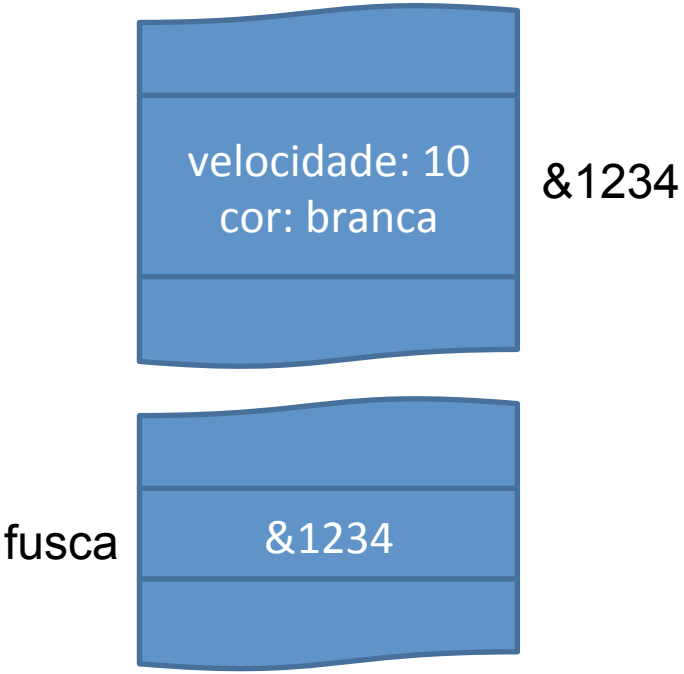
- Variável composta **unidimensional**
  - Contém espaço para armazenar diversos valores de um mesmo tipo
  - É acessada via um índice
- A ideia de vetor é comum na matemática, com o nome de variável subscrita
  - Exemplo: x_1, x_2, ..., x_n
- O que vimos até agora são variáveis com somente um valor
  - Exemplo: x = 123
- No caso de vetores, uma mesma variável guarda ao mesmo tempo múltiplos valores
  - Exemplo: $x_1 = 123$, $x_2 = 456$, ...

Recapitulando: variáveis que contêm tipos primitivos

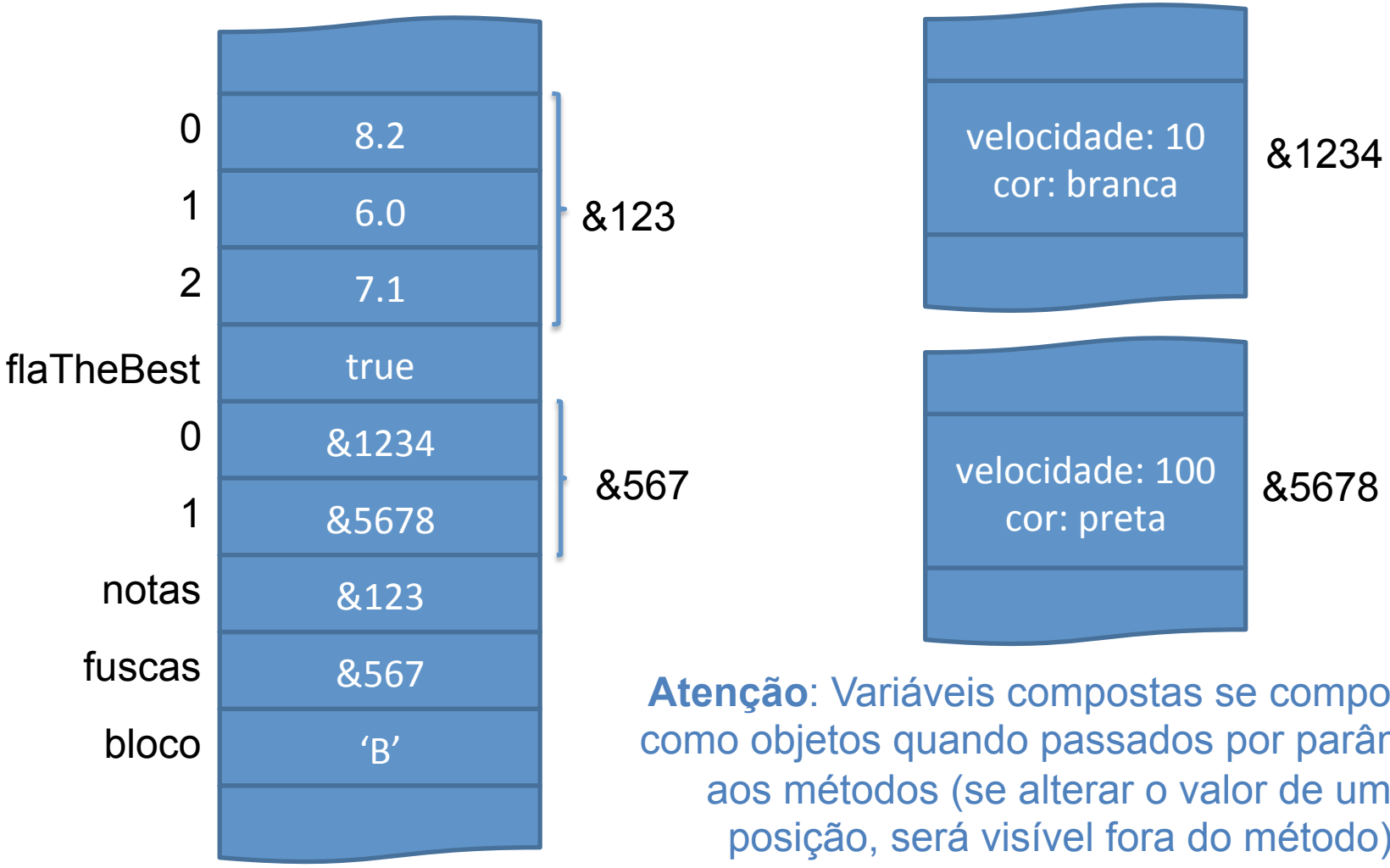
- Até agora, variáveis que contêm tipos primitivos (byte, short, int, long, float, double, char, boolean) sempre ocupam diretamente uma posição na memória

Recapitulando: variáveis que contêm objetos

- Já variáveis que contêm objetos na verdade guardam a posição de memória dos objetos



Retomando: Vetores



Atenção: Variáveis compostas se comportam como objetos quando passados por parâmetro aos métodos (se alterar o valor de uma posição, será visível fora do método)

Declaração de vetores

- Forma geral

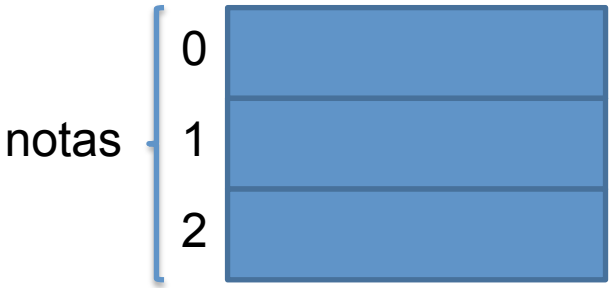
```
TIPO[] NOME = new TIPO[TAMANHO];
ou
TIPO[] NOME;
...
NOME = new TIPO[TAMANHO];
```

- Exemplos

```
String[] nomes = new String[40];
float[] notas = new float[40];
Carro[] fuscas = new Carro[2];
boolean[] presenca;
presenca = new boolean[5];
```

Declaração de vetores

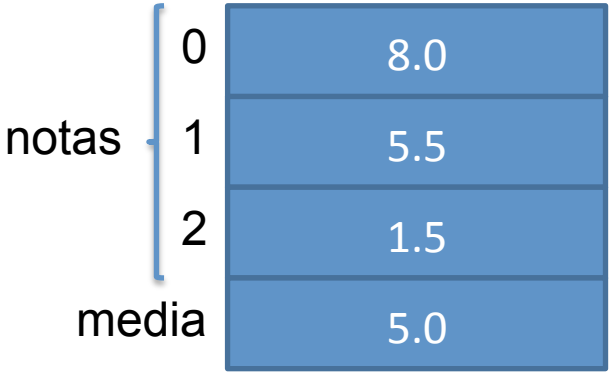
- É possível saber o tamanho de um vetor acessando a propriedade *length*
  - Exemplo: notas. length → 40
- No Java, todo vetor inicia na posição *0* (zero) e termina na posição *length – 1*
  - Exemplo: float[] notas = new float[3];



Utilização de vetores

- Para acessar (ler ou escrever) uma posição do vetor, basta informar a posição entre colchetes

```
notas[0] = 8;
notas[1] = 5.5f;
notas[2] = 1.5f;
media = (notas[0] + notas[1] + notas[2]) / 3;
```



Utilização de vetores

- Também é possível iniciar os valores de vetores diretamente no código, colocando-os entre chaves ({}), separados por vírgula

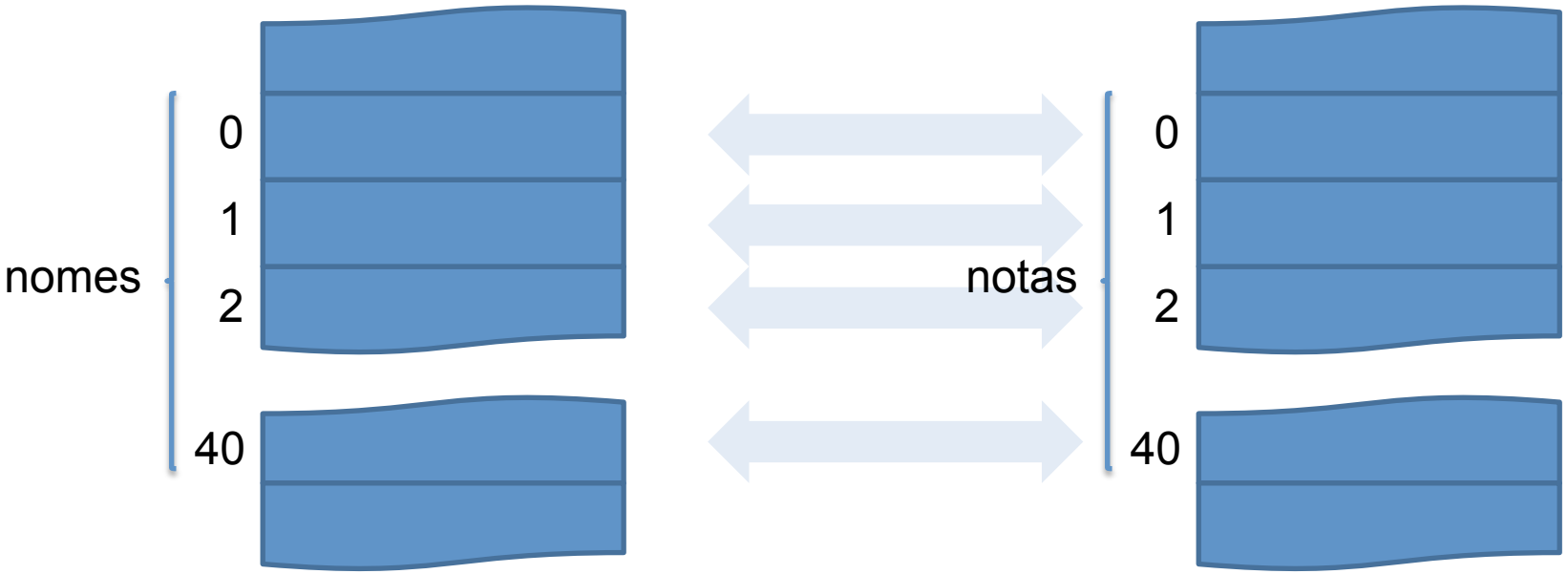
```
notas = { 8, 5.5f, 1.5f};
media = (notas[0] + notas[1] + notas[2]) / 3;
```

- Outra possibilidade é de iterar por todos os seus valores

```
for (int i = 0; i < notas.length; i++) {
  System.out.print(notas[i]);
}
```

Retomando: E se fossem 40 alunos?

- Criaríamos dois vetores (nomes e notas) de 40 posições
- Vincularíamos a posição *N* do vetor de nomes à posição *N* do vetor de notas



Retomando: E se fossem 40 alunos?

```
import java.util.Scanner;

public class Notas {

  public static void main(String[] args) {
    final int NUMERO_ALUNOS = 40;
    Scanner teclado = new Scanner(System.in);
    String[] nomes = new String[NUMERO_ALUNOS];
    float[] notas = new float[NUMERO_ALUNOS];
    float media = 0;

    for (int i = 0; i < NUMERO_ALUNOS; i++) {
      System.out.print("Informe o nome do aluno "+(i+1)+": ");
      nomes[i] = teclado.nextLine();
    }
```

Retomando: E se fossem 40 alunos?

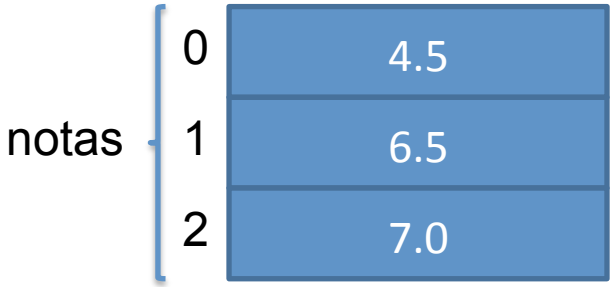
```
    for (int i = 0; i < NUMERO_ALUNOS; i++) {
      System.out.print("Informe a nota de " + nomes[i] + ": ");
      notas[i] = teclado.nextFloat();
      media += notas[i];
    }
    media /= NUMERO_ALUNOS;

    for (int i = 0; i < NUMERO_ALUNOS; i++) {
      if (notas[i] > media)
        System.out.println("Parabéns " + nomes[i]);
    }
  }
}
```

Matrizes

- Variável composta **multidimensional**
  - É equivalente a um vetor, contudo permite a utilização de diversas dimensões acessadas via diferentes índices
  - Pode ser pensada como um vetor cujo tipo é outro vetor, recursivamente
  - Em diversas situações matrizes são necessárias para correlacionar informações

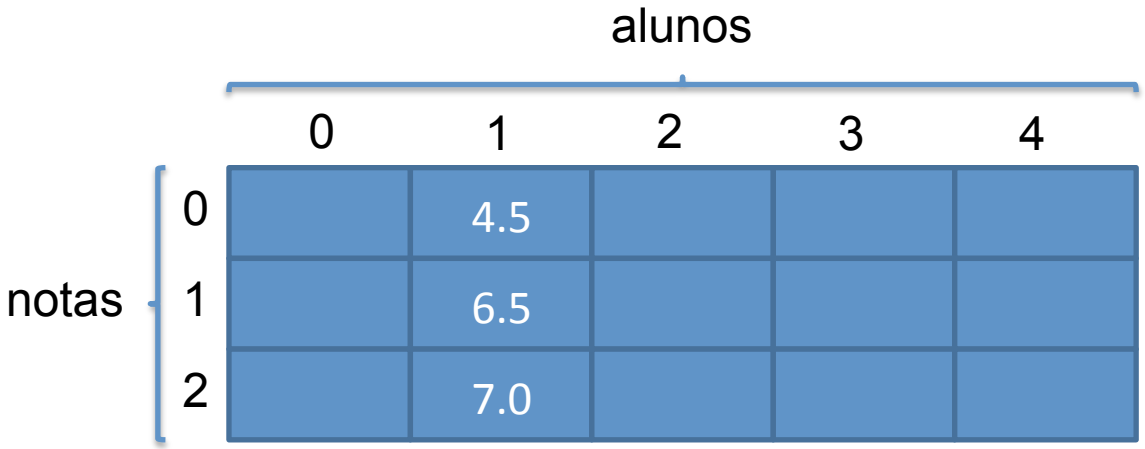
Exemplo motivacional

- Assumindo que **um aluno é avaliado com três notas**, seria necessário um vetor de três posições para guardar as notas de um aluno...



Exemplo motivacional

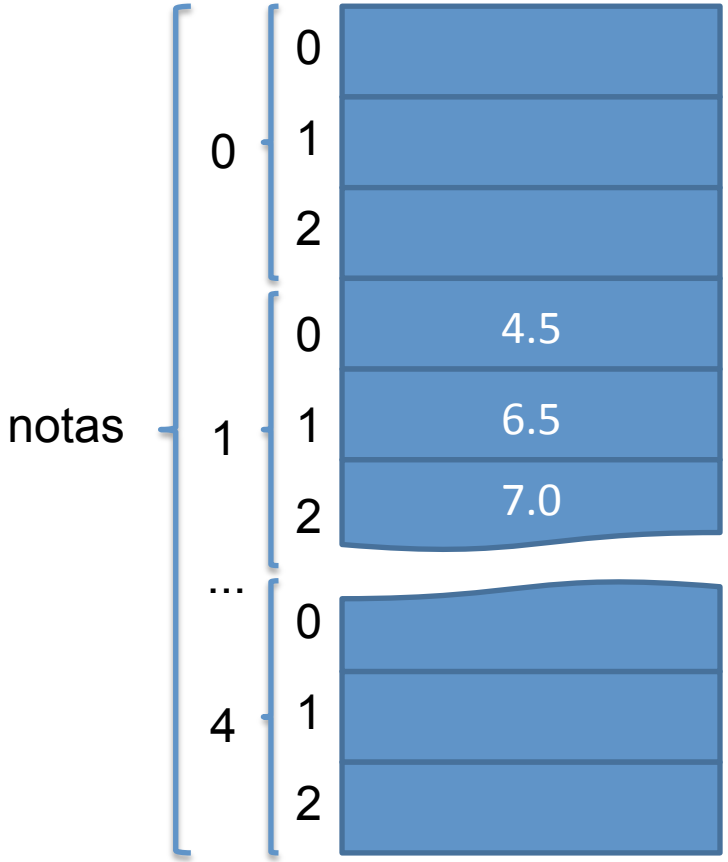
- Contudo, assumindo que **uma turma tem cinco alunos**, seria necessária uma matriz bidimensional para guardar as notas de todos os alunos de uma turma...




```
float[][] notas = new float[5][3]; // Declaração
System.out.println(notas[1][0]);
```

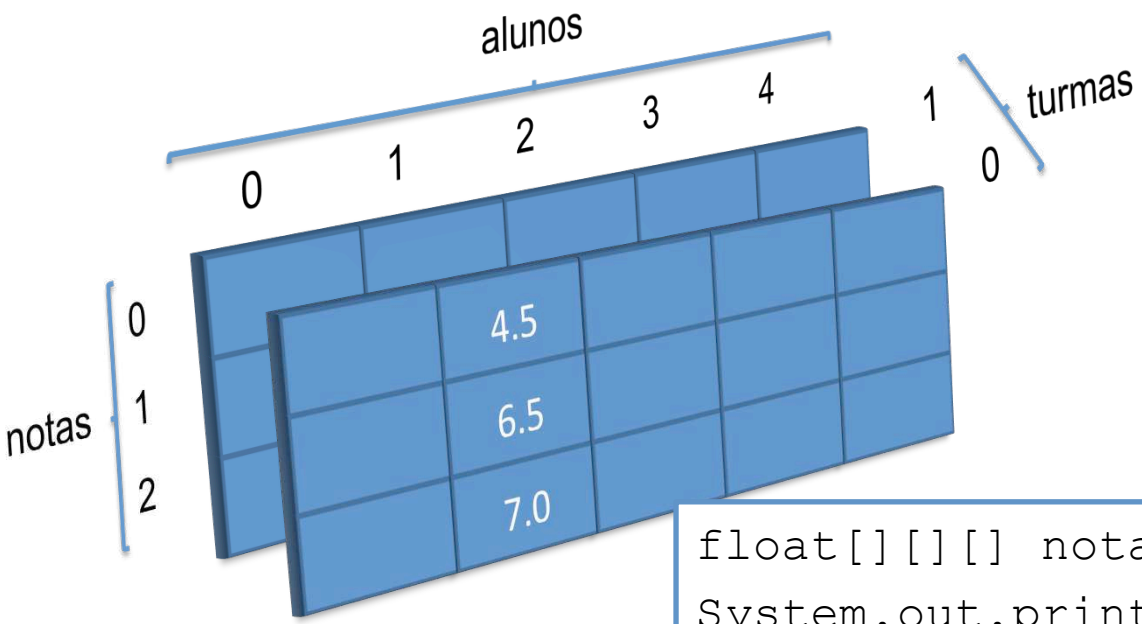
Exemplo motivacional

- Na memória, podemos imaginar que seria algo assim...



Exemplo motivacional

- Ainda, assumindo que **um curso tem duas turmas**, seria necessária uma matriz tridimensional para guardar as notas de todos os alunos de todas as turmas do curso...




```
float[][][] notas = new float[2][5][3];
System.out.println(notas[0][1][0]);
```

Exercício

- Faça um programa que leia dois vetores de 3 posições, que representam forças sobre um ponto no espaço 3D, e escreva a força resultante
- Faça um programa que leia duas matrizes tamanho 2x3 e 3x2 e escreva a matriz resultado da multiplicação dessas matrizes (dica: c[i, j] = Somatório de a[i,k] * b[k,j] para todo k)
- Faça um programa que leia a ordem de uma matriz quadrada (até 100), posteriormente leia os seus valores e finalmente escreva a sua transposta (at[i, j] = a[j, i])

Exercício

- Faça um programa que lê o nome e três notas para cada aluno de cada turma de um curso, onde cada turma tem 5 alunos e o curso tem 3 turmas
- Ao final, permita que o usuário informe...
  - O nome de um aluno e o programa liste a média desse aluno
  - Uma nota e o programa liste todos os alunos que têm médias acima dessa nota

Exercício

- Leia o nome e a idade de 10 pessoas e liste as pessoas em ordem crescente de idade
  - Relembrem a aula que falamos de ordenação do baralho
  - Pense no algoritmo antes de programar!

Exercício

- Ordene um vetor de números inteiros com 10 posições utilizando o algoritmo QuickSort
  - Separe o primeiro elemento do vetor (pivô)
  - Divida o vetor em dois: vetor de menores e vetor de maiores que o pivô
  - Ordene recursivamente cada vetor (menores e maiores)
  - Concatene o vetor de menores ordenado com o pivô e com o vetor de maiores ordenado

Exercício

- Refaça o exercício anterior sem que seja necessário criar vetores separados para menores e maiores
  - Assim não precisaremos alocar memória desnecessariamente
  - Pense no algoritmo antes de programar!

Exercício

- Resolva o mesmo problema de ordenar 10 números utilizando o algoritmo Merge Sort
  - Divida o vetor em dois e chame recursivamente o algoritmo
  - Combine os dois vetores ordenados individualmente em um vetor ordenado por completo
- Refaça o exercício anterior sem que seja necessário criar vetores extras

Instituto de computação uff

Variáveis Compostas

Leonardo Gresta Paulino Murta

leomurta@ic.uff.br