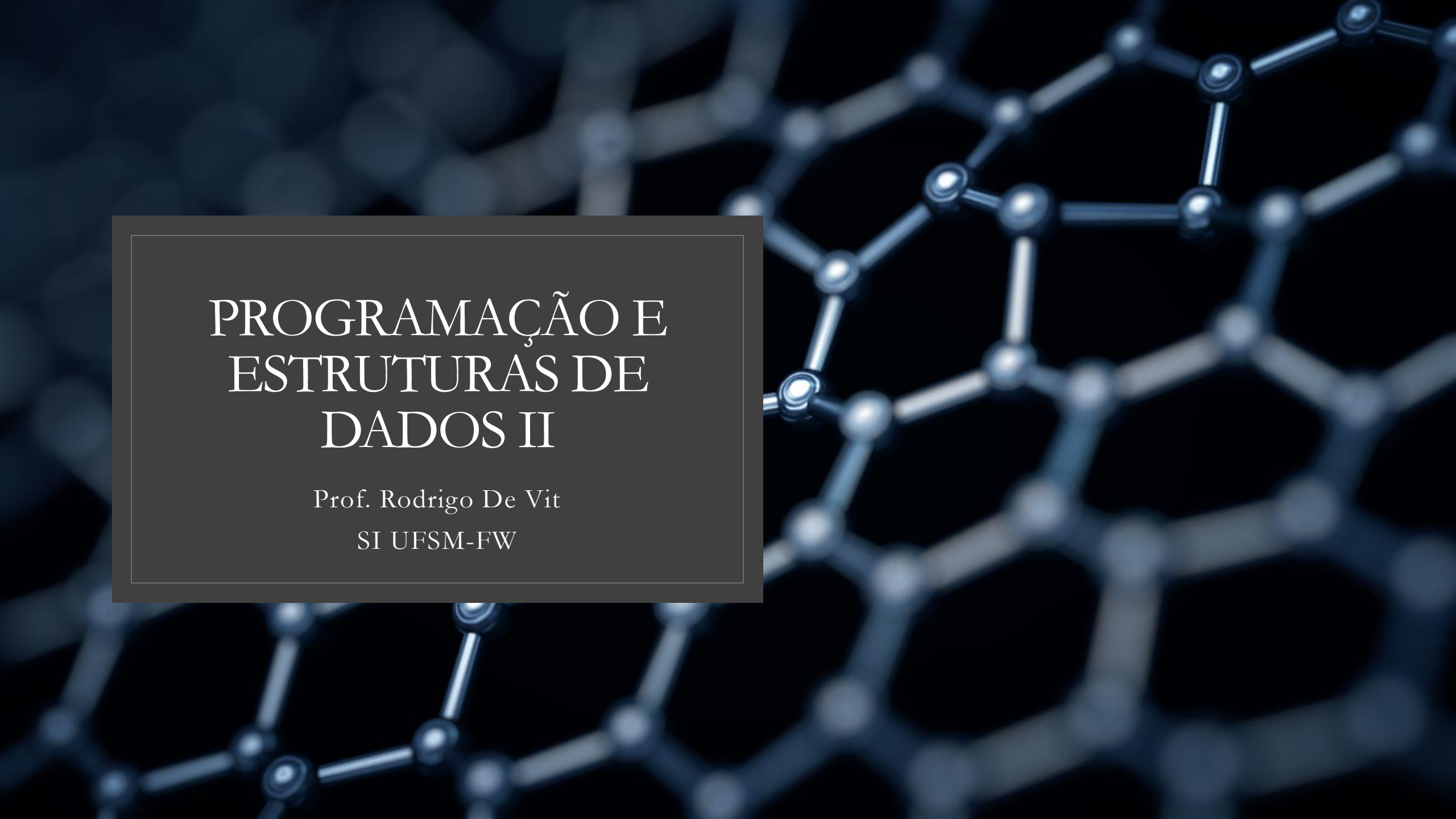

PROGRAMAÇÃO E ESTRUTURAS DE DADOS II

Prof. Rodrigo De Vit

SI UFSM-FW

Revisão: Matrizes, pilhas e filas

- **Aula (05), 3 e 4 de setembro e Aula (06), 10 e 11 de setembro:**
- UNIDADE 1 - ESTRUTURAS LINEARES E ENCADEADAS
- 1.3 - Estrutura Estática - Matrizes.
- 1.3.1 - Conceituação e tipos.
- 1.3.2 - Manipulação de pilhas e filas.
- **Pendências...**
  - Fatorial não recursivo FatorialNaoRecursivo.c
  - Ponteiro de variáveis PonteiroSimples.c
  - Ponteiros para funções PonteiroParaFuncoes.c
  - Fatorial recursivo FatorialRecursivo.c
  - Passagem de vetores para funções VetoresParaFuncoes.c
  - Funções recursivas FuncoesRecursivas.c

Matrizes

- Alocação estática versus dinâmica EstatVsDinam.c
- Matrizes Matrizes.c
- Matrizes Dinâmicas MatrizesDinamicas.c

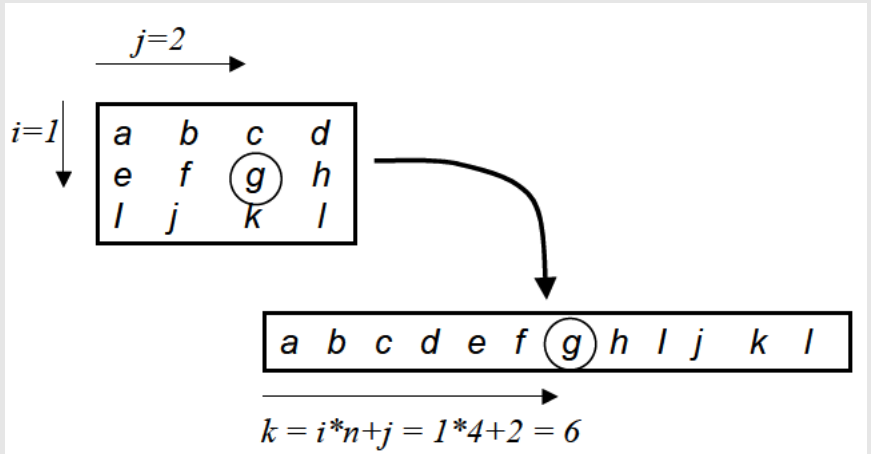

Figura 8.2: Matriz representada por vetor simples.

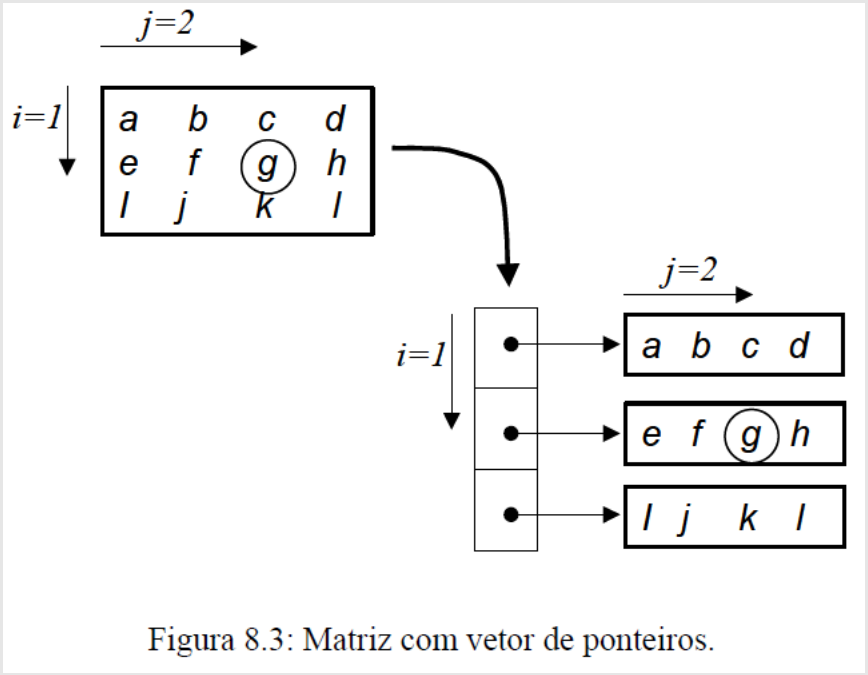

Figura 8.3: Matriz com vetor de ponteiros.

Representação de matrizes

- Para exemplificar o uso de matrizes dinâmicas, vamos discutir a escolha de um tipo para representar as matrizes e um conjunto de operações implementadas sobre o tipo escolhido. Podemos considerar, por exemplo, a implementação de funções básicas, sobre as quais podemos futuramente implementar funções mais complexas, tais como soma, multiplicação e inversão de matrizes.
- Vamos considerar a implementação das seguintes operações básicas:
  - • cria: operação que cria uma matriz de dimensão m por n;
  - • libera: operação que libera a memória alocada para a matriz;
  - • acessa: operação que acessa o elemento da linha i e da coluna j da matriz;
  - • atribui: operação que atribui o elemento da linha i e da coluna j da matriz.
- A seguir, mostraremos a implementação dessas operações usando as duas estratégias para alocar dinamicamente uma matriz, apresentadas na seção anterior.

Representação de matrizes - implementação

- Matriz com vetor simples MatrizCVS.c
- Matriz com vetor de ponteiros MatrizCVP.c

Representação de matrizes simétricas

- Em uma matriz simétrica n por n, não há necessidade, no caso de i≠j, de armazenar ambos os elementos mat[i][j] e mat[j][i], porque os dois têm o mesmo valor. Portanto, basta guardar os valores dos elementos da diagonal e de metade dos elementos restantes – por exemplo, os elementos abaixo da diagonal, para os quais i>j. Ou seja, podemos fazer uma economia de espaço usado para alocar a matriz. Em vez de n2 valores, podemos armazenar apenas s elementos, sendo s dado por:

$$s = n + \frac{(n^2 - n)}{2} = \frac{n\,(n+1)}{2}$$

- Podemos também determinar s como sendo a soma de uma progressão aritmética, pois temos que armazenar um elemento da primeira linha, dois elementos da segunda, três da terceira, e assim por diante.

$$s = 1 + 2 + \ldots + n = \frac{n\,(n+1)}{2}$$

- A implementação deste tipo abstrato também pode ser feita com um vetor simples ou um vetor de ponteiros. A seguir, discutimos a implementação das operações para criar uma matriz e para acessar os elementos, agora para um tipo que representa uma matriz simétrica.

Representação de matrizes simétricas

- Matriz simétrica com vetor simples MatrizSimetricaCVS.c
- Matriz simétrica com vetor de ponteiros MatrizSimetricaCVP.c