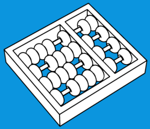
Matrizes

MC102 - Algoritmos e Programação de Computadores

15

05/25



UNICAMP

Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br

"Matrizes agem. Elas não apenas ficam lá."

Gilbert Strang.

Dúvidas da aula anterior

Dúvidas selecionadas

- Por que usar um conjunto ao invés de uma lista ou vice versa? A lista por si não é um conjunto?
- Conjuntos quando chamados em funções (sem serem declarados como globais) funcionam que nem listas? Podem receber valores e alterar o conjunto de verdade?
- Como adicionar um número massivo de chaves-valores em um Dicionário? Toda vez que o programa for executado o Dicionário terá que receber todos dos dados novamente?
- Da pra adicionar listas dentro de um conjunto?
- Seria possível fazer um dicionário que associe 2 chaves a um único valor? Por exemplo: utilizar o RA e o curso para achar o nome do aluno.
- Um dicionário pode ter uma lista como chave? Ou um valor ser uma lista?
- Suponha a lista l = [1,0,1]. Posto que um conjunto não possui elementos repetidos, se a lista l for convertida num conjunto e esse conjunto for impresso, o que será exibido: 0,1 ou 1,0?
- Percorrer (usando for) um conjunto com elementos de diferentes tipos causa algum erro?
- Qual a diferença entre s.discard(x) e s.remove(x)?
- Qual a diferença entre dicionário e conjunto?
- Não entendi exatamente como as chaves no dicionário funcionam.
- Tem como eu fazer um dicionário de dicionários?
- O set guarda elementos repetidos ou as repetições são removidas? Como faço se eu quiser guardar (ou não) repetições?

Matrizes

Matrizes

Em Python, matrizes são representadas como listas de listas.

Isto é, podemos representar a matriz:

$$\begin{bmatrix} 7 & 0 & 2 & 3 \\ 3 & 1 & 4 & 2 \\ 0 & 3 & 2 & 7 \end{bmatrix}$$

Como **m = [[7, 0, 2, 3], [3, 1, 4, 2], [0, 3, 2, 7]]**.

- Isto é uma lista de linhas da matriz.
- Onde cada linha é, também, uma lista.

A célula da linha **i** coluna **j** é acessada escrevendo **m[i][j]**.

- Para leitura ou escrita.
- Ex: **m[0][0]** é **7**, **m[0][1]** é **0**, **m[1][0]** é **3**, etc.

Matrizes

Vamos fazer alguns exercícios?

Exercícios

1. Faça uma função que dados inteiros **m** e **n**, devolve uma matriz **m**×**n** (isto é, **m** linhas e **n** colunas) com algum valor inicial dado (zero por padrão).
2. Faça uma função que, dado um inteiro **n**, devolve a matriz identidade **n**×**n**.
3. Faça uma função que, dada uma matriz, imprime a matriz (em sua representação usual).
4. Faça uma função que, dada uma matriz M e um escalar λ, calcula $\lambda \cdot M$.
5. Faça uma função que soma duas matrizes.
6. Faça uma função que transpõem uma matriz.

Multiplicação de Matrizes

Sejam A uma matriz $m \times n$ e B uma matriz $n \times p$.

O produto $C = A \cdot B$ é a matriz $m \times p$ tal que:

$$\mathbf{C_{i,j}} = \sum_{\mathbf{k=1}}^{\mathbf{n}} \mathbf{A_{i,k}} \cdot \mathbf{B_{k,j}}.$$

Exercício: Faça uma função que multiplica duas matrizes.

Um aplicação

Aplicação: Imagens

Imagens são matrizes de pixels

- Preto-e-Branco: basta um bit por pixel.
- Escala de cinza: um valor entre 0 e 255.
- RGB (24-bit): cada posição tem três valores entre 0 e 255.
- Entre outros modelos.

Existem vários formatos de arquivo de imagem:

- jpg, png, gif, tiff, etc...

Vamos usar um particularmente fácil de trabalhar...

Um arquivo pbm (preto-e-branco)

Exemplo do arquivo:

```
P1
25 7
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 1 0 1 1 1 1 1 0 1 0 1 1 1 1 0 1 1 1 1 0
0 1 1 0 1 1 0 1 0 0 0 0 0 1 0 1 0 0 1 0 0 0 0 1 0
0 1 0 1 0 1 0 1 0 0 0 0 0 1 0 1 0 0 1 0 1 1 1 1 0
0 1 0 0 0 1 0 1 0 0 0 0 0 1 0 1 0 0 1 0 1 0 0 0 0
0 1 0 0 0 1 0 1 1 1 1 1 0 1 0 1 1 1 1 0 1 1 1 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Formato:

- Sempre começa com **P1**.
- Na segunda linha, temos o número de colunas e de linhas.
- E colocamos a matriz de bits separados por espaço:
  - bit 1 indica pixel preto.

Um arquivo pbm (preto-e-branco)

Exemplo do arquivo:

```
P1
25 7
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 0 0 0 1 0 1 1 1 1 1 0 1 0 1 1 1 1 0 1 1 1 1 0
0 1 1 0 1 1 0 1 0 0 0 0 0 1 0 1 0 0 1 0 0 0 0 1 0
0 1 0 1 0 1 0 1 0 0 0 0 0 1 0 1 0 0 1 0 1 1 1 1 0
0 1 0 0 0 1 0 1 0 0 0 0 0 1 0 1 0 0 1 0 1 0 0 0 0
0 1 0 0 0 1 0 1 1 1 1 1 0 1 0 1 1 1 1 0 1 1 1 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Resultado:

Imagens

Vamos fazer alguns exercícios?

Exercícios

1. Faça uma função que lê do terminal o conteúdo de um arquivo **pbm**.
2. Faça uma função que, dada uma matriz, escreve (no terminal) o conteúdo de um arquivo **pbm**.
3. Faça uma função que, dada uma matriz de **0**'s e **1**'s, nega a matriz, isto é, posições que eram **0** viram **1** e vice-versa.
4. Combine os três exercícios anteriores para inverter as cores de uma imagem **pbm**.

Múltiplas dimensões

Matrizes d-dimensionais

As matrizes que vimos têm duas dimensões:

- linhas e colunas.

Mas podemos querer ter matrizes com mais dimensões...

- Ex: Um vídeo é uma sequência de imagens.
- Ex: Em imagens coloridas, em cada linha/coluna, temos três valores.

Mas isso é fácil de resolver!

- Quando tínhamos uma dimensão usamos listas.
- Quando tínhamos duas dimensões usamos listas de listas.
- Quando temos três dimensões usamos listas de listas de listas!
- E assim por diante!

Linearização

Linearizando índices

Às vezes precisamos usar uma lista ao invés de listas de listas:

- Por questões de eficiência.

Para tanto, precisamos de uma função bijetora entre as posições (i, j) da matriz e as posições k da lista:

- Dada uma posição da matriz, queremos a posição da lista.
  - Ex: na hora de acessar o valor.
- E dada uma posição da lista, queremos a posição da matriz.
  - Ex: na hora de imprimir a matriz.

$$m = \begin{bmatrix} 7 & 0 & 2 \\ 3 & 1 & 4 \\ 0 & 3 & 2 \end{bmatrix} \quad \longleftrightarrow \quad l = \begin{bmatrix} 7 & 0 & 2 & 3 & 1 & 4 & 0 & 3 & 2 \end{bmatrix}$$

Linearizando índices

$$m = \begin{bmatrix} 7 & 0 & 2 \\ 3 & 1 & 4 \\ 0 & 3 & 2 \end{bmatrix} \quad \longleftrightarrow \quad l = \begin{bmatrix} 7 & 0 & 2 & 3 & 1 & 4 & 0 & 3 & 2 \end{bmatrix}$$

A posição:

- **(0,0)** deve ir para o índice 0.
- **(0,1)** deve ir para o índice 1.
- **(0, n - 1)** deve ir para o índice $n - 1$.
- **(1,0)** deve ir para o índice n.
- **(1,1)** deve ir para o índice $n + 1$.
- **(1, n - 1)** deve ir para o índice $2n - 1$.
- **(i, j)** deve ir para o índice $i \cdot n + j$.

Mas como voltar da lista para a matriz?

Linearizando índices

$$m = \begin{bmatrix} 7 & 0 & 2 \\ 3 & 1 & 4 \\ 0 & 3 & 2 \end{bmatrix} \quad \longleftrightarrow \quad l = \begin{bmatrix} 7 & 0 & 2 & 3 & 1 & 4 & 0 & 3 & 2 \end{bmatrix}$$

A posição (i, j) deve ir para o índice $i \cdot n + j$.

Mas como ir da posição k da lista para a matriz?

- Precisamos saber quantos linhas completas formamos:
  - Isto é, $i = k \,//\, n$.
- E quantas colunas sobraram:
  - Isto é, $j = k \,\%\, n$.

Exercício: Faça um programa que lê duas matrizes, soma as duas e imprime o resultado usando linearização de índices.

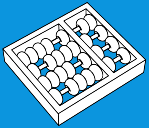
Desafio: Linearize uma matriz tridimensional.

Matrizes

MC102 - Algoritmos e Programação de Computadores

15

05/25



Santiago Valdés Ravelo
https://ic.unicamp.br/~santiago/
ravelo@unicamp.br