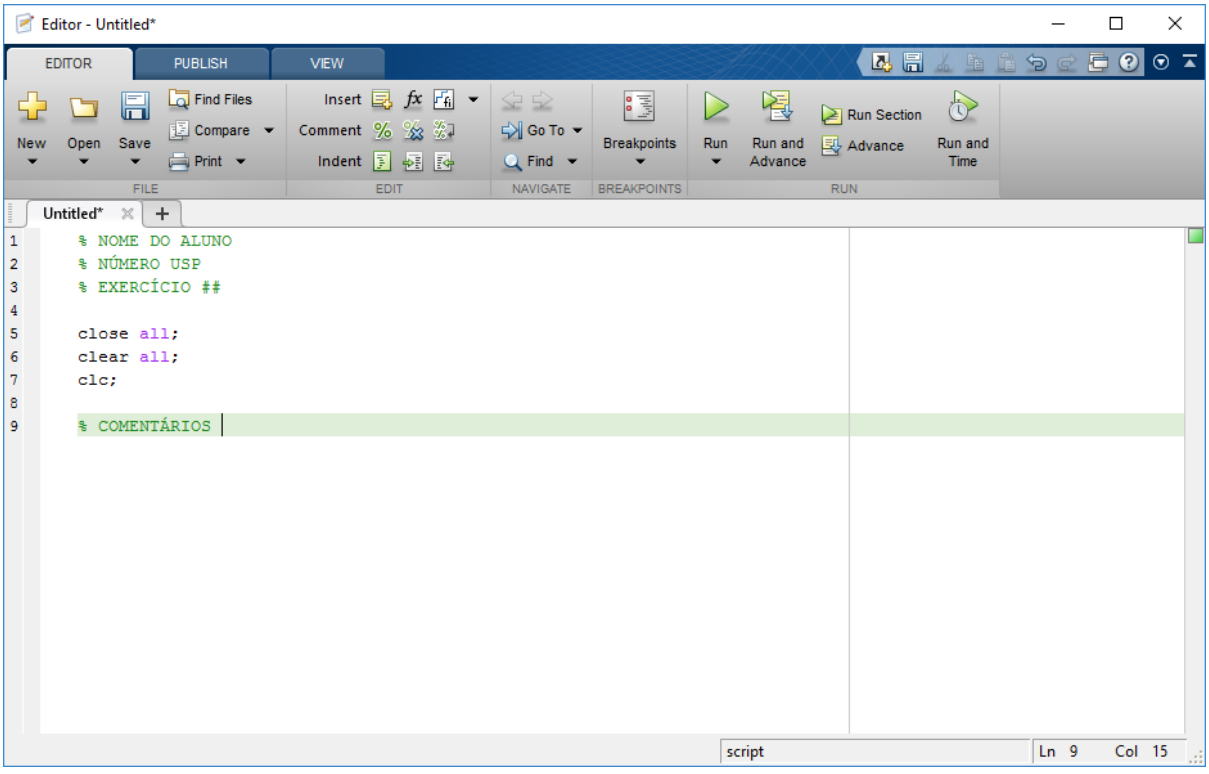
SEL5886 –Visão Computacional
Prof. Dr. Marcelo Andrade da Costa Vieira

Prática 1 - Treinamento no Matlab

Instruções:

- Esse treinamento consiste de 11 exercícios (T_1 a T_11).
- Deve ser gerado um arquivo no editor do Matlab (extensão *.m) para cada exercício pedido.
- Deve-se colocar comentários nos programas desenvolvidos.
- As perguntas devem ser respondidas também como comentários no arquivo.
- Deve-se tornar o diretório onde estão as figuras e os arquivos *.m como um diretório padrão do Matlab.
- Depois de terminado os exercícios, todos os arquivos *.m devem ser comprimidos em um único arquivo e enviado ao professor pelo site de *upload* da IRIS até a data máxima de entrega.

- Utilizar o padrão mostrado na Figura abaixo para seus arquivos *.m:
  - Colocar um cabeçalho contendo seu nome, número USP e o número do exercício correspondente (T1, T2, T3...);
  - Iniciar todos os exercícios com os 3 comandos mostrados na Figura abaixo, que servem para limpar as variáveis e as figuras abertas, além de limpar a tela de comando do Matlab;
  - Colocar comentários nas linhas de programa.



1) Entrando com Matrizes no Matlab.

Matrizes podem ser inseridas no Matlab das seguintes maneiras:

a) Diretamente através de uma lista explícita de elementos.
b) Carregadas através de arquivos externos.
c) Geradas através de funções internas.
d) Criadas através de funções do usuário.

Convenções:

a) Separar os elementos de uma linha com espaços em branco ou vírgulas.
b) Usar ponto e vírgula (`;`) ou "Enter" para indicar o fim de cada linha.
c) Envolver a lista de elementos entre colchetes `[ ]` .
d) Cada linha deve ter sempre o mesmo número de elementos.

T_1: Digitar as Matrizes diretamente nos seguintes formatos e verificar os resultados. Obs.: Os exercícios T_2, T_3, T_4 e T_5 fazem uso das matrizes definidas a seguir.

```
A = [16 3 2 13; 6 9 12 7; 5 10 11 8; 4 15 14 8]

B = [16  8  2  4
     20 30 40 50
     5  7  8  11]

C = [20,30 40;
     50,90,15
     80 30 10]

D = [1,2,3,4;5 6 7 8
     9 8 8 9
     8,7,7,8;
     4 5,9 8]
```

2) Classes de Dados no Matlab.

Classes de Dados:
A tabela da Figura 1 mostra as classes de dados suportadas pelo Matlab.
As coordenadas dos pixels nas imagens são da classe 'inteiro'.
As oito primeiras classes mostradas na tabela são chamadas de numéricas.
Os valores dos pixels nas imagens podem ser de qualquer classe numérica.
As operações numéricas no Matlab são realizadas com a classe 'double'

Name	Description
`double`	Double-precision, floating-point numbers in the approximate range -10^{308} to 10^{308} (8 bytes per element).
`uint8`	Unsigned 8-bit integers in the range [0, 255] (1 byte per element).
`uint16`	Unsigned 16-bit integers in the range [0, 65535] (2 bytes per element).
`uint32`	Unsigned 32-bit integers in the range [0, 4294967295] (4 bytes per element).
`int8`	Signed 8-bit integers in the range [−128, 127] (1 byte per element).
`int16`	Signed 16-bit integers in the range [−32768, 32767] (2 bytes per element).
`int32`	Signed 32-bit integers in the range [−2147483648, 2147483647] (4 bytes per element).
`single`	Single-precision floating-point numbers with values in the approximate range -10^{38} to 10^{38} (4 bytes per element).
`char`	Characters (2 bytes per element).
`logical`	Values are 0 or 1 (1 byte per element).

Figura 1- Classe dos Dados no Matlab

Conversão entre Classes de Dados:

Sintaxe:
```
B = nome_da _classe(A)
```

T_2: Converter as classes de dados das Matrizes digitadas anteriormente.

Usar o comando *whos* para ver a classe de dados em cada caso:

```
AC = uint8(A); BC = uint16(B); CC = uint32(C); DC = single(D);
whos
```

Verificar as classes de dados e o número de bytes de cada matriz antes e depois da conversão.
Comente os resultados

3) Indexando Matrizes.

O Matlab suporta um grande número de esquemas de indexação de Matrizes que facilita a manipulação dos dados. A indexação permite acessar um ou mais elementos diretamente da matriz.

T_3: Verificar o resultado de cada linha abaixo, tentando prever qual será.

Para selecionar um determinado elemento da matriz deve-se utilizar:
`A(i,j)` sendo `i` número da linha e `j` o número da coluna com `i,j = 1, ...n`
Ex.: Elemento da linha 2 e coluna 3:

```
A0 = A(2,3)
```

Usa-se dois pontos (:) para significar "**todos**" ou "**até**".
Ex.: Todos os elementos da quarta coluna:

```
A1 = A(:,4)
```

Ex.: **Todos** os elementos da primeira linha:

```
A2 = A(1,:)
```

Ex.: **Todos** os elementos da linha 1 **até** a linha 2 e da coluna 2 **até** a coluna 3:

```
A3 = A(1:2, 2:3)
```

Ex.: Fazer todos os elementos da coluna 2 iguais a zero:

```
A4 = A
A4(:, 2) = 0
```

Ex.: O elemento da última linha e última coluna:

```
A5 = A(end,end)
```

Ex.: Todos os elementos da última coluna:

```
A6 = A(:, end)
```

Ex.: O elemento da última linha e primeira coluna antes da última:

```
A7 = A(end, end-1)
```

Ex.: Os elementos da linha 2 até a última linha e da última coluna até a segunda coluna com passo de –2:

```
A8 = A(2:end, end:-2:2)
```

Ex.: Matriz transposta de A

```
A9 = A'
```

T_4: Soma, valor máximo e mínimo de uma matriz

O Matlab armazena uma matriz na forma de um vetor alinhado por suas colunas, da primeira até a última. Assim, a indexação pode ser realizada de maneira unidimensional, do primeiro ao último elemento da matriz, na sequência de colunas.

Ex: Para uma matriz A 4x4, indexar o elemento `A(3,3)` é o mesmo que indexar o 11º elemento do vetor:

```
A0 = A(3,3)
A1 = A(11)
```

Logo, pode-se obter a soma de todos os elementos da matriz usando:

```
S1 = sum(A(:))
```

Ou:

```
S2 = sum(sum(A))
```

Vetor mostrando os valores máximos de cada coluna de uma matriz:

```
M0 = max(B)
```

Valor máximo de uma matriz:

```
M1 = max(max(B))
```

Vetor mostrando os valores mínimos de cada coluna de uma matriz:

```
M2 = min(B)
```

Valor mínimo de uma matriz:

```
M3 = min(min(B))
```

4) Tipos de Imagens no Matlab:

O Toolbox de Processamento de Imagens do Matlab trabalha com quatro tipos de Imagens:

- Imagens de Intensidades
- Imagens Binárias
- Imagens Indexadas
- Imagens RGB

Imagens como Matrizes:

O Toolbox de Processamento de Imagens do Matlab considera as imagens como matrizes, adotando a convenção dada na Figura 2b.

(Obs: A Figura 2a denota a convenção adotada para Processamento de Imagens em geral).

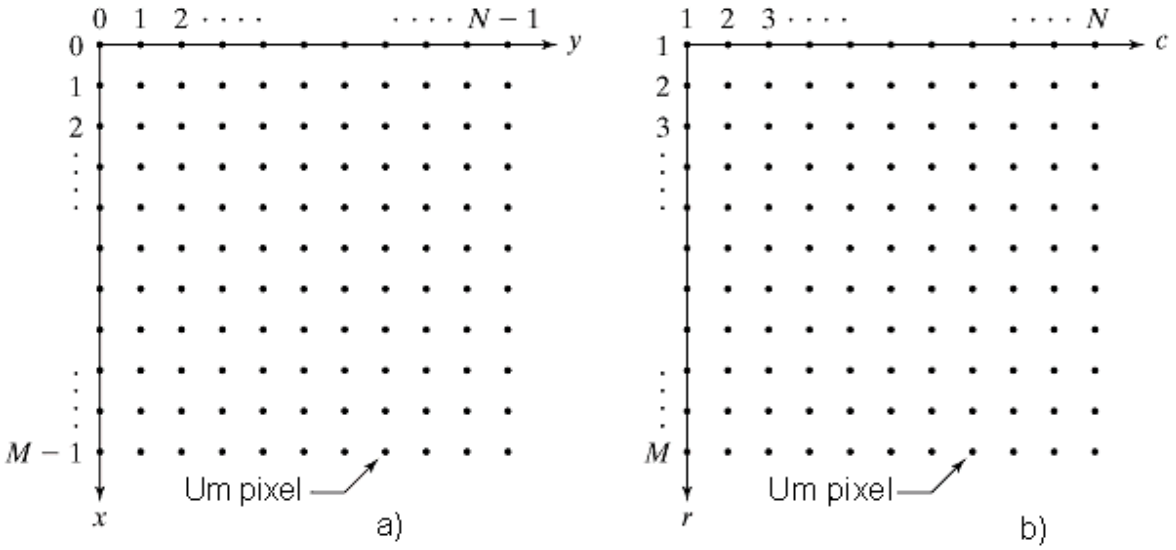


Figura 2 - a) Convenção para Processamento de Imagens. b) Convenção do Matlab

Imagens de Intensidades:

É uma Matriz de dados cujos valores representam as intensidades em cada ponto. Se os elementos de intensidade forem da classe `uint8` seus valores estarão no intervalo [0, 255]. Se forem da classe `uint16` seus valores variarão no intervalo [0, 65535]. Se os elementos forem da classe `double`, seus valores por convenção estarão no intervalo [0,1].

Imagens Binárias:

É um arranjo lógico de zeros e uns onde os dados são da classe `logical`.

5) Convertendo uma Matriz para uma Imagem de Intensidades.

Sintaxe:

```
I = mat2gray(A,[amin amax])
I = mat2gray(A)
```

Descrição:

`I = mat2gray(A,[amin amax])`

Converte a Matriz `A` para a Imagem de Intensidades `I`. A Matriz `A` deve ser `double`.

A Matriz `I` conterá valores entre 0 (preto) e 1 (branco).

Os Parâmetros `amin` e `amax` são os valores na Matriz `A` que corresponderão a 0 e 1 na Imagem `I`.

I = mat2gray(A)

Estabelece os valores de amin e amax como o mínimo e máximo dos valores da Matriz A.

T_5: Converter as Matrizes A, B, C, e D digitadas anteriormente para Imagens de Intensidades e comentar os resultados.

```
IA = mat2gray(A)
IB = mat2gray(B, [4,30])
IC = mat2gray(C, [15,20])
ID = mat2gray(D, [0,20])
```

6) Mostrando uma Imagem de Intensidades.

Sintaxe:

imshow(I, [low high])

Descrição:

imshow(I, [low high])

Mostra a Imagem I em nível de cinza, especificando os limites dos valores de branco e preto. O parâmetro low (e qualquer valor menor do que ele) corresponde ao preto; o parâmetro high (e qualquer valor maior do que ele) é mostrado como branco. Valores intermediários são mostrados em escala de cinza, usando o número padrão de níveis. Se usado uma matriz vazia entre colchetes ([]) para os parâmetros [low high], a função **imshow** usa [min(I(:)) max(I(:))]; ou seja, o menor valor em I é mostrado como preto e o maior valor em I é mostrado como branco.

T_6: Mostrar a imagem dada pela Matriz E em escala de cinza e observar os resultados. Obs.: Os exercícios T7 e T_8 fazem uso das matrizes definidas a seguir.

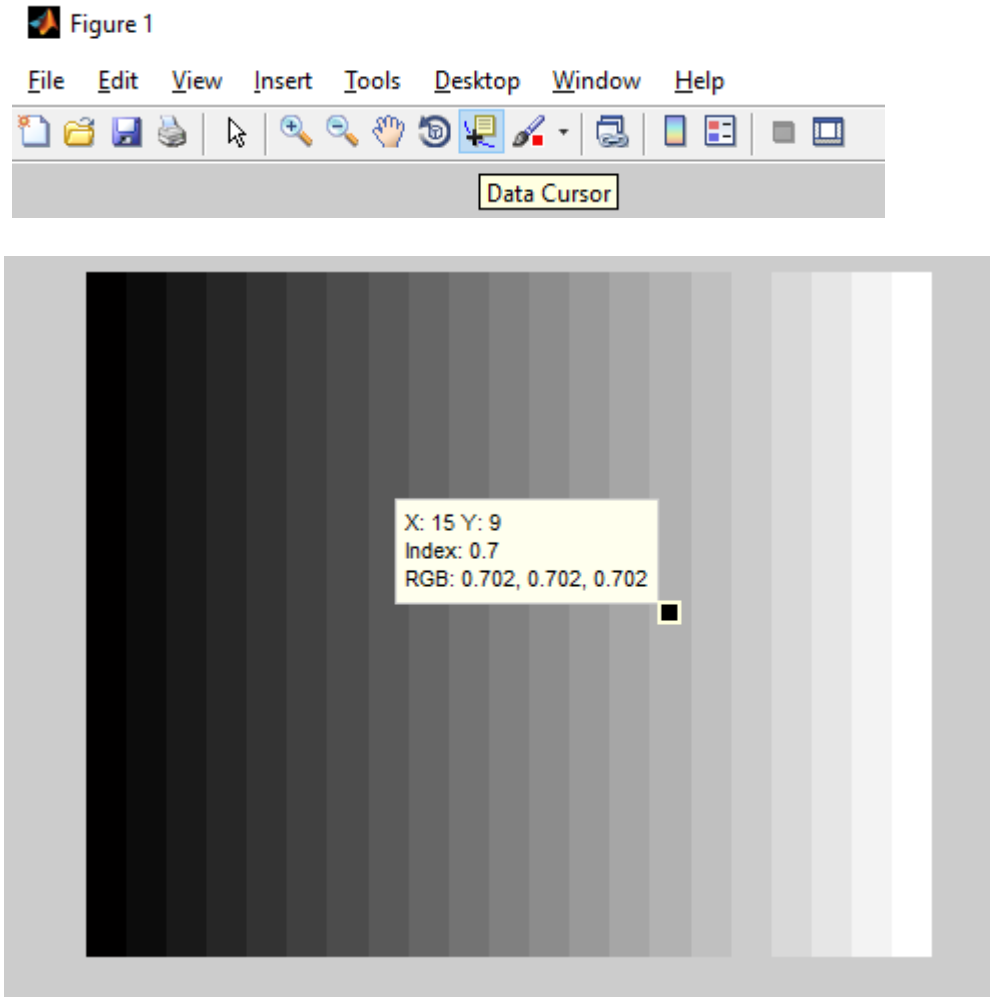
```
E = [0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40
     0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40];

IE = mat2gray(E);

imshow(IE,'initialmagnification','fit')
```

T_7: Verificar os valores de cada pixel diretamente na imagem.

Com o ícone “Data Cursor”, conforme mostra a Figura abaixo, posicionar sobre a Imagem em escala de Cinza e verificar os valores dos níveis. Observe que X refere-se às colunas e Y refere-se às linhas.



Name	Converts Input to:	Valid Input Image Data Classes
im2uint8	uint8	logical, uint8, uint16, and double
im2uint16	uint16	logical, uint8, uint16, and double
mat2gray	double (in range [0, 1])	double
im2double	double	logical, uint8, uint16, and double
im2bw	logical	uint8, uint16, and double

Figura 3 - Funções de conversão entre classes de dados de imagens

7) Convertendo uma Imagem de Intensidades para uma Imagem Binária.

Sintaxe:

```
G = im2bw(f,T)
```

Descrição:
Gera uma Imagem Binária através de Limiarização ("Thresholding"). Os valores serão zero para intensidades menores que `T` e um para os outros pixels. O valor especificado para `T` deve estar no intervalo `[0,1]` independente da classe dos dados de entrada. A Imagem Binária de saída será da classe `logical`. (Se `T` for omitido será considerado `T = 0.5`).

T_8: Converter a Imagem de Intensidades IE para uma imagem Binária.

```
IB = im2bw(IE);
imshow(IB)
whos IE, whos IB
```

Variar o valor de `T` e verificar o resultado. Teste 3 valores diferentes de `T`, mostre os resultados e comente.

Format	Full Name	Variants
'bmp'	Windows Bitmap (BMP)	1-bit, 4-bit, 8-bit, 16-bit, 24-bit, and 32-bit uncompressed images and 4-bit and 8-bit run-length encoded (RLE) images
'cur'	Windows Cursor resources (CUR)	1-bit, 4-bit, and 8-bit uncompressed images
'gif'	Graphics Interchange Format (GIF)	1-bit to 8-bit images
'hdf'	Hierarchical Data Format (HDF)	8-bit raster image data sets, with or without an associated colormap, and 24-bit raster image data sets
'ico'	Windows Icon resources (ICO)	1-bit, 4-bit, and 8-bit uncompressed images
'jpg' or 'jpeg'	Joint Photographic Experts Group (JPEG)	Any baseline JPEG image or JPEG image with some commonly used extensions, including: **Image Type** / **Bitdepth** / **Compression** grayscale / 8- or 12-bit / lossy grayscale / 8-, 12-, or 16-bit / lossless RGB / 24- and 36-bit / lossy or lossless
'pbm'	Portable Bitmap (PBM)	1-bit images using either raw (binary) or ASCII (plain) encoding
'pcx'	Windows Paintbrush (PCX)	1-bit, 8-bit, and 24-bit images
'pgm'	Portable Graymap (PGM)	ASCII (plain) encoding with arbitrary color depth, or raw (binary) encoding with up to 16 bits per gray value
'png'	Portable Network Graphics (PNG)	1-bit, 2-bit, 4-bit, 8-bit, and 16-bit grayscale images; 8-bit and 16-bit indexed images; and 24-bit and 48-bit RGB images
'pnm'	Portable Anymap (PNM)	PNM is not a file format itself. It is a common name for any of the other three members of the Portable Bitmap family of image formats: Portable Bitmap (PBM), Portable Graymap (PGM) and Portable Pixel Map (PPM).
'ppm'	Portable Pixmap (PPM)	ASCII (plain) encoding with arbitrary color depth or raw (binary) encoding with up to 16 bits per color component
'ras'	Sun Raster (RAS)	1-bit bitmap, 8-bit indexed, 24-bit true color and 32-bit true color with alpha data
'tif' or 'tiff'	Tagged Image File Format (TIFF)	Any baseline image, including 1-bit, 8-bit, and 24-bit uncompressed images; 1-bit, 8-bit, and 24-bit images with packbits compression; 1-bit images with CCITT compression; and 16-bit grayscale, 16-bit indexed, and 48-bit RGB images
'xwd'	X Windows Dump (XWD)	1-bit and 8-bit ZPixmaps, XYBitmaps, and 1-bit XYPixmaps

Figura 4 - Tipos de arquivos suportados

8) Lendo Imagens de Arquivos.

Imagens podem ser lidas pelo Matlab através da função *imread:*

Sintaxe:

```
F = imread('arquivo.tipo')
```

A Figura 4 mostra os tipos de arquivos de imagens suportados pelo Toolbox de Processamento de Imagens do Matlab e as variantes de cada um.

T_9: Ler e mostrar imagens verificando informações sobre elas.

(Obs: Se o arquivo não estiver no diretório de trabalho é preciso informar o caminho, ou salvar a imagem no diretório de trabalho)

```
img = imread('rose_gray.tif');
```

Obs.: o ponto e vírgula (;) impede que os valores sejam mostrados na Janela de Comando do Matlab. Use isso quando trabalhar com imagens!

Verificar o número de linhas e colunas da imagem:

```
size(img)
```

Atribuir os valores das linhas e colunas às variáveis M e N:

```
[M,N] = size(img)
```

Mostrar as informações da Matriz (Imagem):

```
whos img
```

Mostrar a Imagem:

```
figure, imshow(img)
```

Ler uma segunda imagem de arquivo e mostrar as duas imagens lidas:

```
img2 = imread('chestxray_gray.jpg');
whos img2
figure, imshow(img2)
```

8) Indexando Imagens.

Como imagens são matrizes, os esquemas de indexação de matrizes podem ser usados diretamente nas imagens.

T_10: Alterar imagens através da indexação.

O arquivo *'chestxray_gray.jpg'* é uma imagem em nível de cinza de 8 bits, classe `uint8`, tamanho de 206 x 499 pixels. Digitar os comandos e verificar o que cada esquema de indexação faz. Explique e comente os resultados de cada um deles.

```
g = imread('chestxray_gray.jpg');
figure, imshow(g)
whos g

fp = g(end:-2:1, :);
figure, imshow(fp)

fl = g(:, end:-2:1);
figure, imshow(fl)

fc = g(10:199, 271:470);
figure, imshow(fc)

fs = g(1:2:end, 1:2:end);
figure, imshow(fs)

figure, plot(g(132,:))
```

9) Arranjos Padrões.

O Matlab pode gerar arranjos padrões que são úteis em diversas aplicações:

- `zeros(M,N)` → matriz de zeros da classe `double`.
- `ones(M,N)` → matriz de uns da classe `double`.
- `true(M,N)` → matriz de uns da classe `logical`.
- `false(M,N)` → matriz de zeros da classe `logical`.
- `magic(M)` → quadrado mágico MxM. A soma dos elementos ao longo de qualquer linha, qualquer coluna ou da diagonal principal é sempre a mesma.
- `rand(M,N)` → matriz de números aleatórios uniformemente distribuídos no intervalo [0,1].
- `randn(M,N)` → matriz de números aleatórios normalmente distribuídos (Gaussiana) com média 0 e variância 1.

T_11: Verificar os seguintes arranjos padrões gerados pelo Matlab.

```
Z = zeros(1, 10)
whos Z

U = ones(4)
whos U

M = magic(5)
whos M
sum(M(:,3)), sum(M(1,:))

R1 = rand(2, 2)
R2 = randn(2, 2)
```

Explique cada um desses comandos e comente os resultados!