MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
CAMPUS SÃO JOSÉ – SANTA CATARINA

Oficina de MATLAB – Nível Básico
Aula 4

Prof. Jeremias Stein Rodriguês
Aluna bolsista: Stephany Padilha Guimarães

Programação no MATLAB

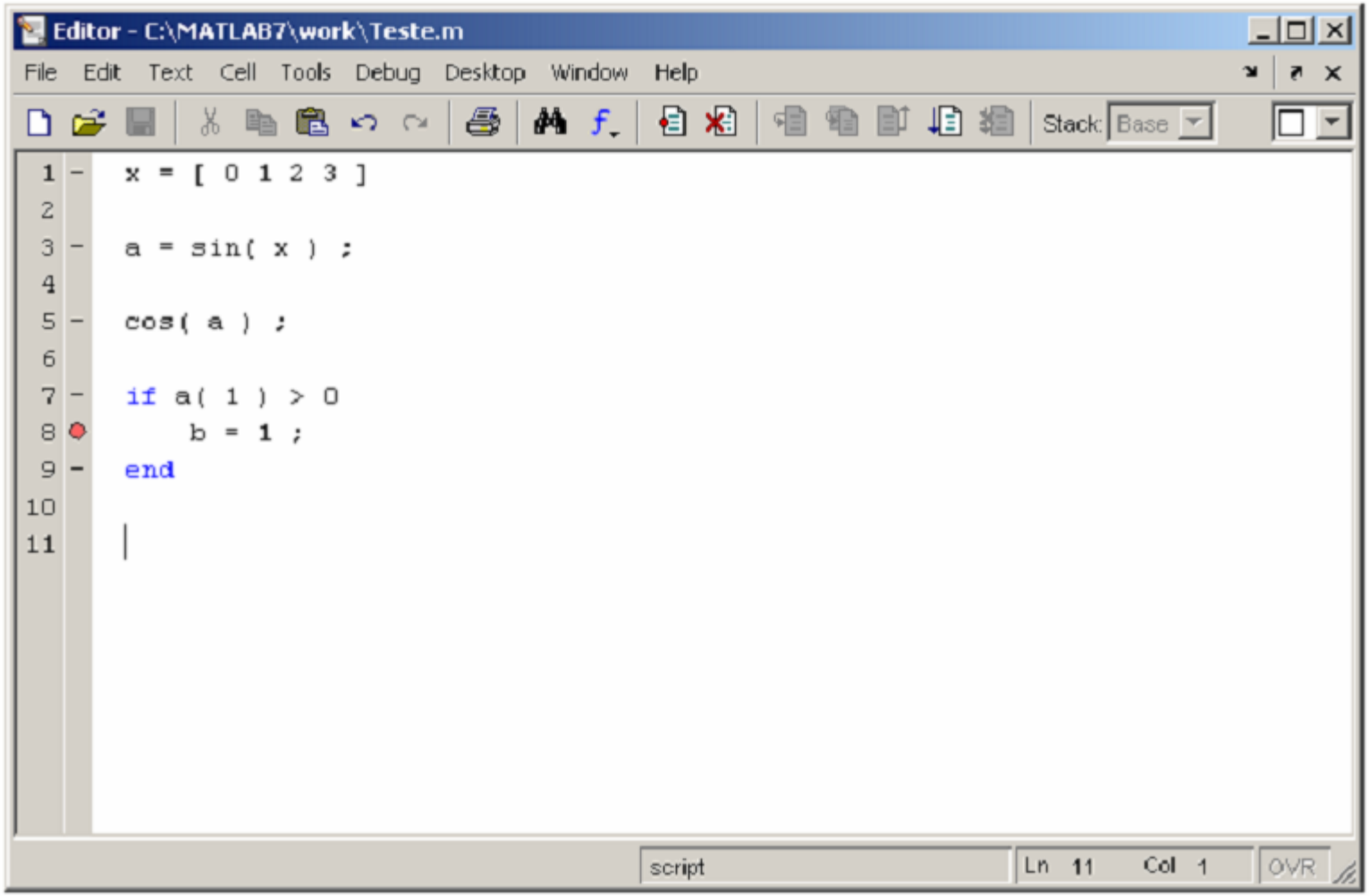
- Para implementar códigos completos e criar programas que sejam funcionais, é necessária a utilização de alguns recursos que o MATLAB fornece.

Editor de Texto

- Para a construção de códigos no MATLAB podemos usar o editor de textos, no qual podemos escrever uma sequência de comandos podendo alterar os comandos previamente utilizados (diferente do ambiente de trabalho).

- O editor de textos pode ser acessado clicando no botão abaixo da aba “file” no MATLAB.

Editor de Texto



Editor de Texto

- Para construir códigos precisamos conhecer os operados lógicos, as estruturas de repetição e condicionais do MATLAB.

- Lembre que tudo escrito depois de “%” é considerado um código para o MATLAB.

Operadores Lógicos

Operador	Significado
<	Menor que
<=	Menor que ou igual a
>	Maior que
>=	Maior que ou igual a
==	Igual
~=	diferente

Operadores Lógicos

Operador	Significado
&	Operador E
|	Operador OU
~	Operador de Negação

Comandos Úteis

- **input ‘ ’:** Permite requisitar (com prompt) fornecimento de dados pelo teclado.

- **disp ‘ ’:** Exibe o conteúdo de uma variável, sem mostrar o seu nome. Também pode ser utilizado para imprimir uma frase no ambiente de trabalho.

- **Sprintf ‘ ’:** Grava dados formatados em uma única linha.

IF e ELSE

- O comando **if** (se) executa o comando “bloco1” somente se a “condição” for satisfeita. Se a condição não for satisfeita ele executa o comando “bloco2” por causa do comando **else** (caso contrário).

```
>>      If <condição>
>>      <bloco1>
>>      else
>>      <bloco2>
>>      end
```

IF e ELSE

- **Exemplo:** criar uma estrutura com **if** e **else** que calcule os valores da função $f(x) = \begin{cases} -x^2 - 3, & x < 0 \\ x + 1, & x \geq 0 \end{cases}$

IF e ELSE

- **Exemplo:** criar uma estrutura com **if** e **else** que calcule os valores da função $f(x) = \begin{cases} -x^2 - 3, & x < 0 \\ x + 1, & x \geq 0 \end{cases}$

```
>>      x=input('a variável x é igual a:')
>>      If  x<0
>>      -x.^2-3
>>      else
>>      x+1
>>      end
```

ELSEIF

- Podemos ainda criar mais de uma condição usando o comando **elseif** (caso contrário, se).

```
>>      If <condição1>
>>      <bloco1>
>>      elseif <condição2>
>>      <bloco2>
>>      elseif <condição3>
>>      <bloco3>
>>      end
```

ELSEIF

```
>>      If <condição1>
>>      <bloco1>
>>      elseif <condição2>
>>      <bloco2>
>>      elseif <condição3>
>>      <bloco3>
>>      end
```

- No comando acima Executa bloco 1 somente se a condição 1 for verdadeira. Caso contrário, executa bloco 2 se a condição 2 for verdadeira. E assim por diante.

ELSEIF

Exemplo: criar uma estrutura com **if** e **else** que calcule os valores da função $f(x) = \begin{cases} -x^2 - 3, & x < -3 \\ x + 1, & x \geq 2 \end{cases}$

ELSEIF

Exemplo: criar uma estrutura com **if** e **else** que calcule os valores da função $f(x) = \begin{cases} -x^2 - 3, & x < -3 \\ x + 1, & x \geq 2 \end{cases}$

```
>>      If  x < -3
>>      -x.^2 - 3
>>      elseif x >= 2
>>      x+1
>>      elseif -3 <= x < 2
>>      disp(‘ não faz parte do domínio da função ’)
>>      end
```

IF e ELSE

Exemplo: fazer uma função que diz se um número dado é par ou ímpar (dica: usar a função mod).

IF e ELSE

Exemplo: fazer uma função que diz se um número dado é par ou ímpar (dica: usar a função mod).

```
>>      x = input(‘ x é o número: ’)
>>      If mod( x , 2) ==0
>>      sprintf(‘ x é par ‘)
>>      else
>>      sprintf(‘ x é ímpar ‘)
>>      end
```

WHILE

- Para executar o mesmo bloco de comandos mais de uma vez, usamos as estruturas de loop (laço). No caso da estrutura **while**, o “bloco” é executado enquanto a condição for verdadeira.

- **OBS:** se a condição for sempre verdadeira, o bloco rodará eternamente, até que alguém aborte o programa.

WHILE

Exemplo: criar uma estrutura com **while** que peça dois números (x e y), enquanto o x for menor que o y adicionar um a x e subtrair um de y.

WHILE

Exemplo: criar uma estrutura com **while** que peça dois números (x e y), enquanto o x for menor que o y adicionar um a x e subtrair um de y.

```
>>    x = input(‘ valor de x: ‘); y = input(‘ valor de y: ‘);
>>    while x<y
>>    x = x + 1
>>    y = y - 1
>>    pause(2)
>>    end
```

WHILE

Exemplo: faça um programa que leia um número e calcule o seu fatorial. O programa deve exibir um erro caso o número seja negativo.

WHILE

```
>>      i=1; prod=1;
>>      n=input('Digite um valor n para o qual se deseja saber o fatorial:');
>>      if n<0
>>      error('n deve ser nao negativo.')
>>      else
            while i<=n
            prod=prod*i;
            i=i+1;
 >>     end
>>      disp(prod)
>>      end
```

FOR

- **for:** Repete comandos por um número de vezes especificado

```
>>      for <variável> = <início>:<passo>:<fim>
>>      <bloco1>
>>      ...
>>      <blocoN>
>>      end
```

FOR

Exemplo: crie um comando que toma os números de 1 até 9 e calcula o quadrado de cada um desses números.

FOR

Exemplo: crie um comando que toma os números de 1 até 9 e calcula o quadrado de cada um desses números.

```
>>      for i = 1 : 5
>>      x(i) = i^2;
>>      end
>>      disp(x)
```

FOR

Exemplo: fazer uma função que gera a sequência de Fibonacci até o n-ésimo termo (onde n é a variável de entrada).

FOR

Exemplo: fazer uma função que gera a sequência de Fibonacci até o n-ésimo termo (onde n é a variável de entrada).

```
>>      n = input(‘ ó valor de n é: ‘)
>>      x(1) = 1; x(2) = 1;
>>      for i = 3 : n
>>      x(i) = x(i - 2) + x(i - 1);
>>      end
>>      disp(x)
```

Exercício

Fazer um código que vai aplicar o método de Newton para a função $f(x) = x^3 - 5x^2 + x + 3$. Lembre que no método de Newton precisamos de um valor inicial aproximado (x_0) e um erro (ϵ) para aplicar a iteração $x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$.

Exercício

```
>>      xo = input(‘ digite o valor de xo: ');
>>      E=input(‘ insira o valor para o erro: ');
>>      while (x1 - xo) > E
>>      xo = x1;
>>       x1 = xo - (xo^3 - 5*xo^2 + xo + 3)/(3*xo^2 - 10*xo + 1);
>>      end
>>      disp(‘ A raiz é ');
>>      disp(x1)
```