

Introdução à Ciência da Computação

Unidade III

Programação de Computadores com Octave

Uso do Octave no Modo de Programação

Prof. Roberto M. de Faria/DSC/UFCG

Criando Programas no Octave

- Arquivos-M
- Saída de Dados
- Entrada de Dados
- Comandos Condicionais
- Comandos de Repetição

Arquivos-M

- Para criar um *arquivo-m* no Octave, utiliza-se um editor de linhas muito semelhante ao “*Bloco de Notas*” do *Windows*, que pode ser acessado de duas maneiras: com o comando `edit` na “Área de Comandos”, clicando-se na aba “Editor”, embaixo da “Área de Comandos”
- Um *arquivo-m*, quando criado, é alocado no “Diretório de Trabalho” mostrado na parte de cima da tela do Octave
- Para se realizar a mudança do Diretório Atual, utiliza-se o “Navegador de Arquivos” ou o comando `cd`

Saída de Dados

- Octave possui várias funções para a apresentação de valores
 - A função `disp()` apresenta o resultado de uma expressão
 - `>> disp('Oi!')`
 - `>> disp(pi)`
 - `>> disp(x^2)`
 - A função `fprintf()` imprime valores na saída
 - `--> fprintf('O valor da Variável é %d\n', var)`

Saída de Dados

- Especificadores de conversão mais comuns
- `%d` ou `%i` → inteiro
- `%f` → real de ponto flutuante
- `%c` → caractere
- `%s` → string
- `%e` → notação científica
- O caractere “`\n`” provoca o cursor ir para a linha seguinte

Entrada de Dados

- A função “`input()`” recebe um valor digitado do teclado pelo usuário e possui dois formatos:
 - Neste formato o usuário digita uma expressão que avaliada e o resultado enviado para a variável

```
>> variavel = input('Digite o valor para a variável:')
```
 - Neste outro formato o usuário digita um valor e este é enviado para a variável
 - ```
>> variavel = input('Digite o valor para a variável:', 's')
```

# Exemplo de Programa

- Programa para calcular a área de um círculo:

```
% Este programa calcula a área de um
% círculo a partir de seu raio em
% centímetros

printf('\nCálculo da área de um círculo:\n');

raio = input('Informe o comprimento do raio em
cm: ');

area = pi * raio ^ 2;

printf('\nA área do círculo de raio %.2f cm é
%.2f cm²!\n\n', raio, area);
```

# Exercícios

Escreva programas que:

- 1) Calcule a área de um triângulo a partir de suas base e altura
- 2) Calcule a distância entre dois pontos no plano a partir das coordenadas de cada ponto
- 3) Calcule o volume de um cone a partir de seus raio da base e altura
- 4) Calcule a superfície de um cilindro a partir de seus raio da base e altura
- 5) Calcule o volume de uma esfera oca a partir dos raios das paredes interna e externa



# Expressões Relacionais

- **Expressões relacionais** são aquelas cujas avaliações recebem o valor **verdadeiro** (*diferente de 0*) ou **falso** (*igual a 0*)
- São chamadas também de **expressões Booleanas** ou **expressões Lógicas**
- Essas expressões podem usar **operadores relacionais** e/ou **operadores lógicos**, que operam sobre **operandos lógicos**
- Um **operando lógico** possui valor **verdadeiro** (*diferente de 0*) ou **falso** (*igual a 0*)
- As **expressões relacionais** definem **condições** para comandos do Octave

# Operadores Relacionais

- Operadores relacionais

- $==$  → igual a
- $\neq$  → diferente de
- $<$  → menor que
- $>$  → maior que
- $\leq$  → menor ou igual a
- $\geq$  → maior ou igual a

- Exemplos

$>> 3 < 5$

# Operadores Relacionais

- Mais exemplos

```
>> 2 > 9
```

```
>> class(ans)
```

```
>> result = 5 < 7
```

```
>> result + 1
```

```
>> 5 + 3 > 7 - 2
```

```
>> a = 10
```

```
>> b = 20
```

```
>> a + 10 ~= b
```

```
>> b - 10 <= 10 + a
```

# Operadores Lógicos

- Operadores lógicos

- `&&` → e
- `||` → ou
- `~` → não

- Exemplos

```
>> 3 > 5 && 9 >= 3
```

```
>> 33 > 5 || 99 <= 23
```

```
>> 9 || 7 - (5 + 2)
```

```
>> 0 && 10 - (5 + 2)
```

# Operadores Lógicos

- Tabelas verdade

| E ( && ) | V | F |
|----------|---|---|
| V        | V | F |
| F        | F | F |

| Ou (    ) | V | F |
|-----------|---|---|
| V         | V | V |
| F         | V | F |

| Não ( ~ ) |   |
|-----------|---|
| V         | F |
| F         | V |

# Regras de Precedência de Operadores

## Operadores

## Precedência

parênteses ()

*maior*

potência ^

negação -, não ~ (*unários*)

multiplicação \*, divisão /, \

adição +, subtração -

relacionais <, <=, >, >=, ==, ~=

and &&

ou ||

atribuição =

*menor*

# Comandos Condicionais

- Comando **if**

```
if condição
 ações
end
```

- As ações serão executadas se a condição for **verdadeira** (*diferente de 0*)

- Exemplo

```
a = input('Digite o valor de a: ');
b = input('Digite o valor de b: ');
if a ~= 0
 c = b / a;
end
```

# Comandos Condicionais

- Comando **if** com a cláusula **else**

```
if condição
 ações_1
else
 ações_2
end
```

- Serão executadas as `ações_1` após a condição, se a condição for **verdadeira** (*diferente de 0*), caso contrário, se for **falsa** (*igual a 0*), serão executadas as `ações_2` após o `else`



# Comandos Condicionais

- Exemplo de **if** com a cláusula **else**

```
a = input('Digite o coeficiente a: ');
b = input('Digite o coeficiente b: ');
c = input('Digite o coeficiente c: ');
delta = b ^ 2 - 4 * a * c;
if delta < 0
 fprintf('Não há raízes reais!\n');
else
 fprintf('As raízes são reais!\n');
end
```

# Comandos Condicionais

- Comandos **if**'s aninhados

```
- sexo = input('Informe o sexo da pessoa(M ou F): ', ...
 's');
idade = input('Informe a idade da pessoa: ');
if sexo == 'f' || sexo == 'F'
 fprintf('Dispensada de alistamento militar!\n');
else
 if idade == 18
 fprintf('Precisa alistar-se!\n');
 else
 fprintf('Não precisa alistar-se!\n');
 end
end
end
```

# Comandos Condicionais

- A cláusula **elseif**

```
if condição_1
 ações_1
elseif condição_2
 ações_2
elseif condição_3
 ações_3
% etc. ... qualquer quantidade de elseif's
else
 ações_n
end
```

# Comandos Condicionais

- Exemplo

```
fprintf('Programa para classificar');
fprintf(' um caractere lido:\n');
carac = input('Digite um caractere: ', 's');
if carac >= 'A' && carac <= 'Z'
 fprintf('O caractere é uma letra maiúscula!\n');
elseif carac >= 'a' && carac <= 'z'
 fprintf('O caractere é uma letra minúscula!\n');
elseif carac >= '0' && carac <= '9'
 fprintf('O caractere é um dígito!\n');
elseif carac == ' '
 fprintf('O caractere é um espaço em branco!\n');
else
 fprintf('É um caractere especial!\n');
end
```

# Comandos Condicionais

- Comando **switch**

```
switch expressão
```

```
 case valor_1
```

```
 ações_1
```

```
 case valor_2
```

```
 ações_2
```

```
% etc. ... qualquer quantidade de case's
```

```
 otherwise
```

```
 ações_n
```

```
end
```

# Comandos Condicionais

- Funcionamento do comando **switch**
  - No início do `switch` a expressão é comparada com cada `valor_i` de `case`
  - Se houver coincidência do valor da expressão com algum `valor_i` de `case`, a execução inicia na `ação_i` e encerra antes do próximo `case` ou do `otherwise`
  - Se não houver coincidência com nenhum `valor_i` de `case`, as `ações_n` após o `otherwise` são executadas

# Comandos Condicionais

- Exemplo

```
ordem = input('Digite a ordem(1, 2 ou 3): ');
switch ordem
 case 1
 fprintf('Primeiro!\n');
 case 2
 fprintf('Segundo!\n');
 case 3
 fprintf('Terceiro!\n');
 otherwise
 fprintf('Ordem inválida!\n');
end
```

# Exercícios

1) Faça um programa que receba valores para as variáveis ***a*** e ***b***. Se o valor de ***a*** for maior que o valor de ***b***, troque os valores dessas variáveis, entre si. Use um comando `if` na solução.



# Exercícios

2) As leituras de pressão arterial sistólica e diastólica são encontradas quando o coração está bombeando e o coração está em repouso, respectivamente. Um experimento biomédico está sendo realizado apenas para os participantes cuja pressão arterial é ideal. Esta é definida como uma pressão arterial sistólica menor ou igual a 120 e uma pressão arterial diastólica menor ou igual a 80. Escreva um programa que irá pedir as pressões sistólica e diastólica de uma pessoa e, em seguida, imprima uma mensagem dizendo se essa pessoa é, ou não, um candidato para este experimento. Use um comando `if` com a cláusula `else` na solução.

# Exercícios

3) Faça um programa que receba os coeficientes (***a***, ***b*** e ***c***) de uma equação do segundo grau e mostre, dependendo do seu ***delta***: uma raiz real, duas raízes reais ou uma mensagem informando que não existe raízes reais para esta equação. Use comandos `if`'s aninhados na solução.

# Exercícios

4) Faça um programa que calcule a área de uma figura geométrica plana (círculo, triângulo, quadrado ou retângulo). O usuário escolherá a figura por meio de um menu, com opções numéricas, e, em seguida, o programa solicitará os dados necessários, para então, mostrar o valor da área da figura. Use cláusulas `elseif`'s na solução.

# Exercícios

5) Faça um programa que receba um número entre 1 e 99, inclusive, e mostre seu numeral ordinal correspondente. Use comandos `switch's` na solução.

# Comandos de Repetição

- Existem dois comandos de repetição (laços) diferentes no Octave: o comando **for** e o comando **while**
- O **for** é usado como um laço contado, e o **while** é usado como um laço condicional
- Em muitas linguagens de programação, laços para manipulação de elementos em um vetor ou matriz é um conceito fundamental
- No Octave, laços para processar elementos de vetores e matrizes, geralmente não é necessário, em vez disso, "código vetorizado" é usado, o que significa substituir os laços de manipulação de vetores e matrizes pelo uso de operadores e funções internas

# Comandos de Repetição

- O comando **for**, ou o laço **for**, é usado quando é necessário repetir comandos em um programa ou função, quando é conhecido previamente quantas vezes os comandos deverão ser repetidos
- As instruções que são repetidas são chamadas de ***ações do laço***.
- Chama-se a variável que é usada para a contagem das iterações do laço, de ***variável do laço*** ou ***variável do iterador***.
- Por exemplo, a variável pode iterar os inteiros de 1 a 5 (por exemplo, 1, 2, 3, 4 e em seguida 5)

# Comandos de Repetição

- A forma geral do laço **for** é:

```
for variável_do_laço = intervalo
 ações_do_laço
end
```

- **intervalo** é o intervalo de valores através do qual a **variável\_do\_laço** itera as **ações\_do\_laço**, que consiste de todas as instruções até o **end**
- As **ações\_do\_laço** são recuadas para torná-las mais fácil de ver
- O **intervalo** pode ser especificado mais facilmente, utilizando o ***operador de dois pontos***.

# Comandos de Repetição

- Por exemplo, será impressa uma coluna de números de 1 a 10:

```
for contador = 1:10
 fprintf('%d\n', contador);
end
```

- O que imprimirá este laço?

```
for controle = 20:-3:1
 fprintf('Eu não devo desobedecer ');
 fprintf('meus pais!\n');
end
```



# Comandos de Repetição

- Outro exemplo:

```
fprintf('\nConta os tipos de 10 caracteres lidos\n');
fprintf('como minúsculo, maiúsculo ou outro:\n\n');
conta_maiusc = 0; conta_minusc = 0; conta_outro = 0;
for vezes = 1:1:10
 carac = input('Digite um caractere: ');
 if carac >= 'A' && carac <= 'Z'
 conta_maiusc = conta_maiusc + 1;
 elseif carac >= 'a' && carac <= 'z'
 conta_minusc = conta_minusc + 1;
 else
 conta_outro = conta_outro + 1;
 end
end
fprintf('Maiúsculas = %d, Minusculas = %d e Outros = %d\n', ...
conta_maiusc, conta_minusc, conta_outro);
```

# Exercícios

- 1) Faça um programa que calcule e imprima o IMC de 5 pessoas
- 2) Modifique o programa anterior para que ele execute para um número determinado de pessoas
- 3) Faça um programa que calcule e mostre o fatorial de um número natural
- 4) Faça um programa que some os números naturais até 20

# Exercícios

- 5) Modifique o programa anterior para que some os números naturais até  $N$
- 6) Faça um programa que some os números naturais no intervalo de  $M$  a  $N$ , inclusive
- 7) Faça um programa que calcule e mostre a soma de 10 números inteiros
- 8) Modifique o programa anterior para que calcule e mostre a soma de  $N$  números inteiros

# Exercícios

- 9) Faça um programa que crie um vetor com 10 valores reais aleatórios, mostre os dados desse vetor com um valor em cada linha
- 10) Estenda o programa anterior para ordenar os dados do vetor em ordem crescente e mostrá-los ordenados
- 11) Faça um programa para desenhar as seguintes figuras, dado o número de linhas:

|      |      |         |         |
|------|------|---------|---------|
| a) * | b) * | c) **** | d) **** |
| **   | **   | ***     | ***     |
| ***  | ***  | **      | **      |
| **** | **** | *       | *       |

# Comandos de Repetição

- O comando `while` é usada como o ***laço condicional*** em FreeMat
- Ele é usado para repetir uma ***ação*** quando, previamente, não se sabe quantas vezes a ***ação*** será repetida
- A forma geral do comando `while` é:

```
while condição
 ação
end
```

- A ***ação***, que consiste de qualquer número de comandos, é executada enquanto a ***condição*** for ***verdadeira***

# Comandos de Repetição

- O modo como funciona, é que primeiro a `condição` é avaliada
- Se a `condição` é logicamente **verdadeira**, a ação é executada
- Até aí, o comando `while` é igual a um comando `if`.
- No entanto, nesse ponto a condição é avaliada novamente
- Se é **verdadeira**, a ação é executada novamente e assim por diante
- Então, eventualmente, algo na ação tem que mudar alguma coisa na `condição` para assim, tornar-se **falsa**
- A `condição` deve, eventualmente, tornar-se **falsa** para evitar um **laço infinito**
- Neste caso, **Ctrl-C** pode forçar a interrupção do laço e, em seguida, pressione  (Stop), para parar o programa

# Comandos de Repetição

- Outro exemplo com `while`:

```
clc
clear all
fprintf('\nPrograma que recebe um inteiro positivo, ');
fprintf('mostra-o e, em seguida, \nmostra os seus ');
fprintf('dígitos separados por espaços em branco.\n\n');
inteiro = input('Digite um número positivo: ');
fprintf('\nNúmero lido = %d\n\n', inteiro);
fprintf('Os dígitos são:\n\n');
while inteiro > 0
 fprintf('%d ', mod(inteiro, 10));
 inteiro = fix(inteiro / 10);
end
fprintf('\n\n');
```

# Comandos de Repetição

- Exemplo com `while`:

```
clc
clear all
fprintf('Programa que soma inteiros positivos e ');
fprintf('para quando encontra um negativo ou 0:\n');
soma = 0;
inteiro = input('Informe um inteiro a somar: ');
while inteiro > 0
 soma = soma + inteiro;
 inteiro = input('Informe um inteiro a somar: ');
end
fprintf('\nA soma dos inteiros positivos é %d\n\n', ...
soma);
```



# Exercícios

- 1) Modifique o programa exemplo com `while` para somar valores e parar quando aparecer um múltiplo de 5.
- 2) Modifique o programa anterior para parar quando aparecer um múltiplo de N
- 3) Faça um programa que calcule o MDC de dois inteiros positivos (Exemplo do método: MDC de 32 e 18)

| Dividendo |   | Divisor  |   | Resto                            |
|-----------|---|----------|---|----------------------------------|
| 32        |   | 18       |   | 14                               |
| 18        | ↙ | 14       | ↙ | 4                                |
| 14        | ↙ | 4        | ↙ | 2                                |
| 4         | ↙ | 2        | ↙ | 0                                |
|           |   | ↓<br>MDC |   | ↓<br>Para quando<br>igual a zero |

# Exercícios

- 4) Modifique o programa anterior para que no mesmo `printf` que mostra o MDC, mostrar também os valores sobre os quais foi calculado o MDC
- 5) Faça um programa para calcular as raízes de várias equações do 2º. grau e parar quando o coeficiente ***a*** for igual a zero
- 6) Faça um programa que mostre que receba um inteiro positivo e mostre os seus dígitos separados por espaços em branco

# Exercícios

7) Modifique o programa anterior para que mostre a soma de dígitos de vários números recebidos e pare quando encontrar um número par

8) Um “Número de Armstrong” de  $n$  dígitos é aquele que é igual a soma das potências de seus dígitos com expoente  $n$

Exemplo:  $153 = 1^3 + 5^3 + 3^3 = 1 + 125 + 27$

Mostre os números de Armstrong menores que M

# Exercícios

- 9) Faça um programa para mostrar se um número positivo é ou não primo, lembrando que 1 é divisor de qualquer número e que os divisores de um número encontra-se entre 1 e a metade desse número
- 10) Faça um programa que gere um número aleatório entre 1 e 100 e aceite palpites do usuário até que o mesmo acerte esse número
- A cada palpite, o programa informa se o palpite foi maior, menor ou se o usuário acertou o número

# Criando Funções

# Gerando Gráficos