Capítulo 2 - Conceitos Básicos

Matemática simples

Exemplo: *Carlos vai à papelaria e compra 4 borrachas a R$.25 cada, 6 blocos a R$.52 cada e 2 fitas adesivas a R$.99 cada. Quantos itens comprou e quanto gastou?*

Matemática simples

Exemplo: *Carlos vai à papelaria e compra 4 borrachas a R$.25 cada, 6 blocos a R$.52 cada e 2 fitas adesivas a R$.99 cada. Quantos itens comprou e quanto gastou?*

Usando calculadora

```
4 + 6 + 2 = 12 items
4 x 25 + 6 x 52 + 2 X 99 =  610 centavos
```

Matemática simples

Exemplo: *Carlos vai à papelaria e compra 4 borrachas a R$.25 cada, 6 blocos a R$.52 cada e 2 fitas adesivas a R$.99 cada. Quantos itens comprou e quanto gastou?*

Usando calculadora

```
4 + 6 + 2 = 12 items
4 x 25 + 6 x 52 + 2 X 99 =  610 centavos
```

Usando MATLAB

```
>> 4 + 4 + 2
ans =
      12
>> 4*25 + 6*52 + 2*99
ans =
      610
```

Sobre expressões matemáticas

- Operações aritméticas básicas:

Operação	Símbolo
Adição	$+$
Subtração	$-$
Multiplicação	$*$
Divisão	$/$ ou $\backslash$
Exponenciação	$\hat{}$

- Avaliação de expressões é feita da esquerda para a direita.
- Ordem de precedência dos operadores é a tradicional:

exponenc. > (multiplic. = divisão) > (adição = subt.)

(A ordem de precedência pode ser alterada através do uso de parêntesis)

Matemática simples

Exemplo: *Carlos vai à papelaria e compra 4 borrachas a R$.25 cada, 6 blocos a R$.52 cada e 2 fitas adesivas a R$.99 cada. Quantos itens comprou e quanto gastou?*

Usando MATLAB com variáveis

```
>> erasers = 4
erasers =
     4
>> pads = 6
pads =
     6
>> tape = 2;
>> items = erasers + pads + tape
items =
    12
>> cost = erasers*25 + pads*52 + tape*99
cost =
   610
```

Reutilização de variáveis

- O **MATLAB** armazena os dados anteriores:

```
>> average_cost = cost / items
average_cost =
    50.883
```

- Alterar variáveis não afeta cálculos anteriores.

```
>> items = erasers + pads + tape
items =
    12
>> erasers = 6
erasers =
     6
>> items
items =
    12
```

Sobre variáveis

- **ans**: Variável padrão para saída de dados quando não são usadas variáveis definidas pelo usuário. Ex.
- O **MATLAB** sempre exibe os resultados de uma sentença (comando). Para inibir essa característica, usamos "**;**" no final da sentença. Ex.
- Regras para nomes de variáveis:
  - Diferencia maiúsculas e mínusculas;
  - Enxerga até 31 caracteres;
  - Os nomes devem *começar* com **letra** que pode ser seguida por qualquer número de **letras**, **dígitos** ou ***'_' (sublinhado)*** . Não é permitido o uso de outros caracteres.

Sobre variáveis

- Palavras reservadas.

```
for   end   while   function   return   try
if   elseif   else   case   continue   switch
catch   global   persistent   break   otherwise
```

- Variáveis especiais. Exemplos:

`pi`	A constante π.
`beep`	Faz o computador soar um *beep*.
`i` ou `j`	$\sqrt{-1}$
`inf`	∞
`realmin`	Menor real positivo que pode ser usado.
`realmax`	Maior real positivo que pode ser usado.
`bitmax`	Maior inteiro positivo que pode ser usado.

Sobre variáveis

- Variáveis especiais podem ter seus valores alterados.

```
>> pi
ans =
    3.1416
>> pi = 233e3
pi =
   233000
>> clear pi
>> pi
ans =
    3.1416
```

Ainda sobre variáveis

- Variáveis criadas pelo usuário são incorporadas ao *Workspace* do **MATLAB** .
- A função `clear <var>` remove `var` do *Workspace* se `var` for uma variável definida pelo usuário, ou reestabelece o valor original de `var` se `var` for uma variável especial. Se `var = all` ou for omitida, todas as variáveis definidas pelo usuário são removidas, e todos os valores de variáveis especiais são reestabelecidos.

Pontuações especiais no MATLAB

- **Comentários:** O símbolo % diz ao MATLAB que ignore o texto que o segue.

```
>> pi % exibe o valor da constante pi ans = 3.1416
```

- **Múltiplos comandos na mesma linha:** Os comandos devem ser separados por vírgula ou ponto-e-vírgula.

```
>> erasers=4, pads = 6; erasers+pads
erasers =
     4
ans =
     10
```

Quebra de linha

- **Quebra de comandos em linhas diferentes:** Para evitar linhas longas pode-se usar **...** para quebrar a linha:

```
>> average_cost = cost/...
items
average_cost =
   50.8333

>> average_cost = cost...
/items
average_cost =
   50.8333

>> average_cost = cost/it...
ems
??? ems
    |
Error: Missing operator, comma, or semicolon.
```

Aritmética de ponto flutuante

- Os números são representados em aritmética de precisão dupla, usando binário como representação interna.
  - Nem todos os números podem ser representados exatamente;
  - Existem limites para os valores que podem ser representados. Ex.
  - Existe um valor-limite inferior que efetivamente pode ser somado a um número de forma a mudar seu valor.

```
>> format long  % exibe mais dígitos
>> eps  % menor num. que somado a 1 gera num > 1
ans =
     2.220446049250313e-16
```

Conseqüências

- Comutatividade da adição: nem sempre vale!

```
>> 0.42 - 0.5 + 0.08
ans =
    -1.38777878078144e-17
>> 0.08 - 0.5 + 0.42
ans =
     0
>> 0.08 + 0.42 - 0.5
ans =
     0
```

- Argumentos e valores de funções nem sempre precisos!

```
>> sin(0)
ans =
     0
>> sin(pi)
ans =
     1.22464679914735e-16
```

Exibição de números

- Depende do *tipo* do número:
  - Inteiro $\longrightarrow$ exibe como inteiro;
  - Real $\longrightarrow$ com 4 dígitos após a vírgula;
    Se os dígitos significativos estão fora do intervalo acima o resultado é exibido em notação científica (como calculadoras).
- Pode-se modificar o padrão:
  - no submenu **`Preferences`** do menu **`File`**;
  - Na *Command Window* digitando comando apropriado.
    Ex.
- O **MATLAB não** muda a representação interna do número quando há modificação no formato de exibição. Todos os cálculos são feitos com aritmética de precisão dupla.

Modificando formatos de números

Comando	***Exemplo usando*** π
`format short`	3.1416 *5 dígitos*
`format short e`	3.1416e+00 *5 dígitos mais expoente*
`format short g`	3.1416 *melhor entre opções* ***short***
`format long`	3.14159265358979 *16 dígitos*
`format long e`	3.14159265358979e+00 *16 dígitos mais expoente*
`format long g`	3.14159265358979 *melhor entre opções* ***long***

Modificando formatos de números

Comando	***Exemplo usando*** π
`format hex`	400921fb54442d18 *hexadecimal com ponto flutuante*
`format bank`	3.14 *2 dígitos*
`format +`	+ *positivo(+), negativo(-) ou zero(0)*
`format rat`	355/113 *aproximação racional*

`format debug`	`Structure address = 26c008`
	`m = 1`
	`n = 1`
	`pr = c60c38`
	`pi = 0`
	`3.1416`
	Informação sobre armazenamento interno

Números complexos

- Não há necessidade de tratamento especial.
  - Definir:

```
>> c1=1-2i
c1 =
   1.0000 - 2.0000i
>> c1=1-2j
c1 =
   1.0000 - 2.0000i
>> c2=3*(2-sqrt(-1)*3)
c2 =
   6.0000 - 9.0000i
>> c3=sqrt(-2)
c3 =
         0 + 1.4142i
>> c4=6+sin(.5)*i
c4 =
   6.0000 + 0.4794i
```

Notas:

- $i = j = \sqrt{-1}$;
- **MATLAB** aceita $2i$, mas **não** aceita que se escreva $sin(0.5)i$;

Números complexos

- Não há necessidade de tratamento especial.
  - Manipular:

```
>> c5=c1/c2
c5 =
   0.2051 - 0.0256i
>> c6=(c1+c2)/c3
c6 =
  -7.7782 - 4.9497i
>> c6r=real(c6)
c6r =
   -7.7782
>> c6i=imag(c6)
c6i =
   -4.9497
```

```
>> c1
c1 =
  1.0000 - 2.0000i
>> % Magnitude
>> mag = abs(c1)
mag =
   2.2361
>> % Ângulo em radianos
>> ang = angle(c1)
ang =
   -1.1071
>> % Ângulo em graus
>> deg = ang*180/pi
deg =
   -63.4349
```

Def