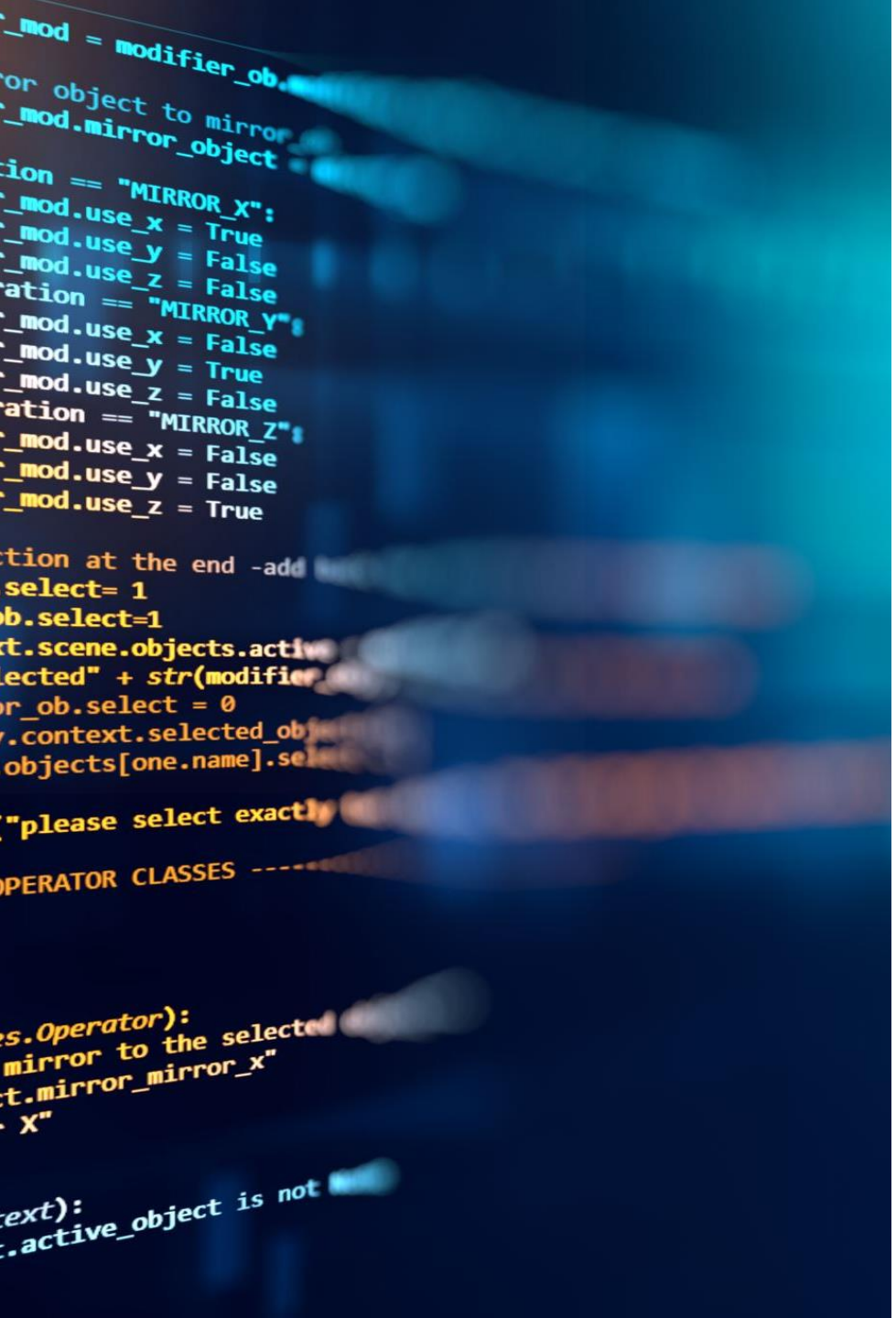


PROGRAMAÇÃO DE COMPUTADORES I

QUARTA PARTE

CAPÍTULO II
CONSTANTES, VARIÁVEIS E TIPOS DE DADOS;

PARTE 3:
TIPO LÓGICO
TIPO COMPLEXO
ENUMERAÇÕES
CONSTANTES

Programação de Computadores I

prof. Marco Villaça

Tipo lógico ou booleano

- Muito simples, assume somente dois valores:
  - verdadeiro ou true e falso ou false
- Variáveis com valores lógicos podem ser parte de expressões lógicas, que usam os operadores lógicos:

Não (Not), Ou (Or) e E (AND)

Tipo lógico ou booleano
Linguagem C

- A linguagem C não tem explicitamente o tipo lógico
- A partir do C99, a linguagem C passou a oferecer um suporte intrínseco para o tipo lógico → <stdbool.h>, que pode ser utilizado pelo C89.
  - O tipo bool, serve para representar os valores lógicos, false e true.
    - ✓ **true e false** podem ser substituídos pelos valores inteiros 0 e 1. **Estes tem como função apenas facilitar a compreensão do código.**
- Operadores lógicos em C:
  - E → && OU → || NOT → !

Tipo lógico ou booleano
Linguagem C

```
#include <stdio.h>
#include <stdbool.h>

int main( )
{
     bool b = false, c = true , a;
     a = b || c;
     printf("A = %d",a);
     return 0;
}
```

SAÍDA: `A = 1`

Tipo lógico ou booleano
Scilab

- Uma variável Scilab pode armazenar também valores lógicos correspondentes a verdadeiro e falso, denotados pelas constantes:
  - %t ou %T → true;
  - %f ou %F → false.
- Variáveis com valores lógicos podem ser parte de expressões lógicas, que usam os operadores lógicos:
  - E → & OU → | NOT → ~

Tipo lógico ou booleano
Scilab

```
-->a = %T;
-->b = %F;
-->~a
 ans  =
  F
-->a|b
 ans  =
  T
```

Tipo complexo em C

- A linguagem C não tem explicitamente o tipo complexo
- A partir do C99, a linguagem C passou a oferecer um suporte intrínseco para o tipo complexo → <complex.h> , que pode ser utilizado no C89.
- Para declarar uma variável complexa real de dupla precisão utiliza-se `double complex` e a constante imaginária $\sqrt{-1}$ é representada por I.

Tipo complexo em C
Algumas funções

- Para uso com double:

`cabs(double complex`	Módulo de um número complexo
`carg(double complex)`	Argumento de um número complexo
`cimag(double complex)`	Parte imaginária de um número complexo
`creal(double complex)`	Parte real de um número complexo
`conj(double complex)`	Conjugado de um número complexo
`cexp`	Exponencial complexa

Tipo complexo em C
Exemplo de uso

```
#include <stdio.h>
#include <complex.h>
#include <math.h>
int main() {
      double complex z;
      double x=3,y=3;
      z = (x + y * I);
      z=z*z;
      printf("%.2f <%.2f \n", cabs(z), carg(z)*180/M_PI);
      return 0;
}
```

Saída: 18.00 <90.00

Tipo complexo no Scilab

- No Scilab as operações com variáveis complexas são tão fáceis como as operações com variáveis reais.

- A constante imaginária $\sqrt{-1}$ é representada por %i.

- Exemplos

```
-->x = 3 + 4*%i;

-->y = 2 - 3*%i;

-->z = x * y

 z  =

    18. - i

-->w = x + y

 w  =

    5. + i
```

Tipo complexo no Scilab
Algumas funções

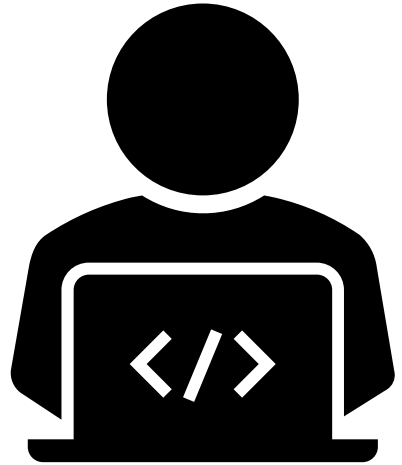
abs(x):	Módulo de um número complexo
real(x):	Parte real de um número complexo
imag(x):	Parte imaginária de um número complexo
atan(imag(x), real(x)):	Argumento de um número complexo
conj(x):	Conjugado de um número complexo

EXEMPLO 1

- Elaborar um programa em C e Scilab que faça a conversão de um número complexo na forma polar para a forma retangular.
- Entrada de dados pelo teclado.
- Utilizar a notação de Euler.
- Em C use as funções da biblioteca `complex`

Exercício 1

- Elaborar um programa em C que calcule as raízes de uma equação de segundo grau com suporte para resultados complexos.
- Entrada de dados pelo teclado.
- Usar variáveis complexas
- Utilizar a função `csqrt`



Outros tipos de variáveis

- Os tipos **compostos** e **definidos pelo usuário** serão apresentados no momento apropriado, assim como os atributos das variáveis **tempo de vida** e **escopo**
- Antes, porém de passar ao próximo capítulo, tratar-se-á das constantes e dos operadores de atribuição reduzida

Constantes numéricas em C

- Constantes inteiras
  - ✓ Valores numéricos sem ponto decimal, precedidos ou não por um sinal.

    Ex: 5, -3, 0, 1955, 0x25(hexadecimal)
  - ✓ Ocupam 4 bytes.
- Constantes Reais
  - ✓São consideradas do tipo double (8 bytes);

    Ex: 4.5, -3.4E+10.
  - ✓ Para constantes single, utilizar o sufixo F

    Ex: 4.5F, -3.4E+10F.

Declaração de variáveis como constantes em C

- Vantagem de guardar constantes em variáveis
  - ✓ Tentativa de alteração gera erro
- Em C:
  - ✓ `const float PI = 3.1415927;`
  - ✓ Esse comando cria uma variável somente de leitura.
- Estilo
  - ✓ Constantes com letras maiúsculas

Constantes no Scilab

- O Scilab oferece também variáveis com valores pré-definidos, como `%pi` (π) ou `%e` (número de neper)
- Valores numéricos são representados no Scilab em ponto flutuante de 64 bits (dupla precisão).

EXEMPLO 2

- Faça um programa em C e Scilabpara calcular o módulo da força elétrica entre duas cargas elétricas separadas por uma distância d.
  - Entrar com os dados via teclado.
  - Carga elétrica em µC e d em mm
  - Definir a constante eletrostática como:
    - $k = 8{,}987552 \times 10^9\ Nm^2/C^2$

Exemplo 3

- Preparação da equação:

$$F = k\frac{q_1 q_2}{d^2}$$

$$F = k\frac{q_1 \times \mathbf{10^{-6}}\ q_2 \times \mathbf{10^{-6}}}{(d \times \mathbf{10^{-3}})^2} = k\frac{q_1\ q_2 \times \mathbf{10^{-12}}}{d^2 \times \mathbf{10^{-6}}}$$

$$F = k\frac{q_1\ q_2 \times \mathbf{10^{-6}}}{d^2}$$

- Em C

```
f = K*q1*q2*1e-6/(d*d);
```

Operadores de atribuição reduzida

- Combinam uma atribuição de valor com uma operação aritmética.

```
total+=2;
```

Equivale a

```
total=total+2;
```

Operator	Shorthand	Equivalent Statement
+=	x += 2;	x = x + 2;
-=	x -= 2;	x = x - 2;
*=	x *= 2;	x = x * 2;
/=	x /= 2;	x = x / 2;
%=	x %= 2;	x = x % 2;

Mais operadores reduzidos

- Incremento ou decremento de 1

```
total++;
```

Equivale a

```
total = total + 1;
```

- Pré ou pós-fixados:

```
--x  ou x--
```

Precedência de operadores

- Todos os operadores do C têm um nível de precedência.
- Os operadores de nível mais elevado são avaliados antes dos operadores de nível mais baixo.
- Por exemplo, a expressão `a + b * c` é avaliada como `a + (b * c)` porque o operador `*` tem maior precedência do que o operador `+`.
- Os operadores do mesmo nível de precedência são geralmente avaliados da esquerda para a direita. Assim `a - b - c` é avaliado como `(a - b) - c`.

Operadores	Associatividade
( ) [ ] -> .	esquerda para a direita
! ~ ++ -- * & (cast) sizeof	direita para a esquerda
* / %	esquerda para a direita
+ -	esquerda para a direita
<< >>	esquerda para a direita
< <= > >=	esquerda para a direita
== !=	esquerda para a direita
&	esquerda para a direita
^	esquerda para a direita
|	esquerda para a direita
&&	esquerda para a direita
||	esquerda para a direita
?:	direita para a esquerda
= += -= *= /= %= &= ^= |= < <= > >=	direita para a esquerda
,	esquerda para a direita

TABELA DE PRECEDÊNCIA E ASSOCIATIVIDADE DE OPERADORES

OPERADORES DO C POR ORDEM DECRESCENTE DE PRIORIDADE E SUA ASSOCIATIVIDADE

EXEMPLO 4

- Qual será a saída do seguinte programa?

```
#include <stdio.h>
int main( ) {
      int i = 1;
      i = 2 + 2 * i++;
      printf("%d",i);
      return 0;
}
```

Exemplo 4

- Resposta: **5**
- Explicação:
  - ✓ Quando o operador de incremento pós-fixado é usado em alguma expressão, primeiro é atribuído o seu valor na expressão e a operação é realizada.
  - ✓Depois da operação ser realizada, ele incrementa a variável em 1. Então:

```
i= 2 + 2 * i++;
```

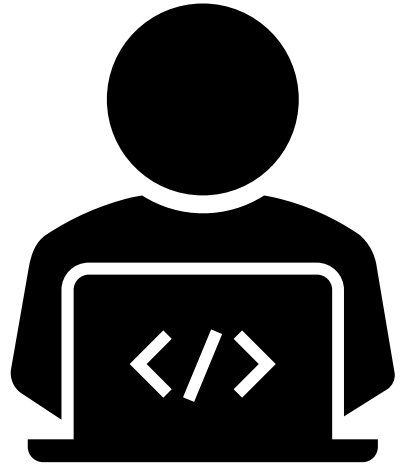
Sequência:

```
i= 2 + 2 * 1
i= 4
i= 4 + 1
```

Exercício 2

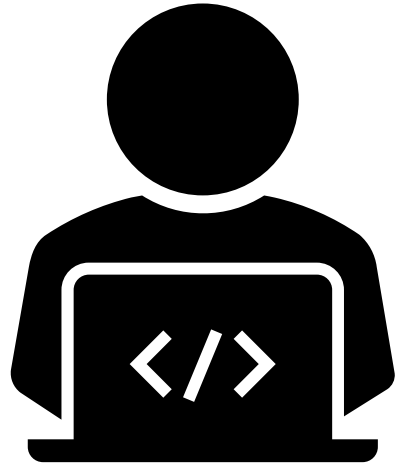
Qual será a saída do seguinte programa?

```
#include<stdio.h>
int main( ){
   int a=2, b=7, c=10;
   c=a==b;
   printf("%d", c);
   return 0;
}
```



Qual será a saída do seguinte programa?

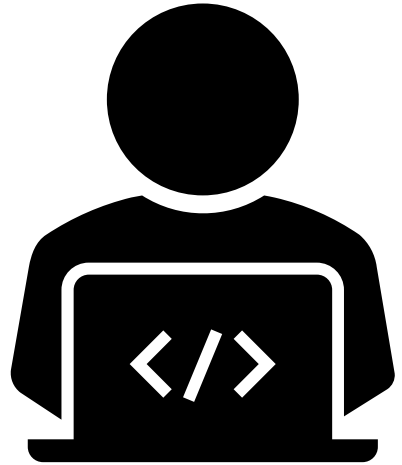
```
#include<stdio.h>
int main(){
   int i=5,j;
   j = ++i+10%4;
   i = ++i*3%4;
   printf("%d %d", i, j);
   return 0;
}
```



Exercício 2

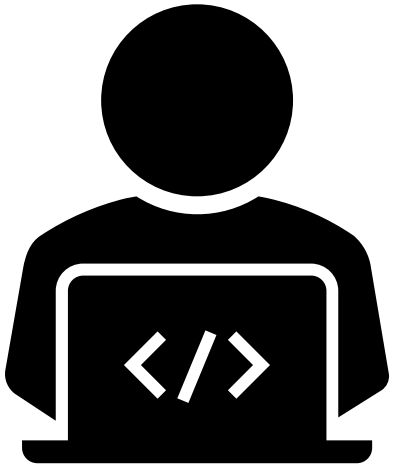
Exercício 2

- Elabore um programa que peça uma letra minúscula ao usuário e imprima o caractere e a letra maiúscula correspondente a este caractere.
- Presuma que o usuário realmente digite uma letra minuscula.
- Pesquise e utilize a função disponível na biblioteca `ctype` do ANSI C.



Exercícios de revisão

1. Uma loja vende seus produtos no sistema entrada mais duas prestações, sendo a entrada maior ou igual as duas prestações; estas devem ser iguais, inteiras e as maiores possíveis. Por exemplo, se o valor da mercadoria for R$ 270,00, a entrada e as duas prestações são iguais a R$ 90,00; se o valor da mercadoria for R$ 302,75, a entrada é de R$ 102,75 e as duas prestações são a iguais a R$ 100,00. Escreva um programa (C e Scilab) que receba o valor da mercadoria e forneça o valor da entrada e das duas prestações, de acordo com as regras acima.
2. Escreva um programa que calcule as médias aritmética e harmônica de 3 valores fornecidos pelo usuário.



Bibliografia e crédito das figuras

OUALLINE, S. Practical C Programming. 3a ed. O'Reilly, 1997.

SEBESTA, R. Conceitos de Linguagens de Programação. 5a ed. Porto Alegre: Bookman, 2003.

FORBELLONE, A. L. V.; Eberspacher, H. F. Lógica de Programação – A construção de Algoritmos e Estrutura de Dados. 2ª. Ed. São Paulo: Pearson/Makron Books, 2000.

ASHLEY, Stephen. The Fundamentals of C. 1a ed. Kindle Edition.

http://help.scilab.org/docs/6.1.0/pt_BR/index.html

Compiler, assembler, linker and loader: a brief story. Disponível em: http://www.tenouk.com/ModuleW.html

http://www.programmingbasics.org