Apresentação da Disciplina

PE-01 – v1.0

Prof. Paulo Joia Filho

Paulo Joia Filho

Email: paulo.joia@ufabc.edu.br

Lattes: http://lattes.cnpq.br/1050961885303209

Graduação

Licenciatura em Matemática - UniFil

Pós-graduação

Especialização em Matemática Superior - UEL
Especialização em Ciência da Computação - UEL
Mestrado em Engenharia Mecânica (Mecânica Computacional) - UNESP
Doutorado em Ciência da Computação e Matemática Computacional - ICMC/USP

Pós-doutorado em andamento

Engenharia da Computação e Sistemas Digitais - Poli/USP

Experiência Profissional

Acadêmica: Unopar (2 anos e meio), UniFil (6 anos), UEL (1 ano e meio)

Empresarial: Analista de Sistemas / Consultor (12 anos)

Paulo Joia Filho

Área de interesse: Segurança da Informação

- Criptografia pós-quântica com aplicações em Internet das coisas
  - Cripto-sistemas pós-quânticos envolvem basicamente as famílias de algoritmos baseados em reticulados (*lattices*), códigos corretores de erros, sistemas multivariados quadráticos (MQ), e esquemas baseados em primitivas criptograficas simétricas em geral, bem como funções de hash.
- Coleta e análise de informações em redes de computadores
  - Relatórios gerados por meio da coleta de dados são de extrema importância para a segurança da informação, permitindo montar uma base de informações a serem utilizadas na modelagem de ameaças, tornando os testes de ataque mais eficientes.
- Análise de vulnerabilidade
  - Executar varreduras que detectam falhas e brechas em sistemas, identificando riscos e providenciando correções.

Objetivos da Disciplina

- Conhecer as principais técnicas de programação estruturada e aplicá-las na resolução de problemas.
- Familiarizar-se com a programação em linguagens compiladas.
- Explorar o enorme potencial da linguagem C para desenvolver programas eficientes com modularidade e clareza.

Objetivos da Disciplina

- Conhecer as principais técnicas de programação estruturada e aplicá-las na resolução de problemas.
- Familiarizar-se com a programação em linguagens compiladas.
- Explorar o enorme potencial da linguagem C para desenvolver programas eficientes com modularidade e clareza.

Objetivos da Disciplina

- Conhecer as principais técnicas de programação estruturada e aplicá-las na resolução de problemas.
- Familiarizar-se com a programação em linguagens compiladas.
- Explorar o enorme potencial da linguagem C para desenvolver programas eficientes com modularidade e clareza.

Motivação

Por que programação estruturada?

Antes da programação estruturada...

- Código espaguete (*spaghetti code*);
- Código com complexidade de fluxo;
- Em geral, um único programa com centenas ou milhares de linhas, onde misturava-se saltos de execução via estruturas de desvio incondicional como GOTO, GOSUB, etc.
- Exemplo com código espaguete em BASIC:

```
10 i = 0
20 i = i + 1
30 PRINT i; " squared = "; i * i
40 IF i >= 10 THEN GOTO 60
50 GOTO 20
60 PRINT "Program Completed."
70 END
```

Antes da programação estruturada...

- Código espaguete (*spaghetti code*);
- Código com complexidade de fluxo;
- Em geral, um único programa com centenas ou milhares de linhas, onde misturava-se saltos de execução via estruturas de desvio incondicional como GOTO, GOSUB, etc.
- Exemplo com código espaguete em BASIC:

```
10 i = 0
20 i = i + 1
30 PRINT i; " squared = "; i * i
40 IF i >= 10 THEN GOTO 60
50 GOTO 20
60 PRINT "Program Completed."
70 END
```

Antes da programação estruturada...

- Código espaguete (*spaghetti code*);
- Código com complexidade de fluxo;
- Em geral, um único programa com centenas ou milhares de linhas, onde misturava-se saltos de execução via estruturas de desvio incondicional como GOTO, GOSUB, etc.
- Exemplo com código espaguete em BASIC:

```
10 i = 0
20 i = i + 1
30 PRINT i; " squared = "; i * i
40 IF i >= 10 THEN GOTO 60
50 GOTO 20
60 PRINT "Program Completed."
70 END
```

Antes da programação estruturada...

- Código espaguete (*spaghetti code*);
- Código com complexidade de fluxo;
- Em geral, um único programa com centenas ou milhares de linhas, onde misturava-se saltos de execução via estruturas de desvio incondicional como GOTO, GOSUB, etc.
- Exemplo com código espaguete em BASIC:

```
10 i = 0
20 i = i + 1
30 PRINT i; " squared = "; i * i
40 IF i >= 10 THEN GOTO 60
50 GOTO 20
60 PRINT "Program Completed."
70 END
```

Hoje...

Quais os principais paradigmas de programação?

Programação Estruturada (PE)

O paradigma da programação estruturada (ou procedural), afirma que qualquer problema pode ser quebrado em problemas menores, de mais fácil solução, chamados de funções, sub-rotinas ou procedimentos.

Além disso...

- O paradigma estruturado preconiza que todos os processamentos possíveis podem ser reduzidos a apenas três tipos de estruturas:
  - Estruturas de sequência;
  - Estruturas de decisão (ou condicionais);
  - Estruturas de repetição (ou iterativas).

Programação Estruturada (PE)

O paradigma da programação estruturada (ou procedural), afirma que qualquer problema pode ser quebrado em problemas menores, de mais fácil solução, chamados de funções, sub-rotinas ou procedimentos.

Além disso...

- O paradigma estruturado preconiza que todos os processamentos possíveis podem ser reduzidos a apenas três tipos de estruturas:
  - Estruturas de sequência;
  - Estruturas de decisão (ou condicionais);
  - Estruturas de repetição (ou iterativas).

Programação Estruturada (PE)

O paradigma da programação estruturada (ou procedural), afirma que qualquer problema pode ser quebrado em problemas menores, de mais fácil solução, chamados de funções, sub-rotinas ou procedimentos.

Além disso...

- O paradigma estruturado preconiza que todos os processamentos possíveis podem ser reduzidos a apenas três tipos de estruturas:
  - Estruturas de sequência;
  - Estruturas de decisão (ou condicionais);
  - Estruturas de repetição (ou iterativas).

Programação Estruturada (PE)

O paradigma da programação estruturada (ou procedural), afirma que qualquer problema pode ser quebrado em problemas menores, de mais fácil solução, chamados de funções, sub-rotinas ou procedimentos.

Além disso...

- O paradigma estruturado preconiza que todos os processamentos possíveis podem ser reduzidos a apenas três tipos de estruturas:
  - Estruturas de sequência;
  - Estruturas de decisão (ou condicionais);
  - Estruturas de repetição (ou iterativas).

Programação Estruturada (PE)

O paradigma da programação estruturada (ou procedural), afirma que qualquer problema pode ser quebrado em problemas menores, de mais fácil solução, chamados de funções, sub-rotinas ou procedimentos.

Além disso...

- O paradigma estruturado preconiza que todos os processamentos possíveis podem ser reduzidos a apenas três tipos de estruturas:
  - Estruturas de sequência;
  - Estruturas de decisão (ou condicionais);
  - Estruturas de repetição (ou iterativas).

Programação Orientada a Objetos (POO)

O paradigma da programação orientada a objetos compreende o problema como uma **coleção de objetos** interagindo por meio de trocas de mensagem. Os objetos são estruturas de dados contendo estado (dados) e comportamento (lógica).

Nesta abordagem...

- Um conjunto de objetos com informações comuns e com o mesmo comportamento dá origem a uma **classe**.
- Deve-se observar que o paradigma orientado a objetos não exclui o estruturado, pelo contrário, eles trabalham juntos, uma vez que grande parte da lógica embutida nos objetos segue o pensamento estruturado.

Programação Orientada a Objetos (POO)

O paradigma da programação orientada a objetos compreende o problema como uma **coleção de objetos** interagindo por meio de trocas de mensagem. Os objetos são estruturas de dados contendo estado (dados) e comportamento (lógica).

Nesta abordagem...

- Um conjunto de objetos com informações comuns e com o mesmo comportamento dá origem a uma **classe**.
- Deve-se observar que o paradigma orientado a objetos não exclui o estruturado, pelo contrário, eles trabalham juntos, uma vez que grande parte da lógica embutida nos objetos segue o pensamento estruturado.

Programação Orientada a Objetos (POO)

O paradigma da programação orientada a objetos compreende o problema como uma **coleção de objetos** interagindo por meio de trocas de mensagem. Os objetos são estruturas de dados contendo estado (dados) e comportamento (lógica).

Nesta abordagem...

- Um conjunto de objetos com informações comuns e com o mesmo comportamento dá origem a uma **classe**.
- Deve-se observar que o paradigma orientado a objetos não exclui o estruturado, pelo contrário, eles trabalham juntos, uma vez que grande parte da lógica embutida nos objetos segue o pensamento estruturado.

Pergunta:

Por que é importante estudar técnicas de programação estruturada?

Estudo de caso 1

Definição (Desigualdade Triangular)

Em todo triângulo, o comprimento de um dos lados é sempre inferior à soma dos comprimentos dos outros dois lados.

- Elaborar um programa para ler três medidas a, b e c. Em seguida, verificar se elas podem ser as medidas dos lados de um triângulo. Se forem, verificar se o triângulo é **equilátero**, **isósceles** ou **escaleno**.

Lembrando que...

- Triângulo equilátero tem três lados de comprimentos iguais.
- Triângulo isósceles tem dois lados de comprimentos iguais.
- Triângulo escaleno os lados não têm comprimentos iguais.

Solução com uso correto da técnica [em Scilab]

```
1  a = input("informe o valor de a: ");
   b = input("informe o valor de b: ");
3  c = input("informe o valor de c: ");

5  // Verifica se as medidas formam um triângulo
   if a < b + c & b < a + c & c < a + b
7     if a == b & b == c
         disp("triângulo equilátero"); // três lados iguais
9     elseif a == b | a == c | b == c
         disp("triângulo isósceles");  // dois lados iguais
11    else
         disp("triângulo escaleno");   // todos os lados são diferentes
13    end
   else
15    // Não satisfaz a propriedade da desigualdade triangular
      disp("não é triângulo");
17 end
```

Solução do aluno A1 (iniciante) [em Scilab]

```
a=input("Informe o lado A: ");
b=input("Informe o lado B: ");
c=input("Informe o lado C: ");

if (a+b>c) then
    if (a+c>b) then
        if (b+c>a) then
            printf("Valores válidos\n");
            if (a==b & b==c) then
                printf("Triângulo equilátero");
            elseif (a~=b & b~=c) then
                printf("Triângulo escaleno");
            else
                printf("Triângulo isósceles");
            end
        end
    end
else
    printf("Medidas incorretas");
end;
```

Solução do aluno A1 (iniciante) [em Scilab]

```
a=input("Informe o lado A: ");
b=input("Informe o lado B: ");
c=input("Informe o lado C: ");

if (a+b>c) then
    if (a+c>b) then
        if (b+c>a) then
            printf("Valores válidos\n");
            if (a==b & b==c) then
                printf("Triângulo equilátero");
            elseif (a~=b & b~=c) then
                printf("Triângulo escaleno");
            else
                printf("Triângulo isósceles");
            end
        end
    end
else
    printf("Medidas incorretas");
end;
```

- Problema de Lógica: criou 4 níveis de aninhamento com estruturas de decisão para resolver um problema simples como este.
- Apesar disso, o programa funciona.

Solução do aluno A1 (iniciante) [em Scilab]

```
a=input("Informe o lado A: ");
b=input("Informe o lado B: ");
c=input("Informe o lado C: ");

if (a+b>c) then
    if (a+c>b) then
        if (b+c>a) then
            printf("Valores válidos\n");
            if (a==b & b==c) then
                printf("Triângulo equilátero");
            elseif (a~=b & b~=c) then
                printf("Triângulo escaleno");
            else
                printf("Triângulo isósceles");
            end
        end
    end
else
    printf("Medidas incorretas");
end;
```

- Problema de Lógica: criou 4 níveis de aninhamento com estruturas de decisão para resolver um problema simples como este.
- Apesar disso, o programa funciona.

Solução do aluno A2 (iniciante) [em Scilab]

```
a = input("Digite o tamanho do lado A: ");
b = input("Digite o tamanho do lado B: ");
c = input("Digite o tamanho do lado C: ");
if a + b < c then
    printf("Medidas incorretas!")
elseif b + c < a
    printf("Medidas incorretas!")
elseif a + c < b
    printf("Medidas incorretas!")
elseif a == b & b == c
    printf("Esse triângulo é equilátero!")
elseif a == b & b ~= c
    printf("Esse triângulo é isóceles!")
elseif b == c & c ~= a
    printf("Esse triângulo é isóceles!")
elseif c == a & a ~=b
    printf("Esse triângulo é isóceles")
else
    printf("Esse triângulo é escaleno!")
end
```

Solução do aluno A2 (iniciante) [em Scilab]

```
a = input("Digite o tamanho do lado A: ");
b = input("Digite o tamanho do lado B: ");
c = input("Digite o tamanho do lado C: ");
if a + b < c then
    printf("Medidas incorretas!")
elseif b + c < a
    printf("Medidas incorretas!")
elseif a + c < b
    printf("Medidas incorretas!")
elseif a == b & b == c
    printf("Esse triângulo é equilátero!")
elseif a == b & b ~= c
    printf("Esse triângulo é isóceles!")
elseif b == c & c ~= a
    printf("Esse triângulo é isóceles!")
elseif c == a & a ~=b
    printf("Esse triângulo é isóceles")
else
    printf("Esse triângulo é escaleno!")
end
```

- Problema de lógica: testou 3 vezes a condição para triângulo isósceles.
- O programa falha no teste da desigualdade triangular.
- Alguém consegue descobrir o erro?

Solução do aluno A2 (iniciante) [em Scilab]

```
a = input("Digite o tamanho do lado A: ");
b = input("Digite o tamanho do lado B: ");
c = input("Digite o tamanho do lado C: ");
if a + b < c then
    printf("Medidas incorretas!")
elseif b + c < a
    printf("Medidas incorretas!")
elseif a + c < b
    printf("Medidas incorretas!")
elseif a == b & b == c
    printf("Esse triângulo é equilátero!")
elseif a == b & b ~= c
    printf("Esse triângulo é isóceles!")
elseif b == c & c ~= a
    printf("Esse triângulo é isóceles!")
elseif c == a & a ~=b
    printf("Esse triângulo é isóceles")
else
    printf("Esse triângulo é escaleno!")
end
```

- Problema de lógica: testou 3 vezes a condição para triângulo isósceles.
- O programa falha no teste da desigualdade triangular.
- Alguém consegue descobrir o erro?

Solução do aluno A2 (iniciante) [em Scilab]

```
a = input("Digite o tamanho do lado A: ");
b = input("Digite o tamanho do lado B: ");
c = input("Digite o tamanho do lado C: ");
if a + b < c then
    printf("Medidas incorretas!")
elseif b + c < a
    printf("Medidas incorretas!")
elseif a + c < b
    printf("Medidas incorretas!")
elseif a == b & b == c
    printf("Esse triângulo é equilátero!")
elseif a == b & b ~= c
    printf("Esse triângulo é isóceles!")
elseif b == c & c ~= a
    printf("Esse triângulo é isóceles!")
elseif c == a & a ~=b
    printf("Esse triângulo é isóceles")
else
    printf("Esse triângulo é escaleno!")
end
```

- Problema de lógica: testou 3 vezes a condição para triângulo isósceles.
- O programa falha no teste da desigualdade triangular.
- Alguém consegue descobrir o erro?

Solução do aluno A3 (iniciante) [em Scilab]

```
a=input("digite o valor de a: ");
b=input("digite o valor de b: ");
c=input("digite o valor de c: ");

k=0;
if a>b then
    k=a
    a=b
    b=k
    k=0
end
if b>c then
    k=b
    b=c
    c=k
end
if (a+b)>c then
    disp("triangulo")
else disp("nao é triangulo")
    return;
end

if a==b & b==c then
    disp("equilatero")
elseif a==b | b==c | a==c
    disp("isóceles")
else disp ("escaleno")
end
```

Solução do aluno A3 (iniciante) [em Scilab]

```
1  a=input("digite o valor de a: ");
   b=input("digite o valor de b: ");
3  c=input("digite o valor de c: ");

5  k=0;
   if a>b then
7     k=a
      a=b
9     b=k
      k=0
11 end
   if b>c then
13    k=b
      b=c
15    c=k
   end
```

```
18 if (a+b)>c then
      disp("triangulo")
20 else disp("nao é triangulo")
      return;
22 end

24 if a==b & b==c then
      disp("equilatero")
26 elseif a==b | b==c | a==c
      disp("isóceles")
28 else disp ("escaleno")
   end
```

- Por incrível que pareça, este programa funciona!
- Você consegue entender a lógica?
- Não vale a pena, imagine um problema com 10 variáveis de entrada!?

Solução do aluno A3 (iniciante) [em Scilab]

```
1  a=input("digite o valor de a: ");
   b=input("digite o valor de b: ");
3  c=input("digite o valor de c: ");

5  k=0;
   if a>b then
7     k=a
      a=b
9     b=k
      k=0
11 end
   if b>c then
13    k=b
      b=c
15    c=k
   end
```

```
18 if (a+b)>c then
      disp("triangulo")
20 else disp("nao é triangulo")
      return;
22 end

24 if a==b & b==c then
      disp("equilatero")
26 elseif a==b | b==c | a==c
      disp("isóceles")
28 else disp ("escaleno")
   end
```

- Por incrível que pareça, este programa funciona!
- Você consegue entender a lógica?
- Não vale a pena, imagine um problema com 10 variáveis de entrada!?

Solução do aluno A3 (iniciante) [em Scilab]

```
1  a=input("digite o valor de a: ");
   b=input("digite o valor de b: ");
3  c=input("digite o valor de c: ");

5  k=0;
   if a>b then
7     k=a
      a=b
9     b=k
      k=0
11 end
   if b>c then
13    k=b
      b=c
15    c=k
   end
```

```
18 if (a+b)>c then
      disp("triangulo")
20 else disp("nao é triangulo")
      return;
22 end

24 if a==b & b==c then
      disp("equilatero")
26 elseif a==b | b==c | a==c
      disp("isóceles")
28 else disp ("escaleno")
   end
```

- Por incrível que pareça, este programa funciona!
- Você consegue entender a lógica?
- Não vale a pena, imagine um problema com 10 variáveis de entrada!?

Estudo de caso 2

Desenvolvimento em série

Muitas funções matemáticas podem ser calculadas por meio de um somatório infinito de termos. Em cada caso, a precisão aumenta à medida que mais termos da série são considerados. Um exemplo é a função cos x:

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots$$

Para cálculos computacionais, no entanto, este somatório deve terminar após um número finito de termos (penalizando a precisão do resultado).

- Escreva a função `coseno(x,n)` com duas variáveis de entrada, onde a primeira variável de entrada `x` representa o ângulo em radianos e a segunda variável de entrada `n`, representa o número de termos a serem utilizados nos cálculos.

Solução com conhecimento das técnicas [em Scilab]

```
1  function y = coseno(x,n)
       y = 0
3      for i = 0 : n-1
           n = x ^ (2*i)
5          d = factorial(2*i)
           y = y + (-1) ^ i * (n/d)
7      end
   endfunction
```

Solução do aluno A1 (iniciante) [em Scilab]

```
function [c]= coseno(x,n)
    M=[];
    i=(-2);
    s=1;
    while n>0
        i=i+2;
        if (s==1) then
            M(n)=(x^i)/factorial(i);
            s=0;
        elseif (s==0) then
            M(n)=-((x^i)/factorial(i));
            s=1;
        end
        n=n-1;
    end;
    c=sum(M);
endfunction
```

Solução do aluno A1 (iniciante) [em Scilab]

```
1  function [c]= coseno(x,n)
       M=[];
3      i=(-2);
       s=1;
5      while n>0
           i=i+2;
7          if (s==1) then
               M(n)=(x^i)/factorial(i);
9              s=0;
           elseif (s==0) then
11             M(n)=-((x^i)/factorial(i));
               s=1;
13         end
           n=n-1;
15     end;
       c=sum(M);
17 endfunction
```

- Observe quantas variáveis e desvios desnecessários!
- Apesar dos problemas de lógica, a série converge.

Solução do aluno A1 (iniciante) [em Scilab]

```
function [c]= coseno(x,n)
    M=[];
    i=(-2);
    s=1;
    while n>0
        i=i+2;
        if (s==1) then
            M(n)=(x^i)/factorial(i);
            s=0;
        elseif (s==0) then
            M(n)=-((x^i)/factorial(i));
            s=1;
        end
        n=n-1;
    end;
    c=sum(M);
endfunction
```

- Observe quantas variáveis e desvios desnecessários!
- Apesar dos problemas de lógica, a série converge.

Solução do aluno A2 (iniciante) [em Scilab]

```
function c = coseno(x,n)
    z = 2 * n;
    s = 0;
    while n >=1
        if modulo(n,2) == 1 then
            s = s - ((x^z)/factorial(z));
        else
            s = s + ((x^z)/factorial(z));
        end
        z = z - 2;
        n = n -1;
    end
    c = 1 + s;
endfunction
```

Solução do aluno A2 (iniciante) [em Scilab]

```
1  function c = coseno(x,n)
       z = 2 * n;
3      s = 0;
       while n >=1
5          if modulo(n,2) == 1 then
               s = s - ((x^z)/factorial(z));
7          else
               s = s + ((x^z)/factorial(z));
9          end
           z = z - 2;
11         n = n -1;
       end
13     c = 1 + s;
   endfunction
```

- Complexidade desnecessária do código: excesso de váriáveis e desvios.
- A série converge, mas o programa está implementado para n+1 termos da série, alguém sabe explicar por quê?

Solução do aluno A2 (iniciante) [em Scilab]

```
1  function c = coseno(x,n)
       z = 2 * n;
3      s = 0;
       while n >=1
5          if modulo(n,2) == 1 then
               s = s - ((x^z)/factorial(z));
7          else
               s = s + ((x^z)/factorial(z));
9          end
           z = z - 2;
11         n = n -1;
       end
13     c = 1 + s;
   endfunction
```

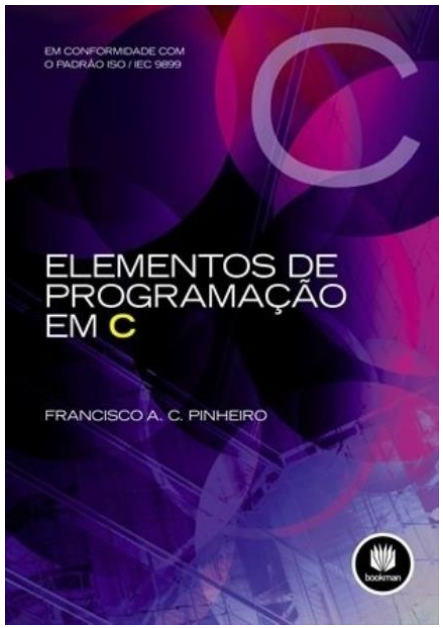
- Complexidade desnecessária do código: excesso de váriáveis e desvios.
- A série converge, mas o programa está implementado para n+1 termos da série, alguém sabe explicar por quê?

Foco deste curso

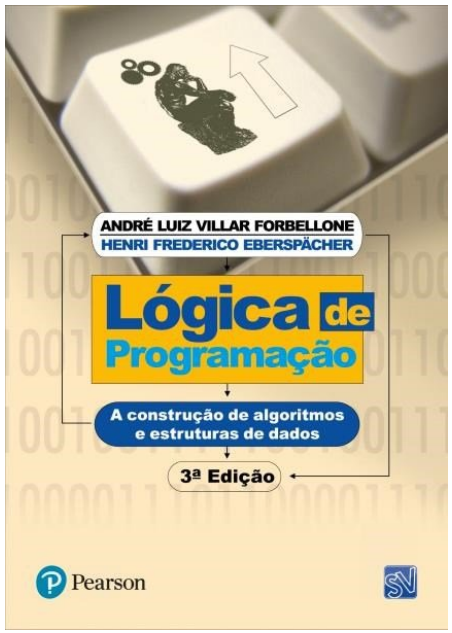
- Explorar as técnicas de programação estruturada para a resolução de problemas.
- Primeiro desafio: após familiarizar-se com a linguagem C, converter os dois estudos de casos apresentados em Scilab para ANSI C.

Bibliografia Básica

- Pinheiro, F. A. C. (2012). Elementos de Programação em C. Porto Alegre: Bookman.

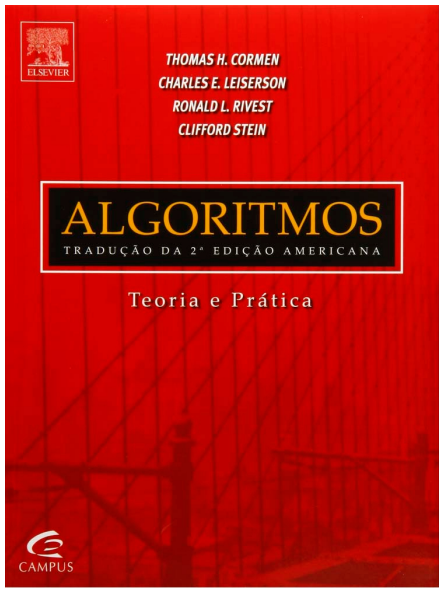


- Forbellone, A. L. V., & Eberspacher, H. F. (2005). Lógica de Programação: A Construção de Algoritmos e Estrutura de Dados (3^{a}ed). São Paulo: Pearson Prentice Hall.



Bibliografia Básica

- Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2002). Algoritmos: Teoria e Prática (3^a edição). Rio de Janeiro: Elsevier.



Sobre sites e livros...

Na Internet existe farto material sobre a linguagem C, com bons exemplos e dicas. No entanto, alguns sites e livros (autores muito "populares") costumam apresentar informações erradas ou vagas. Desde já é preciso desenvolver o senso crítico para filtrá-las. No caso de dúvidas, analise a informação comparando-a com outras fontes (outros autores, professores, colegas, etc).

Ementa

- Apresentar noções básicas e intermediárias sobre:
  - Algoritmos;
  - Programação em linguagens compiladas;
  - Compilação;
  - Programas em execução (processos);
  - Ponteiros;
  - Alocação estática e dinâmica de memória;
  - Vetores e matrizes;
  - Funções e passagem de parâmetros;
  - Registros;
  - Arquivos;
  - Recursividade.
- Aplicar os conceitos apresentados no contexto da resolução de problemas clássicos e novos da computação.

Cronograma [Parte 1/2]

Data	Dia sem.	Tipo	Conteúdo
17/09	2a-feira	Teórica	Apresentação da disciplina. Noções sobre algoritmos e programação estruturada. Visão geral da linguagem C.
20/09	5a-feira	Prática	Expressões em C. Entrada/Saída pelo console.
24/09	2a-feira	Teórica	Comandos de controle do programa.
27/09	5a-feira	Prática	Comandos de controle do programa.
01/10	2a-feira	Teórica	Arrays e strings.
04/10	5a-feira	Prática	Arrays e strings.
08/10	2a-feira	Teórica	Ponteiros: definição, operadores, expressões.
11/10	5a-feira	Prática	Ponteiros: definição, operadores, expressões.
15/10	2a-feira	Teórica	Ponteiros. Alocação estática e dinâmica de memória.
18/10	5a-feira	Prática	Ponteiros. Alocação estática e dinâmica de memória.
22/10	2a-feira	Teórica	Revisão / Resolução de exercícios.
25/10	5a-feira	Prática	Prova 1 e entrega da lista 1.

Cronograma [Parte 2/2]

Data	Dia sem.	Tipo	Conteúdo
29/10	2a-feira	Teórica	Estruturas, enumerações, tipos definidos pelo usuário. Passagem de parâmetros. Vista de prova.
01/11	5a-feira	Prática	Estruturas, enumerações, tipos definidos pelo usuário. Passagem de parâmetros.
05/11	2a-feira	Teórica	Entrada/Saída com arquivo.
08/11	5a-feira	Prática	Entrada/Saída com arquivo.
12/11	2a-feira	Teórica	Algoritmos recursivos.
~~15/11~~	~~5a-feira~~	~~Prática~~	Feriado
~~19/11~~	~~2a-feira~~	~~Teórica~~	Recesso
22/11	5a-feira	Prática	Recursividade em C.
26/11	2a-feira	Teórica	Revisão / Resolução de exercícios.
29/11	5a-feira	Prática	Prova 2 e entrega da lista 2.
03/12	2a-feira	Teórica	Ordenação e pesquisa. Vista de prova.
06/12	5a-feira	Prática	Prova substitutiva
10/12	2a-feira	Teórica	Aplicações em problemas clássicos e novos da Computação.
13/12	5a-feira	Prática	Prova de recuperação

Conteúdo das Aulas

O material da disciplina estará disponível no seguinte endereço:

http://paulojoia.orgfree.com

Algumas observações:

- O site está em desenvolvimento . . .
- A ideia é inserir novos conteúdos semanalmente.
- Avisos e informações importantes serão colocados na seção de avisos (página principal do site).

☞ *Portanto, fiquem atentos e consultem a página semanalmente!*

Conteúdo das Aulas

O material da disciplina estará disponível no seguinte endereço:

`http://paulojoia.orgfree.com`

Algumas observações:

- O site está em desenvolvimento . . .
- A ideia é inserir novos conteúdos semanalmente.
- Avisos e informações importantes serão colocados na seção de avisos (página principal do site).

☞ *Portanto, fiquem atentos e consultem a página semanalmente!*

Conteúdo das Aulas

O material da disciplina estará disponível no seguinte endereço:

http://paulojoia.orgfree.com

Algumas observações:

- O site está em desenvolvimento . . .
- A ideia é inserir novos conteúdos semanalmente.
- Avisos e informações importantes serão colocados na seção de avisos (página principal do site).

☞ *Portanto, fiquem atentos e consultem a página semanalmente!*

Conteúdo das Aulas

O material da disciplina estará disponível no seguinte endereço:

`http://paulojoia.orgfree.com`

Algumas observações:

- O site está em desenvolvimento . . .
- A ideia é inserir novos conteúdos semanalmente.
- Avisos e informações importantes serão colocados na seção de avisos (página principal do site).

☞ *Portanto, fiquem atentos e consultem a página semanalmente!*

Sistema de Avaliação

A média final (MF) será calculada do seguinte modo:

$$MF = \frac{3,5 \cdot P1 + 4,0 \cdot P2 + 2,5 \cdot AC}{10}$$

Onde:

P1 = Prova 1, representa 35% da média final

P2 = Prova 2, representa 40% da média final

AC = Atividades complementares, representam 25% da média final (explicado a seguir)

Lembrando que:

- Cada avaliação receberá um valor entre 0 e 10,0.
- Não será tolerado nenhuma forma de plágio.

Atividades Complementares (AC)

Serão constituídas por:

- Duas listas de exercícios (L1 e L2)
  - Atividade individual;
  - Entrega na data das provas P1 e P2, respectivamente;
  - As listas visam complementar as aulas teóricas e práticas, além de auxiliar no estudo para as provas.

Nota das atividades:

$$AC = \frac{L1 + L2}{2}$$

Relação Nota - Conceito

A: $\text{MF} \geqslant 8{,}5$

B: $7{,}0 \leqslant \text{MF} < 8{,}5$

C: $6{,}0 \leqslant \text{MF} < 7{,}0$

D: $5{,}0 \leqslant \text{MF} < 6{,}0$

F: $\text{MF} < 5{,}0$

O: reprovação, acima de 25% de faltas

Relação Nota - Conceito [Resolução ConsEPE nº 147]

A: Excelente compreensão da disciplina.

B: Bom desempenho, demonstrando boa capacidade de uso dos conceitos da disciplina.

C: Desempenho mínimo satisfatório, demonstrando capacidade de uso adequado dos conceitos da disciplina e habilidade para enfrentar problemas relativamente simples e capacidade adequada para seguir adiante em estudos mais avançados.

D: Aproveitamento mínimo não satisfatório dos conceitos da disciplina, com familiaridade parcial do assunto e alguma capacidade para resolver problemas simples, mas demonstrando deficiências que exigem trabalho adicional para prosseguir em estudos avançados.

F: Reprovado. A disciplina deve ser cursada novamente para a obtenção de crédito.

O: Reprovado por falta. A disciplina deve ser cursada novamente para a obtenção de crédito.

Prova de Recuperação (PR) [Resolução ConsEPE nº 182]

- Essa avaliação abrangerá todo o conteúdo da disciplina, sendo destinada aos alunos que tenham obtido conceito D ou F.
- A nota dessa prova substituirá a nota final da disciplina. O novo conceito será atribuído da seguinte forma:

$PR \geqslant 8{,}5$ sobe dois conceitos

$PR \geqslant 7{,}0$ sobe um conceito

Desempenho na Disciplina

Para obter bom desempenho na disciplina é importante...

- Estar presente nas aulas e atento ao material apresentado.
- Fazer os exercícios em sala de aula e em casa.
- Procurar entender, **refletir** e questionar.
- Associar o conteúdo com sua própria experiência.
- Associar com o conteúdo das aulas anteriores.
- Consultar as referências bibliográficas sugeridas.
- Procurar ajuda dos colegas (estudo em grupo), monitoria, atendimento extra-classe.

Desempenho na Disciplina

Para obter bom desempenho na disciplina é importante...

- Estar presente nas aulas e atento ao material apresentado.
- Fazer os exercícios em sala de aula e em casa.
- Procurar entender, **refletir** e questionar.
- Associar o conteúdo com sua própria experiência.
- Associar com o conteúdo das aulas anteriores.
- Consultar as referências bibliográficas sugeridas.
- Procurar ajuda dos colegas (estudo em grupo), monitoria, atendimento extra-classe.

Desempenho na Disciplina

Para obter bom desempenho na disciplina é importante...

- Estar presente nas aulas e atento ao material apresentado.
- Fazer os exercícios em sala de aula e em casa.
- Procurar entender, **refletir** e questionar.
- Associar o conteúdo com sua própria experiência.
- Associar com o conteúdo das aulas anteriores.
- Consultar as referências bibliográficas sugeridas.
- Procurar ajuda dos colegas (estudo em grupo), monitoria, atendimento extra-classe.

Desempenho na Disciplina

Para obter bom desempenho na disciplina é importante...

- Estar presente nas aulas e atento ao material apresentado.
- Fazer os exercícios em sala de aula e em casa.
- Procurar entender, **refletir** e questionar.
- Associar o conteúdo com sua própria experiência.
- Associar com o conteúdo das aulas anteriores.
- Consultar as referências bibliográficas sugeridas.
- Procurar ajuda dos colegas (estudo em grupo), monitoria, atendimento extra-classe.

Desempenho na Disciplina

Para obter bom desempenho na disciplina é importante...

- Estar presente nas aulas e atento ao material apresentado.
- Fazer os exercícios em sala de aula e em casa.
- Procurar entender, **refletir** e questionar.
- Associar o conteúdo com sua própria experiência.
- Associar com o conteúdo das aulas anteriores.
- Consultar as referências bibliográficas sugeridas.
- Procurar ajuda dos colegas (estudo em grupo), monitoria, atendimento extra-classe.

Desempenho na Disciplina

Para obter bom desempenho na disciplina é importante...

- Estar presente nas aulas e atento ao material apresentado.
- Fazer os exercícios em sala de aula e em casa.
- Procurar entender, **refletir** e questionar.
- Associar o conteúdo com sua própria experiência.
- Associar com o conteúdo das aulas anteriores.
- Consultar as referências bibliográficas sugeridas.
- Procurar ajuda dos colegas (estudo em grupo), monitoria, atendimento extra-classe.

Desempenho na Disciplina

Para obter bom desempenho na disciplina é importante...

- Estar presente nas aulas e atento ao material apresentado.
- Fazer os exercícios em sala de aula e em casa.
- Procurar entender, **refletir** e questionar.
- Associar o conteúdo com sua própria experiência.
- Associar com o conteúdo das aulas anteriores.
- Consultar as referências bibliográficas sugeridas.
- Procurar ajuda dos colegas (estudo em grupo), monitoria, atendimento extra-classe.

Atendimento extra-classe

- Para quem precisar tirar dúvidas sobre a disciplina ou conversar sobre algum projeto de pesquisa, estarei à disposição na minha sala em alguns horários pré-estabelecidos.
- Os horários serão fixados no site:
  `http://paulojoia.orgfree.com`
- Lembrando que, agendamentos por email evitam transtornos:
  paulo.joia@ufabc.edu.br

Regras de Conduta

Para disciplinas com prática em laboratório:

- Não é permitido comer nem beber no laboratório.
- Os equipamentos devem ser preservados!

Uso de aparelhos celulares:

- Manter o aparelho desligado ou em modo silencioso durante as explicações.
- Se precisar atendê-lo, favor sair da sala.

☞ *Contudo, espera-se que essa não seja uma prática comum a todas as aulas.*

Regras de Conduta

Somos todos adultos...

Portanto, evitem:

- Entrar e sair da sala frequentemente.
- Chegar após a aula ter iniciado e interromper a explicação.
- Chegar no final da aula.

Além disso:

- Durante as explicações recomenda-se o silêncio.
- Espera-se iniciativa e responsabilidade na execução das tarefas.
- Procurem participar das aulas.
- Ninguém será exposto a situação constrangedora por perguntar.

Referências Bibliográficas I

Aguilar, L. J. (2008).

Programação em C++: Algoritmos, Estruturas de Dados e Objetos.
McGraw-Hill, São Paulo.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., e Stein, C. (2002).

Algoritmos: Teoria e Prática.
Elsevier, Rio de Janeiro.

Drozdek, A. (2009).

Estrutura de Dados e Algoritmos em C++.
Cengage Learning, São Paulo.

Forbellone, A. L. V. e Eberspacher, H. F. (2005).

Lógica de Programação: A Construção de Algoritmos e Estrutura de Dados.
Pearson Prentice Hall, São Paulo, 3 edition.

Knuth, D. E. (2005).

The Art of Computer Programming.
Addison-Wesley, Upper Saddle River, NJ, USA.

Pinheiro, F. d. A. C. (2012).

Elementos de Programação em C.
Bookman, Porto Alegre.

Referências Bibliográficas II

Sedgewick, R. (1998).

Algorithms in C: Parts 1-4, Fundamentals, Data Structures, Sorting, Searching.
Addison-Wesley, Boston, 3rd edition.

Szwarcfiter, J. L. e Markenzon, L. (1994).

Estruturas de Dados e Seus Algoritmos.
LTC, Rio de Janeiro.

Tenenbaum, A. A., Langsam, Y., e Augenstein, M. J. (1995).

Estruturas de Dados Usando C.
Makron Books, São Paulo.