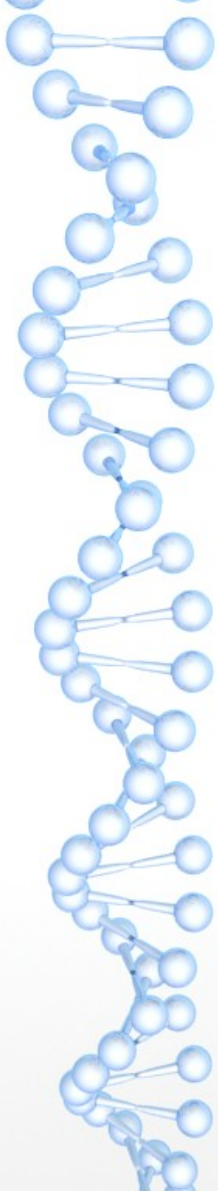


UniCesumar

Algoritmos e Lógica de Programação II

Introdução

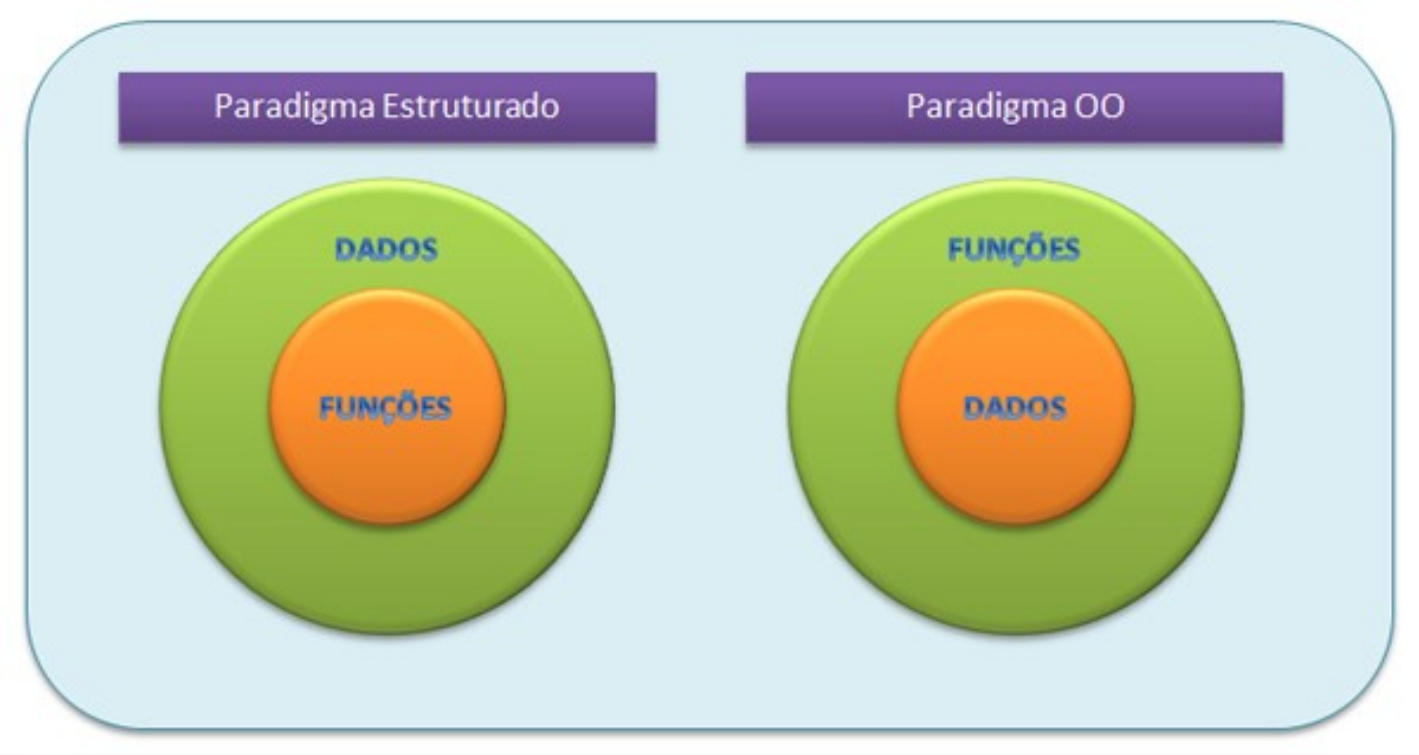
Prof. MSc. Rafael Staiger Bressan

Introdução

Introdução
Paradigmas de Programação

- Intimamente ligado com a **forma de pensar do programador** e como ele **busca soluções para os problemas**.
  - É o paradigma que permite ou proíbe a utilização de algumas técnicas de programação.
- Paradigma estruturado:
  - Conhecido como *imperativo* ou *procedural*;
  - Qualquer problema pode ser quebrado em problemas menores;
  - Podem ser reduzidos a apenas três estruturas: **sequência**, **decisão** e **iteração**
- Paradigma orientado a objeto:
  - Compreende o problema como uma coleção de objetos interagindo por meio de trocas de mensagem.

Introdução
Paradigmas de Programação



Introdução
Termologias

- ANSI (American National Standards Institute).
  - ("Instituto Nacional Americano de Padrões");
  - Serve como guia na escrita de **compiladores** e de programas nesta **linguagem de programação**.
- Compiladores.
  - Um compilador é um programa de computador (ou um grupo de programas) que, a partir de um código fonte escrito em uma linguagem compilada, cria um programa semanticamente equivalente.

Introdução
Termologias

- Linguagem de Programação.
  - É um método padronizado para comunicar instruções para um computador;
  - É um conjunto de regras sintáticas e semânticas usadas para definir um programa de computador.
- Erros de sintaxe.
  - A escrita dos programas contém erros.
- Erros de lógica.
  - A estratégia “gizada” no algoritmo está errada.

Introdução
Termologias

- Linguagem de Programação.
  - É um método padronizado para comunicar instruções para um computador;
  - É um conjunto de regras sintáticas e semânticas usadas para definir um programa de computador.
- Erros de sintaxe.
  - A escrita dos programas contém erros.
- Erros de lógica.
  - A estratégia “gizada” no algoritmo está errada.

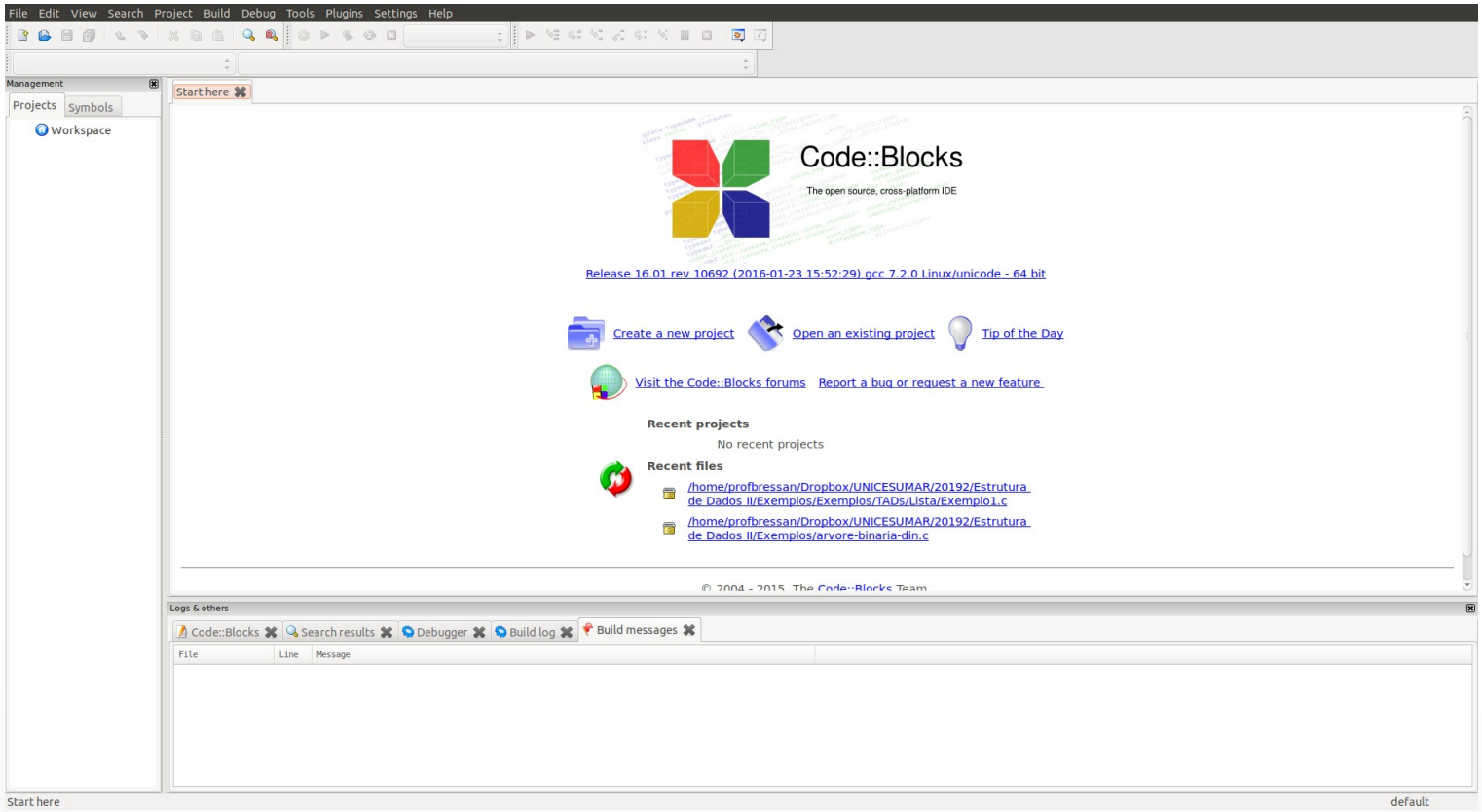
Linguagem C

- 1970 (Denis Ritchie),
  - Uso em um computador DEC PDP-11 em Unix
- BCPL → B → C → C++
- C++ é uma extensão da linguagem C que aceita a programação orientada a objetos.
- O sistema Unix é escrito em C e C++

Linguagem C

- Encontra-se na literatura diversos compiladores para linguagem C
  - **Code::Blocks IDE**
  - GCC
  - Dev C++
  - C++ Builder
  - Turbo C
  - ...

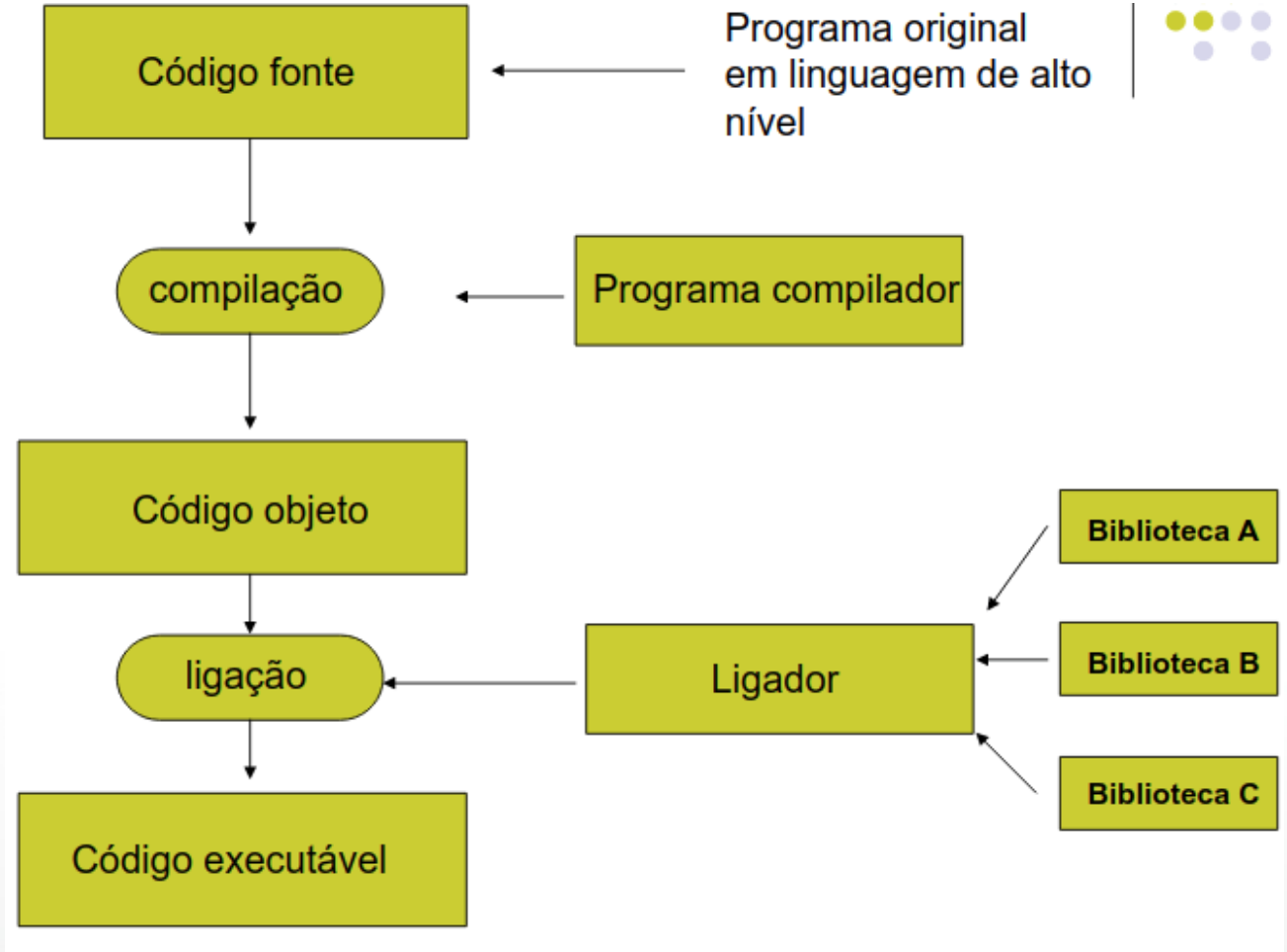
Linguagem C


File Edit View Search Project Build Debug Tools Plugins Settings Help
Management
Projects
Symbols
Workspace
Start here
Code::Blocks
The open source, cross-platform IDE
Release 16.01 rev 10692 (2016-01-23 15:52:29) gcc 7.2.0 Linux/unicode - 64 bit
Create a new project
Open an existing project
Tip of the Day
Visit the Code::Blocks forums
Report a bug or request a new feature
Recent projects
No recent projects
Recent files
/home/profbressan/Dropbox/UNICESUMAR/20192/Estrutura de Dados II/Exemplos/Exemplos/TADs/Lista/Exemplo1.c
/home/profbressan/Dropbox/UNICESUMAR/20192/Estrutura de Dados II/Exemplos/arvore-binaria-din.c
© 2004 - 2015 The Code::Blocks Team
Logs & others
Code::Blocks
Search results
Debugger
Build log
Build messages
File
Line
Message
Start here
default

Linguagem C

- Download
  - http://www.codeblocks.org/downloads/binaries
- Tutorial para instalação do Code Blocks.
  - http://linguagemc.com.br/tutorial-para-instalacao-do-code-blocks/

Conceitos Iniciais de Programação



Estrutura de um Programa em C

```
1  main(){
2      // Conjunto de instruções
3
4      return 0;
5  }
```

Estrutura de um Programa em C

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

main(){
   setlocale(LC_ALL, "Portuguese");
   printf("Alá mundo!");

   return 0;
}
```

Identificadores

- Os identificadores são os **nomes das variáveis**, dos programas, das constantes, das rotinas etc. As regras básicas são:
- Os caracteres que você pode utilizar são: os números, as letras maiúsculas, as letras minúsculas e o caractere sublinhado.
- O primeiro caractere deve ser sempre uma letra ou caractere sublinhado.
- Não são permitidos espaços em branco e caracteres especiais (@,$, +,-,%,!).
- Não podemos usar palavras reservadas nos identificadores, ou seja, palavras que pertençam a uma linguagem de programação.

Tipos de Dados

Tipo	Num de bits	Formato para leitura com scanf	Intervalo	
			Inicio	Fim
Char	8	%c	-128	127
unsigned char	8	%c	0	255
Signed char	8	%c	-128	127
Int	16	%i ou %d	-32.768	32.767
unsigned int	16	%u	0	65.535
signed int	16	%i ou %d	-32.768	32.767
Short int	16	%hi ou %hd	-32.768	32.767
unsigned short int	16	%hu	0	65.535
signed short int	16	%hi ou %hd	-32.768	32.767
Long int	32	%li ou %ld	-2.147.483.648	2.147.483.647
signed long int	32	%li ou %ld	-2.147.483.648	2.147.483.647
unsigned long int	32	%lu	0	4.294.967.295
Float*	32	%f, %e ou %g	3,4E-38	3.4E+38
Double*	64	%lf, %le ou %lg	1,7E-308	1,7E+308
long double*	80	%Lf	3,4E-4932	3,4E+4932

Palavras reservadas

auto	extern	sizeof
break	float	static
case	for	struct
char	goto	switch
const	if	typedef
continue	int	union
defaukt	long	void
do	register	volatile
double	return	while
else	short	
enum	signed	

Variáveis

- Em C, as variáveis são declaradas **após a especificação de seus tipos**.
- Ex.: <tipo> identificador;
  - int numero;
  - float total;
  - int n1, n2, n3;
  - char sexo;
  - char endereco[30];

Constantes

- Constantes armazenam informações que não variam com o tem. Em C temos a seguinte sintaxe:
  - #define <itendificador> valor
    - Não é utilizado o (;) no final.
  - Exemplo:
    - #define valorPI 3.1416

Expressões e Operadores

Operação	Operador	Prioridade
Soma	+	1 (menor)
Subtração	-	1
Multiplicação	*	2
Divisão	/	2
Exponenciação	pow(variável, expoente)	3 (maior)
Resto	%	3

Expressões e Operadores

Operação	Operador	Exemplo
Igual	==	A==B ---- B==A
Diferente	!>	A !> B
Maior	>	A > 5
Menor	<	A < 5
Maior ou igual a	>=	A >= 5
Menor ou igual a	<=	A <= 5

Expressões e Operadores

Operação	Operador	Exemplo
Conjunção	&&	((A>B) && (B<C))
Disjunção	||	((A>B) || (B<C))
Negação	!	!A

Expressões e Operadores

Operador	Exemplo	Equivalencia
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y
++	x++ y = ++x y = x++	x = x + 1 y = x + 1 depois y = x y = x depois x = x + 1
--	x-- y = --x y = x--	x = x - 1 y = x - 1 depois y = x y = x depois x = x - 1

Funções Intrínsecas

- http://linguagemc.com.br/a-biblioteca-math-h/
- Função Descrição do comando
  - floor( ) arredonda para baixo
  - ceil( ) arredonda para cima
  - sqrt( ) Calcula raiz quadrada
  - pow(variável, expoente) potenciação
  - sin( ) seno
  - cos( ) cosseno
  - tan( ) Tangente
  - log( ) logaritmo natural
  - log10( ) logaritmo base 10

Atribuições

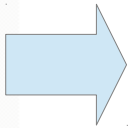
- O comando de atribuição é usado para conceder valores ou operações a variáveis.
  - Idendificador = expressão
  - X = 3
  - X = X + 5
  - X = 34.78
  - nome = “Rafael”

Entrada de Dados

- Permite receber os dados digitados pelo usuário armazenando-os em variáveis.
- Existem diversas funções para entrada de dados
  - cin
  - gets
  - scanf

Entrada de Dados

```c
#include <stdio.h>
int main(){
    int idade;
    printf("Digite a sua idade: ");
    scanf("%d", &idade);
    printf("A idade digitada foi %d \n", idade);
    return 0;
}
```

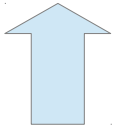


Entrada de Dados

CÓDIGO	SIGNIFICADO
%c	Leitura de um único caractere.
%d	Leitura de um número decimal inteiro.
%i	Leitura de um decimal inteiro.
%u	Leitura de um decimal sem sinal.
%e	Leitura de um número em ponto flutuante com sinal opcional.
%f	Leitura de um número em ponto flutuante com ponto opcional.
%g	Leitura de um número em ponto flutuante com expoente opcional.
%o	Leitura de um número em base octal.
%s	Leitura de uma string.
%x	Leitura de um número em base hexadecimal.
%p	Leitura de um ponteiro.

Saída de Dados

- A saída de dados permite mostrar dados ao usuário. Em C utilizaremos a função (**printf**).
  - printf(“expressão de controle”, lista de argumentos);
  - Exemplo:
    - printf("Olá Mundo! ");
    - printf("A idade digitada foi **%d \n**", **idade**);

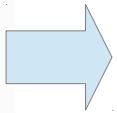


Saída de Dados

- Códigos especiais
  - \n → Nova linha
  - \t → TAB
  - \b → Retrocesso
  - \” → Aspas
  - ...

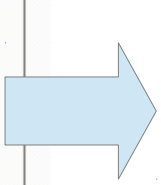
Construindo um Programa

```c
#include <stdio.h>
int main(){
    char nome[50];
    int idade;
    printf("Digite o seu nome: ");
    scanf("%s", nome);
    printf("Digite a sua idade: ");
    scanf("%d", &idade);
    printf("\n Olá %s. \n A idade digitada foi %d \n\n", nome, idade);
    return 0;
}
```



Construindo um Programa

```c
#include <stdio.h>
int main(){
    char nome[50];
    int idade;
    printf("Digite o seu nome: ");
    scanf("%[^\n]s", nome);
    printf("Digite a sua idade: ");
    scanf("%d", &idade);
    printf("\n Olá %s. \n A idade digitada foi %d \n\n", nome, idade);
    return 0;
}
```



Atividades

- Faça um programa que leia quatro números, calcule e apresente a média.
- Faça um programa que leia dois números inteiros (num1, num2) calcule e apresente:
  - Soma
  - Subtração
  - Multiplicação
  - Divisão.

```c
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>

int main() {
    setlocale(LC_ALL, "Portuguese");
    float num1, num2, num3, num4, media;
    printf("Digite o primeiro número : ");
    scanf("%f", &num1);
    printf("Digite o segundo número  : ");
    scanf("%f", &num2);
    printf("Digite o terceiro número : ");
    scanf("%f", &num3);
    printf("Digite o quarto número   : ");
    scanf("%f", &num4);
    media = (num1 + num2 + num3 + num4) / 4;
    printf("Média aritmética         : %.2f", media);
    return 0;
}
```

```c
int main() {
    setlocale(LC_ALL, "Portuguese");
    int num1, num2, soma, subtracao, multiplicacao;
    float divisao;
    printf("Digite o primeiro número : ");
    scanf("%d", &num1);
    printf("Digite o segundo número  : ");
    scanf("%d", &num2);
    soma = num1 + num2;
    subtracao = num1 - num2;
    multiplicacao = num1 * num2;
    divisao = (float) num1 / num2;
    printf("Soma           : %d \n", soma);
    printf("Subtração      : %d \n", subtracao);
    printf("Multiplicação : %d \n", multiplicacao);
    printf("Divisão        : %.2f \n", divisao);
    return 0;
}
```

Atividades

- Exercícios Resolvidos e Propostos.
  - “Fundamentos da programação de computadores: algoritmos, Pascal, C/C++ e Java”
  - Livro Base (Algoritmos e Lógica de Programação II)

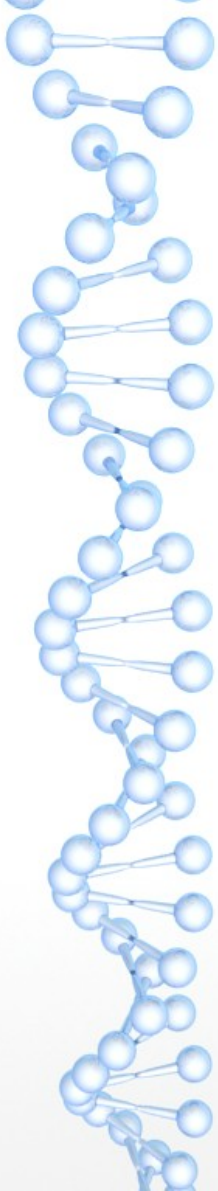
Referencias

MANZANO, José Augusto NG. **Algoritmos lógica para desenvolvimento de programação de computadores**. Saraiva Educação SA, 2010.

ASCENCIO, Ana Fernandes Gomes; CAMPOS, Edilene Aparecida Veneruchi de. **Fundamentos da programação de computadores: algoritmos, Pascal e C/C++ e Java**. São Paulo: Pearson Prentice Hall, 2010

CORMEN, Thomas H. et al. **Algoritmos: teoria e prática**. Editora Campus, v. 2, p. 2, 2002.

XAVIER, Gley Fabiano Cardoso. **Lógica de programação**. Senac, 2018.



“Só existem dois dias no ano que nada pode ser feito.
Um se chama ontem e o outro se chama amanhã”

Dalai Lama