

Fundamentos de Programação

Aula 02 - Conceitos Preliminares

Prof^a. Elisa de Cássia Silva Rodrigues

Sumário

- Introdução e motivação.
- Conceitos básicos.
- Desenvolvendo um programa.
- Por onde devo começar?
- Noções de lógica.
- Análise de problemas.
- Noções de sistemas de computação.
- Paradigmas de programação.
- Representação de dados.

Introdução

- O que vamos aprender em Fundamentos de Programação?

Introdução

- O que vamos aprender em Fundamentos de Programação?
 - ▶ Escrever programas!!!

Introdução

- O que vamos aprender em Fundamentos de Programação?
 - ▶ Escrever programas!!!
- Para escrever programas precisamos saber o que?

Introdução

- O que vamos aprender em Fundamentos de Programação?
 - ▶ Escrever programas!!!
- Para escrever programas precisamos saber o que?
 - ▶ Definir e resolver problemas.
 - ▶ Formalizar soluções através de algoritmos.
 - ▶ Usar o computador.
 - ▶ Utilizar linguagens de programação.

Motivação

- Temos um **problema** a ser resolvido quando existe um **conjunto de informações** e um **objetivo** a ser alcançado.
- **Problema 1:**

Você tem uma raposa, uma galinha e um milho. Seu objetivo é atravessar um rio levando todos com você. Para isso você dispõe de uma canoa, porém você pode transportar apenas um item por vez. Note que você não pode deixar a raposa e a galinha sozinhos nem a galinha e o milho, senão um dos itens será devorado antes que você possa atravessá-los.

Como você resolveria este problema?

Motivação

- Qual é a sequência de passos para que o **Problema 1** seja resolvido?

Motivação

- Qual é a sequência de passos para que o **Problema 1** seja resolvido?
 1. Leve a galinha para o outro lado.
 2. Volte sozinho.
 3. Leve a raposa para o outro lado.
 4. Volte com a galinha.
 5. Leve o milho para o outro lado.
 6. Volte sozinho.
 7. Leve a galinha para o outro lado.

Motivação

- Qual é a sequência de passos para que o **Problema 1** seja resolvido?
 1. Leve a galinha para o outro lado.
 2. Volte sozinho.
 3. Leve a raposa para o outro lado.
 4. Volte com a galinha.
 5. Leve o milho para o outro lado.
 6. Volte sozinho.
 7. Leve a galinha para o outro lado.

Isto é um ALGORITMO!

Conceitos Básicos

- **Problema** é um conjunto de informações e um objetivo.
- **Algoritmo** é uma sequência de instruções para resolver um problema.
- **Programação** é ensinar o computador a resolver um problema.
- **Programa** é um algoritmo escrito de forma que o computador entenda.
- **Linguagem de programação** você e o computador entendem.
- **Computador** é uma máquina que executa as suas instruções.

Desenvolvendo um Programa

- Etapas para desenvolvimento de um programa:

- ▶ **Análise:**

- ★ Estudar o enunciado do problema.
 - ★ Definir dados de entrada.
 - ★ Definir o processamento.
 - ★ Definir os dados de saída.

- ▶ **Algoritmo:**

- ★ Ferramenta utilizada para descrever o problema e a solução.

- ▶ **Codificação:**

- ★ Transformar um algoritmo em códigos de uma linguagem de programação.

- **Problema 2:** Somar dois números.

- ▶ Dados de entrada: primeiro número ($n1$) e segundo número ($n2$)
- ▶ Processamento: somar os números ($s = n1 + n2$).
- ▶ Dados de saída: resultado da soma (s).

- **Exemplo:**

- ▶ $n1 = 5$
- ▶ $n2 = 4$
- ▶ $s = 9$

Algoritmo

- O que é necessário para construir um algoritmo?
 - ▶ Compreender o problema.
 - ▶ Destacar pontos importantes e objetos que o compõem.
 - ▶ Definir os dados de entrada, processamento e saída.
 - ▶ Escolher um tipo de algoritmo.
 - ▶ Construir o algoritmo.
 - ▶ Testar o algoritmo realizando simulações.

Algoritmo

- Tipos de algoritmos

- ▶ Descrição narrativa: utiliza linguagem natural.
- ▶ Fluxograma: utiliza símbolos gráficos predefinidos.
- ▶ Pseudocódigo: utiliza regras predefinidas.

Algoritmo

- Tipos de algoritmos
 - ▶ Descrição narrativa: utiliza linguagem natural.
 - ▶ Fluxograma: utiliza símbolos gráficos predefinidos.
 - ▶ Pseudocódigo: utiliza regras predefinidas.
- Como testar se o algoritmo está correto?

Algoritmo

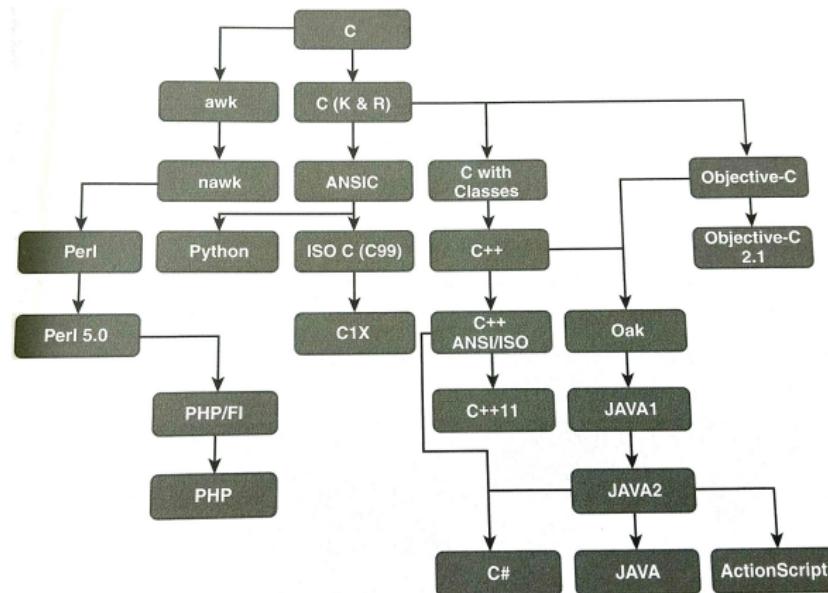
- Tipos de algoritmos
 - ▶ Descrição narrativa: utiliza linguagem natural.
 - ▶ Fluxograma: utiliza símbolos gráficos predefinidos.
 - ▶ Pseudocódigo: utiliza regras predefinidas.
- Como testar se o algoritmo está correto?
 - ▶ Teste de mesa.

Codificação

- Transformação do pseudocódigo em programa.
 - ▶ Escolher uma linguagem de programação.
- Por que vamos aprender **linguagem C?**
 - ▶ Linguagem de programação de alto-nível mais utilizada.
 - ▶ Possui estrutura simples e grande portabilidade.
 - ▶ Compilador C gera códigos mais enxutos e velozes.
 - ▶ Oferece acesso de baixo-nível à memória.

Codificação

- Influência da linguagem C.



Por onde vamos começar?

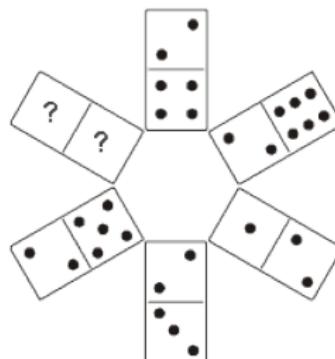
- Definir e resolver problemas.
 - ▶ Noções de lógica.
 - ▶ Análise de problemas.



Noções de Lógica

- **Objetivo:**

- ▶ Desenvolver capacidade cognitiva para compreender fundamentos básicos da solução de problemas de forma algorítmica.
- ▶ Desenvolver e estimular o **raciocínio lógico**.



Noções de Lógica

- **Problema:**

- ▶ Temos um conjunto de informações e um objetivo a ser alcançado.

Informações do Problema ⇒ Raciocínio Lógico ⇒ Solução

- **Exemplo 1:**

- ▶ Eu quero assistir aula de COM110.
- ▶ A porta do laboratório está fechada.
- ▶ Logo, preciso abrir a porta para assistir aula de COM110.

- **Exemplo 2:**

- ▶ Matemática é mais legal que português.
- ▶ Programação é mais legal que matemática.
- ▶ Logo, programação é mais legal que português.

Raciocínio Lógico

- Processo de estruturação do pensamento de forma analítica.
- Usado para justificar, analisar, argumentar ou confirmar raciocínios.
- É exato pois é fundamentado em dados que podem ser comprovados.

Não pode ser ensinado e sim trabalhado por meio da resolução de problemas lógicos!

<https://rachacuca.com.br/jogos/tags/raciocinio/>.

Tipos de Raciocínio Lógico

- **Dedutivo:**

- ▶ Parte de fatos verdadeiros para chegar a uma conclusão verdadeira.
- ▶ **Exemplo 3:**
 - ★ Toda ave tem asas.
 - ★ A coruja é uma ave.
 - ★ Logo, toda coruja tem asas.

- **Indutivo:**

- ▶ Parte de fatos particulares e chega a uma conclusão genérica.
- ▶ **Exemplo 4:**
 - ★ Todas as corujas observadas possuíam asas.
 - ★ Logo, todas as corujas possuem asas.

Problemas de Lógica

- Envolve enunciados com afirmações entrelaçadas entre si.
- Algumas afirmações são verdadeiras e outras falsas.
- O desafio é descobrir o fato correto.
- **Exemplo 5:**
 - ▶ Se durante uma corrida de carros, você deixa o segundo colocado pra trás, qual é a sua colocação após a ultrapassagem?
- **Exemplo 6:**
 - ▶ Uma família resolveu passear de carro. Nele entraram 1 avô, 2 pais, 2 filhos e 1 neto. Qual o número mínimo de pessoas dentro do veículo, afinal?

[https://rachacuca.com.br/logica/problemas/.](https://rachacuca.com.br/logica/problemas/)

Noções de Sistemas de Computação

- O que é um computador?

- ▶ Conjunto de dispositivos eletrônicos interligados, que conseguem executar automaticamente uma tarefa, orientado por programa e em grande velocidade.



- Alguns tipos de computador.



Desktop computer

Laptop

Netbook

Hybrid

Tablet

Smartphone

Noções de Sistemas de Computação

- O que é **hardware**?
 - ▶ Parte física (peças, cabos, componentes e dispositivos).



- O que é **software**?
 - ▶ Parte lógica (sistema operacional, programas e aplicativos.).



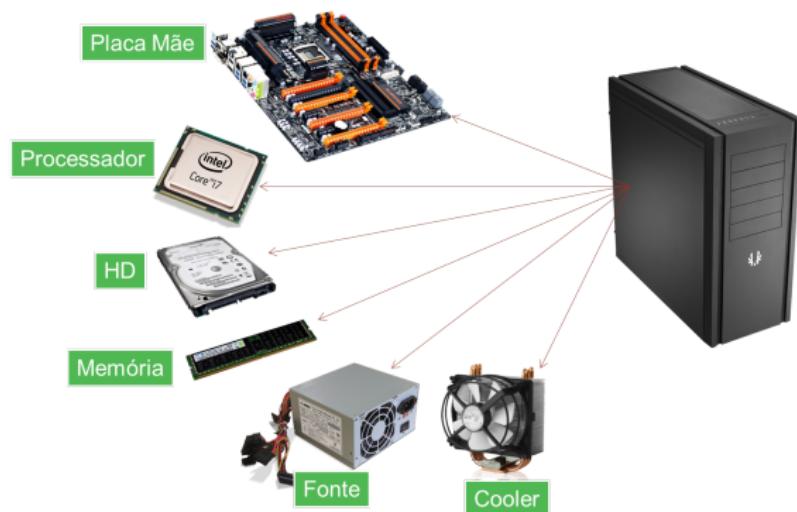
Noções de Sistemas de Computação

- Exemplos de dispositivos de entrada e saída.



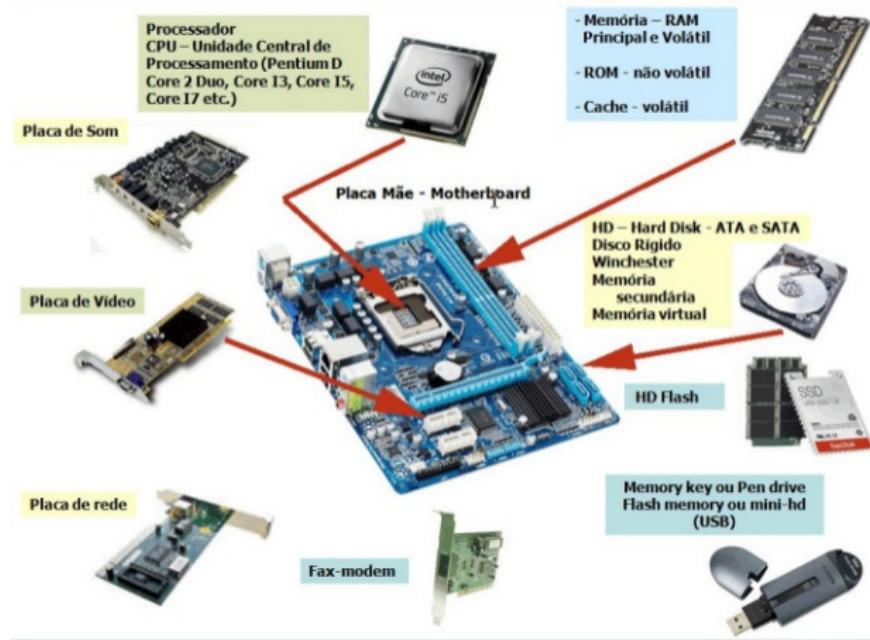
Noções de Sistemas de Computação

- Principais componentes internos do computador.



Noções de Sistemas de Computação

- Principais componentes da placa-mãe.



Noções de Sistemas de Computação

- O que é **sistema operacional**?

- ▶ Software responsável pelo gerenciamento, funcionamento e execução de todos os programas.
- ▶ Fica instalado no HD e é executado sempre que o computador é ligado.



- Exemplos:



Windows



Linux



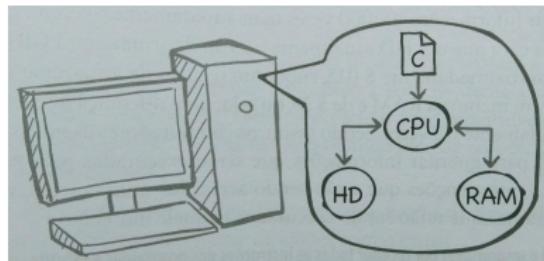
Mac™
OS



android

Introdução

- Como o computador executa um programa?
 - ▶ A CPU executa as instruções e acessa a memória, se preciso.



- Como o computador entende as instruções do programa?
 - ▶ O **compilador** converte as instruções para linguagem de máquina.
 - ▶ Logo, precisa de uma **representação da informação**.

Noções de Sistemas de Computação

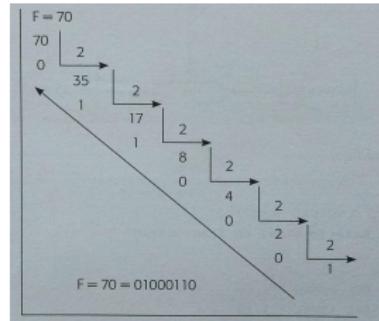
- Uma informação é representada por um conjunto de símbolos.
 - ▶ Ex: língua portuguesa => alfabeto (A, B, C, ..., Z).
 - ▶ Ex: quantidades => sistema decimal (0, 1, 2, ..., 9).
- O computador é feito de componentes elétricos.
 - ▶ Identifica corrente alta (1) e corrente baixa (0).
- Quais símbolos o computador entende?
 - ▶ 0 e 1 => Sistema Binário.
- Sistema binário x Memória.
 - ▶ Cada dígito binário (0 ou 1) ocupa uma posição da memória (bit)
 - ★ 8 bits = 1 byte.
 - ▶ Cada byte é identificado e acessado por meio de um endereço.

Noções de Sistemas de Computação

- Todo caractere possui um **código ASCII** (representação de caracteres por número decimal).

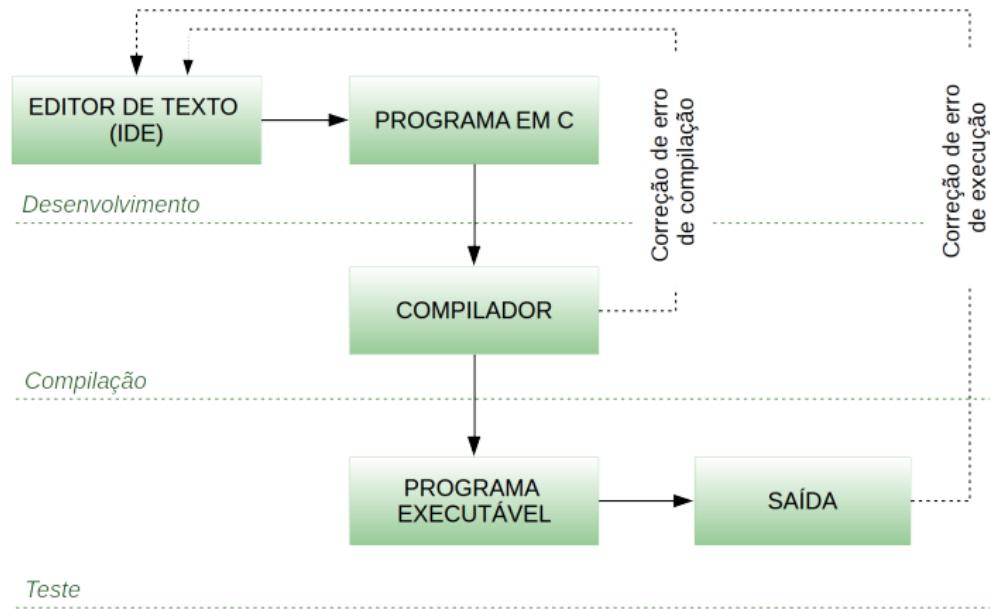
Caractere	Valor decimal na tabela ASCII	Valor binário
A	65	01000001
B	66	01000010
C	67	01000011

- Um código ASCII é transformado em binário pelo **Método da Divisão**.



Noções de Sistemas de Computação

- Ciclo de desenvolvimento de um programa.



Paradigmas de Programação

- Um **paradigma de programação** está relacionado a **forma de pensar do programador** e como ele busca a solução para os problemas.
 - ▶ Pode ser implementado em qualquer linguagem de programação.
 - ▶ Cada linguagem atende a pelo menos um paradigma de programação.
 - ▶ O uso de um paradigma não exclui os outros, podem ser combinados.
- **Exemplos:**
 - ▶ Estruturado.
 - ▶ Orientado a objeto.
 - ▶ Lógico.
 - ▶ Funcional.

Paradigma Estruturado

- Quando o problema é dividido em problemas menores mais fáceis de resolver (funções).



- Todo processamento pode ser realizado pelo uso de três estruturas:
 - ▶ Sequencial.
 - ▶ Condicional.
 - ▶ Repetição.

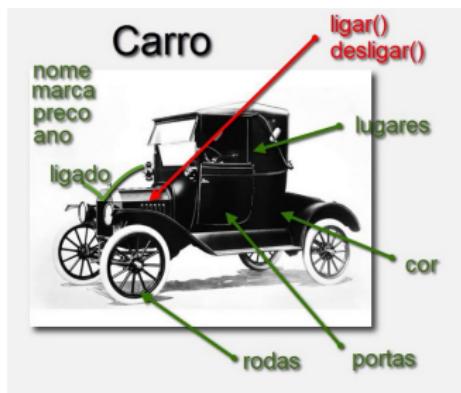
É objetivo da disciplina Fundamentos de Programação
desenvolver o pensamento lógico usando o Paradigma Estruturado!

Paradigma Estruturado

- A análise do problema tenta:
 - ▶ Relacionar as ações.
 - ▶ Subdividir cada ação em módulos.
- **Exemplo:** Calcule a área e o perímetro de um retângulo.
 - ▶ Detalhar as ações necessárias.
 - ① Obter o valor da altura do retângulo.
 - ② Obter o valor da largura do retângulo.
 - ③ Calcular a área.
 - ④ Calcular o perímetro.
 - ⑤ Mostrar os resultados.
 - ▶ Modularizar a solução (cada módulo deve realizar uma tarefa).
 - ① `calcularArea(altura, largura);`
 - ② `calcularPerimetro(altura, largura);`
 - ③ `mostrarResultados();`

Paradigma Orientado a Objeto

- Compreende o problema como uma coleção de objetos que interagem entre si através de mensagens.
- Objetos são estruturas de dados que contém:
 - ▶ Estado (dados).
 - ▶ Comportamento (responsabilidades).
- Um conjunto de objetos com informações comuns e mesmo comportamento fazem parte de uma Classe.



Paradigma Orientado a Objeto

- A análise do problema tenta:
 - ▶ Identificar os objetos que compõem o problema.
 - ▶ Identificar como os objetos interagem entre si.
- **Exemplo:** Calcule a área e o perímetro de um retângulo.
 - ▶ Procurar objetos existentes no problema.
 - 1 Retângulo.
 - 2 Interface do usuário.
 - ▶ Determinar características e responsabilidades do Retângulo.
 - 1 Armazenar e manipular valores de altura e largura.
 - 2 Calcular a área.
 - 3 Calcular o perímetro.
 - ▶ Determinar características e responsabilidades da Interface do usuário.
 - 1 Receber valores iniciais (altura e largura).
 - 2 Enviar para o retângulo.
 - 3 Solicitar os valores de área e perímetro ao objeto retângulo.
 - 4 Mostrar os resultados.

Representação de dados

- Toda informação precisa ser armazenada na memória do computador.
- Uma posição da memória pode ser:
 - ▶ Constante
 - ★ Conteúdo NÃO VARIA durante a execução do programa.
 - ▶ Variável
 - ★ Conteúdo que VARIA durante a execução do programa.
 - ▶ Ponteiro.
 - ★ Variável cujo conteúdo é uma posição da memória (endereço).

Representação de dados

- Toda posição de memória possui um **nome** e um **tipo**.
- Nome ou identificador.
 - ▶ **Caracteres permitidos:** letras, números e sublinhado (*underline*).
 - ▶ **Primeiro caractere:** NÃO PODE ser um número.
 - ▶ **Palavras reservadas:** NÃO PODE usar palavras da linguagem.
- Tipos primitivos de dados.
 - ▶ **Numéricos:** inteiros ou reais (parte decimal separada por ponto).
 - ★ Ex: **78** e **9.43**.
 - ▶ **Lógicos:** dados booleanos (álgebra de Boole).
 - ★ Ex: **V** e **F**.
 - ▶ **Caracteres:** um caractere (letra, número ou caractere especial).
 - ★ Ex: '**a**', '@' e '**1**'. (*Obs: esse tipo de dado não faz cálculos*)
 - ▶ **Literais:** uma cadeia de caracteres (*string*).
 - ★ Ex: "**aluno aprovado**", "**123**" e "**#prova**".

Representação de dados

- Operadores aritméticos.

Operador	Significado	Exemplo
+	adição de dois valores	$z = x + y$
-	subtração de dois valores	$z = x - y$
*	multiplicação de dois valores	$z = x * y$
/	quociente de dois valores	$z = x / y$
%	resto de uma divisão	$z = x \% y$

- Operadores lógicos.

Operador	Significado	Exemplo
&&	Operador E	$(x >= 0 \&\& x <= 9)$
	Operador OU	$(a == 'F' b != 32)$
!	Operador NEGAÇÃO	$!(x == 10)$

- Operadores relacionais.

Operador	Significado	Exemplo
>	Maior do que	$x > 5$
>=	Maior ou igual a	$x >= 10$
<	Menor do que	$x < 5$
<=	Menor ou igual a	$x <= 10$
==	Igual a	$x == 0$
!=	Diferente de	$x != 0$

Atividade Prática

- Treinar raciocínio lógico:

- ▶ [https://rachacuca.com.br/jogos/tags/raciocinio/.](https://rachacuca.com.br/jogos/tags/raciocinio/)
- ▶ [https://www.geniol.com.br/.](https://www.geniol.com.br/)

- Treinar lógica de programação:

- ▶ [http://silentteacher.toxicode.fr/.](http://silentteacher.toxicode.fr/)

Próxima aula...

- Apresentação de IDEs para desenvolvimento.



- Contextualização dos conceitos na prática.



Sugestões de pesquisa e leitura:

- ▶ **Code::Blocks** ⇒ <http://www.codeblocks.org/>.
- ▶ **Scratch** ⇒ <https://scratch.mit.edu/>.

Referências Bibliográficas

- ① ASCÊNCIO, A. F. G.; CAMPOS, E. A. V. ***Fundamentos da Programação de Computadores.*** 2002.
- ② PAES, R. B.; ***Introdução à Programação com Linguagem C.*** 2016.
- ③ BACKES, A.; ***Linguagem C: Completa e Descomplicada.*** 2013.
- ④ CELES, W.; Cerqueira, R.; Rangel, J. L. ***Introdução a Estruturas de Dados.*** 2016.

Obrigada pela atenção!