

## **UMA SEQUÊNCIA DIDÁTICA PARA O ENSINO DE ALGORITMOS NO CURSO TÉCNICO INTEGRADO EM INFORMÁTICA**



Crédito: Freepik

**BRUNO SOUZA**

**MOSSORÓ/RN**  
**2024**

## FICHA TÉCNICA DO PRODUTO EDUCACIONAL

**Título da Dissertação:** Proposta didática para o ensino de algoritmos no curso técnico integrado em informática.

**Vínculo do Produto:** O produto educacional é relacionado ao Programa de Pós-Graduação em Educação Profissional e Tecnológica – ProfEPT.

**Origem do Produto:** Produto educacional do *Campus* Mossoró, do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, como requisito para a obtenção do grau de Mestre em Educação Profissional e Tecnológica em Rede Nacional (ProfEPT).

**Autor:** Bruno Souza.

**Orientador:** Prof. Dr. Demóstenes Dantas Vieira.

**Público-alvo:** Estudantes do Curso Técnico Integrado em Informática.

**Finalidade:** Promover a melhoria da aprendizagem para a disciplina Algoritmos por meio da lógica de programação, do pensamento computacional e da resolução de problemas.

**Local:** Mossoró, Rio Grande do Norte, Brasil. Ano: 2024.

**Palavras-chave:** Sequência Didática. Lógica de programação. Pensamento computacional. Educação Profissional e Tecnológica.

FICHA CATALOGRÁFICA  
Biblioteca IFRN – Campus Mossoró

S729	<p>Souza, Bruno.</p> <p>Uma sequência didática para o ensino de algoritmos no curso técnico integrado em informática / Bruno Souza, Demóstenes Dantas Vieira. – 2024.</p> <p>44 f.</p> <p>Produto Educacional integrante da Dissertação: Proposta didática para o ensino de algoritmos no curso técnico integrado em informática. (Mestrado em Educação Profissional e Tecnológica) – Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, 2024.</p> <p>1. Sequência didática. 2. Lógica de programação. 3. Produto educacional. I. Vieira, Demóstenes Dantas. II. Título.</p> <p style="text-align: right;">CDU: 377:004.421(0.078)</p>
------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Ficha catalográfica elaborada pela bibliotecária  
Viviane Monteiro da Silva CRB15/758

BRUNO SOUZA

**PRODUTO EDUCACIONAL: UMA SEQUÊNCIA DIDÁTICA PARA O ENSINO DE ALGORITMOS NO CURSO TÉCNICO INTEGRADO EM INFORMÁTICA.**

Este produto educacional foi julgado, validado e aprovado para obtenção do título de mestre em Educação Profissional e Tecnológica do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, *Campus Mossoró*.

Mossoró, 21 de junho de 2024.

**BANCA EXAMINADORA**

---

Prof. Dr. Demóstenes Dantas Vieira (IFRN)

Orientador

---

Prof. Dra. Carla Katarina de Monteiro Marques (IFRN)

(Examinadora interna)

---

Prof. Dra. Aleksandra Nogueira de Oliveira Fernandes (IFRN)

(Examinadora externa)

## SUMÁRIO

<b>APRESENTAÇÃO .....</b>	<b>7</b>
<b>OBJETIVO GERAL .....</b>	<b>8</b>
<b>OBJETIVOS ESPECÍFICOS.....</b>	<b>8</b>
<b>O PRODUTO EDUCACIONAL: SEQUÊNCIA DIDÁTICA .....</b>	<b>9</b>
MÓDULO I .....	10
MÓDULO II.....	11
MÓDULO III.....	11
MÓDULO IV .....	12
<b>CONSIDERAÇÕES FINAIS.....</b>	<b>14</b>
<b>REFERÊNCIAS .....</b>	<b>15</b>
Apêndice A – módulo I - material de apoio .....	16
Apêndice B – módulo I - lista de situações-problema sobre lógica .....	18
Apêndice C – módulo I - situações-problema sobre pensamento computacional.....	24
Apêndice D – módulo I – exemplos de aplicação do pensamento computacional.....	25
Apêndice E – módulo I – atividade para casa – pensamento computacional.....	27
Apêndice F – módulo I – modelo de avaliação .....	28
Apêndice G – módulo II – material introdutório.....	29
Apêndice H – módulo II – situações-problema .....	30
Apêndice I – módulo II – slides utilizados .....	32
Apêndice J – módulo III – material introdutório.....	34
Apêndice K – módulo III – situação-problema .....	35
Apêndice L – módulo III – slides utilizados .....	36
Apêndice M – módulo III – modelo de avaliação .....	38
Apêndice N – módulo IV– situação problema - jogo <i>Lightbot</i> .....	39
Apêndice O – módulo IV– slides parte um .....	40
Apêndice P – módulo IV– slides parte dois .....	42

Apêndice Q – módulo IV– modelo de avaliação.....	44
--------------------------------------------------	----

## APRESENTAÇÃO

Este produto educacional, em formato de Sequência Didática, foi aplicado com 13 estudantes em dependência curricular na disciplina Algoritmos, do Curso Técnico Integrado em Informática, do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, no *Campus* Santa Cruz. Segundo Zabala, uma Sequência Didática é "um conjunto de atividades ordenadas, estruturadas e articuladas para a realização de certos objetivos educacionais" (Zabala, 1998, p. 18), ou seja, essa estratégia de ensino permite organizar as atividades de maneira lógica e coerente.

A Sequência Didática aqui apresentada utiliza a teoria de ensino construtivista, onde os estudantes são os sujeitos ativos que constroem o conhecimento a partir da experiência, interações sociais e discussões (Carretero, 2002), em conjunto com a abordagem cognitivista, que permite uma aprendizagem estruturada e significativa, onde novas informações são relacionadas aos conhecimentos prévios (Moreira, 2011).

Nesse sentido, a Sequência Didática foi elaborada com o objetivo de potencializar a aprendizagem da disciplina Algoritmos, tendo como finalidade ensinar a natureza elementar da lógica de programação (ordenação, sequenciação e repetição), do pensamento computacional (decomposição, identificação de padrões, abstração e algoritmo), que serve "para a resolução de problemas de forma inteligente" (Bittencourt, 2018, p. 100), e dos laços de repetição *while*, *do/while* e *for*.

Para atingir esse objetivo, foram desenvolvidos quatro módulos. O primeiro trata sobre o conceito, a importância e a usabilidade da lógica de programação e do pensamento computacional por meio de rodas de conversa e exercícios que envolvem situações-problema lógicos e matemáticos. O segundo e terceiro módulo possibilitam ensinar o funcionamento das estruturas de repetição *while* e *do/while* por meio da aprendizagem baseada em problemas, que busca "desenvolver competências que tornem os alunos aptos a analisar fatos, identificar evidências, buscar soluções, argumentar e a apresentar seus resultados" (Silva, Kalhil, 2015, p. 3).

Por último, o quarto módulo almeja ensinar o funcionamento e a utilidade do laço de repetição *for* por meio do jogo *Lightbot* e exercícios práticos que funcionam como meios para solidificar o conhecimento dos estudantes. Além disso, o uso do jogo como recurso educacional permite ratificar a importância da lógica de programação e do pensamento computacional para a resolução de problemas.

## OBJETIVO GERAL

Compreender a natureza dos algoritmos, com ênfase na lógica de programação, no pensamento computacional e na resolução de problemas.

## OBJETIVOS ESPECÍFICOS

- Desenvolver uma compreensão sólida sobre a importância da lógica de programação e do pensamento computacional para a resolução de problemas;
- Potencializar a compreensão dos laços de repetição *while*, *do/while* e *for* e suas respectivas importâncias nos problemas algorítmicos;
- Desenvolver a autonomia, a reflexão crítica, a análise de problemas, a colaboração e comunicação entre sujeitos.



## O PRODUTO EDUCACIONAL: SEQUÊNCIA DIDÁTICA

**Da importância:** a sequência didática foi elaborada visando o desenvolvimento da lógica de programação e do pensamento computacional, competência fundamental que facilita a resolução de problemas, a inovação e a participação ativa na sociedade moderna (Bittencour, 2018). Essas habilidades proporcionam aos estudantes uma capacidade de resolução de problemas mais eficaz que potencializa os processos de ensino e aprendizagem para a disciplina Algoritmos.

**Do formato:** cada módulo da sequência didática foi elaborado para facilitar a construção gradual do conhecimento. Embora tenha sido determinado esse formato, os módulos permitem a adaptação dos conteúdos e exercícios de acordo com as necessidades dos estudantes.

**Da Educação Profissional e Tecnológica:** esta sequência didática está diretamente vinculada à Educação Profissional e Tecnológica, pois promove a autonomia, o pensamento crítico, a iniciativa e a indissociabilidade entre teoria e prática nos processos de ensino e aprendizagem, o que estabelece relação direta com o mundo do trabalho, preceitos que estão no Documento Base/SETEC (Brasil, 2007).

**Aplicação:** durante os encontros, o docente deve incentivar a participação ativa dos estudantes por meio de discussões e situações-problema. Além disso, deve ser feita uma avaliação contínua com a finalidade de monitorar o progresso dos discentes com foco a ajustar os conteúdos conforme a necessidade dos sujeitos envolvidos.

**Expansão:** a sequência didática poderá ser expandida para quaisquer outros assuntos/disciplinas que necessitam da lógica de programação e do pensamento computacional, como a programação estruturada e orientada a objetos; programação com acesso a banco de dados; e programação para internet; pois todos esses conteúdos dependem da lógica de programação e dos pilares do pensamento computacional para um melhor desenvolvimento das habilidades de programação.

**Material utilizado (Apêndices):** o material utilizado nas aulas foi elaborado de acordo com o resultado de uma avaliação diagnóstica que delineou os perfis e os conhecimentos prévios dos estudantes, como forma de tornar significativo o percurso formativo. Sugere-se que essa ação seja implementada antes da aplicação da sequência didática para adaptar o material de acordo com a necessidade dos sujeitos, principalmente sobre o entendimento da lógica de programação e do pensamento computacional.

## MÓDULO I

<b>Tema</b>	A importância da lógica de programação e do pensamento computacional para a resolução de problemas.
<b>Horas/aula</b>	4 horas aula.
<b>Objetivo Geral</b>	Desenvolver uma compreensão sólida sobre a importância da lógica de programação e do pensamento computacional para a resolução de problemas.
<b>Objetivos específicos</b>	Desenvolver habilidades de resolução de problemas, incluindo a capacidade de analisar informações, identificar padrões e propor soluções; Identificar e aplicar princípios lógicos básicos, como indução e dedução, em situações práticas e cotidianas; Trabalhar de forma colaborativa com colegas e docente.
<b>Conteúdo</b>	<ul style="list-style-type: none"> <li>✓ O que é um problema;</li> <li>✓ Lógica de Programação;</li> <li>✓ Pensamento Computacional.</li> </ul>
<b>Metodologia</b>	<p>Roda de conversa e proposição de atividades sobre lógica e pensamento computacional para os discentes. Para tanto, devem ser realizadas as seguintes ações:</p> <p>Encontro 1</p> <ul style="list-style-type: none"> <li>• Apresentação da sequência didática para os estudantes e os objetivos a serem alcançados em cada módulo, assim como responder as dúvidas;</li> <li>• Desenvolvimento de uma roda de conversa sobre o significado do que é um “problema” (como a base de tudo para se buscar uma solução), em seguida o que é a “lógica” no sentido humano, e por fim o que é a lógica de programação;</li> <li>• Proposição de exercícios de lógica para os discentes resolverem abrindo espaço para o debate após cada situação-problema apresentado e fazendo analogia com a estrutura dos códigos de programação, no sentido lógico (sequência, ordem, identificação de padrões para solução);</li> <li>• Reforço dos principais conceitos abordados.</li> </ul> <p>Encontro 2</p> <ul style="list-style-type: none"> <li>• Desenvolvimento de uma roda de conversa sobre o significado do “pensamento” no sentido humano e o que é o pensamento computacional;</li> <li>• Apresentação de exemplos de aplicação do pensamento computacional ligados ao cotidiano (Ex. Organização de uma festa, Manutenção de um carro);</li> <li>• Proposição de exercícios de pensamento computacional para os discentes resolverem abrindo espaço para o debate após cada situação-problema apresentado, fazendo analogia com a estrutura dos códigos de programação (Decomposição, padrões, abstração, algoritmo);</li> <li>• reforço dos principais conceitos abordados.</li> </ul>
<b>Recursos pedagógicos</b>	Projetor, Pincel, Quadro Branco, Material impresso de apoio com o resumo da sequência didática, perguntas norteadoras, e exercícios que envolvem a utilização do raciocínio lógico.
<b>Avaliação</b>	Formativa (Participação nas rodas de conversa (20% da nota), prática durante a aula / engajamento (40% da nota)), somativa (avaliação (40% da nota)).

Fonte: Autoria própria (2023)

## MÓDULO II

<b>Tema</b>	A importância da estrutura de repetição <i>while</i> .
<b>Horas/aula</b>	2 horas aula.
<b>Objetivo Geral</b>	Desenvolver uma compreensão sólida sobre o laço de repetição <i>while</i> e sua importância na resolução de problemas algorítmicos.
<b>Objetivos específicos</b>	Aprender a trabalhar em equipe para resolver problemas, desenvolvendo habilidades de colaboração e comunicação; Resolver situações-problemas com a ajuda da estrutura de repetição <i>while</i> ; Explorar e descobrir novos conhecimentos.
<b>Conteúdo</b>	<ul style="list-style-type: none"> <li>➤ O que é o laço de repetição <i>while</i>;</li> <li>➤ Importância do laço de repetição <i>while</i>;</li> <li>➤ Estrutura e aplicação do laço de repetição <i>while</i>.</li> </ul>
<b>Metodologia</b>	<p>Aula no laboratório de informática e proposição de um desafio problema para os discentes. Para tanto, devem ser realizadas as seguintes ações:</p> <ul style="list-style-type: none"> <li>• Divisão da turma em pequenos grupos ou pares;</li> <li>• Fornecimento aos estudantes de material introdutório com instruções passo a passo para criar programas utilizando a estrutura de repetição <i>while</i>;</li> <li>• Proposição de situações-problemas que possam ser resolvidos com a ajuda da estrutura de repetição <i>while</i>;</li> <li>• Observação e suporte docente durante a implementação do desafio;</li> <li>• Discussão sobre o problema proposto;</li> <li>• Apresentação da aplicação do laço de repetição <i>while</i> em softwares do cotidiano;</li> <li>• Apresentação do funcionamento da estrutura de repetição <i>while</i>, explicando sua sintaxe básica, o fluxo de controle e o papel da condição de repetição;</li> <li>• Apresentação de exemplos de códigos que utilizam a estrutura <i>while</i>;</li> <li>• Discussão de exemplos simples com a turma de situações em que a estrutura <i>while</i> pode ser útil;</li> <li>• reforço dos principais conceitos abordados.</li> </ul>
<b>Recursos Pedagógicos</b>	Projetor, Pincel, Quadro Branco, Exercícios Práticos em Ambiente de Desenvolvimento Integrado.
<b>Avaliação</b>	Formativa (Discussão sobre o problema (20% da nota), prática durante a aula / engajamento (50% da nota)), somativa (avaliação (30% da nota)).

Fonte: Autoria própria (2023)

## MÓDULO III

<b>Tema</b>	A importância da estrutura de repetição <i>do/while</i> .
<b>Horas/aula</b>	2 horas aula.
<b>Objetivo Geral</b>	Desenvolver uma compreensão sólida sobre o laço de repetição <i>do/while</i> e sua importância na resolução de problemas algorítmicos.

<b>Objetivos específicos</b>	Aprender a trabalhar em equipe para resolver problemas, desenvolvendo habilidades de colaboração e comunicação; Resolver situações-problemas com a ajuda da estrutura de repetição <i>while</i> ; Explorar e descobrir novos conhecimentos.
<b>Conteúdo</b>	<ul style="list-style-type: none"> <li>➤ O que é o laço de repetição <i>do/while</i>;</li> <li>➤ Importância do laço de repetição <i>do/while</i>;</li> <li>➤ Estrutura e aplicação do laço de repetição <i>do/while</i>.</li> </ul>
<b>Metodologia</b>	<p>Aula no laboratório de informática e proposição de um desafio problema para os discentes. Para tanto, devem ser realizadas as seguintes ações:</p> <ul style="list-style-type: none"> <li>• Divisão da turma em pequenos grupos ou pares;</li> <li>• Fornecimento aos estudantes de material introdutório com instruções passo a passo para criar programas utilizando a estrutura de repetição <i>do/while</i>;</li> <li>• Proposição de situações-problemas que possam ser resolvidos com a ajuda da estrutura de repetição <i>do/while</i>;</li> <li>• Observação e suporte docente durante a implementação do desafio;</li> <li>• Discussão sobre o problema proposto;</li> <li>• Apresentação do funcionamento da estrutura de repetição <i>do/while</i>, explicando sua sintaxe básica, o fluxo de controle e o papel da condição de repetição;</li> <li>• Apresentação da aplicação do laço de repetição <i>do/while</i> em softwares do cotidiano;</li> <li>• Apresentação de exemplos de códigos que utilizam a estrutura <i>do/while</i>;</li> <li>• Discussão de exemplos simples com a turma de situações em que a estrutura <i>do/while</i> possa ser útil;</li> <li>• Reforço dos principais conceitos abordados.</li> </ul>
<b>Recursos Pedagógicos</b>	Projektor, Pincel, Quadro Branco, Exercícios Práticos em Ambiente de Desenvolvimento Integrado.
<b>Avaliação</b>	Formativa (Discussão sobre o problema (20% da nota), prática durante a aula / engajamento (50% da nota)), somativa (avaliação (30% da nota)).

Fonte: Autoria própria (2023).

#### MÓDULO IV

<b>Tema</b>	A importância da estrutura de repetição <i>for</i> .
<b>Horas/aula</b>	4 horas aula.
<b>Objetivo Geral</b>	Desenvolver uma compreensão sólida sobre o laço de repetição <i>for</i> e sua importância na resolução de problemas algorítmicos.
<b>Objetivos específicos</b>	Utilizar ferramentas e recursos digitais para apoiar o pensamento lógico e ordenado, incluindo jogos digitais; Demonstrar autonomia e iniciativa ao enfrentar desafios e resolver problemas utilizando o laço de repetição <i>for</i> .
<b>Conteúdo</b>	<ul style="list-style-type: none"> <li>➤ O que é o laço de repetição <i>for</i>;</li> <li>➤ Importância do laço de repetição <i>for</i>;</li> <li>➤ Estrutura e aplicação do laço de repetição <i>for</i>.</li> </ul>

<b>Metodologia</b>	<p>Aula no laboratório de informática e proposição de um desafio problema para os discentes. Para tanto, devem ser realizadas as seguintes ações:</p> <ul style="list-style-type: none"><li>• Utilização do jogo <i>LightBot</i> para demonstrar a utilidade da estrutura de repetição <i>for</i>, reforçando a lógica de programação (sequência, ordem) e o pensamento computacional (decomposição, padrão e algoritmo) na resolução do problema;</li><li>• Discussão de exemplos com a turma de utilidades em que a estrutura <i>for</i> seja útil, além do jogo;</li><li>• Apresentação de exemplos de códigos que utilizam a estrutura <i>for</i> explicando a sintaxe, o fluxo de controle e o papel da condição de repetição;</li><li>• Proposição de situações-problemas que possam ser resolvidos com a ajuda da estrutura de repetição <i>for</i>;</li><li>• Observação e suporte e durante a implementação do desafio;</li><li>• Reforço dos principais conceitos abordados.</li></ul>
<b>Recursos Pedagógicos</b>	Jogo Digital, Projetor, Pincel, Quadro Branco, Exercícios Práticos em Ambiente de Desenvolvimento Integrado.
<b>Avaliação</b>	Formativa (Discussão sobre o problema (10% da nota), prática durante a aula / engajamento (50% da nota)), somativa (avaliação (40% da nota)).

Fonte: Autoria própria (2023)

## CONSIDERAÇÕES FINAIS

A sequência didática aplicada junto aos estudantes em dependência curricular na disciplina Algoritmos, do Curso Técnico Integrado em Informática, no *Campus* Santa Cruz, apresentou resultados positivos quanto a melhoria nos índices de motivação, engajamento e facilidade na aprendizagem das estruturas de repetição: *while*, *do/while* e *for*, conteúdos pertencentes à disciplina Algoritmos.

Esse resultado é direcionado, principalmente, pelo desenvolvimento da lógica de programação e pensamento computacional, conteúdos presentes no primeiro módulo da sequência didática e continuados no decorrer do percurso formativo. Essa aprendizagem possibilita aos estudantes compreenderem elementos importantes para a elaboração dos algoritmos, como ordenação, sequenciação, repetição, decomposição, identificação de padrões e abstração, conhecimentos que contribuem para que os programas desenvolvidos sejam executados de maneira lógica e previsível, ou seja, mais eficientes.

De igual modo, a aprendizagem baseada em problemas e a aprendizagem significativa são importantes durante a execução das aulas, pois a primeira possibilita proporcionar desafios que podem ser superados em equipe por meio da colaboração, investigação e discussão, sendo seguido de posterior compreensão e resolução dos problemas. Da mesma maneira, a significação do conhecimento, que organiza o ensino e a aprendizagem de acordo com os conhecimentos prévios dos estudantes, é de suma importância para motivá-los a resolverem os problemas, pois eles compreendem melhor a importância e finalidade dos conteúdos (Moreira, 2011).

Por fim, o produto educacional busca proporcionar aos estudantes o desenvolvimento da autonomia e a possibilidade de emancipação na construção do próprio conhecimento por meio das discussões, aprendizagem baseada em problemas e colaboração em equipe, tornando-os mais capazes de criar soluções não apenas para o mundo informatizado, mas ligados aos seus respectivos cotidianos, características pertencentes à Educação Profissional e Tecnológica.

## REFERÊNCIAS

BITTENCOURT, R. L. de (organizador). **Políticas e práticas na formação de professores**. - São Paulo: Pimenta Cultural, 2018. 164p.

CARRETERO, M. **Construtivismo e Educação**. Porto Alegre: Artes Médicas, 2002.

DANTE, L. R. **Formulação e resolução de problemas de matemática: teoria e prática**. 1ª ed. São Paulo: Ática, 2009.

DE QUEIROZ, A. P. C. **Avaliação formativa: ferramenta significativa no processo de ensino e aprendizagem**. Anais VI CONEDU... Campina Grande: Realize Editora, 2019. Disponível em: <<https://editorarealize.com.br/artigo/visualizar/59424>>. Acesso em: 1 jun. 2023.

DE QUEIROZ, A. P. C. de; MACIEL, F. A. S. **Avaliação formativa: instrumento de formação contínua do professor em serviço**. Brazilian Journal of Development, v. 6, n. 5, p. 29784-29794, 2020.

FORBELLONE, A. L. V.; EBERSPACHER, H. F. **Lógica de programação: a construção de algoritmos e estruturas de dados**. 3.ed. São Paulo: Prentice Hall, 2005.

HAAN, L.; HOMAN, Z.; SURT; HEKTIKMUSIC. **Lightbot**. 2024. Original concept: lightbot.com. Disponível em: <https://www.lightbot.lu/>. Acesso em: 05 jun. 2023.

MINISTÉRIO DA EDUCAÇÃO. **Documento Base da Educação Profissional Técnica de Nível Médio Integrada ao Ensino Médio**. Brasília: MEC/Setec, dez. 2007.

MOREIRA, M. A. **Aprendizagem Significativa: a teoria e textos complementares**. São Paulo: Editora Livraria da Física, 2011.

O QUE É PENSAMENTO COMPUTACIONAL. [S. l.], 2022. Disponível em: <https://www.youtube.com/watch?v=pRpjYrdb9UY>. Acesso em: 22 ago. 2023.

PAIVA, S. R. **Pensamento Computacional e o Desenvolvimento de Competências para a Resolução de Problemas no Ensino Básico**. Rio de Janeiro: Ciência Moderna Ltda., 2022. 84p.

PUGA, S. **Lógica de programação e estruturas de dados, com aplicações em Java** / Sandra Puga, Gerson Rissetti. – 2. ed. – São Paulo: Pearson Prentice Hall, 2009.

SILVA, E. E. S.; KALHIL, J. B. Uma discussão sobre a utilização da Problem Based Learning (PBL) no ensino da Física para o nível médio. **Lat. Am. J. Sci. Educ**, v. 2, E. 22072, 2015.

ZABALA, A. **A Prática Educativa: Como ensinar**. Porto Alegre: ArtMed, 1998. 223p.



Instituto Federal de Educação, Ciência e Tecnologia  
do Rio Grande do Norte – IFRN  
Programa de Pós-Graduação em Educação

---

## MATERIAL DE APOIO

### O que é um problema?

De acordo com Lester (1983), um problema é "uma situação que um indivíduo ou um grupo quer ou precisa resolver e para a qual não dispõe de um caminho rápido e direto que o leve à solução" (LESTER, 1982, apud DANTE, 2009, p.12), OU SEJA, **qualquer situação a ser resolvida que não tenha uma solução imediata, é considerada um problema.**

### O que é Lógica de Programação?

É a maneira pela qual os programadores estruturam as instruções e os comandos em um programa de computador de forma lógica e sequencial, permitindo que o computador execute tarefas específicas (Forbellone, Eberspächer 2005), OU SEJA, **é a forma de organizar as instruções e comandos em um programa de computador de maneira lógica e em uma sequência ordenada.**

### O que é Pensamento Computacional?

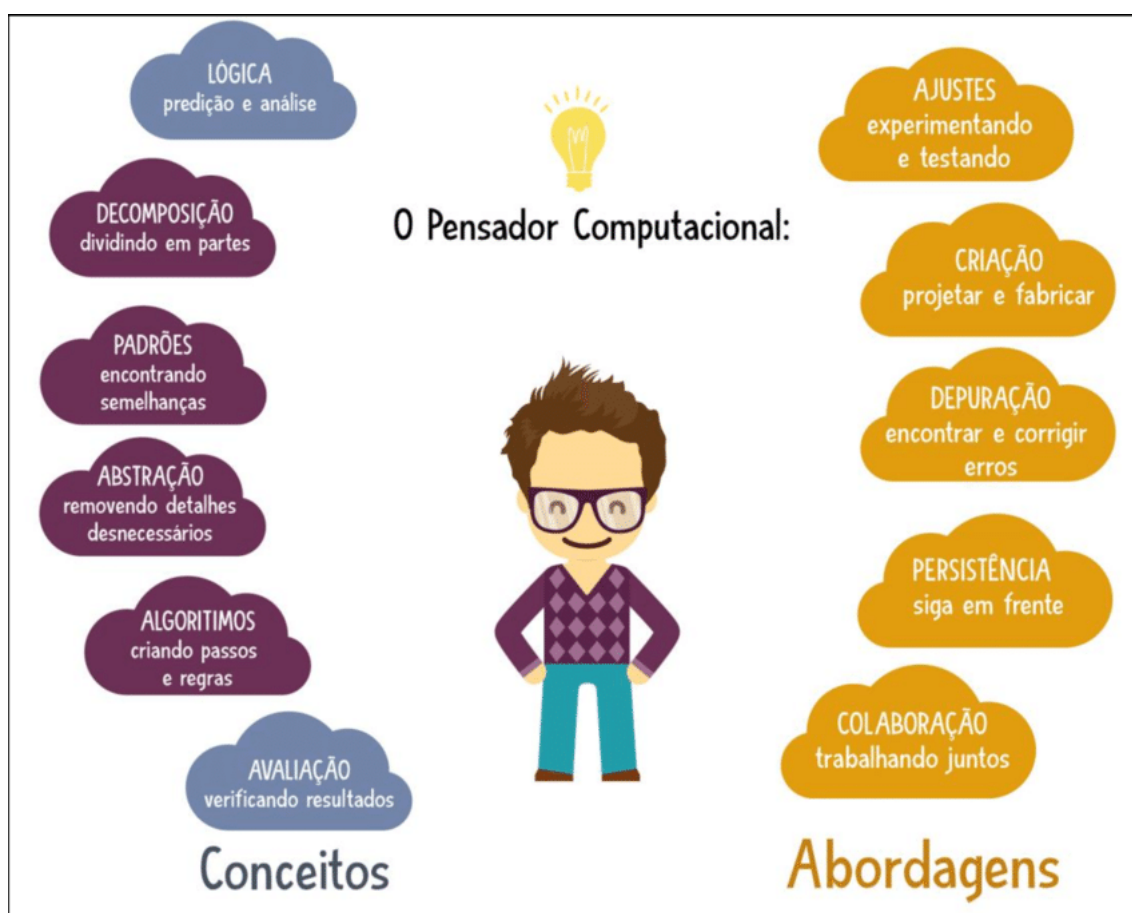
O pensamento computacional é uma **ABORDAGEM/METODOLOGIA PARA RESOLVER PROBLEMAS** e processar informações de maneira **semelhante** a como um computador abordaria essas tarefas (Paiva, 2022).

- Vídeo didático sobre o que é pensamento computacional:  
<https://www.youtube.com/watch?v=pRpjYrdb9UY>





## MATERIAL DE APOIO



Fonte: Bittencourt (2018).

Apêndice B – módulo I - lista de situações-problema sobre lógica



Instituto Federal de Educação, Ciência e Tecnologia  
do Rio Grande do Norte – IFRN  
Programa de Pós-Graduação em Educação

**Situação-problema 1**

a) Organize as ações abaixo de **maneira lógica e sequencial** para a correta obtenção do resultado!

Sacar dinheiro em um caixa eletrônico		
Ordem N°	AÇÕES	ORDEM CORRETA
	Pegue o dinheiro, o cartão e o comprovante.	
	Digite sua senha.	
	Caminhe até o caixa eletrônico.	
	Selecione a opção “Saque” no menu principal.	
	Digite a quantia que gostaria de sacar.	
	Pressione o botão ou toque na tela "Sim" ou "Não" para definir se quer ou não um comprovante.	
	Insira o seu cartão no caixa eletrônico.	

Pegar um livro emprestado na biblioteca do <i>Campus</i>		
Ordem N°	AÇÕES	ORDEM CORRETA
	Escolha o livro e pegue-o.	
	Receba o livro.	
	Escolha a seção do livro.	
	Digite sua matrícula no terminal de empréstimo.	
	Dirija-se à seção de empréstimo.	
	Digite sua senha no terminal de empréstimo.	



Instituto Federal de Educação, Ciência e Tecnologia

do Rio Grande do Norte – IFRN

Programa de Pós-Graduação em Educação

### Situação-problema 2

b) Descubra qual é o próximo número de cada sequência e qual é o **padrão** utilizado para cada resposta!

Sequência 1	Próximo número?	Padrão encontrado
10, 20, 30, 40, 50...		
Sequência 2	Próximo número?	Padrão encontrado
100, 90, 81, 73, 66, ...		
Sequência 3 - Fibonacci	Próximo número?	Padrão encontrado
1, 1, 2, 3, 5, 8, ...		

### Situação-Problema 3

Identifique os erros lógicos nos códigos abaixo.

#### 1- Erro lógico de execução sequencial do programa!

##### Algoritmo ErroDeSequencial1 (Pseudocódigo)

```
Var
nota1, nota2, nota3, media: real

media <- (nota1 + nota2 + nota3) / 3

nota1 <- 5
nota2 <- 7
nota3 <- 8

Se media >= 7 Entao
    Escreva ("Aprovado! Média: ", media)
Senao
    Escreva ("Reprovado! Média: ", media)
FimSe
FimAlgoritmo
```

##### Algoritmo ErroDeSequencial1

```
public class Main {
    public static void main(String[] args) {
        double nota1, nota2, nota3, media;

        media = (nota1 + nota2 + nota3) / 3;

        nota1 = 5;
        nota2 = 7;
        nota3 = 8;

        if (media >= 7) {
            System.out.println("Aprovado! Média: " + media);
        } else {
            System.out.println("Reprovado! Média: " + media);
        }
    }
}
```

**2- Erro lógico de execução sequencial do programa!****Algoritmo ErroDeSequencia2 (Pseudocódigo)**

```
Var  
numero1, numero2, resultado: inteiro  
  
resultado <- numero1 + numero2  
  
numero1 <- 10  
numero2 <- 5  
  
Escreva ("A soma de ", numero1, " e ", numero2, " é: ", resultado)  
  
FimAlgoritmo
```

**Algoritmo ErroDeSequencia2**

```
public class Main {  
  
    public static void main(String[] args) {  
  
        int numero1, numero2, resultado;  
  
resultado = numero1 + numero2;  
  
        numero1 = 10;  
        numero2 = 5;  
  
        System.out.println("A soma de " + numero1 + " e " + numero2 + " é: " + resultado);  
  
    }  
  
}
```

### 3 - Erro lógico na leitura das variáveis

#### Algoritmo ErroLeituraVariaveis (Pseudocódigo)

```
Var
    idade, nome: caractere

Escreva("Olá! Qual é o seu nome? ")
Leia(idade)

Escreva("Digite sua idade: ")
Leia(nome)

Escreva("Olá, ", nome, "! Você tem ", idade, " anos.")

FimAlgoritmo
```

#### Algoritmo ErroLeituraVariaveis

```
import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        Scanner leia = new Scanner(System.in);
        String nome;
        int idade;

        System.out.print("Olá! Qual é o seu nome? ");
        idade = leia.nextInt();

        System.out.print("Digite sua idade: ");
        nome = leia.next();

        System.out.println("Olá, " + nome + "! Você tem " + idade + " anos.");

    }

}
```

**4 - Erro de lógica na estrutura condicional (se-senão)****Algoritmo ErroTesteCondicional (Pseudocódigo)**

```
Var
    numero: inteiro

Escreva("Digite um número: ")
Leia(numero)

Se numero > 20 E numero < 10 Entao
    Escreva("O número está entre 20 e 10.")

Senao
    Escreva("O número não está entre 10 e 20.")

FimSe

FimAlgoritmo
```

**Algoritmo ErroTesteCondicional**

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner leia = new Scanner(System.in);

        int numero;
        System.out.print("Digite um número: ");
        numero = leia.nextInt();

        if (numero > 20 && numero < 10) {

            System.out.println("O número está entre 20 e 10.");

        } else {

            System.out.println("O número não está entre 10 e 20.");

        }

    }
}
```

Apêndice C – módulo I - situações-problema sobre pensamento computacional



Instituto Federal de Educação, Ciência e Tecnologia  
do Rio Grande do Norte – IFRN  
Programa de Pós-Graduação em Educação

1 - Resolva os problemas utilizando o pensamento computacional:

**Problema 1** – Desenvolva a adição de 2 números informados pelo teclado.

**Problema 2** – Desenvolva uma solução que leia um número inteiro pelo teclado e exiba o seu sucessor:

**Problema 3** – Desenvolva uma solução para calcular a área de um quadrado de lado L

*Um quadro para cada problema*

1 - DECOMPOSIÇÃO	2- RECONHECIMENTO DE PADRÕES
Quebrar o problema em partes menores.	Identificar os padrões/similaridades que ajudam a solucionar o problema.
3 - ABSTRAÇÃO	4- ALGORITMO
Focar apenas no que é importante para construir a solução	Sequência de passos para resolver o problema





Instituto Federal de Educação, Ciência e Tecnologia  
do Rio Grande do Norte – IFRN  
Programa de Pós-Graduação em Educação

---

## EXEMPLOS DE APLICAÇÃO DO PENSAMENTO COMPUTACIONAL (Para casa)

**Exemplo 1:** Desenvolva uma solução para um estudante que está com três provas na recuperação final e uma delas está com notas mais baixas do que as outras.

**Decomposição** (Quebrar o problema maior em problemas menores)

Estudar para a prova um;

Estudar para a prova dois;

Estudar para a prova três.

**Reconhecimento de padrões** (Experiência que ajuda a resolver o problema)

Estudar mais as provas com as notas menores;

Estudar mais as provas mais difíceis.

**Abstração** (Focar apenas no que é importante para auxiliar na solução)

Estudar o dobro do tempo para as provas mais difíceis e com menor nota

**Algoritmo** (Passo a passo para resolver o problema)

Estudar para a prova um;

Estudar para a prova dois;

Estudar para a prova três;

Estudar para a prova mais difícil e com nota mais baixa;

Estudar para a prova um;

Estudar para a prova dois;

Estudar para a prova três;

Estudar para a prova mais difícil e com nota mais baixa.

**Exemplo 2:** Desenvolva uma solução que calcule as 4 operações básicas pelo teclado do computador

**Decomposição** (Quebrar o problema maior em problemas menores)

Ler primeiro número do teclado

Selecionar a operação básica

Ler segundo número do teclado

Calcular a operação básica

Apresentar o resultado

**Reconhecimento de padrões** (Experiência que ajuda a resolver o problema)

Operação de adição

Operação de subtração

Operação de multiplicação

Operação de divisão

**Abstração** (Focar apenas no que é importante para auxiliar na solução)

Validar os dados de entrada para saber se são números

**Algoritmo** (Passo a passo para resolver o problema)

Ler o primeiro número

Validar o número

Selecionar a operação básica

Ler o segundo número

Validar o número

Calcular a operação básica

Apresentar o resultado

Apêndice E – módulo I – atividade para casa – pensamento computacional

---



**Instituto Federal de Educação, Ciência e Tecnologia**  
**do Rio Grande do Norte – IFRN**  
**Programa de Pós-Graduação em Educação**

---

1. Desenvolva uma solução que calcule o dobro de um número.
2. Desenvolva uma solução que junte nome, sobrenome e último nome em um texto só.
3. Desenvolva uma solução para calcular o total da conta de um lanche entre três amigos e dívida essa conta por 3.



**Instituto Federal de Educação, Ciência e Tecnologia**  
**do Rio Grande do Norte – IFRN**  
**Programa de Pós-Graduação em Educação**

---

## **EXERCÍCIO AVALIATIVO**

1 - Diga com suas palavras o que você entende por lógica de programação e para que ela serve.

2 - Diga com suas palavras o que você entende por pensamento computacional e para que ele serve.

## Apêndice G – módulo II – material introdutório

---



Instituto Federal de Educação, Ciência e Tecnologia  
do Rio Grande do Norte – IFRN  
Programa de Pós-Graduação em Educação

---

### Exemplo: fonte 1

Reference > Language > Structure > Control structure > While

## while

[Control Structure]

### Descrição

Um loop `while` irá se repetir continuamente, e infinitamente, até a expressão dentro dos parênteses (), se torne falsa. Algo deve mudar a variável testada, ou o loop `while` nunca irá encerrar. Isso pode ser no seu código, por exemplo, uma variável incrementada, ou uma condição externa, como a leitura de um sensor.

### Sintaxe

```
while (condição) {  
    // código a ser executado repetidamente  
}
```

A condição é uma expressão booleana que resulta em `true` ou `false`.

### Código de Exemplo

```
var = 0;  
while (var < 200) {  
    // faz algo repetitivo 200 vezes  
    var++;  
}
```

### Ver Também

**EXEMPLO** [Loop While \(Em Inglês\)](#)

**Fonte 1** <https://www.arduino.cc/reference/pt/language/structure/control-structure/while/>

**Fonte 2:** <https://www.guj.com.br/t/laco-de-repeticao-while/372766/2>

**Fonte 3:** <https://www.javaprogressivo.net/2012/08/loop-infinito-e-controlando-lacos-e.html>



Instituto Federal de Educação, Ciência e Tecnologia

do Rio Grande do Norte – IFRN

Programa de Pós-Graduação em Educação

## CONTEXTUALIZAÇÃO DO DESAFIO

Um programador escreveu um código que consegue somar apenas 04 números. Diante disso, ele decidiu ampliar a funcionalidade do código utilizando a estrutura de repetição **WHILE**. Para isso, você deve compreender o funcionamento da estrutura de repetição **WHILE**.

- Quando o laço de repetição **WHILE** deve ser usado?
- Qual a sintaxe(escrita) do laço de repetição **WHILE**?

1 - Utilize o laço de repetição **WHILE** para melhorar o código abaixo permitindo que possam ser somados 10 números, ao invés de 4. (**Utilize uma variável para controlar as iterações**).

```
Main.java
1 import java.util.Scanner;
2
3 public class Main {
4     public static void main (String[] args){
5         Scanner leia = new Scanner (System.in);
6
7         int soma = 0;
8         int numero = 0;
9
10        System.out.println ("Digite um número: ");
11        numero = leia.nextInt();
12        soma = soma + numero;
13
14        System.out.println ("Digite um número: ");
15        numero = leia.nextInt();
16        soma = soma + numero;
17
18        System.out.println ("Digite um número: ");
19        numero = leia.nextInt();
20        soma = soma + numero;
21
22        System.out.println ("Digite um número: ");
23        numero = leia.nextInt();
24        soma = soma + numero;
25
26        System.out.println("O resultado da soma, foi: " + soma);
27    }
28 }
29
30
```

2 - Utilize o laço de repetição *WHILE* para melhorar o código abaixo permitindo controlar o encerramento do laço quando o usuário digitar o número 0 no teclado.

```
Main.java
1 import java.util.Scanner;
2
3 public class Main {
4     public static void main (String[] args){
5         Scanner leia = new Scanner (System.in);
6
7         int soma = 0;
8         int numero = 0;
9
10        System.out.println ("Digite um número: ");
11        numero = leia.nextInt();
12        soma = soma + numero;
13
14        System.out.println ("Digite um número: ");
15        numero = leia.nextInt();
16        soma = soma + numero;
17
18        System.out.println ("Digite um número: ");
19        numero = leia.nextInt();
20        soma = soma + numero;
21
22        System.out.println ("Digite um número: ");
23        numero = leia.nextInt();
24        soma = soma + numero;
25
26        System.out.println("O resultado da soma, foi: " + soma);
27    }
28 }
29
30
```

**Laços de repetição (LOOPS)**

Prof. Bruno H. Souza

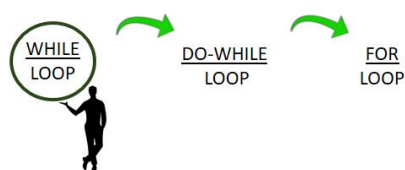
1

## Representação



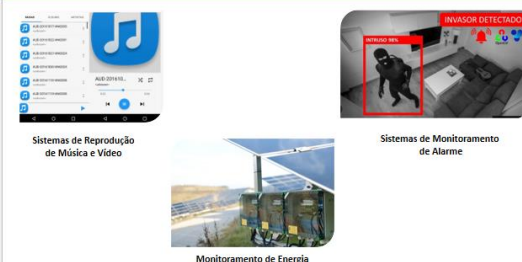
2

## Caminho!

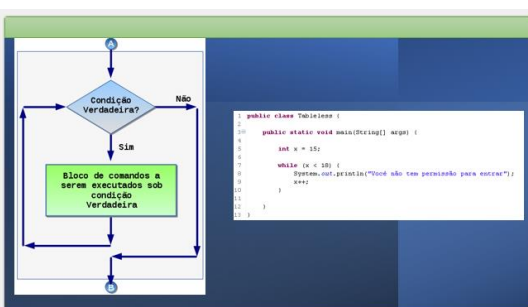


3

## Aplicação



4



5

## Sintaxe do WHILE

```

Main.java
1 public class Main
2 {
3     public static void main(String[] args) {
4         while (teste condicional) {
5             instruções
6         }
7     }
8 }
9
10
11
12
13

```

6

## Quando usar o WHILE

Quando não sabemos quantas vezes um determinado bloco de instruções precisa ser repetido



Quando é preciso validar uma condição antes de executar as instruções dentro do Loop



7

## Quando não sabemos quantas vezes um determinado bloco de instruções precisa ser repetido

```

Calculadora.java
1 import java.util.Scanner;
2
3 public class CalculadoraMediana {
4     public static void main(String[] args) {
5         Scanner leia = new Scanner(System.in);
6         int totalMedias = 0;
7         double somaMedias = 0.0;
8         while (true) {
9             System.out.println("Calculadora de Média Escolar");
10            System.out.print("Digite a nota (ou -1 para calcular a média): ");
11            double nota = leia.nextDouble();
12            if (nota == -1) {
13                break;
14            }
15            somaMedias += nota;
16            totalMedias++;
17        }
18        if (totalMedias == 0) {
19            System.out.println("Você não escolheu a média");
20        } else {
21            System.out.println("Médias nota inserida. Média não calculada.");
22        }
23    }
24 }

```

Calculadora de Média Escolar  
Digite a nota (ou -1 para calcular a média): 10  
Digite a nota (ou -1 para calcular a média): 8  
Digite a nota (ou -1 para calcular a média): 9  
Digite a nota (ou -1 para calcular a média): 6  
Digite a nota (ou -1 para calcular a média): -1  
Média escolar é: 8,75  
...Program finished with exit code 0  
Press ENTER to exit console.

8



### Quando é preciso validar uma condição antes de executar o código do loop

```
ValidacaoDeNota.java: 1
1 import java.util.Scanner;
2
3 public class ValidacaoDeNota {
4
5     public static void main(String[] args) {
6         Scanner leia = new Scanner(System.in);
7
8         double nota;
9
10        System.out.println("Validação de Nota Escolar");
11
12        while (true) {
13            System.out.print("Digite uma nota (de 0 a 10): ");
14
15            nota = leia.nextDouble();
16
17            if (nota >= 0 && nota <= 10) {
18                break;
19            } else {
20                System.out.println("Nota inválida. Por favor, digite uma nota entre 0 e 10.");
21            }
22        }
23
24        System.out.println("Você inseriu a nota: " + nota);
25    }
26 }
27
28
```

input

```
Validação de Nota Escolar
Digite uma nota (de 0 a 10): 11
Nota inválida. Por favor, digite uma nota entre 0 e 10.
Digite uma nota (de 0 a 10): 15
Nota inválida. Por favor, digite uma nota entre 0 e 10.
Digite uma nota (de 0 a 10): 8
Você inseriu a nota: 8.0

...Program finished with exit code 0
Press ENTER to exit console.
```

## Apêndice J – módulo III – material introdutório

---



Instituto Federal de Educação, Ciência e Tecnologia  
do Rio Grande do Norte – IFRN  
Programa de Pós-Graduação em Educação

---

### Exemplo: fonte 1

Reference > Language > Structure > Control structure > Dowhile

## do...while

[Control Structure]

### Descrição

O loop `do...while` funciona da mesma forma que o loop `while`, com a exceção de a condição ser testada no final do loop, tal que o loop será executado pelo menos uma vez.

### Sintaxe

```
do {  
  // bloco de comandos  
} while (condição);
```

A condição é uma expressão booleana que é avaliada como verdadeiro ou falso, respectivamente `true` ou `false` na linguagem Arduino.

### Código de Exemplo

```
int x = 0;  
do {  
  delay(50);           // espera os sensores estabilizarem  
  x = readSensors();   // checa os sensores  
} while (x < 100);
```

### Ver Também

**Fonte 1** <https://www.arduino.cc/reference/pt/language/structure/control-structure/dowhile/>

**Fonte 2:** <https://www.javaprogressivo.net/2012/09/o-laco-do-while-o-laco-que-sempre.html>

**Fonte 3:** <https://javatpoint.com/pt/declaração-break-em-java>



Instituto Federal de Educação, Ciência e Tecnologia  
do Rio Grande do Norte – IFRN  
Programa de Pós-Graduação em Educação

---

### CONTEXTUALIZAÇÃO DO DESAFIO

Um programador precisa escrever um algoritmo que calcule operações de adição e multiplicação utilizando o laço de repetição **DO/WHILE**. Para isso, você deve compreender o funcionamento da estrutura de repetição **DO/WHILE**.

- a) Quando o laço de repetição **DO/WHILE** deve ser usado?
- b) Qual a sintaxe(escrita) do laço de repetição **DO/WHILE**?


1 - Desenvolva um código em *Java* que calcule as operações de **adição** e **multiplicação**.

2 - Os números devem ser inseridos a partir do teclado.

3 - O código deve ainda apresentar um menu (1 - Adição, 2 - Multiplicação, 3 - Sair)

(Utilize o laço de repetição **DO/WHILE** para criar o menu e o comando **BREAK** para encerrar/sair do laço).

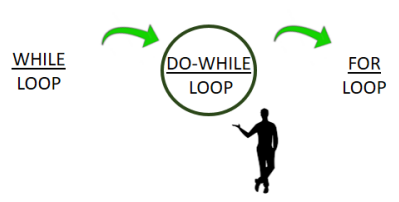
## Apêndice L – módulo III – slides utilizados



# Laços de repetição (LOOPS)

Prof. Bruno H. Souza


### Caminho!



**WHILE**  

```

graph TD
    A[enquanto condição] -- verdade --> B[bloco de instruções]
    B -- fim enquanto --> C[ ]
    A -- falso --> C
        
```



**DO-WHILE**  

```

graph TD
    A[repita] --> B[bloco de instruções]
    B -- verdade --> A
    B -- falso --> C[ ]
        
```

**WHILE**  

```

graph TD
    A[enquanto condição] -- verdade --> B[bloco de instruções]
    B -- fim enquanto --> C[ ]
    A -- falso --> C
        
```

```
while ("for verdade") {
    "instruções"
}
```

```
do {
    "Instruções"
}while ("for verdadeiro");
```

**DO-WHILE**  

```

graph TD
    A[repita] --> B[bloco de instruções]
    B -- verdade --> A
    B -- falso --> C[ ]
        
```

### Quando usar DO-WHILE

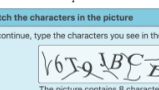
Quando se deseja que um **conjunto de instruções** seja executado **por pelo menos uma vez** e, em seguida, repetido enquanto uma condição específica for verdadeira!

### Exemplos de Aplicação

**Validação de dados**

Match the characters in the picture

To continue, type the characters you see in the picture. Why?




The picture contains 8 characters.

Characters:

Continue

**Menu de opções**



### Exemplo 1 – Validação dos dados

```
do {
    num1 = random.nextInt(10); // Gera um número aleatório de 0 a 9
    num2 = random.nextInt(10); // Gera outro número aleatório de 0 a 9

    soma = num1 + num2;

    System.out.println("Por favor, resolva o CAPTCHA abaixo:");
    System.out.println(num1 + " + " + num2 + " = ?");

    resposta = scanner.nextInt();

    if (resposta == soma) {
        System.out.println("CAPTCHA resolvido com sucesso!");
        break; // Sai do loop se a resposta estiver correta
    } else {
        System.out.println("Resposta incorreta. Tente novamente.");
    }
} while (true);
```

## Exemplo 2 – Menu de opções

```
do {
    System.out.println("Menu de Comida:");
    System.out.println("1. Pizza");
    System.out.println("2. Hambúrguer");
    System.out.println("3. Sushi");
    System.out.println("4. Sair");
    System.out.print("Escolha uma opção: ");

    escolha = scanner.nextInt();

    if (escolha == 1) {
        System.out.println("Você escolheu Pizza!");
    } else if (escolha == 2) {
        System.out.println("Você escolheu Hambúrguer!");
    } else if (escolha == 3) {
        System.out.println("Você escolheu Sushi!");
    } else if (escolha == 4) {
        System.out.println("Saindo do menu.");
    } else {
        System.out.println("Opção inválida. Tente novamente.");
    }
} while (escolha != 4);
```

```
Menu de Comida:
1. Pizza
2. Hambúrguer
3. Sushi
4. Sair
Escolha uma opção: 1
Você escolheu Pizza!

Menu de Comida:
1. Pizza
2. Hambúrguer
3. Sushi
4. Sair
Escolha uma opção: 4
Saindo do menu.
```

9

## INTERROMPENDO UM LAÇO

# Break;



10

## INTERROMPENDO UM LAÇO

```
do {
    System.out.print("Digite a senha (somente números): ");
    int senhaDigitada = scanner.nextInt();
    tentativas++;

    if (senhaDigitada == 12345) {
        System.out.println("Acesso permitido!");
        break; // Sai do loop imediatamente se a senha estiver correta
    } else {
        System.out.println("Senha incorreta. Tente novamente.");
    }

    if (tentativas >= 3) {
        System.out.println("Número máximo de tentativas atingido. Acesso bloqueado.");
        break; // Sai do loop se o número máximo de tentativas for atingido
    }
} while (true);
```

11

## Dúvidas?

Bruno.souza@ifrn.edu.br

12



Instituto Federal de Educação, Ciência e Tecnologia  
do Rio Grande do Norte – IFRN  
Programa de Pós-Graduação em Educação

---

## EXERCÍCIO AVALIATIVO

- 1 – Escreva a sintaxe do laço de repetição WHILE!
- 2- Quando usar o laço de repetição WHILE?
- 3 – Escreva a sintaxe do laço de repetição DO-WHILE!
- 4- Quando deve ser usado o laço de repetição DO-WHILE?

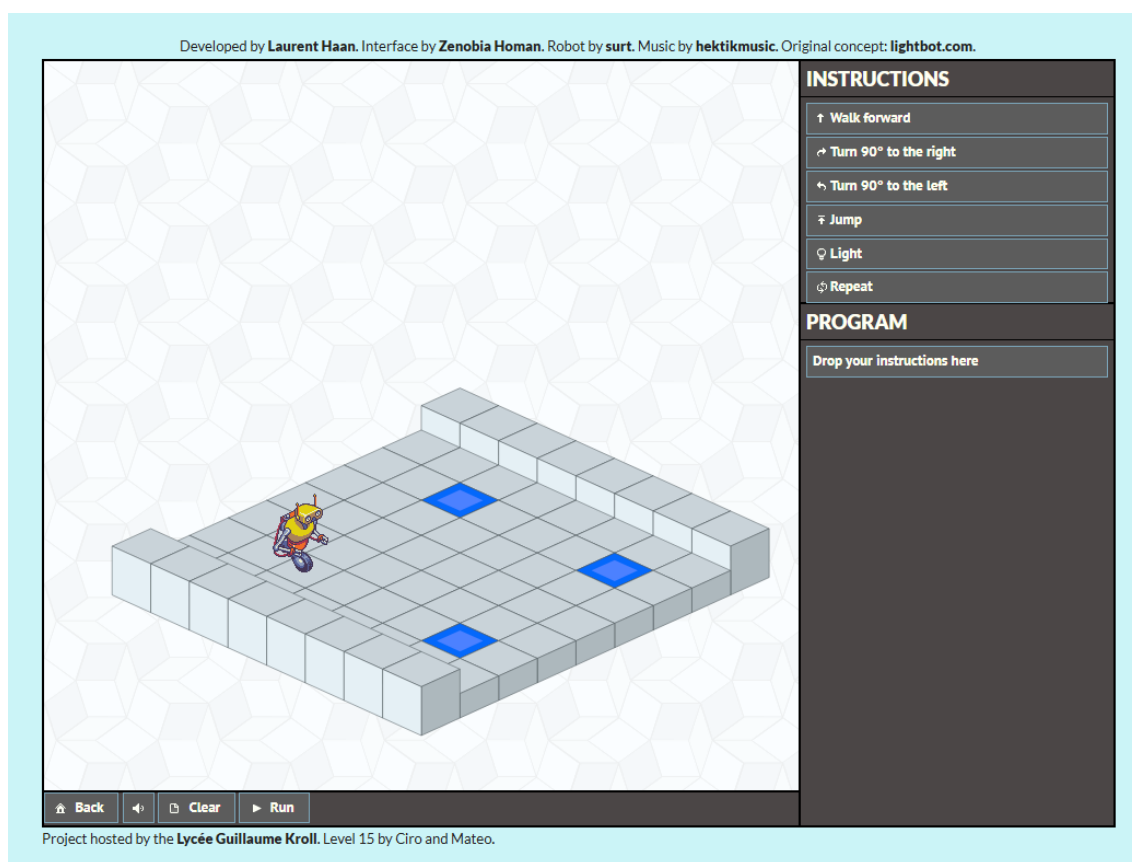
Apêndice N – módulo IV – situação problema - jogo *Lightbot*



Instituto Federal de Educação, Ciência e Tecnologia


do Rio Grande do Norte – IFRN

Programa de Pós-Graduação em Educação



Fonte: <https://www.lightbot.lu/>

## Apêndice O – módulo IV – slides parte um



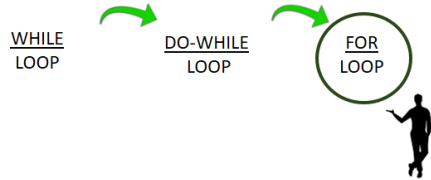
# Laços de repetição (LOOPS)

Prof. Bruno H. Souza

### Caminho!

```

  WHILE LOOP  →  DO-WHILE LOOP  →  FOR LOOP
  
```



### Sintaxe

```

for (inicialização; condição; atualização) {
    // Bloco de código a ser repetido
}
  
```

Inicialização	Inicializa uma variável de controle.
Condição	determina quando o loop será encerrado
Atualização (Incremento / decremento)	atualiza a variável de controle após cada iteração

### Componentes do FOR

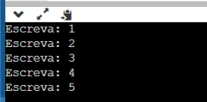
	Inicialização;	Condição;	Atualização*	Iterações (repetições)
FOR 1	( X = 1;	X <= 5;	X++	5
FOR 2	( int X = 10;	X > 0;	X--	10
FOR 3	( X = variavelUsuario;	X > 0;	X--	Usuário escolhe

\* Final da iteração!

### Exemplo for 1

```

9 public class Main
10 {
11     public static void main(String[] args) {
12         for (int x = 1; x <= 5; x++) {
13             System.out.println("Escreva: " + x);
14         }
15     }
16 }
  
```



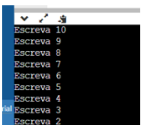
#### Saída

Valor X 1ª iteração	Escreva 1
Valor X 2ª iteração	Escreva 2
Valor X 3ª iteração	Escreva 3
Valor X 4ª iteração	Escreva 4
Valor X 5ª iteração	Escreva 5

### Exemplo for 2

```

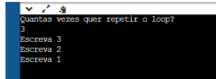
4- for (int x=10; x>1; x--) {
5-     System.out.println("Escreva " + x);
6- }
  
```



### Exemplo for 3

```

7
8 System.out.println("Quantas vezes quer repetir o loop? ");
9
10 variavelUsuario = leia.nextInt();
11
12 for (int x = variavelUsuario; x>0; x--) {
13     System.out.println("Escreva " + x);
14 }
15
16
17
  
```



### Quando usar o FOR LOOP

Quando **sabemos a quantidade** de vezes que um determinado bloco de código será repetido.



## Exemplos de Aplicação

### Jogos eletrônicos



ÓRDENS EXTRA SCZ x R0

01	02	03	04	05	06	07	08	09	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

COMANDOS

01	02	03	04	05	06	07	08	09	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50

Reservas de assentos

9

### Pulando uma iteração!

# Continue;



10

#### EXEMPLO 1

```
1- public class PularNumeros {
2-     public static void main(String[] args) {
3-         System.out.println("Números pares de 1 a 20:");
4-         for (int i = 1; i <= 20; i++) {
5-             if (i % 2 != 0) {
6-                 continue; // Pular números ímpares e continuar com a próxima iteração
7-             }
8-             System.out.println(i);
9-         }
10-    }
11- }
12- }
13- }
```

% -> Operador MOD (Resto da divisão)

```
Números pares de 1 a 20:
2
4
6
8
10
12
14
16
18
20
```

#### EXEMPLO 2

```
1- public class ExemploContinue2 {
2-     public static void main(String[] args) {
3-         System.out.println("Números pares de 1 a 10:");
4-         int i = 1;
5-         while (i <= 10) {
6-             if (i % 2 != 0) {
7-                 i++; // Incrementa o contador e pula números ímpares
8-                 continue; // Pula a iteração atual e continua com a próxima
9-             }
10-            System.out.println(i);
11-            i++; // Incrementa o contador após imprimir um número par
12-        }
13-    }
14- }
15- }
```

```
Números pares de 1 a 10:
2
4
6
8
10
```

#### EXEMPLO 1

```
1- public class PularNumeros {
2-     public static void main(String[] args) {
3-         System.out.println("Números pares de 1 a 20:");
4-         for (int i = 1; i <= 20; i++) {
5-             if (i % 2 != 0) {
6-                 continue; // Pular números ímpares e continuar com a próxima iteração
7-             }
8-             System.out.println(i);
9-         }
10-    }
11- }
12- }
13- }
```

% -> Operador MOD (Resto da divisão)

```
Números pares de 1 a 20:
2
4
6
8
10
12
14
16
18
20
```

11

#### EXEMPLO 2

```
1- public class ExemploContinue2 {
2-     public static void main(String[] args) {
3-         System.out.println("Números pares de 1 a 10:");
4-         int i = 1;
5-         while (i <= 10) {
6-             if (i % 2 != 0) {
7-                 i++; // Incrementa o contador e pula números ímpares
8-                 continue; // Pula a iteração atual e continua com a próxima
9-             }
10-            System.out.println(i);
11-            i++; // Incrementa o contador após imprimir um número par
12-        }
13-    }
14- }
15- }
```

```
Números pares de 1 a 10:
2
4
6
8
10
```

12

## Dúvidas?

Bruno.souza@ifrn.edu.br

13

## Situação-Problema

### Utilizando FOR

1 - Escreva um programa que faça a contagem regressiva para o ano novo.

2 - escreva um programa que imprima os múltiplos de 2 (2, 4, 6, 8, 10, 12, 14, 16, 18, 20)

```
3
2
1
Feliz Ano Novo!
```

```
2
4
6
8
10
12
14
16
18
20
```

14

Apêndice P – módulo IV– slides parte dois

Prof. Bruno H. Souza

### Variável

Variável numero1

### Variável

resultado = numero1 + numero2 ;

### Array

1. É uma coleção de elementos do mesmo tipo que são armazenados em uma única variável.
2. São acessados por meio dos índices. (0, 1, 2, 3 ...)

### Array

### Array – Sintaxe (escrita)

```

tipo[] nomeDaVariavel

int[] numeros = {1, 2, 3, 4, 5};

double[] precos = {19.99, 29.99, 39.99, 49.99};

String[] nomes = {"Alice", "Bob", "Carol", "David"};

char[] vogais = {'a', 'e', 'i', 'o', 'u'};

```

### Array / Sintaxe (escrita)

```

int[] numeros = {1, 2, 3, 4, 5};

int [] arrayVazio = new int[5];

```

### Como acessar os valores/índice de um Array?

Variáveis	Arrays
numero = 20;	numero[2] = 20;
nome = "Alice";	nome[4] = "Alice";

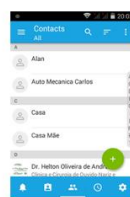
Imprimir uma variável	Imprimir um Array
System.out.println (numero);	System.out.println (numero[2]);
System.out.println (nome);	System.out.println (nome[4]);

## Quando usar um Array?

Quando precisamos armazenar uma coleção fixa de **elementos do mesmo tipo** e quando desejamos acessar esses elementos **por índices numéricos** de forma eficiente!

9

## Exemplos de Aplicação



Agenda de Contatos



Galeria de fotos

10

## Como percorrer um Array?

```
ImprimirArray.java:
1 public class ImprimirArray{
2     public static void main (String [] args){
3
4         String[] nomes = {"Alice", "Bob", "Carol", "David"};
5
6         for (int i = 0; i < nomes.length; i++) {
7             System.out.println(nomes[i]);
8         }
9     }
10 }
11
12 }
```

Dúvidas?

Bruno.souza@ifrn.edu.br

## Apêndice Q – módulo IV – modelo de avaliação



Instituto Federal de Educação, Ciência e Tecnologia

do Rio Grande do Norte – IFRN

Programa de Pós-Graduação em Educação

### EXERCÍCIO AVALIATIVO

```

1- public class Main {
2-     public static void main (String [] args){
3-
4-         int[] vetorNumeros = {1, 2, 3, 4};
5-         String[] listaDeNomes = {"Alice", "Bianca", "Carla", "David"};
6-         String[] listaDeNomes2 = {"Elisa", "Fátima", "Gabriel", "Humberto"};
7-         int [] vetorVazio = new int [4];
8-         int variavel = 0;
9-         vetorVazio[2] = vetorNumeros[3];
10-
11-         System.out.println (vetorNumeros[3]);
12-         System.out.println (listaDeNomes[2]);
13-         System.out.println (variavel);
14-
15-         for (int i = 0; i < listaDeNomes2.length; i++){
16-             System.out.println (listaDeNomes2[i]);
17-         }
18-
19-         System.out.println (vetorVazio[2]);
20-     }
21- }

```

- Qual o valor da variável de inicialização do laço for?
- Quantas vezes o laço vai repetir?
- A cada iteração (repetição), o que acontecerá com a variável de inicialização?

**Qual a saída dos comandos abaixo?**

System.out.println (vetorNumeros[3]);	
System.out.println (listaDeNomes[2]);	
System.out.println (variavel);	
System.out.println (listaDeNomes2[i])	
System.out.println (vetorVazio[2])	