Um Mapeamento da Literatura sobre o Uso do Pensamento Computacional e da Teoria da Carga Cognitiva no Processo de Ensino e Aprendizagem da Programação

Helder Daniel de Azevedo Dias [1], Josivaldo de Souza Araújo [1]

[1]Instituto de Ciências Exatas e Naturais - ICEN – Universidade Federal do Pará (UFPA)

helder.dias@icen.ufpa.br, josivaldo@ufpa.br

Abstract. *The undergraduate courses in Computing have high retention and evasion rates in their initial periods, mainly due to difficulties in understanding disciplines such as Algorithm and Programming. This is because learning to program isn't a simple task, much less trivial. It is a highly cognitive skill, where one needs to have multiple domains such as abstraction, logical reasoning and algorithmic. This work presents a Literature Mapping on the teaching of programming using concepts from Cognitive Load Theory on the pillars of Computational Thinking. The research used seven international bases and, after applying the defined criteria, seven primary studies were selected that answered the research questions.*

Resumo. *Os cursos de graduação em Computação apresentam elevados índices de retenção e evasão nos seus períodos iniciais, principalmente por dificuldades no entendimento de disciplinas como Algoritmo e Programação. Isso porque aprender a programar não é uma tarefa simples, e muito menos trivial. É uma habilidade altamente cognitiva, onde se necessita ter múltiplos domínios como abstração, raciocínio lógico e algorítmico. Este trabalho apresenta um Mapeamento da Literatura sobre o ensino da Programação utilizando conceitos da Teoria da Carga Cognitiva sobre os pilares do Pensamento Computacional. A pesquisa utilizou sete bases internacionais e, após a aplicação dos critérios definidos, foram selecionados sete estudos primários que responderam às questões de pesquisa.*

1. Introdução

Apesar de suas relevâncias, as disciplinas de Algoritmo e Programação, não representam sozinhas a realidade de uma matriz curricular de um curso de graduação em Computação, no entanto, apresentam baixa motivação, altos índices de retenção e até mesmo evasão. Estudos apontam taxas de insucessos que ficam em torno de 30%, apenas considerando essas disciplinas dos períodos iniciais [Berssanette and Francisco 2022]. Isso se deve, principalmente, às dificuldades na compreensão dos conceitos algorítmicos para a resolução dos problemas, no elevado grau de abstração e na construção de modelos mentais que exigem, algumas vezes, raciocínio lógico aprimorado [Almeida Jr. and Araújo 2022].

Para tentar contornar essa realidade, várias técnicas, teorias, metodologias e abordagens de ensino vêm sendo estudadas, aplicadas e apresentadas, com o objetivo de auxiliar uma melhor compreensão e entendimento dos conceitos por parte do aluno, assim como motivá-los, de forma a evitar a retenção e a evasão. Porém, o uso de qualquer dessas

propostas, não vai, de forma isolada ou até mesmo em conjunto, solucionar as dificuldades que são inerentes ao aprendizado da Programação, pois existem dificuldades que são próprias da natureza do aluno, fazendo com que os professores necessitem identificar e acompanhar esse aprendizado [Berssanette and Francisco 2022].

O objetivo deste trabalho é apresentar um Mapeamento Sistemático da Literatura (MSL) sobre o ensino da Programação nos cursos de graduação em Computação, utilizando a Teoria da Carga Cognitiva sobre os pilares do Pensamento Computacional. Dessa forma, pretende-se definir, entre outras informações, as principais abordagens, atividades e ferramentas utilizadas.

2. Trabalhos Relacionados

O trabalho de [Rijo-García et al. 2022] apresenta uma revisão bibliográfica sobre como a experiência do usuário impacta o desenvolvimento do Pensamento Computacional. O autor conclui que é necessário se aprofundar mais nos efeitos causados pelas tecnologias que são usadas para desenvolver o Pensamento Computacional, sendo esta uma linha de pesquisa digna de consideração.

O trabalho de [Duran et al. 2022] realiza uma Revisão Sistemática da Literatura (RSL), pesquisando de que forma a Teoria da Carga Cognitiva (TCC) vêm sendo utilizada em Educação Computacional desde 2010. O artigo retrata que a razão mais comum para citar a TCC é somente mencioná-la, brevemente, como uma influência de *design* de material instrucional, e não uma aplicação prática da TCC em sala de aula.

Já [Zhan et al. 2025] apresenta um estudo que contemplou os temas Pensamento Computacional e Carga Cognitiva, utilizando-se de Realidade Aumentada e Aprendizagem baseada em Jogos no ensino de ”Cultura de *Guangzhow*”em uma Universidade na China. O estudo, inclusive, prepara os alunos para o ”Teste Internacional Bebras de Pensamento Computacional”[1], demonstrando quão valiosa pode ser a aplicação destes conceitos em áreas transversais da Educação, não somente no ensino de Programação.

A relevância deste estudo reside na ausência de sínteses sistemáticas que integrem e associem o Pensamento Computacional com a Teoria da Carga Congnitiva, oferecendo uma visão estruturada sobre um campo que ainda precisa ser explorado. Ao sistematizar estudos dispersos na literatura, esse mapeamento contribui tanto para o avanço da pesquisa sobre educação em computação quanto na prática docente, ao apoiar professores a compreenderem melhor a natureza de cada aluno. Além disso, o estudo aponta subsídios para o desenvolvimento de intervenções que podem ser mais alinhadas no processo de ensino da Programação.

3. Pensamento Computacional

Segundo [Navarro et al. 2024], o termo Pensamento Computacional foi citado pela primeira vez na Educação Matemática por Seymor Papert, um dos idealizadores da linguagem Logo, nos anos de 1960. Muitos anos depois, Jeannette Wing publicou um artigo com o objetivo de abordar o Pensamento Computacional no contexto educacional [Wing 2006]. Hoje este conceito é descrito como uma capacidade de compreender, identificar e explicar as dimensões quantitativas de um problema que tenha solução algorítmica,

[1]https://www.bebras.org/

resolvendo-o em ambientes de Programação [MEC 2016a]. O conceito do Pensamento Computacional ganhou tanta relevância que, em conjunto com o Mundo Digital e a Cultura Digital, tornaram-se eixos da Base Nacional Comum Curricular da Computação, a BNCC Computação [BNCC 2022].

Na BNCC Computação, o Pensamento Computacional possui competências e habilidades que vão desde a Educação Infantil até o Ensino Médio, e tem como objetivo fazer com que os estudantes compreendam que são agentes ativos do mundo em que vivem, além de poderem aplicar os conceitos de Computação para desenvolverem soluções para o cotidiano [BNCC 2022]. Este aprendizado proporciona também o desenvolvimento de uma maior autonomia, flexibilidade, criatividade e uma melhor capacidade de raciocínio. Tudo isso, somado às metodologias e ferramentas adequadas, favorece positivamente o desenvolvimento interdisciplinar dos alunos [Machado and Dutra 2023].

4. Teoria da Carga Cognitiva

A Teoria da Carga Cognitiva (TCC), formulada por Sweller, visa compreender a influência da capacidade da memória de trabalho na aprendizagem [Kouam 2025]. Qualquer tarefa que precise ser realizada, individualmente ou em equipe, traz consigo uma certa quantidade de carga cognitiva [Sweller 1994]. Parte dessa carga é intrínseca à tarefa em si e é determinada pela complexidade da mesma, ou seja, o número de elementos que precisam ser processados pela pessoa ou grupo que realizam a tarefa e a interatividade entre estes elementos [Kirschner et al. 2008].

Para [Sweller 1994], existem três diferentes tipos de cargas cognitivas:

- **Carga Intrínseca**, que é a carga inerente à dificuldade da tarefa de aprendizagem;
- **Carga Pertinente**, que é composta de recursos necessários na memória de trabalho para gerenciar a carga intrínseca, que, por sua vez, dá suporte à aprendizagem;
- **Carga Estranha**, que surge do *design* instrucional e dificulta à aprendizagem.

A TCC afirma que a carga estranha deve ser minimizada para que os recursos pertinentes possam ser maximizados. Portanto, os *designers* instrucionais devem considerar a redução da quantidade de informações que não contribuem para a aprendizagem (ou seja, carga cognitiva estranha), enquanto aumentam a carga pertinente, concentrando-se em estabelecer conexões entre o conhecimento prévio dos alunos e o novo conteúdo.

5. Mapeamento Sistemático da Literatura

Um Mapeamento Sistemático da Literatura (MSL) consiste em fornecer uma visão geral e formativa, permitindo identificar, classificar e sintetizar evidências de estudados dentro de uma determinada área de pesquisa [Cabrejos et al. 2018]. Desta forma, este MSL apresenta o uso das diretrizes do Pensamento Computacional e da Teoria da Carga Cognitiva no processo de ensino e aprendizagem de Programação nos cursos de Computação. Para isso, foi desenvolvido um protocolo metodológico baseado em [Petersen et al. 2008] e [Kitchenham et al. 2015], visando garantir rigor e reprodutibilidade ao processo.

5.1. Questões de Pesquisa

A definição da Questão de Pesquisa (QP) e das Questões Secundárias permite guiar a condução do estudo, definindo o que deve ser considerado quanto ao tópico abordado [Cabrejos et al. 2018]. Para este trabalho, foram formuladas às questão de pesquisa definidas no Quadro 1.

ID	Questões de Pesquisa
QP	Quais são as abordagens utilizadas no Processo de Ensino e Aprendizagem da Programação que fazem uso da Teoria da Carga Cognitiva e dos Pilares do Pensamento Computacional?
QS.1	Quais são as atividades apoiadas ou desenvolvidas nas Instituições de Ensino Superior (ensino, pesquisa, extensão) relacionadas ao ensino de Programação usando a Teoria da Carga Cognitiva e os Fundamentos do Pensamento Computacional?
QS.2	Qual foi o público alvo?
QS.3	Quais foram os itens de aprendizagem envolvidos no processo de ensino?
QS.4	Quais foram as técnicas, ferramentas e/ou tecnologias (sistemas operacionais, bibliotecas, linguagens) utilizadas?

Quadro 1. Definição das Questões de Pesquisa

5.2. Estratégia de Busca

As questões apresentadas foram organizadas conforme a estrutura PICOC (*Population, Intervention, Context, Outcomes, Comparison*), recomendada por [Kitchenham 2007]. No entanto, apenas os itens População, Intervenção, Contexto e Resultados foram considerados relevantes para este estudo. O item Comparação não foi avaliado, já que não há interesse em comparar os resultados, apenas identifica-los [Cabrejos et al. 2018]. Para a pesquisa e extração de dados dos estudos primários, as bases pesquisadas foram: *IEEE, ACM Digital Library, Compendex, Science Direct, Web of Science , Scopus* e *Springer*. A *string* de busca segue o padrão estabelecido no Quadro 2.

***String* de Busca**
(("*Students*" OR "*Professor*") AND ("*Technique*" OR "*Method*" OR "*Methodology*" OR "*Approach*") AND ("*Computational Thinking*" AND "*Cognitive Load*" AND "*Programming*") AND ("*Teaching*" OR "*Learning*"))

Quadro 2. *String* de busca utilizada nas bases de dados.

Para esta pesquisa, foi delimitado o período compreendido entre os meses de janeiro de 2017 a fevereiro de 2026. Para isso, foi utilizado como marco temporal, a Resolução CNE/CES nº 5, de 16 de novembro de 2016, que institui as Diretrizes Curriculares Nacionais para os Cursos de Graduação na área da Computação [MEC 2016b].

5.3. Processo de Seleção dos Estudos

A seleção dos estudos foi realizada em duas etapas distintas: Na etapa inicial, os trabalhos retornados foram avaliados através da análise dos títulos e *abstract*, utilizando para isso os Critérios de Inclusão (CI), apresentados no Quadro 3, e os Critérios Exclusão (CE), definidos no Quadro 4. Esses critérios são utilizados para filtrar os resultados, analisar os estudos encontrados e, assim criar, uma lista de possíveis artigos primários que respondam às questões de pesquisa [Cabrejos et al. 2018]. Foram excluídos editoriais e outros resultados textuais encontrados nas bases (CE.3), os quais, de alguma forma, mencionavam as palavras chave em seu texto, mas não são artigos.

Critérios de Inclusão (CI)
CI.1. Estudos que apresentem abordagens primárias, no contexto das metodologias e ferramentas utilizadas no Processo de Ensino-aprendizagem da Programação, que fazem uso da Teoria da Carga Cognitiva e dos fundamentos do Pensamento Computacional.
CI.2. Experiências de Instituições de Ensino superior.

Quadro 3. Critérios de Inclusão

Na segunda etapa, os estudos avaliados como relevantes na etapa anterior, foram analisados e lidos de na sua integralidade. Essa etapa, tem como objetivo responder às Questões de Pesquisa definidas no Quadro 1.

Critérios de Exclusão (CE)
CE.1. Estudos que não estejam disponíveis livremente para consulta ou *download*, em versão completa, através das fontes de pesquisa.
CE.2. Estudos que claramente não atendam as questões de pesquisa.
CE.3. Estudos que não sejam artigos completos.
CE.4. Trabalhos que estejam fora do período estabelecido.
CE.5 Estudos que não tiverem sido desenvolvidos em Instituições de Ensino Superior.
CE.6. Estudos repetidos (terão apenas sua primeira ocorrência considerada).
CE.7. Estudos Secundários.

Quadro 4. Critérios de Exclusão

As pesquisas nas bases utilizadas retornaram **905** resultados. Estes trabalhos retornados por base, assim como a quantidade de estudos incluídos e excluídos, após passarem pelos Critérios de Inclusão e Exclusão, podem ser visualizados na Tabela 1.

BASES	**ESTUDOS RETORNADOS**	**CRITÉRIOS DE INCLUSÃO/EXCLUSÃO**									**ESTUDOS SELECIONADOS**
		CE-1	**CE-2**	**CE-3**	**CE-4**	**CE-5**	**CE-6**	**CE-7**	**CI-1**	**CI-2**	
ACM	121	2	52	31	-	24	-	10	1	1	2
COMPENDEX	8	-	5	1	-	1	1	-	-	-	
IEEE	4	-	3	-	-	-	-	-	1	-	1
SCIENCE DIRECT	160	-	88	2	-	33	-	36	1	-	1
SCOPUS	237	3	150	-	-	55	1	26	1	-	1
SPRINGER	373	1	201	78	-	46	2	44	2	-	2
WEB OF SCIENCE	2	-	-	-	-	2	-	-	-	-	

TOTAL	**905**	**SELECIONADOS**	**7**

Tabela 1. Análise das Bases

5.3.1. Extração e Síntese

As listas contendo os resultados primários foram importadas e unificadas utilizando a plataforma *Parsifal*[2]. Para cada estudo primário, foram coletadas informações relativas à identificação do artigo, abordagens utilizadas, público-alvo, conteúdos de aprendizagens abordados e ferramentas e tecnologias envolvidas no processo de ensino da Programação que utilizem os pilares do Pensamento Computacional com a a Teoria da Carga Cognitiva.

[2]https://parsif.al/

6. Resultados.

Os resultados são apresentados e organizados em duas partes: Em primeiro lugar, são apresentados os estudos selecionados após o processo de busca, seleção e extração. Os trabalhos podem ser visualizados no Quadro 5, onde são listados os sete estudos selecionados. Cada Estudo Primário (EP) está organizado de forma que ser identificado. Na segunda parte, foi realizada a análise e às questões de pesquisa foram respondidas.

ID	Título do Artigo
EP.1	An Analysis of the Effects of Learner-Centered Software Education and Required Support Strategies [Ahn and Oh 2024] Scopus.
EP.2	Differentiated Measurement of Cognitive Loads in Computer Programming [Quintero-Manes and Vieira 2024]. Springer.
EP.3	DBox: Scaffolding Algorithmic Programming Learning through Learner-LLM Co-Decomposition [Ma et al. 2025] ACM.
EP.4	Rethinking Computer Science Education in the Age of GenAI [Hazzan and Erez 2025]. ACM.
EP.5	Transforming programming education: a comparative study of traditional and techenhanced online learning for undergraduates and graduates [Kouam 2025]. Springer.
EP.6	A Study of the Effects of Differentiated Hints in Online Pair Programming on College Students' Computational Thinking [Zhao et al. 2025] IEEE
EP.7	SageJavon: A scalable AI tutor for personalized programming learning [Zhao et al. 2026] Science Direct

Quadro 5. Artigos Selecionados

6.1. Análise dos Resultados

Esta etapa do trabalho responde a Questão Primária (QP) e as Questões Secundárias (QS), definidas no protocolo deste Mapeamento Sistemático da Literatura, utilizando para isso os estudos selecionados no Quadro 5.

[QP] Quais são as abordagens utilizadas no Processo de Ensino e Aprendizagem de Programação que fazem uso da Teoria da Carga Cognitiva e dos Pilares do Pensamento Computacional?

Em **[EP.1]**, os autores ofereceram uma disciplina chamada Pensamento Computacional (PC), utilizando um modelo de resolução criativa de problemas, baseado em PC (CT-CPS) e *Design Thinking* (DT), que é um outro método de resolução de problemas pelo qual os alunos identificam e resolvem problemas da vida real. As cargas cognitivas foram analisadas antes e depois da aplicação da disciplina com um diário de bordo de cada aluno e, também, foi realizada uma análise para determinar o caminho entre o aprendizado de PC pelo aluno antes e após a disciplina e também a Carga Cognitiva de Programação. Em **[EP.2]** os autores traduziram e aprimoraram um questionário, adaptado para a língua espanhola, capaz de avaliar os três tipos de cargas cognitivas e sua relação com a autoeficácia, o autoconceito e o interesse em Programação dos alunos de um curso introdutório em Computação. Este questionário foi aplicado após a conclusão dos testes, avaliando a carga cognitiva com base nas percepções dos alunos em relação às atividades.

Já **[EP.3]** trabalhou a interação dos alunos com uma ferramenta criada pelos autores, que é uma sistema interativo Web baseado em LLM (*Large Language Model*), capaz de reconhecer padrões, gerando textos, traduzindo, resumindo e realizando outras tarefas semelhantes. Os autores notaram melhoria no Pensamento Algorítmico, e mediram as cargas cognitivas utilizando questionários de avaliação. **[EP.4]** trabalhou a Inteligência Artificial Generativa (GenIA), fazendo uso do ChatGPT como ferramenta de auxílio a

aprendizagem de Programação, utilizando-se de dez teorias diferentes listadas no estudo, entre elas a TCC e o PC, onde a GenIA funciona como agente de redução na Carga Cognitiva e oferece *feedback* para personalização do ensino de Programação para cada aluno em separado. As cargas cognitivas foram avaliadas utilizando um questionário reflexivo, aplicando a Taxonomia de Bloom.

O estudo **[EP. 5]** tratou da utilização de Realidade Virtual (RV) e sistemas de tutoria baseados em Inteligência Artificial (IA) e seus impactos na motivação, no envolvimento e na carga cognitiva dos alunos. Foram aplicados questionários quantitativos e qualitativos para medir o aprendizado e as cargas cognitivas. O estudo **[EP.6]** demonstrou o uso de dicas de código com preenchimento de lacunas e dicas de pseudocódigo, provenientes da TCC para aprimorar o desenvolvimento do Pensamento Computacional dos alunos. Foram coletadas transcrições de discursos e códigos gerados pelos alunos para avaliar a eficácia do aprendizado. Por fim, em **[EP.7]** os autores propuseram uma estrutura de ensino: Aprender-Praticar-Avaliar-Suportar (LPES), utilizando IA como tutor, com o objetivo de reduzir a carga cognitiva e desenvolver habilidades do Pensamento Computacional, foram aplicados questionários padrão NASA-TLX [3], que mede a carga de trabalho, e também, entrevistas com os alunos para avaliar a aceitação do método e o esforço durante o aprendizado.

[QS.1]: Quais são as atividades apoiadas ou desenvolvidas nas Instituições de Ensino Superior (ensino, pesquisa, extensão) relacionadas ao ensino de Programação usando a Teoria da Carga Cognitiva e os Fundamentos do Pensamento Computacional?

O trabalho **[EP.1]** trata de um curso de Educação em Software, atividade de ensino. Neste estudo, são aplicados os modelos CT-CPS e DT (*Design Thinking, Computational Thinking* e resolução criativa de problemas baseado em Teoria Cognitiva). Esses modelos são integrados para elaborar um plano de ensino e aplicá-los a um curso básico de software denominado "Pensamento Computacional". Já **[EP.2]** contribuiu para os processos educacionais desenvolvidos por professores e pesquisadores em Educação utilizando questionários para medir cargas cognitivas diferenciadas em sua versão em espanhol e avaliar os três tipos de cargas cognitivas e sua relação com a autoeficácia, o autoconceito e o interesse em Programação com alunos de um curso Introdutório em Programação.

O estudo **[EP.3]** trabalhou o processo de ensino e aprendizagem com o desenvolvimento de uma ferramenta Web: DBox (*Decomposition Box*), com foco em suporte ao aprendizado de programação algorítmica, percepções dos alunos e a experiência do usuário. Em **[EP.4]** é utilizada a Inteligência Artificial Generativa (GenAI) como apoio ao ensino de Programação, e seus efeitos no aprendizado dos alunos. O estudo **[EP.5]** tratou da integração de tecnologias digitais emergentes, como Inteligência Artificial e Realidade Virtual, e como estas tecnologias podem afetar o ensino de Programação, comparando o ensino tradicional em sala de aula e ambientes *on-line* que incorporam estas tecnologias emergentes. No **[EP.6]** são usadas as dicas de códigos e pseudocódigos, oriundos da TCC, como auxiliares no ensino de programação. Já em **[EP.7]**, os autores criaram o SageJavon, utilizando conceitos da TCC, apontado como ferramenta para otimização do ensino de Programação.

[3] https://humansystems.arc.nasa.gov/groups/tlx/

[QS.2]: Qual foi o público alvo?

Em **[EP.1]** foram 190 estudantes da Universidade Konkuk, em Seul, capital da Coréia do Sul. No segundo semestre de 2023. Em **[EP.2]** foram 1157 alunos da Universidade del Norte, Barranquilla, Colômbia. Já em **[EP.3]**, foram dois momentos de pesquisa: no primeiro, foram 15 estudantes universitários, sendo 10 de graduação e 5 de pós-graduação. No segundo momento, foram 24 participantes selecionados por *e-mail* e redes sociais em universidades locais. Em **[EP.4]** o estudo teve a participação de 1.019 calouros do Instituto de Tecnologia de Israel, durante o semestre de inverno de 2023-2024. O estudo **[EP.5]** teve a participação de 120 estudantes de Engenharia de uma universidade do norte da França, não citando o nome no texto. Em **[EP.6]** foram 42 alunos do segundo ano de Ciência da Computação, da Universidade de Guizhou, na China. E no estudo **[EP.7]**, foram 85 alunos programação Orientada a Objetos em Java para alunos do segundo ano da Universidade Normal do Leste da China, em Xangai entre 2024 e 2025.

[QS.3]: Quais foram os itens de aprendizagem envolvidos no processo de ensino?

O estudo **[EP.1]** trabalhou itens como: Pensamento Computacional, coleta, análise e representação de dados, definição do problema, decomposição do problema, abstração, algoritmo, operadores, *strings* e listas, expressões condicionais e instruções de seleção, declarações de repetição e funções, aplicando-se o conteúdo e utilizando a linguagem de programação *Pyhton*. Já em **[EP.2]**, a equipe de pesquisa utilizou dois instrumentos de avaliação para este estudo. O primeiro, focado na mensuração da carga cognitiva, foi traduzido e adaptado do Questionário de Avaliação de Ingenuidade [Klepsch et al. 2017], e o segundo mediu a autoeficácia dos alunos, o interesse e o autoconceito em programação, adaptado de [Magana et al. 2016].

O estudo **[EP.3]** mediu os efeitos nos resultados da aprendizagem: o ganho de aprendizagem percebido, a confiança na resolução de problemas semelhantes, a melhoria no pensamento algorítmico e a autoeficácia. Os efeitos nas percepções e experiência do usuário: engajamento cognitivo, o pensamento crítico, o senso de realização, o sentimento de trapaça, a adequação da ajuda percebida, a utilidade, a demanda mental, o esforço, a frustração, a facilidade de uso, a satisfação e a intenção de uso futuro, os padrões de uso da ferramenta DBox e foram aplicadas entrevistas para investigar os motivos subjacentes às percepções, padrões de uso e reações dos participantes a erros de IA. Em **[EP.4]** as atividades exigiram conhecimento prévio de linguagem C e conceitos básicos de programação (por exemplo, variáveis globais, recursão e *strings*).

No estudo **[EP.5]** , trabalhou-se assuntos como algoritmos, Programação Orientada a Objetos (POO), metodologia de desenvolvimento de software e tópicos como Programação em Realidade Virtual, Inteligência Artificial em Programação e ferramentas de colaboração *on-line*. Em **[EP.6]** , foram utilizados Conceitos Computacionais: Sequências, Laços, Condições, Operadores e Dados; Prática Computacional: Identificação do problema, Análise, Projeto, Codificação, Teste e Avaliação; e Perspectiva Computacional, de uma forma mais subjetiva, com reflexões e levantamento de questões sobre possibilidades e limitações da tecnologia educacional. Em **[EP.7]** foram tratados os conceitos fundamentais de Programação Java, incluindo configuração do ambiente Java, estruturas básicas de classes e Programação Orientada a Objetos: princípios, herança,

interfaces e classes abstratas.

QS 4 - Quais foram as técnicas, ferramentas e/ou tecnologias (sistemas operacionais, bibliotecas, linguagens) utilizadas?

Em **[EP.1]** os autores aplicaram uma modalidade de ensino denominada D-CT-CPS, baseada nos modelos de Pensamento de *Design* e Resolução Criativa de Problemas baseada em Pensamento Computacional, utilizando a linguagem de programação Python. Já o trabalho **[EP.2]**, por ser um estudo subjetivo e conceitual que analisa as práticas educacionais, não apresentou informações sobre os ambientes utilizados para o ensino da Programação. Em **[EP.3]**, foi utilizado o DBox (*Decomposition Box*), uma ferramenta de Inteligência Artificial desenvolvida para auxiliar os alunos no aprendizado de Programação, através de uma interface WEB. Segundo os autores, a ferramenta fornece *feedback* passo a passo e orientação em camadas sem comprometer a autonomia do aluno. Em **[EP.4]** foi utilizado o ChatGPT, e as linguagens de programação Pyhton e C, onde os alunos realizavam tarefas de conversão de código de Pyhton, que é uma linguagem que eles pouco conheciam, para C, que é a linguagem que eles mais estudam no curso e, por isso, estavam mais familiarizados.

Já em **[EP.5]** , nas aulas presenciais, foram usados *Codecademy* e *Blockly* como ferramentas de software instrucionais que proporcionaram aos alunos uma experiência de programação imersiva e interativa. Os ambientes *GitHub* e *Scratch* foram utilizados para ampliar projetos de programação colaborativa e sessões de Programação entre pares. Nas aulas *on-line* foram usados, *Google Cardboard* e o *Unity3D* para experiências de programação imersivas, soluções de tutoria baseadas em IA, como o *Knewton*, juntamente com recursos educacionais, como *OpenClassrooms* e vídeos do *YouTube*, e fóruns de discussão Zoom e Piazza. Em **[EP.6]**, o estudo menciona formação prévia de 14 semanas em linguagem C, antes do experimento, sem maiores informações de plataformas ou ferramentas utilizadas. Já no estudo **[EP.7]**, é apresentado o *SageJavon*, um tutor de IA projetado pelos autores para o ensino de Programação. Construído sobre a estrutura LPES, também desenvolvida pelos autores, integra Modelos de Aprendizagem Baseados em Aprendizagem (LLMs), rastreamento de conhecimento e estruturas avançadas de avaliação de IA para proporcionar uma experiência de aprendizagem mais interativa, adaptativa e personalizada.

7. Discussão dos Resultados

Neste trabalho, foram analisados estudos que aplicaram conceitos da Teoria da Carga Cognitiva associados às diretrizes do Pensamento Computacional objetivando o aprendizado da Programação, e visando reduzir o esforço dos alunos. Foram identificados métodos como *Design Thinking, Computational Thinking* e otimização de *Design* Instrucional utilizando efeitos da TCC, principalmente o Efeito do Desvanecimento da Orientação. Esse efeito é quando o aluno inicia o aprendizado em um assunto com muita orientação, por parte do instrutor e/ou da plataforma, e conforme vai adquirindo expertise, essa orientação vai sendo diminuída/retirada. Essa orientação pode ser realizada a partir da adoção de ferramentas de IA, diretamente como assistente pessoal de aprendizagem, ou em interfaces desenvolvidas pelos próprios autores. Todos os trabalhos fizeram uso de questionários e entrevistas, medindo quantitativamente e qualitativamente o esforço e o aprendizado dos alunos, fechando com isso, o ciclo ensino-aprendizagem. Um es-

tudo aplicou questionários adaptados para a realidade da cultura de seu país, buscando resultados, mais adaptados à sua realidade, em sua análise. Notou-se que as avaliações aconteciam de forma contínua em todos os estudos, assim os autores puderam ir ajustando as ferramentas de ensino de forma que os alunos tivessem maior exito na aprendizagem.

Cinco estudos selecionados foram desenvolvidos no continente asiático, o que demonstra a vanguarda do continente em aplicar metodologias de ensino. Todos os estudos foram publicados entre 2024 e 2026, confirmando o atual interesse em inserir o aluno como protagonista no processo de aprendizagem. Apesar da pesquisa inicial trazer 905 resultados, ressalta-se que a grande maioria dos estudos retornados, citavam os termos de pesquisa apenas em suas bibliografias, ou apenas uma vez no texto, não tratando de fato do Pensamento Computacional e Carga Cognitiva. Além disso, muitos dos estudos retornados eram aplicações na Educação Infantil ou na Educação Básica, não se aplicando ao ensino da Programação na Educação Superior.

8. Ameaça à Validade da Pesquisa

Pode-se destacar, como possíveis ameaças a esse MSL, que o processo de avaliação não adotou a análise de dados quantitativos dos artigos, no qual seria utilizada uma pontuação para que cada estudo pudesse ser aceito. Isso não foi necessário, já que a finalidade deste MSL foi de apenas identificar trabalhos que utilizassem a Teoria da Carga Cognitiva e os Pilares do Pensamento Computacional no processo de ensino e aprendizagem da Programação. Dessa forma, os estudos foram aceitos apenas considerando os critérios de inclusão e exclusão. Visando proporcionar uma maior credibilidade à pesquisa, foi utilizada uma busca automática nas bases, para a comprovação da eficácia da *string* através de vários testes. Soma-se a isso, foi elaborado um protocolo onde foram definidos os objetivos, escopo, restrições, critérios de seleção (inclusão e exclusão) e avaliação de forma que se possa ter transparência e replicabilidade. Vale ressaltar que seis trabalhos foram excluídos por não estarem disponíveis para *download* nas bases pesquisadas, e não foi possível, também, consegui-los com os respectivos autores.

9. Conclusão e Trabalhos Futuros

O ensino de Programação sempre se mostrou um desafio devido o elevado grau de abstração e raciocínio lógico necessários à compreensão dos conceitos envolvidos para a solução dos problemas. Os resultados encontrados pelo Mapeamento são, não apenas animadores, mas também promissores para todas as áreas do conhecimento. Vários trabalhos relataram o uso de Inteligência Artificial Generativa (GenIA) em Sistemas Tutores como uma forma de auxiliar e dar mais autonomia ao aluno no processo de aprendizagem da Programação, oferecendo um *feedback* personalizado para cada aluno. Ferramentas como o Dbox, *Knewton* e *SageJavon* foram desenvolvidas para atuar como redutores da carga cognitiva dos alunos e potencializar o pensamento algorítmico a partir de uma experiência mais interativa e adaptativa. Outras tecnologias de Inteligência Artificial, como o ChatGPT e seus correlatos, também estão sendo alvo de estudo e aplicações de sucesso na área de Educação.

Como trabalhos futuros, pretende-se propor e analisar ferramentas de ensino, utilizando o resultado deste mapeamento, para aprimorar e adaptar às diretrizes nacionais, e à realidade do ensino local, visando melhorias no processo de ensino e aprendizagem da Programação Paralela.

Declaração de uso de Ferramentas de Inteligência Artificial

Os autores declaram que não foram utilizadas ferramentas de Inteligência Artificial (IA) no processo de elaboração ou revisão deste artigo, bem como em nenhuma etapa do Mapeamento da Literatura. Todo o conteúdo científico, análise de dados, interpretação dos resultados e a seleção bibliográfica foram realizados integralmente pelos autores.

Referências

Ahn, S. and Oh, K. (2024). An analysis of the effects of learner-centered software education and required support strategies. *Front. Educ.*, 9.

Almeida Jr., R. B. and Araújo, J. S. (2022). Uma avaliação do uso das placas gráficas no ensino da programação paralela nos cursos de computação no brasil. In *Anais do Workshop sobre Educação em Computação (WEI)*, volume 30, pages 357–368.

Berssanette, J. H. and Francisco, A. C. d. (2022). Avaliação de uma qualificação docente para o ensino de programação por meio do uso de metodologias ativas de aprendizagem e a teoria da carga cognitiva. In *Anais do XXX Workshop sobre Educação em Computação (WIE)*, pages 1–12.

BNCC (2022). *Computação Complemento à BNCC*. Acesso em: 1 fev. 2026.

Cabrejos, L. J. E. R., Viana, D., and Santos, R. P. (2018). Planejamento e execução de estudos secundários em informática na educação: Um guia prático baseado em experiências. In *Proceedings of the VII Congresso Brasileiro de Informática na Educação (CBIE) / VII Jornada de Atualização em Informática na Educação (JAIE)*, Fortaleza, CE.

Duran, R., Rurenko, A., and Sorva, J. (2022). Cognitive load theory in computing education research: A review. *ACM Transactions on Computing Education*.

Hazzan, O. and Erez, Y. (2025). Rethinking computer science education in the age of genai. *ACM Trans. Comput. Educ.*, 25(3).

Kirschner, P., Beers, P., Boshuizen, H., and Gijselaers, W. (2008). Coercing shared knowledge in collaborative learning environments. *Computers in Human Behavior*, 24:403–420.

Kitchenham, B. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE-2007-01, Department of Computer Science, Keele University.

Kitchenham, B., Brereton, P., Budgen, D., Turner, M., Bailey, J., and Linkman, S. (2015). Systematic literature reviews in software engineering – a tertiary study. *Information and Software Technology*, 52(8):792–805.

Klepsch, M., Schmitz, F., and Seufert, T. (2017). Development and validation of two instruments measuring intrinsic, extraneous, and germane cognitive load. *Frontiers in Psychology*, 8.

Kouam, A. W. F. (2025). Transforming programming education: a comparative study of traditional and tech-enhanced online learning for undergraduates and graduates. *Discover Education*, 4.

Ma, S., Wang, J., Zhang, Y., Ma, X., and Wang, A. Y. (2025). Dbox: Scaffolding algorithmic programming learning through learner-llm co-decomposition. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25, New York, NY, USA. Association for Computing Machinery.

Machado, K. K. and Dutra, A. (2023). Desenvolvimento do pensamento computacional. *Rev. diálogo educ.*, 23(77):945–956.

Magana, A. J., Falk, M. L., Vieira, C., and Reese, M. J. (2016). A case study of undergraduate engineering students' computational literacy and self-beliefs about computing in the context of authentic practices. *Computers in Human Behavior*, 61:427–442.

MEC (2016a). Resolução nº 5, de 16 de novembro de 2016. (2016). In 220, editor, *Diário Oficial da União (DOU)*, 1, page 22.

MEC, CNE, C. (2016b). Resoluções ces 2016. Disponível em: https://www.gov.br/mec/pt-br/cne/resolucoes/resolucoes-ces-2016. Acesso em: 17 de junho de 2025.

Navarro, E. R. N., Sousa, M. d. C. d., and Rolkouski, E. (2024). O movimento lógico-histórico do conceito de pensamento computacional. *Obutchénie. Revista de Didática e Psicologia Pedagógica*, 8(Contínua):1–21.

Petersen, K., FELDT, R., MUJTABA, S., and MATTSSON, M. (2008). Systematic mapping studies in software engineering. In *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering (EASE)*, pages 68–77, Italy. Bournemouth University.

Quintero-Manes, R. and Vieira, C. (2024). Differentiated measurement of cognitive loads in computer programming. *J. Comput. High. Educ.*

Rijo-García, S., Segredo, E., and León, C. (2022). Computational thinking and user interfaces: A systematic review. *IEEE Trans. on Educ.*, 65(4):647–656.

Sweller, J. (1994). Cognitive load theory, learning difficulty, and instructional design. *Learning and Instruction*, 4(4):295–312.

Wing, J. M. (2006). Computation thinking. In *Communications of the ACM*, volume 49, pages 33–35.

Zhan, Z., Zhou, X., Cai, S., and Lan, X. (2025). Exploring the effect of competing mechanism in an immersive learning game based on augmented reality. *J. Comput. Educ.*, 12(2):449–475.

Zhao, H., Wu, Y., Lu, Z., Yu, X., Miao, W., and Chen, L. (2026). Sagejavon: A scalable ai tutor for personalized programming learning. *Information Processing Management*, 63(5):104605.

Zhao, J., Lei, Y., Fu, X., and Yi, B. (2025). A study of the effects of differentiated hints in online pair programming on college students' computational thinking. In *2025 International Conference on Distance Education and Learning (ICDEL)*, pages 436–441.