



# **Análise dos Cursos de Computação no Brasil: Ensino de Programação sob a perspectiva das Necessidades do Século XXI**

**Katyeydo Karlos de S. Oliveira<sup>1</sup>, William S. de Deus<sup>1</sup>, Gustavo Martins N. Avellar<sup>1</sup>  
Taciana Pontual Falcão<sup>2</sup>, Ellen Francine Barbosa<sup>1</sup>**

<sup>1</sup>Universidade de São Paulo (USP) – São Carlos, SP, Brasil

<sup>2</sup>Universidade Federal Rural de Pernambuco (UFRPE) – Recife, PE, Brasil

{karlos\_oliveira, williamsimao, gustavo.avellar}@usp.br,  
taciana.pontual@ufrpe.br, francine@icmc.usp.br

**Abstract.** *Programming skills are essential in the contemporary technological landscape, besides fostering essential skills for the 21st century. This article examines the pedagogical approaches employed in introductory programming courses within Brazilian higher education and their alignment with the demands of the modern era. A case study was undertaken, focusing on Software Engineering and Computer Science programs, employing machine learning techniques to categorize the covered topics. The findings unveiled three distinct clusters of disciplines exhibiting varying degrees of programming concepts integration. These results provoke pertinent inquiries regarding the congruence of instructional practices with the requisites of programming education in the 21st century. By offering empirical evidence pertaining to programming education and the cultivation of 21st-century skills in Brazilian higher education, this article makes a contribution to the existing scholarly literature.*

**Resumo.** *As habilidades de programação são essenciais no mundo tecnológico atual, além de ajudar a desenvolver habilidades fundamentais para o século XXI. Este artigo analisa o ensino de disciplinas introdutórias de programação em cursos de ensino superior do Brasil e o seu alinhamento com as necessidades do século XXI. Um estudo de caso foi realizado em cursos de Engenharia de Software e Ciência da Computação. Técnicas de aprendizado de máquina foram utilizadas para agrupar os tópicos abordados. Os resultados revelaram três grupos de disciplinas com diferentes níveis de conceitos de programação. As descobertas levantam questões sobre o alinhamento das práticas de ensino com as demandas da educação em programação no século XXI. O artigo contribui para a literatura existente, fornecendo evidências empíricas sobre o ensino de programação e habilidades do século XXI no ensino superior brasileiro.*

## **1. Introdução**

A programação é considerada fundamental no mundo tecnológico atual [Avcı 2022]. Especialistas defendem sua inclusão no sistema educacional global em todos os níveis [Salleh *et al.* 2021; Sun *et al.* 2022]. Aprender a codificar vai além da criação de software [DePryck *et al.* 2016], é uma habilidade que também contribui para o desenvolvimento de habilidades de resolução de problemas, pensamento lógico [Kampylis *et al.* 2023], consideradas essenciais no século XXI [Oliveira e Souza 2022].

O relatório *Skills for the Digital Transition* da OECD [OECD 2022] mostra o crescimento das ocupações digitais em 10 países, com destaque para desenvolvedores de software, cientistas de dados e engenheiros de dados. A automação abrange 34% das tarefas globais, de acordo com o relatório *Future of Jobs* de 2023 do Fórum Econômico

Mundial [WEF 2023]. Ambos os relatórios enfatizam a importância das habilidades digitais e destacam a necessidade de capacitação para o futuro do trabalho.

O ensino de programação introdutória é crucial para preparar os alunos em carreiras STEM (*Science, Technology, Engineering and Mathematics*) [Sobral 2021] e enfrentar a crescente automação [WEF 2023]. No entanto, com as diretrizes técnicas mudando a visão contemporânea do ensino de programação [ACM-IEEE 2020; SBC 2017], é necessário repensar o modelo de ensino e aprendizagem para que os alunos possam desenvolver as habilidades necessárias para atuar como cidadãos críticos no século XXI [Silva *et al.* 2021].

No Brasil, todo o sistema educacional vem sendo atualizado, com a determinação de que todos os alunos sejam apresentados aos fundamentos do ensino de programação [Brasil 2022]. Seguindo essa tendência, a Sociedade Brasileira de Computação (SBC) elaborou referências técnicas para desenvolver e atualizar programas de graduação e pós-graduação no ensino superior [SBC 2017, Araujo *et al.* 2019].

Ainda, cabe destacar outro problema atual - o relatório *The Future of Education and Skills 2030* [OECD 2018] destaca a necessidade de combinar conhecimento e habilidades exigidas para o aprendizado do século XXI. A literatura atual sobre o ensino de programação carece de estudos práticos e evidências que descrevam o apoio ao desenvolvimento de habilidades de programação e habilidades do século XXI nos alunos [Wang *et al.* 2019]. Tais habilidades referem-se ao aperfeiçoamento do pensamento analítico, resolução de problemas e colaboração, entre outras, sendo tais habilidades diferentes das habilidades acadêmicas tradicionais, apoiadas por conhecimentos conceituais [Dede 2010, OECD 2018].

Considerando o contexto supracitado, é pertinente questionar se os cursos de computação do Brasil estão ensinando programação em consonância com as necessidades do século XXI. Assim, visando responder esta questão, foi realizado um estudo exploratório analisando 104 disciplinas de programação introdutória de cursos de graduação em Engenharia de Software e Ciência da Computação do Brasil. Para a coleta de dados, foi analisado o plano de ensino de cada disciplina e o Projeto Pedagógico do Curso (PPC). Nessa análise, adotou-se o aprendizado de máquina para realizar o agrupamento dos tópicos das disciplinas, e o conteúdo das disciplinas foi analisado em relação às necessidades do século XXI.

O restante deste artigo está organizado como se segue: A Seção 2 discute os trabalhos relacionados. A Seção 3 descreve o método de pesquisa. A Seção 4 resume os principais resultados do estudo. A Seção 5 apresenta propostas para solucionar os problemas identificados. Conclusões e perspectivas para trabalhos futuros são discutidas na Seção 6.

## **2. Trabalhos Relacionados**

De modo geral, são poucas as pesquisas sobre o ensino de programação no contexto das necessidades do século XXI, especialmente no contexto brasileiro. Wang *et al.* (2019) identificaram habilidades do século XXI nos planos de ensino de ciência K-12 na China, enquanto o presente estudo, por sua vez, aborda o ensino superior brasileiro, com foco na programação introdutória.

Vinnervik (2022) analisou os planos de ensino K-12 sueco e destacou a necessidade de formação adequada para os professores de programação. Embora nossas perspectivas geográficas e níveis educacionais sejam diferentes do estudo de Vinnervik (2022), ambos destacam a importância da programação no contexto educacional.

Enquanto Vinnervik (2022) investigou como os planos de ensino podem apoiar os professores no processo de ensino, este estudo fornece uma visão geral para professores e alunos sobre o ensino de programação para o século XXI.

Marques *et al.* (2016) analisaram os planos de ensino de Engenharia de Software em universidades chilenas, encontrando alta cobertura de tópicos, mas baixo número de cursos e abordagem predominantemente prática ou teórico-prática. O presente estudo investigou a programação introdutória e habilidades do século XXI em universidades brasileiras, utilizando diretrizes atualizadas e focando em cursos introdutórios de programação. Embora haja semelhanças, nosso estudo aborda diferentes aspectos do ensino de ciência da computação em contextos distintos.

Por fim, Silva *et al.* (2022) analisaram disciplinas de CS1 em universidades federais brasileiras, identificando os tópicos mais abordados na programação. O estudo utilizou uma lista de instituições do *Wikipedia* e enfatizou a reformulação das ementas e a granularidade dos conteúdos. Nosso estudo também investigou o ensino de programação em cursos de graduação, mas diferiu na metodologia de coleta de dados e nos tópicos analisados, focando em universidades selecionadas de acordo com um *ranking* de universidades *University Rankings* [Rankings 2023] e se concentrou em cursos de Engenharia de Software e Ciência da Computação.

### **3. Método de Pesquisa**

Para orientar a presente pesquisa, foi elaborada a seguinte Questão de Pesquisa (QP): os cursos de computação do Brasil estão ensinando programação de acordo com as necessidades do século XXI? Para responder a esta QP, foi realizado um estudo de caso exploratório seguindo as premissas propostas por Yin (2009). A seguir, são apresentados os detalhes do protocolo.

#### **3.1. Definição e Planejamento**

A QP proposta tem uma intenção exploratória de mapear o ensino e a aprendizagem de programação introdutória e habilidades do século XXI no ensino superior brasileiro. No entanto, pretende-se realizar uma investigação limitando a inserção de vieses originados por atores educacionais (ou seja, professores ou alunos).

Assim, foram identificados cursos de graduação que incluem disciplinas introdutórias de programação e habilidades do século XXI em seus planos de ensino. Para tanto, adotou-se a diretriz de Currículos de Computação fornecida por ACM-IEEE (2020) e SBC (2017), que trazem diretrizes técnicas para cursos de graduação em computação.

As diretrizes da ACM-IEEE (2020) e da SBC (2017) estabelecem critérios importantes para a análise de cursos de graduação em computação. Para o presente estudo, foram selecionados dois cursos de graduação: Engenharia de Software e Ciência da Computação. Esses cursos foram considerados importantes por estarem relacionados à interseção entre os fundamentos da computação e as necessidades das organizações. Isso quer dizer que tanto os cursos de Engenharia de Software quanto os de Ciência da Computação possuem como requisito para a formação dos alunos o domínio das habilidades de programação introdutória, uma vez que são disciplinas fundamentais nessas áreas de estudo. Além disso, esses cursos também destacam a importância do desenvolvimento de habilidades organizacionais que são relevantes para o século XXI, uma vez que estão diretamente relacionadas às necessidades das organizações.

Os demais cursos de computação, como Engenharia de Computação, Ciência de Dados, Análise e Desenvolvimento de Sistemas e Sistemas de Informação focam em aspectos mais específicos da programação (como programação para chips, uso de grande

base de dados, etc...) ou direcionam sua atenção para aprofundar as habilidades organizacionais. Por isso, não foram analisados.

A escolha das universidades para este estudo foi baseada no *Times Higher Education World University Rankings 2023* [Rankings 2023]. Este ranking possui parceria com a empresa editorial Elsevier e inclui 1.799 universidades em 104 países e regiões, e é considerado o maior e mais diversificado ranking universitário da atualidade [Rankings 2023]. No ranking, as universidades são avaliadas de acordo com o desempenho em quatro áreas: ensino, pesquisa, transferência de conhecimento e perspectiva internacional. Assim, foi aplicado um filtro pelo país, selecionando as melhores classificações para as universidades no Brasil. Em seguida, foram analisados se os cursos de Ciência da Computação ou Engenharia de Software estavam disponíveis em cada universidade selecionada. Com base nesses critérios, foram criados dois grupos distintos de universidades, cada um com cursos específicos para análise.

Para garantir uma abordagem viável, foram selecionadas 6 universidades em Engenharia de Software e 6 em Ciências da Computação, com base no número de pesquisadores envolvidos. Cada pesquisador analisou o plano de ensino das disciplinas e o PPC dos cursos, permitindo uma análise detalhada e uma comparação justa entre as instituições selecionadas. A escolha cuidadosa dessas universidades foi feita para garantir dados consistentes e significativos sobre as disciplinas de programação introdutória. A lista das instituições selecionadas é apresentada na Tabela 1.

**Tabela 1. Cursos e instituições investigados**

Curso	Instituição	Abreviação
Ciências da Computação	Universidade de São Paulo – Campus São Paulo, Instituto de Matemática e Estatística	USP-IME
	Universidade de São Paulo – Campus São Carlos, Instituto de Ciências Matemáticas e Computação	USP-ICMC
	Universidade Estadual de Campinas	UNICAMP
	Universidade Federal do Rio Grande do Sul	UFRGS
	Universidade Federal de São Paulo	UNIFESP
	Universidade Federal de Minas Gerais	UFMG
	Universidade Federal de Sergipe	UFS
Engenharia de Software	Pontifícia Universidade Católica do Rio Grande do Sul	PUCRS
	Universidade de Brasília	UNB
	Universidade Federal de Goiás	UFG
	Universidade Federal de Lavras	UFLA
	Universidade Federal do Rio Grande do Norte	UFRN
	Pontifícia Universidade Católica do Paraná	PUCPR

### 3.2. Coleta de Evidências e Dados

Neste estudo, a coleta de dados foi realizada em duas fontes distintas: o plano de ensino da disciplina e o PPC. No primeiro caso, o plano de ensino foi analisado pelos pesquisadores, identificando os tópicos de cada disciplina, os objetivos de aprendizagem, os pré-requisitos bem como a dependência da disciplina para prosseguir no curso. Ou seja, se o aluno repetir na disciplina, não poderá cursar outra disciplina.

Já o PPC conta com dados mais subjetivos sobre o perfil de formação dos estudantes. O PPC foi usado como estratégia complementar, tendo em vista que algumas instituições não disponibilizam o plano de ensino para acesso e verificação. Para coletar as informações dos planos de ensino e PPC, utilizou-se um formulário digital composto por três seções distintas. A primeira seção capturou informações básicas sobre o curso, enquanto a segunda seção abordou as competências de programação introdutória ensinadas. Na terceira seção, foram coletadas informações sobre as habilidades do século

XXI. Foram realizadas comparações entre essas habilidades e os planos de ensino das instituições selecionadas, analisando planos de ensino e PPC. O formulário digital continha questões abertas, numéricas e fechadas para obter informações padronizadas. O formulário pode ser acessado em: <https://doi.org/10.5281/zenodo.8298060>.

### 3.3. Análise e Interpretação

Para analisar os dados coletados, foram adotadas duas estratégias. A primeira delas foi o uso de Aprendizado de Máquina para realizar uma análise exploratória dos dados. Para isso, os dados coletados foram submetidos à análise do algoritmo PCA [Weingessel e Hornik 2000]. De forma resumida, esse algoritmo gerou uma clusterização dos dados, a qual permitiu a identificação de padrões de acordo com a terminologia adotada em cada disciplina.

Em paralelo, os pesquisadores realizaram reuniões virtuais para interpretar os resultados obtidos. Esse processo foi iterativo, realizado na presença de três pesquisadores, que conduziram a coleta de dados. As conclusões foram debatidas e esboçadas no presente trabalho de acordo com a literatura atual, como é exposto na Seção 4.

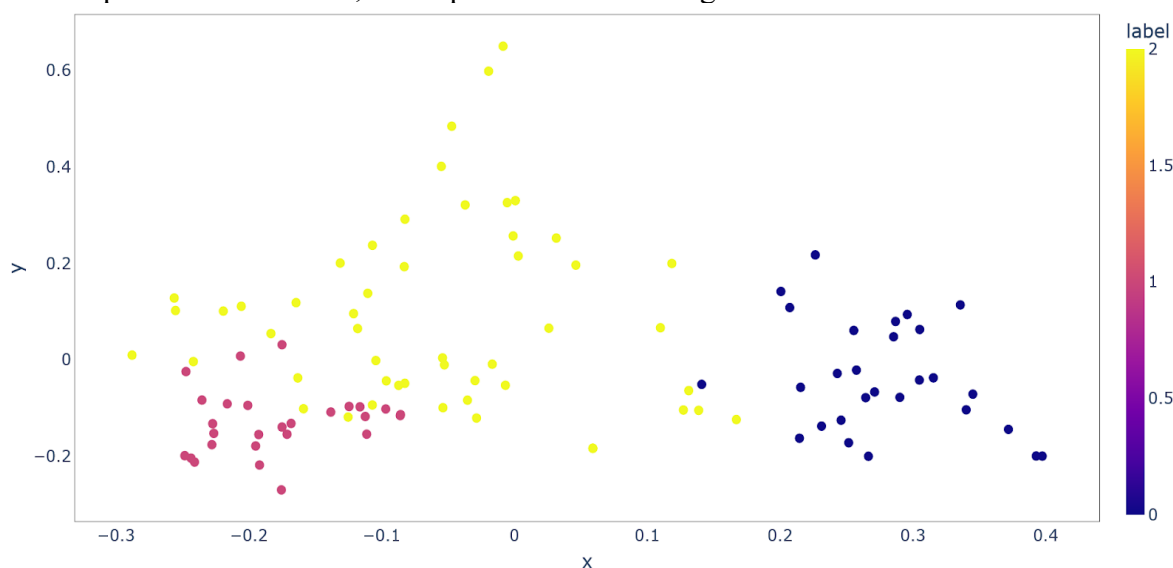
### 3.4. Caso Piloto

Para avaliar a qualidade do nosso processo de pesquisa, foi realizado um estudo de caso piloto com duas instituições. Os pesquisadores utilizaram o formulário digital para extrair informações dos cursos e discutiram os resultados obtidos. Após análise e discussão, conclui-se que o formulário era eficiente e poderia ser adotado em outras instituições de ensino superior.

## 4. Resultados

### 4.1. Análise PCA

Inicialmente, foi utilizada a técnica PCA para analisar a estrutura interna dos tópicos ensinados por cada disciplina. Assim, os tópicos semelhantes foram agrupados de acordo com os padrões observados, como pode ser visto na Figura 1.



**Figura 1 – Tópicos semelhantes agrupados de acordo com os padrões: Amarelo, disciplinas com tópicos mais elementares de programação; Azul, tópicos mais avançados; Lilás, disciplinas com foco em programação orientada a objetos.**

Nota-se a existência de três grupos ou *clusters* de disciplinas. A primeira, que recebeu coloração amarela, refere-se às disciplinas que apresentam tópicos fundamentais

de programação. Aborda, por exemplo, tópicos como “línguagens”, “problemas” e “código”, entre outros. Como pode ser observado, essas disciplinas estão espalhadas na parte central do gráfico. Elas apresentam conceitos introdutórios de programação e podem ser consideradas como disciplinas para os alunos que estão matriculados no primeiro semestre de um curso de computação.

Já o segundo grupo, de coloração azul, está concentrado na parte inferior direita do gráfico. Este grupo representa disciplinas que exploram conceitos mais avançados de técnicas de programação, como por exemplo, “árvores”, “ordenação”, “filas”, “busca”, etc. O foco é desenvolver habilidades mais sofisticadas de programação, como técnicas de algoritmos recursivos, busca e ordenação, estruturas de dados, etc. Geralmente, são disciplinas introduzidas a partir do segundo semestre e que possuem algum pré-requisito. Neste caso, para cursar a disciplina, o estudante deve ter sido aprovado em uma ou mais disciplinas de programação.

Por fim, o terceiro grupo, que recebeu coloração lilás, está concentrado na parte inferior esquerda do gráfico. Este grupo compreende disciplinas voltadas ao ensino e aprendizado de conceitos introdutórios de orientação a objetos, como “classes”, “software”, “objetos”, “métodos”, etc. Aqui, o aluno está aprendendo o paradigma de orientação a objetos, algumas vezes, com foco no desenvolvimento *web* e por meio de projetos acadêmicos. Novamente, para cursar uma disciplina dessa natureza, é essencial que o estudante já tenha tido aprovação nos dois grupos anteriores de disciplinas. Na maioria dos casos, os estudantes são apresentados a esses conceitos enquanto se aproximam da metade do curso.

Para ilustrar esse comportamento, a Figura 2 mostra os *clusters* de acordo com a frequência dos termos mais comuns. O que se pode inferir é que a programação introdutória perpassa, ao menos, 3 tipos de disciplinas. Na primeira são explorados conceitos de programação, introduzindo os estudantes aos recursos mais básicos. Logo em seguida, os alunos são apresentados a conceitos mais sofisticados. Neste caso, eles precisam unificar o ensino introdutório anterior para criar estruturas mais complexas. Por exemplo, conceber a ideia de uma estrutura simples de dados e depois evoluir para estruturas complexas, envolvendo busca e ordenação. Por fim, após concluir essa etapa, o estudante é apresentado ao paradigma de orientação a objetos. Sendo nessa parte o ensino introdutório de recursos bem mais sofisticados, como abstração, herança, polimorfismo, entre outros.



Figura 2 – Clusters com a frequência dos termos mais comuns encontrados

O que emerge dessa constatação é uma dúvida: será que o atual processo de ensino está dialogando, intrinsecamente, com o que é esperado para o ensino e aprendizado de programação no século XXI?

#### **4.2. Habilidades para o Século XXI**

Como observado anteriormente, as instituições de ensino superior brasileiras enfrentam um grande desafio quando se trata de capacitar os alunos com as habilidades de programação necessárias para o século XXI. Infelizmente, pelo menos nas instituições analisadas, há falhas em se fornecer um ensino adequado que prepare os alunos para as demandas da indústria de tecnologia atual. O diálogo da educação é inspirado em uma estrutura rígida de conceitos e aprovação/reprovação com base na dependência dos assuntos. Não há, essencialmente, um diálogo com as necessidades do século XXI.

Um ensino ideal de programação no século XXI deve ir além do ensino de conceitos básicos de linguagens de programação [Negoro *et al.* 2023]. Os programadores do século XXI precisam dominar não apenas uma linguagem específica, mas também ter um conjunto abrangente de habilidades que os capacitem a lidar com os desafios tecnológicos em constante evolução [De Vega 2022].

Um programador do século XXI deve ser versátil e ter conhecimentos em diversas áreas [Vogel *et al.* 2020]. Além de ter um domínio sólido de linguagens de programação populares, como *Python*, *Java* e *JavaScript*, é essencial ter habilidades relevantes para as necessidades tecnológicas atuais [Eteng *et al.* 2022]. No entanto, mais do que apenas dominar uma determinada linguagem ou tecnologia, é crucial ter uma mentalidade de aprendizado contínuo e estar disposto a aprimorar e requalificar constantemente [Wang *et al.* 2023]. No contexto do rápido avanço tecnológico, é importante reconhecer que as habilidades adquiridas hoje podem se tornar obsoletas em um curto espaço de tempo [Alam 2021]. Portanto, os programadores do século XXI precisam estar preparados para desaprender e reaprender, atualizando constantemente seus conhecimentos e adaptando-se às mudanças do setor [Ayinde e Kirkwood 2020].

Além das habilidades técnicas específicas, os programadores do século XXI também devem desenvolver habilidades transversais, como resolução de problemas complexos, pensamento crítico, colaboração efetiva em equipes multidisciplinares e comunicação clara [WEF 2023]. A capacidade de entender as necessidades do cliente, trabalhar em equipe e comunicar-se de forma eficaz é fundamental para o sucesso na indústria de tecnologia [OECD 2022]. Os programadores devem estar preparados para enfrentar os desafios tecnológicos em constante evolução, adotando uma abordagem holística que combine competência técnica com habilidades de comunicação, colaboração, ética e segurança [Kelly *et al.* 2023].

É preocupante constatar que até mesmo as melhores instituições de ensino superior brasileiras analisadas neste estudo não têm conseguido cumprir esse objetivo. Se as principais instituições, que geralmente têm mais recursos e reputação, não estão fornecendo um ensino adequado em programação para o século XXI, é difícil imaginar como as outras instituições estão enfrentando esse desafio.

A falta de capacitação adequada em programação nas instituições de ensino superior brasileiras pode comprometer a formação dos futuros profissionais de tecnologia. Isso coloca o país em desvantagem em um mercado global cada vez mais competitivo e inovador. Além disso, dificulta o desenvolvimento do setor de tecnologia no Brasil, limitando sua capacidade de acompanhar as tendências e contribuir de forma significativa para a inovação e o crescimento econômico.

Portanto, urge a necessidade de um esforço significativo para reformular os planos e métodos de ensino das disciplinas de programação em todas as instituições de ensino superior. Tal empreendimento exige investimento em capacitação docente, atualização de conteúdos e estabelecimento de uma integração mais estreita com a indústria de tecnologia. Estratégias e ferramentas que se propõem a orientar as instituições e os professores no ensino introdutório de programação no ensino superior são imprescindíveis, com o objetivo de desenvolver as habilidades fundamentais nessa área e promover as competências necessárias para o século XXI dos estudantes de disciplinas de programação nos cursos superiores de Computação.

Além disso, é imperativo que a capacitação dos professores seja uma prioridade. Eles devem manter-se atualizados em relação às tecnologias e metodologias mais recentes, o que requer investimentos em programas de desenvolvimento profissional, bem como a implementação de incentivos para aprimorar suas habilidades. Além disso, pesquisas que busquem orientar as instituições de ensino superior e os educadores no planejamento de disciplinas de programação introdutória são necessárias, a fim de proporcionar não apenas o aprendizado de programação, mas também o desenvolvimento das habilidades do século XXI.

#### **4.2.1 Competências para o Século XXI**

Em 2017, a SBC publicou as Referenciais de Formação para Cursos de Graduação em Computação [SBC 2017]. Esse referencial estabeleceu as competências que os formados em cursos de computação devem possuir. Ao mesmo tempo, demonstrou a necessidade de migrar o modelo de ensino e aprendizado de computação para acompanhar os desafios do século XXI.

Algumas competências dialogam diretamente com esse cenário. Por exemplo, a competência “Reconhecer a importância do pensamento computacional no cotidiano e sua aplicação em circunstâncias apropriadas e em domínios diversos”, alinhando o pensamento computacional a domínios diversos de aplicação. E, esse é o ponto chave para o século XXI: não basta apenas “ensinar” pensamento computacional aos estudantes de programação, mas ir além, explorando novas formas e estratégias de ensino com a atualidade.

Infelizmente, esse diálogo não é simples de ser feito. E, com base nos dados coletados, nota-se ainda que existe uma diferença entre a formação esperada e o que vem sendo praticado. Os desafios são grandes para superar esse cenário, e por isso mesmo é necessário focar em ideias e práticas que possam auxiliar nesse processo.

### **5. Propostas para solucionar os problemas identificados**

Tendo em vista os problemas levantados neste estudo, propõe-se a seguinte agenda para reduzir os desafios identificados.

1. ***Desenvolver pesquisas sobre como tornar o ensino e aprendizado de programação coerente com as habilidades e competências do século XXI:*** Como dito anteriormente, esse diálogo não é simples de ser feito, mas os pesquisadores podem ajudar os professores a resolverem esse problema. A construção de guias, frameworks, estratégias de ensino, entre outras práticas educacionais podem apoiar os professores de programação;
2. ***Divulgar boas iniciativas que ocorrem em disciplinas e/ou instituições:*** Até o melhor do nosso conhecimento, não há uma iniciativa estruturada que apresente ideias ou projetos de programação que alcançaram bons resultados. Nesse sentido, a criação de repositórios, sites ou de alguma estratégia de disseminação de ideias



(para um país de dimensão continental) pode ser uma ajuda interessante aos professores;

3. **Promover parcerias entre instituições de ensino superior e a indústria de tecnologia:** Estabelecer um diálogo mais próximo entre as instituições de ensino e as empresas do setor de tecnologia pode fornecer insights valiosos sobre as necessidades e demandas atuais do mercado de trabalho. Isso pode ser feito por meio de programas de estágio, parcerias de pesquisa e projetos conjuntos, permitindo que os alunos tenham uma experiência mais alinhada com o ambiente profissional e adquiram habilidades relevantes para as necessidades do século XXI;
4. **Investir em capacitação docente e atualização de conteúdos:** Oferecer programas de desenvolvimento profissional para os professores de programação, com foco nas metodologias pedagógicas atualizadas, uso de tecnologias modernas e abordagens inovadoras de ensino. Além disso, é importante investir na atualização dos conteúdos das disciplinas de programação, incluindo tópicos relevantes para a educação e as demandas do século XXI;
5. **Estimular a integração de habilidades transversais no ensino de programação:** Promover a inclusão de habilidades como resolução de problemas complexos, pensamento crítico, colaboração em equipes multidisciplinares e comunicação eficaz nos currículos de programação. Isso pode ser feito por meio da criação de projetos interdisciplinares, trabalhos em grupo e atividades que incentivem a aplicação prática das habilidades transversais;
6. **Aplicar abordagens centradas nas necessidades reais dos usuários no ensino de programação:** Utilizar abordagens centradas nos usuários (*Human Centered Design, Design Thinking, etc.*) para colocar as necessidades e experiências dos usuários no centro do processo de aprendizagem. Essa abordagem pode ser aplicada ao ensino de programação, buscando compreender as necessidades, motivações e desafios dos estudantes na aprendizagem de programação. Isso pode envolver a realização de pesquisas e entrevistas com os alunos para identificar suas necessidades específicas, bem como a criação de experiências de aprendizagem personalizadas, que sejam envolventes, relevantes e significativas para eles. Além disso, pode ajudar a incorporar elementos de usabilidade, acessibilidade e experiência do usuário no ensino de programação, ajudando com que os alunos estejam preparados para desenvolver soluções para as necessidades do século XXI.

## 6. Conclusão e Trabalhos Futuros

Este artigo teve como objetivo investigar se os cursos de computação do Brasil estão ensinando programação de acordo com as necessidades do século XXI. Através de uma análise exploratória de 104 disciplinas de programação introdutória em cursos de graduação em Engenharia de Software e Ciência da Computação, verificou-se que o ensino introdutório de programação está organizado em três grupos distintos.

O primeiro grupo é composto por disciplinas que apresentam conceitos fundamentais de programação, adequadas para alunos matriculados no primeiro semestre de um curso de computação. O segundo grupo abrange disciplinas que exploram técnicas de programação mais avançadas, como estruturas de dados e algoritmos complexos. Essas disciplinas geralmente são introduzidas a partir do segundo semestre e possuem pré-requisitos. O terceiro grupo engloba disciplinas que focam no ensino introdutório de orientação a objetos, com ênfase no desenvolvimento web e projetos acadêmicos.

Contudo, os resultados obtidos apontaram que as instituições de ensino superior brasileiras enfrentam um desafio significativo quando se trata de capacitar os alunos com

as habilidades de programação necessárias para o século XXI. Infelizmente, os cursos analisados não têm conseguido promover um ensino adequado que prepare os alunos para as demandas da indústria de tecnologia atual. De fato, a estrutura rígida e desatualizada do ensino não dialoga com as necessidades do século XXI.

O ensino de programação no século XXI precisa ir além das linguagens de programação básicas. Os programadores do século XXI devem possuir um conjunto abrangente de habilidades, incluindo a capacidade de adaptar-se às constantes mudanças tecnológicas, desenvolvendo uma mentalidade de aprendizado contínuo. Além das habilidades técnicas, é crucial que os programadores desenvolvam habilidades transversais, como resolução de problemas complexos, pensamento crítico, colaboração efetiva em equipes multidisciplinares e comunicação clara.

Foi possível verificar que as instituições de ensino superior brasileiras precisam reformular seus planos e métodos de ensino das disciplinas de programação. É urgente investir em capacitação docente, atualização de conteúdos e uma integração mais estreita com a indústria de tecnologia. Estratégias e ferramentas que orientem os professores no ensino introdutório de programação, desenvolvendo habilidades fundamentais e promovendo competências para o século XXI, são imprescindíveis.

**Ameaças à validade:** Uma das principais limitações deste estudo é o tamanho da amostra utilizada, o que pode afetar a validade dos resultados obtidos. Para tentar minimizar essa limitação, as instituições de ensino foram selecionadas com base em indicadores técnicos relevantes, como os propostos pela ACM-IEEE (2020) e SBC (2017). Outra potencial ameaça à validade é a estratégia de coleta de dados utilizada. Foram analisados planos de ensino acadêmicos, o que significa que nossas conclusões são baseadas nas informações disponíveis nesses documentos. Pode ser que os professores das disciplinas analisadas adotem práticas coerentes com os desafios do século XXI, mesmo que o plano de ensino ou PPC não apresentem esses dados de forma consistente.

**Trabalhos futuros:** Nossa agenda propõe medidas para reduzir os desafios identificados: incentivar pesquisas sobre métodos coerentes com as habilidades do século XXI, divulgar boas iniciativas em disciplinas e instituições, promover parcerias entre instituições de ensino e a indústria de tecnologia, investir em capacitação docente e atualização de conteúdos, estimular a integração de habilidades transversais no ensino de programação e aplicar abordagens centradas nas necessidades reais dos usuários.

## Agradecimentos

Os autores agradecem ao apoio financeiro da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código Financeiro 001, do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), da Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP - Processo 2019/26871-4), e da Especialização em Computação Aplicada à Educação (CAE - ICMC/USP).

## Referências

- ACM and IEEE-CS (2020). “Computing Curricula 2020: CC2020”. <https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2020.pdf>.
- Alam, A. (2021). Possibilities and apprehensions in the landscape of artificial intelligence in education. In *2021 International Conference on Computational Intelligence and Computing Applications (ICCICA)* (pp. 1-8). IEEE.

- Araujo, R., Simão, A., Malucelli, A., Zorzo, A., Monteiro, J. A. and Chaimowicz, L. (2019). Referenciais de formação para os cursos de pós-graduação stricto sensu em computação 2019. *Sociedade Brasileira de Computação*.
- Avcı, Ü. (2022). A predictive analysis of learning motivation and reflective thinking skills on computer programming achievement. *Computer Applications in Engineering Education*, 30(4), 1102-1116.
- Ayinde, L. and Kirkwood, H. (2020). Rethinking the roles and skills of information professionals in the 4th Industrial Revolution. *Business Information Review*, 37(4), 142-153.
- Brasil. (2022). “Base Nacional Comum Curricular”. [http://basenacionalcomum.mec.gov.br/images/BNCC\\_EI\\_EF\\_110518\\_versaofinal\\_site.pdf](http://basenacionalcomum.mec.gov.br/images/BNCC_EI_EF_110518_versaofinal_site.pdf).
- Dede, C. (2010). Comparing frameworks for 21st century skills. *21st century skills: Rethinking how students learn*, 20(2010), 51-76.
- DePryck, K., Vermeersch, J. and Tytgat, A. (2016). If Learning to Code is Not About Coding, Then What is it About?. In *EDEN Conference Proceedings* (No. 1, pp. 50-52).
- De Vega, F. F. (2022). Teaching Programming in the 21st Century. *Journal of Computer Information Systems*, 1-12.
- Eteng, I., Akpotuzor, S., Akinola, S. O. and Agbonlahor, I. (2022). A review on effective approach to teaching computer programming to undergraduates in developing countries. *Scientific African*, e01240.
- Kampylis, P., Dagienė, V., Bocconi, S., Chiocciariello, A., Engelhardt, K., Stupurienė, G., ... and Earp, J. (2023). Integrating Computational Thinking into Primary and Lower Secondary Education. *Educational Technology & Society*, 26(2), 99-117.
- Kelly, W., McGrath, B. and Hubbard, D. (2023). Starting from ‘scratch’: Building young people’s digital skills through a coding club collaboration with rural public libraries. *Journal of Librarianship and Information Science*, 55(2), 487-499.
- Marques, M., Ochoa, S. F. and Bastarrica, M. C. (2016). Software engineering education in Chile-Status report. In *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education* (pp. 180-185).
- Negoro, R. A., Rusilowati, A. and Aji, M. P. (2023). Scratch-Assisted Waves Teaching Materials: ICT Literacy and Students' Critical Thinking Skills. *Journal of Turkish Science Education*, 20(1).
- OECD. (2022). “Skills for the Digital Transition: Assessing Recent Trends Using Big Data”. OECD Publishing, Paris. <https://doi.org/10.1787/38c36777-en>.
- OECD. (2018). “The future of education and skills: Education 2030. OECD education 2030”. OECD Education 2030. [https://www.oecd.org/education/2030/E2030%20Position%20Paper%20\(05.04.2018\).pdf](https://www.oecd.org/education/2030/E2030%20Position%20Paper%20(05.04.2018).pdf).
- Oliveira, K. K. D. S., and SOUZA, R. A. (2022). Digital transformation towards education 4.0. *Informatics in Education*, 21(2), 283-309.
- Rankings, Times Higher Education World University. (2023). World university rankings 2023. [https://www.timeshighereducation.com/world-university-rankings/2023/world-ranking#!/length/25/locations/BRA/sort\\_by/rank/sort\\_order/asc/cols/stats](https://www.timeshighereducation.com/world-university-rankings/2023/world-ranking#!/length/25/locations/BRA/sort_by/rank/sort_order/asc/cols/stats).
- Salleh, F. H. M., Dewi, D. A. and Liyana, N. A. (2021). Issues and challenges for teaching successful programming courses at national secondary schools of Malaysia. In *Computational Science and Technology: 7th ICCST 2020*, Pattaya, Thailand, 29–30 August, 2020 (pp. 501-513). Springer Singapore.

- SBC. Sociedade Brasileira de Computação. (2017). “Referenciais de Formação para os Cursos de Graduação em Computação 2017”. <https://www.sbc.org.br/documentos-da-sbc/send/127-educacao/1155-referenciais-de-formacao-para-cursos-de-graduacao-em-computacao-outubro-2017>.
- Silva, L., Mendes, A. J., Gomes, A. and Cavalcanti de Macêdo, G. F. (2021). Regulation of learning interventions in programming education: A systematic literature review and guideline proposition. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education* (pp. 647-653).
- Silva, E. P., Caceffo, R. E. and de Azevedo, R. J. (2022). Análise dos Tópicos Mais Abordados em Disciplinas de Introdução à Programação em Universidades Federais Brasileiras. In *Anais do II Simpósio Brasileiro de Educação em Computação* (pp. 29-39). SBC.
- Sobral, S. R. (2021). Teaching and Learning to Program: Umbrella Review of Introductory Programming in Higher Education. *Mathematics*, 9(15), 1737.
- Sun, L., Guo, Z. and Zhou, D. (2022). Developing K-12 students’ programming ability: A systematic literature review. *Education and Information Technologies*, 27(5), 7059-7097.
- Vinnervik, P. (2022). An in-depth analysis of programming in the Swedish school curriculum—rationale, knowledge content and teacher guidance. *Journal of Computers in Education*, 1-35.
- Vogel, S., Hoadley, C., Castillo, A. R. and Ascenzi-Moreno, L. (2020). Languages, literacies and literate programming: can we use the latest theories on how bilingual people learn to help us teach computational literacies?. *Computer Science Education*, 30(4), 420-443.
- Wang, Y., Lavonen, J. and Tirri, K. (2019). Twenty-first century competencies in the Chinese science curriculum. *Nordic-Chinese Intersections within Education*, 151-171.
- Wang, W., Zhao, Y., Wu, Y. J., & Goh, M. (2023). Interaction strategies in online learning: Insights from text analytics on iMOOC. *Education and Information Technologies*, 28(2), 2145-2172.
- WEF. (2023). “Future of Jobs Report 2023”. [https://www3.weforum.org/docs/WEF\\_Future\\_of\\_Jobs\\_2023.pdf](https://www3.weforum.org/docs/WEF_Future_of_Jobs_2023.pdf).
- Weingessel, A. and Hornik, K. (2000). Local PCA algorithms. *IEEE Transactions on neural Networks*, 11(6), 1242-1250.
- Yin, R. K. (2009). “Case study research: Design and methods (Vol. 5)”. sage.