Proposta de Abordagem Prática para o Ensino de Programação Baseada em Ausubel

João Henrique Berssanette[1], Antonio Carlos de Francisco[2]

[1]Instituto Federal do Paraná – IFPR – Campus Telêmaco Borba – PR – Brasil
[2]Universidade Tecnológica Federal do Paraná – UTFPR – PR – Brasil

joao.berssanette@ifpr.edu.br, acfrancisco@utfpr.edu.br

Abstract. *This article aims to present the results of a classroom application of a teaching proposal that emphasizes the interaction with the computer and exposes the students earlier to practical programming tasks, taking the theory of meaningful learning Ausubel as a backdrop. The data were collected through the record observations in class during the application, and official documents such as the class diarys of the teachers of the discipline (researcher and holder) and their evaluations. The qualitative analysis of the data collected indicated that the proposed approach can contribute to the programming teaching/learning process.*

Resumo. *Este artigo se objetiva a apresentar os resultados de uma aplicação em sala de aula de uma proposta de ensino de que enfatiza a interação com o computador e expõe os estudantes mais cedo a tarefas práticas de programação, tomando a teoria da aprendizagem significativa de Ausubel como pano de fundo. Os dados foram coletados por meio de registro observações em aula durante a aplicação, e documentos oficiais como os diários de classe dos professores da disciplina (pesquisador e titular) e suas avaliações. A análise qualitativa dos dados coletados indicou que a proposta de abordagem desenvolvida pode contribuir no processo de ensino/aprendizagem de programação.*

1. Introdução

Os cursos de nível superior da área de computação e informática tem como uma de suas metas capacitar estudantes a apresentar soluções computadorizadas para diversos problemas do mundo real. A aprendizagem de programação é essencial para todas as carreiras ligadas à computação e também é importante em áreas relacionadas.

Esta aprendizagem ocorre em disciplinas como algoritmos, lógica de programação, linguagem de programação, técnicas de programação, estrutura de dados, entre outras. Estas disciplinas podem ser consideradas fundamentais para formação de estudantes que terão no desenvolvimento de softwares o produto final de seu trabalho.

Para produzir estas soluções os estudantes devem utilizar comandos definidos a partir de uma linguagem de programação. Uma linguagem de programação possui um conjunto de símbolos e regras de sintaxe e semântica que permite descrever um

DOI: 10.5753/cbie.sbie.2018.398

conjunto de processos de forma precisa e que possam ser executadas por um computador.

No entanto, percebe-se que o processo de ensino/aprendizagem de programação tem se demonstrado difícil para estudantes e professores, acarretando grandes índices de reprovação, desistência e abandono de cursos em instituições de ensino [Pereira Júnior and Rapkiewicz 2004; Silva et al. 2015; Zanetti et al. 2016]. Isto ocorre devido a diversos problemas, a literatura aponta causas como a falta de competências na resolução de problemas, poucas habilidades matemáticas, baixo nível de abstração, dificuldades de interpretação do problema e de compreensão de texto por parte dos estudantes [Gomes, Anabela et al. 2008; Jenkins 2002; Sheard et al. 2009].

Este artigo apresenta parte dos resultados de uma pesquisa de mestrado realizada entre 2013 e 2016 que buscou elaborar uma proposta de abordagem prática baseada na teoria da aprendizagem significativa para o ensino de programação. Discute-se em particular as implicações da exposição dos estudantes mais cedo ao uso prático do computador e a conteúdos que normalmente são vistos de maneira teórica conceitual na abordagem tradicional destas disciplinas.

2. Ensino de programação

No Brasil o ensino de programação ocorre principalmente por cursos superiores nas áreas de Computação e Informática, embora esse conteúdo também seja trabalhado em cursos técnicos profissionalizantes subsequentes ou na modalidade integrada ao ensino médio.

Geralmente o processo inicial de ensino/aprendizagem de programação ocorre por meio de um conjunto de disciplinas introdutórias que são identificadas por nomes como: Algoritmos, Lógica de programação, Linguagem de programação, Técnicas de programação. Estas disciplinas têm como objetivo fornecer aos estudantes os conceitos básicos de programação, geralmente representado por um pequeno conjunto de comandos e conceitos, que os estudantes devem utilizar para implementar soluções para problemas.

Alguns aspectos fundamentais em um curso introdutório seriam: 1º Como resolver problemas; 2º Como descrever uma solução algorítmica; 3º Como verificar se um algoritmo está correto [Gomes, A. et al. 2008; Gries 1974].

De modo geral nestas disciplinas introdutórias o conteúdo tratado inclui: a descrição dos passos necessários para se solucionar um problema; entradas e saídas; constantes e variáveis; tipos primitivos de dados; instrução de atribuição; operadores aritméticos, relacionais e lógicos; estruturas simples de controle de fluxo, decisão e repetição. Nesta etapa inicial em muitas salas de aula evita-se o contato com uma linguagem de programação nesse instante.

Neste contexto, atividades práticas podem ter uma importante função, pois conforme apontam Bennedssen e Caspersen (2008) e Haberman e Muller (2008), um dos entraves no ensino de programação é a forte carga de conceitos abstratos presentes no processo da elaboração e implementação da solução do problema no ambiente computacional.

Sendo assim, é importante oportunizar ao estudante que observe a execução de um programa e seus resultados. Isso lhe oferece mais um caminho para tratar dificuldades encontradas em aspectos teóricos, uma vez que outra limitação encontrada em algumas salas de aula é justamente deixar de enfatizar a solução de problemas para se concentrar em conceitos abstratos [Gomes and Mendes 2007; Koliver et al. 2004; Nobre and Menezes 2002].

Dunican (2002), identifica três categorias de estudantes iniciantes em programação:

i. estudantes que não têm a aptidão para compreender os conceitos básicos, este é muitas vezes resultado de uma escolha equivocada do curso;

ii. estudantes que podem captar os conceitos essenciais se expostos a abordagens de ensino eficazes; e

iii. estudantes que são totalmente confortáveis com a natureza abstrata de conceitos de programação.

Seguindo essa classificação e assumindo que esteja correta, o professor deve ficar atento para identificar os estudantes pertencentes à segunda categoria, e assegurar condições adequadas de ensino/aprendizagem para que a maioria destes possam progredir. Entretanto, esperar que o processo de ensino/aprendizagem de programação, possa ser considerado apenas uma receita didática é seguramente um erro. Aprender a programar computadores é como corolário, ensinar essa habilidade não é uma tarefa simples, tampouco trivial [Jenkins 2002; Robins et al. 2003].

Ao longo dos anos, o processo de ensino/aprendizagem dos fundamentos de programação de computadores, tem repetido as mesmas dificuldades. Isto levou muitos professores e pesquisadores a estudar as causas e propor soluções variadas que visam de alguma maneira atenuar estes problemas. Grande parte destas pesquisas são voltadas especialmente para estudantes novatos em programação, como em [Brusilovsky et al. 1994; Delgado et al. 2004; Pereira Júnior and Rapkiewicz 2004], apenas para citar alguns trabalhos.

A literatura disponibiliza uma série de propostas que visam contribuir com o processo de ensino/aprendizagem de programação, porém nenhuma destas se mostrou completa ou mesmo genérica a ponto de sanar os problemas de aprendizado de programação que ainda persistem.

Em geral, para tratar este problema, as propostas presentes na literatura podem ser qualificadas em três vertentes, sendo: Ferramentas, Metodologias e a união de ambas [Pereira Júnior and Rapkiewicz 2004].

3. Aprendizagem significativa

Aprendizagem significativa é o conceito central da teoria de aprendizagem de David Ausubel, proposta na década de 60. Nela, o autor destaca que a aprendizagem significativa é o mecanismo humano para adquirir e armazenar a vasta quantidade de ideias e informações representadas em qualquer campo de conhecimento [Ausubel 1963; Ausubel et al. 1968].

Essa teoria postula que o conhecimento humano é armazenado de maneira estruturada, em conjuntos de informações que o indivíduo possui sobre um determinado assunto. A aprendizagem torna-se significativa se novos conteúdos são relacionados de alguma maneira com conhecimentos anteriores. Emprega-se o termo ancoragem para destacar a necessidade desses conhecimentos pré-existentes. A nova relação – ou conceito – não consiste numa simples associação, havendo uma interação entre os elementos que dela participam [Moreira 2011].

Sendo assim, a aprendizagem significativa é um processo de mudança do conhecimento de um indivíduo. Esse processo modifica o conhecimento já existente, pois cria novas relações ou conexões com os fatos novos que são adquiridos. Pode-se dizer que a configuração da estrutura cognitiva passa de um estado a outro.

Para que a aprendizagem seja significativa se faz necessário atender a três condições: predisposição do indivíduo; material potencialmente significativo e estrutura cognitiva capaz de assimilar a nova informação [Ausubel et al. 1980].

A predisposição para aprender diz respeito à experiência afetiva, ao interesse do aprendiz em relação ao evento de instrução [Novak et al. 2000]. Além disso, a aprendizagem significativa propõe a participação ativa do estudante na aquisição de conhecimento, de maneira a evitar-se uma mera reprodução de conceitos formulados pelo professor ou pelo livro-texto; espera-se uma reelaboração do aluno [Pelizzari et al. 2002].

No que diz respeito ao material potencialmente significativo, cabe observar que cada disciplina possui uma estrutura de conceitos, ideias, exemplos [Moreira and Masini 2001]. Entretanto, a sequência em que tais informações são apresentadas nem sempre é a mais adaptada para facilitar a interação com o conhecimento do estudante. Para evitar isso, é essencial um planejamento que considere a estrutura da disciplina e dos conteúdos, a organização desse material ao longo do tempo e, em particular, manter foco no estudante e estimar seu conhecimento prévio. A linguagem pode ser um facilitador importante ou um empecilho, e deve pesar no planejamento [Moreira 1983]. É importante também não sobrecarregar o estudante de informações e saber filtrar aquelas desnecessárias.

Outra questão a ser ponderada é o momento para tratar um dado conteúdo. O adiamento da experiência de aprendizagem para além da maturidade do estudante desperdiça oportunidades valiosas. Inversamente, avançar demasiadamente a introdução de certos conceitos pode levar o estudante até a temer, desgostar e evitar uma tarefa [Ausubel 2003].

4. Metodologia

Este trabalho foi conduzido como parte de uma pesquisa de mestrado. Conforme exposto, o objetivo central foi elaborar uma proposta de abordagem de ensino de programação de computadores com uma forte conotação prática, alicerçada pela teoria da aprendizagem significativa de Ausubel.

A pesquisa foi executada no Instituto Federal do Paraná – IFPR, campus Telêmaco Borba, a população do estudo compreende 38 (trinta e oito) estudantes. O foco da análise são os conteúdos apresentados aos estudantes, comparando-se a

abordagem elaborada com uma que se poderia chamar de tradicional, muito empregada na disciplina.

Para isso, realizou-se a coleta de dados por meio de documentos oficiais como os diários de classe dos professores da disciplina (pesquisador X titular que ministrou a disciplina no ano anterior) e suas avaliações. Além disso foram registradas pelo pesquisador observações em sala de aula durante a aplicação.

Com a posse destes dados buscou-se comparar a aplicação da abordagem proposta e a abordagem aqui dita tradicional. Para isso foram elencados os seguintes critérios:

i. Cumprimento da ementa e objetivos da disciplina;
ii. Comparação da época em que os assuntos foram introduzidos;
iii. Comparação da época de realização de avaliações e provas;
iv. Comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina;

4.1. Aplicação em sala

A aplicação ocorreu na disciplina de Algoritmos e Lógica de Programação I – ALP-I, do curso superior em Análise e Desenvolvimento de Sistemas - ADS do IFPR – campus Telêmaco Borba. A disciplina ALP-I está presente no 1º semestre do curso com 80 horas aula.

Todas as aulas ministradas pelo professor pesquisador foram práticas em laboratório. Foram utilizadas, em sequência as seguintes ferramentas:

- Primeiro bimestre, as linguagens de programação *BASIC* pelo interpretador *Decimal Basic* e *Python* pelo *IDLE Python*.
- Segundo bimestre, a linguagem de programação C, utilizando o *Codeblocks*.

A introdução de diferentes linguagens foi planejada propositalmente na proposta de abordagem elaborada, pois, por meio da percepção dos estudantes durante a transição entre as linguagens, esperava-se que o foco dos estudantes fosse direcionado aos conteúdos de programação, e não nas características especificas de sintaxe e semântica das linguagens de programação.

4.2. Proposta de abordagem elaborada

De acordo com a aprendizagem significativa: um dos pontos fundamentais consiste em determinar aquilo que o aprendiz já sabe ou conhece [Moreira and Masini 2001], ou seja, o estado atual da sua estrutura cognitiva (conjunto de ideias que, no aluno, preexistem à nova aprendizagem), para que a proposta de ensino seja baseada nestes conhecimentos.

No entanto, para a maioria dos estudantes iniciantes, a programação de computadores é um assunto completamente novo [Dijkstra 1988]. Sendo assim, os estudantes na maioria das vezes não possuem os subsunçores necessários para que a aprendizagem ocorra de maneira significativa, fazendo com que a aprendizagem de programação de computadores ocorra com pouco sentido. Pode então acontecer a

aprendizagem mecânica, em que os estudantes são capazes eventualmente de repetir exatamente o que viram, mas não conseguem aplicar o conhecimento em uma situação nova.

Nestes casos onde o aprendiz não possui o conjunto de conhecimentos prévios necessários, Ausubel sugere os organizadores prévios. Exemplificando, no presente caso a ideia de "repetição" já é conhecida dos estudantes. Ao apresentar o computador realizando uma repetição, em uma linguagem interpretada, não se exige do estudante imaginar o funcionamento de um comando como *FOR*: a execução de uma repetição pela máquina adquire um caráter quase concreto. Ao mudar os valores limites do laço em seu computador quando questionado pelo professor pesquisador (por exemplo, em lugar de 10 iterações realizar 20), o estudante pode ainda experimentar hipóteses e sedimentar mais os conceitos. Em comparação, o estudante na aula teórica só irá confirmar o que viu mais tarde – ou outro dia – em laboratório, quando poderá ainda ser frustrado por erros de sintaxe e questionar a si mesmo, se entendeu ou não a matéria.

Na proposta de abordagem elaborada os conteúdos foram apresentados em pequenas estruturas e códigos simples, que ao longo da disciplina vão se tornando maiores e mais complexos. Por meio de atividades práticas, os estudantes podem desenvolver seu próprio repertório de conhecimentos e habilidades para resolver problemas e representá-los no computador.

Os conteúdos na abordagem proposta foram apresentados de forma inter-relacionados e de maneira cíclica. O estudante poderia assim, ganhar maturidade para assimilar estes conteúdos, o que geralmente não ocorre na abordagem mais tradicional, pois nesta os conteúdos são distribuídos ao longo do tempo, apresentados isoladamente. É comum pensar em uma aula para variáveis, outra para estruturas de decisão, etc. Isto faz com que o estudante sinta dificuldades para combinar conceitos na solução de problemas quando necessário.

5. Resultados

Visando a analisar os resultados desta pesquisa buscou-se comparar a aplicação da proposta de abordagem a uma abordagem tradicional de acordo com os critérios estabelecidos na metodologia.

Conforme os registros de conteúdo dos diários de classes do professor titular da disciplina, observa-se que no ano anterior, o professor titular cumpriu com todos os elementos da ementa com a inserção de tópicos não contemplados na ementa como a integração da programação com banco de dados no último bimestre.

O professor pesquisador não contemplou alguns elementos da disciplina que dizem respeito mais à metodologia de apresentação do que ao desenvolvimento de habilidade de programação; isso incluiu 'Fundamentos de algoritmos e programação'; 'Fluxogramas'; 'Pseudocódigo'. A não apresentação destes elementos foi opção do pesquisador por contrastar com a estratégia didática proposta, que busca evitar a discussão e apresentação de conceitos abstratos ainda não conhecidos pelos estudantes. Desta forma assume-se que nesse item o cumprimento da ementa foi parcial.

Por outro lado, o professor pesquisador inseriu itens mais avançados não contemplados na disciplina como: recursividade, busca e ordenação de vetores, desta

forma o cumprimento da ementa foi parcial, porém com inserção de tópicos não contemplados originalmente. Uma vez que um dos objetivos da disciplina é levar os estudantes a produzir códigos, esse resultado foi positivo.

Para comparar o momento em que assuntos foram introduzidos por cada professor e os conteúdos abordados em avaliações e provas, observaram-se os seguintes tópicos da ementa: 1. Programação sequencial; 2. Instruções de seleção; 3. Instruções de repetição; 4. Vetores; 5. Matrizes; 6. Funções e procedimentos; 7. Implementação de problemas em uma linguagem de programação. A seguir é apresentado o Quadro 1, comparando a época em que os assuntos foram introduzidos.

Quadro 1: Comparação da época em que os assuntos foram introduzidos

Tópicos	Titular	Pesquisador
Programação sequencial	Mais tarde	Mais cedo
Instruções de seleção	Mais tarde	Mais cedo
Instruções de repetição	Mais tarde	Mais cedo
Vetores	Mais tarde	Mais cedo
Matrizes	Igual	Igual
Funções e procedimentos	Mais tarde	Mais cedo
Implementação de problemas em uma linguagem de programação	Mais tarde	Mais cedo

Conforme o Quadro 1, observa-se que com a exceção do assunto matrizes que foi visto na mesma época todos os assuntos foram introduzidos mais cedo pelo professor pesquisador. Os registros de conteúdos indicam também que a época de realização de avaliações, provas e trabalhos, foram semelhantes entres os professores.

A seguir é apresentado o Quadro 2, onde é realizada a comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina pelo professor titular e pesquisador.

Quadro 2: Conteúdos abordados em avaliações e provas referentes a ementa da disciplina

TÓPICOS	Titular				Pesquisador			
	1o Bim		2o Bim		1o Bim		2o Bim	
	Contempla		Contempla		Contempla		Contempla	
	Sim	*Não*	*Sim*	*Não*	*Sim*	*Não*	*Sim*	*Não*
Programação sequencial	X		X		X		X	
Instruções de seleção	X		X		X		X	
Instruções de repetição		X	X		X		X	
Vetores		X		X		X	X	
Matrizes		X		X		X		X
Funções e procedimentos		X	X		X		X	
Implementação de problemas em uma linguagem de programação		X	X		X		X	
Total de tópicos contemplados	**2**		**5**		**5**		**6**	

As comparações dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina realizadas indicam que de um modo geral a disciplina ministrada pelo professor pesquisador contemplou mais tópicos no primeiro e segundo bimestre.

Além destes resultados apresentados, cabe mencionar que percebeu-se que as aulas práticas dos conteúdos introdutórios de programação de computadores produziram um engajamento maior por parte dos estudantes. Observou-se também alguns indícios de que estas atividades práticas podem diminuir a lacuna de abstração, um dos grandes

problemas apontados por pesquisas sobre o ensino/aprendizagem de programação de computadores.

Do ponto de vista investigativo, conclui-se que esta aplicação obteve êxito e diversos fatores foram identificados ao longo de sua execução. Apesar do fato da disciplina ministrada pelo pesquisador abordar em grande maioria os conteúdos mais cedo, não foi possível mensurar com todo rigor metodológico o quanto isso impactou na aprendizagem dos estudantes.

Os resultados da aplicação permitiram também observar indícios de que adiantar a exposição dos estudantes a assuntos que normalmente são apresentados de forma abstrata e conceitual, por meio de atividades práticas no computador, não interferiu negativamente no desempenho da disciplina e dos estudantes. Do ponto de vista da abordagem proposta, a definição de conceitos teóricos acontece depois que os subsunçores foram estabelecidos na estrutura cognitiva do estudante. Isso é feito exatamente pelas atividades práticas em laboratório, que anulam o caráter abstrato ao apresentar exemplos reais de execução de código interpretado no computador.

Além disso, pode-se observar que a não apresentação de forma teórica dos fundamentos de algoritmos e programação, e o não uso de representações como fluxogramas e pseudocódigo não influenciou a capacidade de os estudantes assimilarem os conteúdos envolvidos em programação e desenvolverem seus próprios códigos. Evidentemente, com a construção de códigos de maiores dimensões existirá a necessidade de tratar a estruturação de soluções (por exemplo na forma de módulos). Entretanto essa etapa só acontece depois que os estudantes já estão programando.

6. Considerações finais

O processo de ensino/aprendizagem dos fundamentos de programação de computadores, tem se mostrado difícil para estudantes e professores. A literatura disponibiliza propostas que visam de alguma maneira contribuir com o processo de aprendizagem de programação ou atenuar as dificuldades existentes no processo.

Deste modo, esta pesquisa teve como objetivo elaborar uma proposta de abordagem prática baseada na teoria da aprendizagem significativa, enfatizando/valorizando a interação com a máquina e expondo os estudantes mais cedo ao uso prático do computador para o ensino de programação.

A síntese dos resultados da aplicação da proposta de abordagem desenvolvida indica que a carga horária utilizada para apresentação dos tópicos da disciplina foi menor comparativamente a carga horária habitual, desta forma, o professor pode aproveitar o tempo para trabalhar mais vezes estes conteúdos permitindo uma melhor fixação destes por parte do estudante.

Além disso, a proposta oportuniza aos estudantes verem os conteúdos mais vezes de maneira inter-relacionada e de maneira cíclica onde o estudante pode ganhar maturidade para assimilar estes conteúdos.

Percebeu-se também que as aulas práticas dos conteúdos introdutórios de programação de computadores podem produzir um engajamento maior por parte dos estudantes e podem diminuir a lacuna no quesito abstração, um dos grandes problemas

apontados por pesquisas sobre o ensino /aprendizagem de programação de computadores.

Por fim, destaca-se que a exposição dos estudantes mais cedo ao uso prático do computador e a assuntos que normalmente são vistos de maneira conceitual primeiramente na abordagem tradicional, pode não interferir negativamente no desempenho da disciplina e dos estudantes.

Agradecimentos

Os autores agradecem ao apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico – CNPq, e ao professor André Koscianski pela colaboração.

Referências

Ausubel, D. P. (1963). The psychology of meaningful verbal learning.

Ausubel, D. P. (2003). Aquisição e retenção de conhecimentos: uma perspectiva cognitiva. *Lisboa: Plátano*, *1*.

Ausubel, D. P., Novak, J. D., & Hanesian, H. (1968). *Educational psychology: A cognitive view* (Vol. 6). New York: Holt, Rinehart and Winston.

Ausubel, D. P., Novak, J. D., & Hanesian, H. (1980). *Psicologia educacional*. Interamericana.

Bennedssen, J., & Caspersen, M. E. (2008, September). Abstraction ability as an indicator of success for learning computing science?. In *Proceedings of the Fourth international Workshop on Computing Education Research* (pp. 15-26). ACM.

Brusilovsky, P. (1994). Teaching Programming to Novices: A Review of Approaches and Tools.

Delgado, C., Xexeo, J. A. M., SOUZA, I. F., Campos, M., & Rapkiewicz, C. E. (2004). Uma abordagem pedagógica para a iniciação ao estudo de algoritmos. In *XII Workshop de Educação em Computação*.

Dijkstra, E. W. (1988). On the cruelty of really teaching computing science. *Communications of the ACM, 32*(12), 1398-1404.

Dunican, E. (2002). Making the analogy: Alternative delivery techniques for first year programming courses.

Gomes, A., Areias, C., Henriques, J., & Mendes, A. J. (2008). Aprendizagem de programação de computadores: dificuldades e ferramentas de suporte. *Revista Portuguesa de Pedagogia*, 161-179.

Gomes, A., Henriques, J., & Mendes, A. (2008). Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores. *Educação, Formação & Tecnologias-ISSN 1646-933X, 1*(1), 93-103.

Gomes, A., & Mendes, A. J. (2007). Learning to program-difficulties and solutions. In *International Conference on Engineering Education–ICEE* (Vol. 2007).

Gries, D. (1974). What should we teach in an introductory programming course?. *ACM SIGCSE Bulletin, 6*(1), 81-89.

Haberman, B., & Muller, O. (2008). Teaching abstraction to novices: Pattern-based and ADT-based problem-solving processes. In *Frontiers in Education Conference, 2008. FIE 2008. 38th Annual* (pp. F1C-7). IEEE.

Jenkins, T. (2002). On the difficulty of learning to program. In *Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences* (Vol. 4, No. 2002, pp. 53-58).

Koliver, C., DORNELES, R. V., & CASA, M. E. (2004). Das (muitas) dúvidas e (poucas) certezas do ensino de algoritmos. In *XII Workshop de Educação em Computação*.

Moreira, M. A. (1983). *Uma abordagem cognitivista ao ensino da Física*. Universidade Portucalense.

Moreira, M. A. (2011). *Teorias de aprendizagem*. 2. ed. amp. São Paulo: EPU.

Moreira, M. A., & Masini, E. F. (2001) *Aprendizagem Significativa: A Teoria de David Ausubel*. São Paulo: Centauro.

Nobre, I. A. M., & de Menezes, C. S. (2002). Suporte à Cooperação em um Ambiente de aprendizagem para Programação (SAmbA). In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)* (Vol. 1, No. 1, pp. 337-347).

Novak, J. D., Rabaça, A., & Valadares, J. (2000). *Aprender criar e utilizar o conhecimento: Mapas conceptuais TM como ferramentas de facilitação nas escolas e empresas*.

Pelizzari, A., Kriegl, M. D. L., Baron, M. P., Finck, N. T. L., & Dorocinski, S. I. (2002). Teoria da aprendizagem significativa segundo Ausubel. *revista PEC*, *2*(1), 37-42.

Pereira Júnior, J. C. R., & Rapkiewicz, C. E. (2004). O processo de ensino-aprendizagem de fundamentos de Programação: uma visão crítica da pesquisa no Brasil. In *Anais do XII Workshop sobre Educação em Computação (SBC)*.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer science education*, *13*(2), 137-172.

Sheard, J., Simon, S., Hamilton, M., & Lönnberg, J. (2009, August). Analysis of research into the teaching and learning of programming. In *Proceedings of the fifth international workshop on Computing education research workshop* (pp. 93-104). ACM.

Silva, T. R., Medeiros, T., Medeiros, H., Lopes, R., & Aranha, E. (2015). Ensino-aprendizagem de programação: uma revisão sistemática da literatura. *Revista Brasileira de Informática na Educação*, *23*(1).

Zanetti, H., Borges, M., & Ricarte, I. (2016, November). Pensamento Computacional no Ensino de Programação: Uma Revisão Sistemática da Literatura Brasileira. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)* (Vol. 27, No. 1, p. 21).