Um modelo para a aprendizagem do pensamento computacional aliado à autorregulação

Rozelma Soares de França, Patrícia Cabral de Azevedo Restelli Tedesco

Centro de Informática– Universidade Federal de Pernambuco (CIn - UFPE)
Recife – PE – Brasil

{rsf2, pcart}@cin.ufpe.br

Resumo. *A literatura recente sobre ensino do pensamento computacional enfatiza a necessidade de engajar os estudantes em atividades de reflexão durante a aprendizagem de conceitos norteadores dessa habilidade. Neste âmbito, este trabalho apresenta um modelo colaborativo concebido para introduzir práticas de autorregulação no contexto do ensino e da aprendizagem do pensamento computacional no ensino médio. O modelo foi implementado e avaliado por meio de um quasi-experimento durante um curso de desenvolvimento de jogos digitais que introduziu conceitos de lógica de programação em iniciantes na área. Seus resultados sugerem que a estratégia proposta é benéfica ao aprendizado do pensamento computacional e pode contribuir para a autorregulação da aprendizagem dos estudantes.*

1. Introdução

Há uma consciência crescente sobre a necessidade de ensinar Computação, enquanto ciência, na educação básica. Este fato dá à escola a oportunidade de rever e aprimorar o seu currículo a fim de abordar desafios e oportunidades oferecidas pelo mundo tecnologicamente rico em que vivemos.

A introdução de conceitos de Computação na educação básica é fundamental pelo seu caráter transversal às demais áreas do conhecimento (CSTA, 2011). Neste contexto, desafios provenientes das Ciências Exatas, Humanas, Artes e da realidade cotidiana, por exemplo, poderão ser resolvidos com o auxílio da Computação. Complementarmente, profissionais de tais áreas poderão interagir com aqueles da Computação por meio de um pensamento interdisciplinar, o pensamento computacional, fazendo-se necessário o seu ensino na formação básica dos estudantes, assim como a leitura, a escrita e a aritmética (WING, 2006).

O pensamento computacional baseia-se em fundamentos da Ciência da Computação e constitui uma habilidade fundamental para todos e não apenas para os cientistas da computação (WING, 2006), influenciando a pesquisa em várias áreas do conhecimento (BUND, 2007).

Percebendo a importância em desenvolver desde cedo habilidades como o pensamento computacional, os Estados Unidos e países da Europa têm implantado um currículo mínimo de Computação em suas escolas (CSTA, 2005). No Brasil, o debate ainda é incipiente. Contudo, é possível observar iniciativas em diversos estados como Rio de Janeiro (DE SOUZA *et al.,* 2014), Minas Gerais (CARVALHO *et al.*, 2013), Paraíba (SCAICO *et al.*, 2013), Rio Grande do Sul (ANDRADE *et al.*, 2013), Amazonas (VIEIRA *et al.*, 2013), Pernambuco (FRANÇA *et al.*, 2012) e Bahia (SOUSA *et al*., 2010).

Por outro lado, a formação em Computação apresenta desafios diversos, a exemplo das dificuldades enfrentadas por programadores iniciantes na aplicação de conceitos básicos de programação já compreendidos (LAHTINEN *et al.*, 2005). Além disso, outros problemas podem ser desencadeados pela inabilidade dos estudantes de estabelecer metas de estudo, em monitorar e refletir sobre sua própria aprendizagem, processos estes, associados à aprendizagem autorregulada, considerada uma das competências-chave para iniciar e manter o aprendizado ao longo da vida (COUNCIL, 2002).

Nesse contexto, os resultados da revisão de literatura realizada por Lye e Koh (2014) apontam para a necessidade de engajar estudantes do ensino médio em atividades de reflexão durante a aprendizagem do

pensamento computacional. Sob a ótica da aprendizagem autorregulada (ZIMMERMAN, 2002), a reflexão, especificamente a autorreflexão, desempenha um papel primordial na aprendizagem por possibilitar aos aprendizes que avaliem o próprio conhecimento na tentativa de identificar as causas dos seus próprios erros e acertos acadêmicos. Tais atividades de monitoramento e avaliação da aprendizagem podem ocorrer individualmente, como também em colaboração com os pares.

Adicionalmente, o documento produzido no APEC'2008 *Education Reform Symposium*[1], destaca que os sistemas educacionais existentes devem ser modificados visando a integração das competências requeridas dos estudantes no século XXI, a saber: i) aprender ao longo da vida; ii) resolver problemas; iii) autogerenciar a aprendizagem e iv) trabalhar em equipe. Espera-se, com isto, que os estudantes aprendam a participar de forma apropriada em uma sociedade cada vez mais diversificada, utilizar as novas tecnologias e lidar com as rápidas mudanças nos locais de trabalho algo que, indubitavelmente, está ancorado nos quatro pilares da educação apresentados pela Comissão Internacional sobre Educação para o Século XXI à UNESCO (DELORS *et al*., 1996).

No contexto brasileiro, ressalta-se ainda que um dos grandes desafios científicos da Computação apontado pela Sociedade Brasileira de Computação (SBC) para o decênio 2006-2016 é o "Acesso participativo e universal do cidadão brasileiro ao conhecimento". Tal desafio tem como objetivo criar condições para vencer barreiras tecnológicas, educacionais, culturais, sociais e econômicas que impedem o acesso e a interação por meio da concepção de sistemas, ferramentas, modelos, métodos, procedimentos e teorias capazes de endereçar, de forma competente, a questão do acesso do cidadão brasileiro ao conhecimento (SBC, 2006).

Paralelo a isso, há também o Plano Nacional de Educação (BRASIL, 2014) o qual prevê metas e estratégias com relação à educação nacional a serem implementadas no decorrer do decênio 2014-2024. Dentre as estratégias traçadas para atingir as metas do plano, pode-se destacar: i) o desenvolvimento de tecnologias educacionais e de inovação das práticas pedagógicas, bem como a seleção e divulgação de tecnologias que sejam capazes de alfabetizar e de favorecer a melhoria do fluxo escolar e a aprendizagem dos estudantes (Estratégias 5.3 e 5.4) e ii) o desenvolvimento de tecnologias para correção de fluxo, para acompanhamento pedagógico individualizado e para recuperação e progressão parcial (Estratégia 8.1). Em tais estratégias, fica evidente o incentivo à construção de recursos tecnológicos que auxiliem na prática pedagógica e favoreçam a aprendizagem de estudantes da educação básica.

Considerando o exposto, é mister investigar de que maneira o processo de ensino-aprendizagem do pensamento computacional pode ser promovido na educação básica brasileira. Mais que isso, como a autorregulação pode contribuir com o desenvolvimento dessa habilidade, considerando o contexto em que os estudantes estão inseridos e suas dificuldades na aprendizagem de conceitos computacionais. Complementarmente a tecnologia pode ser utilizada para apoiar o desenvolvimento de tais atividades.

Neste contexto, esta pesquisa visa investigar de que forma a aprendizagem do pensamento computacional, em especial a programação, pode ser promovida no ensino médio por meio da autorregulação apoiada por tecnologias. Para tanto, um modelo conceitual foi definido, o qual é baseado na noção de que instigar a autorreflexão dos estudantes durante os processos de resolução e avaliação de problemas de lógica de programação desencadeia o desenvolvimento da autorregulação e tem potencial de melhorar a aprendizagem do pensamento computacional.

O restante do artigo está organizado como segue: na seção 2 é apresentado o modelo proposto para promover a aprendizagem do pensamento computacional, destacando suas fases e processo de concepção. Na seção 3 um estudo experimental com o uso do modelo proposto é delineado, juntamente com resultados obtidos em um curso de desenvolvimento de jogos digitais com iniciantes em programação. Por fim, na seção 4 são feitas as considerações finais da pesquisa, destacando as contribuições do estudo e apontando algumas direções para trabalhos futuros.

[1] APEC'2008 Education Reform Symposium in Xi'na, China. 21st Century Competencies. Disponível em: <http://bit.ly/1jLooj4 >. Acesso em: 01 de abr. 2014.

2. O modelo penC[2]

Considerando-se o desafio científico "Acesso participativo e universal do cidadão brasileiro ao conhecimento" da SBC e o PNE 2014-2024, atrelado à necessidade de formar os educandos para que construam habilidades que lhe permitam aplicar o pensamento computacional na resolução de problemas provenientes de diversas áreas, bem como regular a própria aprendizagem, o modelo penC foi definido.

Sua concepção baseou-se em estudos sobre metacognição, ensino de Ciência da Computação e aprendizagem no nível médio. Em relação à instrução metacognitiva apoiada por tecnologia, o trabalho de Gama (2004) foi a principal referência, contribuindo na definição das fases para resolução de problemas. Estudos selecionados em mapeamentos sistemáticos realizados durante a pesquisa deram subsídio à criação de atividades de avaliação de problemas pelos estudantes. Em especial, os resultados da pesquisa de Sitthiworachart *et al.* (2004) foram considerados, os quais indicam a importância do anonimato e discussão em grupo no processo de avaliação por pares em programação.

Tendo em vista que a maioria dos estudos mapeados teve como público-alvo o ensino superior, recorreu-se à literatura sobre aprendizagem no ensino médio para apoiar os estudantes de tal nível de ensino em seu processo de aprendizagem do pensamento computacional. Nesse sentido, as atividades de autoavaliação propostas foram influenciadas pelo trabalho de Long e Aleven (2013) o qual aponta para a necessidade de instruções explícitas que auxiliem estudantes do ensino médio a autoavaliarem o conhecimento.

O penC tem a intenção de criar condições adequadas para que estudantes do ensino médio desenvolvam habilidades e competências requeridas na atualidade pensando sobre si mesmos (consciência metacognitiva) como solucionadores de problemas e refletindo sobre suas experiências contínuas de aprendizagem.

A consciência metacognitiva permite ao estudante planejar, sequenciar e monitorar a própria aprendizagem de modo a melhorar diretamente o seu desempenho acadêmico. Todavia, segundo Gama (2004), a metacognição é necessária, mas não suficiente para o sucesso acadêmico. O ponto mais importante é que por meio da prática da autorregulação os estudantes possam desenvolver o controle sobre a própria aprendizagem. Os professores podem aumentar a consciência e o controle sobre a aprendizagem dos estudantes, ensinando-lhes a refletir sobre como eles pensam, aprendem, lembram e realizam tarefas acadêmicas. Isto pode ocorrer antes, durante e após a execução de uma tarefa de aprendizagem.

O penC foi concebido para ser integrado a ambientes de resolução de problemas de Lógica de Programação. Desse modo, não deve ser usado isoladamente. Em vez disso, deve ser acoplado a algum ambiente computacional, atuando como assistente no processo de aprendizagem do pensamento computacional.

2.1. Fases do penC

O modelo penC é constituído de quatro fases, a saber: pré-reflexão, resolução, avaliação por pares e pós-reflexão. Elas são instanciadas enquanto os estudantes resolvem cada novo problema de Lógica de Programação, e detalhadas a seguir.

A primeira fase é a *pré-reflexão*. Ela antecede a resolução de um problema e é constituída de duas atividades. A fim de estimular a reflexão dos estudantes sobre seu processo de aprendizagem como um todo, a primeira atividade na fase de pré-reflexão apresenta o estado atual do aprendiz com relação aos resultados de aprendizagem que devem ser alcançados em um curso introdutório de programação. A segunda atividade tem como objetivo apoiar a reflexão dos estudantes sobre o problema a ser resolvido, ajudando-os a reconhecer as metas e identificar os dados, com base na sua confiança para resolvê-lo. Para isso, esta atividade apresenta um conjunto de questões de autoavaliação que devem ser respondidas pelo estudante antes dele começar a resolver o problema. As respostas coletadas são utilizadas na última fase do modelo penC a fim de avaliar o nível de confiança do aprendiz.

[2] Pronuncia-se 'pense'.

A segunda fase do modelo penC é chamada de *resolução*. Nesta fase os estudantes resolvem um problema de programação seguindo padrões de codificação (por exemplo, nomenclatura de variáveis). Durante a resolução de problemas, há *scaffolds*[3] que ajudam os aprendizes a refletir sobre o problema atual com base na sua experiência anterior na resolução de problemas de programação. No final desta fase, os estudantes apresentam a sua solução antes do término do prazo definido pelo professor, que pode monitorar os aprendizes com dificuldades na resolução do problema e fornecer *feedback* de acordo com as necessidades dos estudantes.

A *avaliação por pares* é a terceira fase do modelo penC. Nesta fase, as soluções apresentadas pelos estudantes são avaliadas por seus pares. No início desta fase, estas soluções são atribuídas a, pelo menos, outros três estudantes que devem entender os critérios de avaliação. Esta atividade é importante em dois aspectos. O primeiro aspecto é porque uma compreensão imprecisa dos critérios pode interferir na forma de avaliação e, consequentemente, afetar a aprendizagem dos estudantes que receberão os comentários. O outro aspecto é que o professor será notificado desses estudantes-avaliadores que não entenderam os critérios e poderá guiá-los em sua avaliação.

A lista de critérios para a avaliação da solução pelos aprendizes fornece uma análise quantitativa e qualitativa da solução. Para cada critério de avaliação, o avaliador deve fornecer uma pontuação e também um *feedback* escrito para ajudar o estudante a refletir sobre o que ele realizou. Neste sentido, *scaffolds* foram projetados para ajudar os avaliadores a escrever comentários que demonstrem os pontos fortes da solução, os erros e sugiram melhorias. No final desta fase, cada avaliador envia de volta a sua análise para o criador da solução.

A quarta fase do penC é a *pós-reflexão*, que visa envolver os estudantes na reflexão sobre a resolução de problemas de programação. No início desta fase, os estudantes podem ver o *feedback* dado à sua solução na fase anterior. Depois de ver o seu *feedback*, os aprendizes também avaliam a qualidade do *feedback* recebido. Em seguida, eles podem optar por compartilhar com outros estudantes suas soluções e receber novos comentários, um processo que estimula uma nova discussão sobre a solução compartilhada.

Ainda na fase pós-reflexão, os estudantes também pode monitorar seu processo de aprendizagem através de diferentes atividades. Primeiro, eles podem analisar gráficos que apresentam dois aspectos da capacidade de monitoramento: precisão no monitoramento do conhecimento (*Knowledge Monitoring Accuracy* – KMA) definido por Tobias e Everson (2002) e o viés no monitoramento do conhecimento (*Knowledge Monitoring Bias* – KMB) definido por Gama (2004).

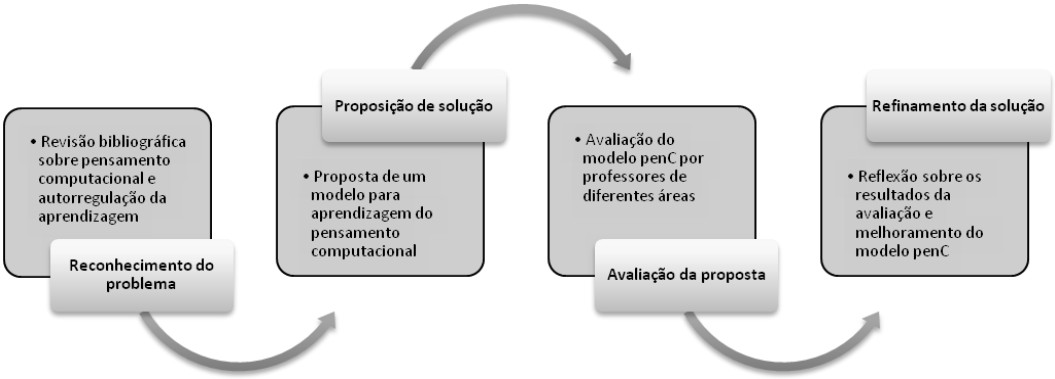
O KMA refere-se à capacidade do estudante de inferir como ele irá realizar uma tarefa de aprendizagem e, dependendo do resultado obtido a partir do KMA, o KMB indica se o aprendiz é pessimista, otimista ou aleatória. Analisando os gráficos apresentados, os estudantes podem refletir e acompanhar o seu próprio processo de aprendizagem do pensamento computacional. Além disso, eles podem interagir com os gráficos apresentados e interpretar os resultados em um processo de autoavaliação. Em segundo lugar, os estudantes, na visão de avaliadores, podem comparar os seus resultados com os resultados de seus pares. Finalmente, todos podem discutir sobre as diferentes soluções de problemas compartilhadas com e pelos seus pares.

2.2. Processo de concepção

O processo de concepção do penC envolveu várias etapas, conforme Figura 1, as quais são descritas a seguir.

[3] Fundamentalmente a ideia de *scaffold* baseia-se na teoria de Vygotsky, sobre a Zona de Desenvolvimento Proximal. O objetivo é ajudar o aprendiz a fazer uma ponte entre o que ele sabe e o que ele pretende aprender, a partir de modelos e orientações. Quando a lacuna acaba, as orientações podem ser removidas (CEZAR *et al.*, 2006).

Figura 1. Processo de concepção do penC



- ***Reconhecimento do problema***

Esta etapa compreendeu a realização da revisão bibliográfica. Inicialmente a literatura sobre pensamento computacional na educação básica foi consultada, seguida da análise de estudos sobre autorregulação da aprendizagem em diferentes áreas. Nesse processo, relatórios de pesquisas, artigos, capítulos de livro, dissertações e teses foram analisados.

Além disso, um mapeamento sistemático das publicações do *Workshop on Self-Regulated Learning in Educational Technologies* (SRL@ET) foi realizado. Segundo Kitchenham *et al.* (2010), os mapeamentos sistemáticos representam uma forma particular de revisão sistemática e identificam e analisam, de maneira mais geral, a pesquisa sobre um tópico específico fornecendo evidências para questões mais amplas relacionadas a tendências de pesquisa. Quanto ao SRL@ET, ele vem se tornando um importante evento de discussão para a comunidade científica que pesquisa sobre metacognição e aprendizagem autorregulada em tecnologias educacionais. Desse modo, considerando que o mapeamento realizado não é exaustivo, optou-se pela análise da principal conferência da área no intuito de ter-se uma visão mais próxima da realidade.

A partir dos resultados obtidos, elegeram-se duas estratégias que apoiam a autorregulação: a autoavaliação e a avaliação por pares. Desse modo, a literatura sobre essas estratégias no contexto de programação para iniciantes foi analisada. Este estudo compreendeu outro mapeamento sistemático no contexto nacional e internacional o qual possibilitou identificar lacunas de pesquisa na área investigada.

- ***Proposição de solução***

A partir do estudo exploratório descrito na etapa anterior, um modelo conceitual para promoção da aprendizagem do pensamento computacional, especificamente introdução à programação, atrelado à prática de autorregulação foi definido.

- ***Avaliação da proposta***

Após a definição do modelo, um estudo piloto foi realizado com professores formados e atuantes em áreas como Educação em Computação, Educação em Programação, Pensamento Computacional, Educação Matemática, Informática na Educação, Psicologia Cognitiva e Interação Humano-Computador.

O objetivo dessa diversidade de perfis de participantes foi obter-se *insights* sobre os conceitos estruturantes do modelo, compreendidos em pensamento computacional na educação básica, programação para iniciantes, autorregulação da aprendizagem, metacognição, autoavaliação da aprendizagem e avaliação por pares.

Dessa forma, era esperado que cada participante fizesse a avaliação sob a ótica de sua formação e/ou atuação, integrando-se tais resultados ao final do processo e tendo-se uma avaliação sob a perspectiva de especialistas nas áreas contempladas no penC.

Tal estudo objetivou identificar os pontos fortes do modelo proposto, bem como os aspectos que requeriam melhorias. Como resultado, teve-se uma avaliação com observações categorizadas em Educação em Computação, *Design* de Interfaces e Metacognição.

- ***Refinamento da solução***

A partir dos resultados obtidos na etapa 3, o modelo foi refinado e um protótipo que incorpora as fases e atividades definidas foi implementado e analisado por meio de um quasi-experimento, que é descrito na seção a seguir.

3. Estudo Experimental

Buscando-se identificar a contribuição do modelo à autorregulação e à aprendizagem do pensamento computacional no ensino médio, um quasi-experimento (SHADISH *et al*., 2002) foi realizado em janeiro de 2015.

A abordagem *Goal/Question/Metric* (VAN SOLINGEN; BERGHOUT, 1999) foi utilizada como apoio à organização e à elaboração do experimento. Neste estudo, o objetivo foi analisar o modelo penC com o propósito de caracterizá-lo com respeito à contribuição na autorregulação e na aprendizagem do pensamento computacional do ponto de vista de estudantes no contexto de um curso de desenvolvimento de jogos digitais que introduziu conceitos de lógica de programação em iniciantes na área.

No estudo aqui apresentado, o perfil dos 22 participantes foi traçado, a população foi dividida em grupos controle e experimental, questionários foram aplicados e pré e pós-testes foram administrados. Adicionalmente, o professor do grupo experimental foi entrevistado visando-se identificar sua percepção sobre o modelo proposto e usado com sua turma. Ainda, os artefatos produzidos pelos estudantes, em formato de jogos, foram analisados sendo possível identificar o impacto da proposta sobre a aprendizagem do pensamento computacional, bem como sobre a jogabilidade dos *games* criados.

As duas turmas que formavam os grupos controle e experimental possuíam professores distintos. No entanto, de acordo com o levantamento de dados com os educadores, o planejamento pedagógico era definido conjuntamente. As aulas ministradas eram expositivas com a realização de exercícios práticos. Em tais exercícios, havia o desenvolvimento de jogos digitais com o uso do Stencyl (www.stencyl.com), um *software* que possibilita a criação de jogos para dispositivos móveis, web e desktop através do encaixe de blocos de comando, tendo o usuário que preocupar-se apenas com a lógica de funcionamento do projeto.

Antes da realização do quasi-experimento, componentes do Stencyl (atores, cenários, fontes e outros) haviam sido apresentados às turmas. Além disso, conceitos fundamentais de programação haviam sido ensinados como operadores lógicos, eventos e variáveis. Os estudantes eram avaliados ao fim dos exercícios propostos e realizados no curso, sendo cada professor responsável por corrigir as atividades de sua turma.

Na etapa de intervenção, ocorrida no encontro seguinte ao preenchimento dos questionários e à realização do pré-teste, os professores de ambas as turmas apresentaram Estruturas Condicionais aos estudantes seguido da proposição de exercícios, em formato de jogos, que exploravam o conteúdo abordado. Ao grupo controle, cabia responder os exercícios propostos no Stencyl e apresentar a solução para avaliação pelo professor. Ao grupo experimental, as atividades também foram realizadas em formato de jogos, no Stencyl, contudo a resolução e avaliação dos problemas seguiu as atividades do modelo penC, havendo momentos de reflexão antes e após o desenvolvimento dos jogos. Além disso, a avaliação da produção foi realizada pelos estudantes de acordo com critérios definidos previamente.

Considerando-se a importância da qualidade da interação do usuário com jogos digitais, heurísticas de jogabilidade foram apresentadas para ambos os grupos, incentivando que os estudantes atentassem para tais aspectos no momento da construção de seus jogos. Para o grupo controle, as heurísticas propostas por Barcelos *et al.* (2011) foram explanadas oralmente e entregues em formato impresso a cada um dos participantes. Para o grupo experimental, as heurísticas além de discutidas e entregues à turma, foram também utilizadas para avaliação de jogos produzidos pelos estudantes. Desse modo, a avaliação por pares contemplou critérios para análise de aspectos de programação, especificamente Estruturas Condicionais, bem como aqueles relacionados à jogabilidade.

No encontro seguinte, os estudantes de ambos os grupos responderam a um pós-teste que foi estruturado nos moldes do pré-teste, visando identificar o desempenho dos participantes no conteúdo Estruturas Condicionais. Assim como no pré-teste, na primeira questão do pós-teste era requisitado detectar

e corrigir erros em um jogo; na segunda era necessário analisar outro *game* e implementar as funcionalidades que estavam faltando; e a terceira questão requeria que os participantes usassem a criatividade para implementar um jogo. Novamente, para solucionar os problemas propostos os estudantes deveriam mobilizar seus conhecimentos sobre Estruturas Condicionais e codificar os jogos, arrastando blocos de comando no Stencyl.

A aprendizagem do pensamento computacional foi inferida a partir do desempenho dos estudantes no pré e pós-teste sobre o conteúdo Estruturas Condicionais. Três questões integraram cada teste e a pontuação máxima considerada para cada uma delas foi dez. Desse modo, a nota máxima que os estudantes poderiam obter em cada teste era trinta. A Tabela 1 apresenta as médias obtidas pelos participantes deste estudo em sua avaliação da aprendizagem. Como pode ser observado, o grupo experimental apresentava inicialmente uma média superior ao grupo controle. Após as intervenções didáticas em que o conteúdo Estruturas Condicionais foi apresentado, ambos os grupos demonstraram acréscimo de seu conhecimento sobre o assunto. Contudo, o grupo experimental demonstrou um ganho mais expressivo, se comparado ao controle.

Objetivando-se averiguar a significância estatística dos resultados apresentados, recorreu-se ao teste *t* de *Student* para as hipóteses $H0_1$ e $H1_1$. A hipótese nula afirma que os resultados obtidos pelo grupo experimental são iguais aos do grupo controle no que diz respeito à aprendizagem do pensamento computacional. Todavia, a hipótese alternativa declara que tais resultados são melhores.

Tabela 1. Resultado da aprendizagem do pensamento computacional

GRUPOS	PRÉ-TESTE		PÓS-TESTE	
	Média	Desvio Padrão	Média	Desvio Padrão
Grupo Experimental	13,75	7,71	24,17	3,42
Grupo Controle	3,93	6,43	7,86	10,04

Tendo em vista as médias obtidas pelos estudantes no pós-teste utilizou-se o teste *t* de *Student* sobre as duas amostras presumindo variâncias diferentes e nível de significância de 5%. O valor-P resultante é igual a 0,0038 ou 0,38%. Como este valor é inferior a 5%, a hipótese nula é rejeitada e há aceitação da hipótese alternativa. Isto implica dizer que existe relação entre a abordagem alternativa e a melhora da aprendizagem do pensamento computacional.

É importante ressaltar que estudantes dos grupos controle e experimental declararam ter cursado alguma disciplina introdutória de programação antes de ingressar no curso de *games* e isto pode ter impactado nos resultados obtidos. Nesse contexto, a pesquisa de Byrne e Lyons (2001) sugere que experiência anterior em programação pode favorecer o aprendizado na disciplina. Nesta pesquisa, a partir do questionário de perfil foi identificado que estudantes já haviam cursado alguma disciplina na área. Contudo, cinco de cada grupo afirmaram isto e, mesmo assim, o grupo experimental apresentou melhor desempenho, o que sugere que o penC tem relação com esse resultado e não apenas o conhecimento prévio em programação.

Os aspectos relativos à autorregulação foram inferidos pelas respostas dos estudantes a um questionário de estratégias de aprendizagem, o *Motivated Strategies for Learning Questionnaire – MSLQ* (PINTRICH *et al,* 1991). De modo complementar, as medidas da precisão e do viés no monitoramento do conhecimento, referenciadas como KMA (TOBIAS; EVERSON, 2002) e KMB (GAMA, 2004), respectivamente, foram consideradas.

Em relação à autorregulação, três escalas foram avaliadas: autorregulação metacognitiva, pensamento crítico e aprendizagem por pares. Todas elas foram beneficiadas após a intervenção didática com o modelo. Os resultados apontam que o valor-P é igual a 0,0428. Como este valor é menor que 5%, nível de significância adotado, a hipótese nula foi rejeitada e a hipótese alternativa foi aceita. Ou seja, há relação entre a abordagem de ensino com o uso do penC e as estratégias de aprendizagem utilizadas pelos estudantes durante sua formação em conceitos norteadores do pensamento computacional. Tal resultado pode estar atrelado às atividades de reflexão e avaliação do penC. Nas fases de pré e pós-reflexão objetivou-se envolver os estudantes em atividades metacognitivas, incentivando que eles pensassem sobre

o próprio conhecimento. Ainda, *scaffolds* foram projetados para dar suporte à resolução dos problemas propostos. Isto pode ter contribuído para a melhoria dos educandos na escala autorregulação metacognitiva.

Na fase de avaliação por pares houve também suporte à reflexão e era esperado que os estudantes, na função de revisor, avaliassem crítica e construtivamente a solução dos pares, evidenciando os pontos fortes, diagnosticando equívocos e sugerindo melhorias ao solucionador do problema. Com essa atividade, a reflexão sobre o próprio conhecimento a partir da revisão das soluções dos pares poderia aumentar a consciência da qualidade do próprio trabalho. Desse modo, a execução dessas tarefas pode ter contribuído para a melhoria dos estudantes nas escalas autorregulação metacognitiva e pensamento crítico.

Ao solucionador do problema havia a possibilidade refletir sobre os próprios erros e acertos, além de avaliar a qualidade dos comentários providos pelos pares. Esperava-se, com isso, colaborar com a capacidade de aceitar críticas à solução proposta e corrigir erros identificados. O estudante ainda poderia compartilhar com a turma a solução proposta a um determinado problema e as avaliações recebidas, abrindo espaço para discussão. Tais atividades podem ter contribuído com as escalas autorregulação metacognitiva e aprendizagem por pares.

Para complementar a análise da autorregulação da aprendizagem foram analisados o KMA e o KMB dos participantes do estudo. Notou-se no grupo experimental um ganho na precisão e no viés no monitoramento do conhecimento. Em suma, dois estudantes que eram otimistas passaram a ser realistas após a intervenção com o penC. Por outro lado, após a intervenção com a abordagem tradicional, o grupo controle teve o número de realistas reduzido. Este resultado pode ser atribuído às constantes atividades de reflexão definidas no penC que podem ter aumentado a consciência sobre o próprio conhecimento, possibilitando ao grupo experimental julgar com maior precisão a sua aprendizagem.

Não era objeto desta pesquisa abordar conceitos associados ao *design* de artefatos digitais construídos pelos estudantes. Contudo, tendo em vista o contexto em que o estudo experimental foi realizado, foi verificada a necessidade de contribuir com a formação dos participantes nesse aspecto. Desse modo, as heurísticas de jogabilidade propostas por Barcelos *et al.* (2011) foram utilizadas para avaliar os jogos produzidos pelos estudantes no pré e pós-teste. Tais heurísticas foram também utilizadas na análise, pelos estudantes, dos *games* produzidos pelos pares. Os resultados obtidos apontam para uma melhoria da qualidade dos jogos construídos pelo grupo experimental, algo que pode ser atribuído à estratégia de avaliação por pares utilizada durante a intervenção.

Como forma de identificar as estratégias que os participantes tiveram consciência utilizar no desenvolvimento de jogos digitais, foram coletados seus autorrelatos, tanto no pré, como no pós-teste. Nesta pesquisa, os autorrelatos foram analisados e pôde-se identificar dois tipos de estratégias: programação e jogabilidade. O grupo experimental demonstrou ser mais consciente no uso de suas estratégias, se comparado ao grupo controle. Contudo, considera-se importante informar que nem todos responderam ao questionamento. Assim, apesar do uso de narrativas reflexivas poder capturar informações relevantes quanto às estratégias de aprendizagem utilizadas pelos estudantes, sugere-se que elas sejam complementadas com outro instrumento de coleta de modo a conseguir captar as percepções de todos os envolvidos no estudo.

Para ter-se um quadro ainda mais completo sobre os resultados alcançados com o quasi-experimento, uma entrevista foi realizada com o professor do grupo experimental. Em suma, pôde-se identificar que o penC foi satisfatoriamente avaliado pelo docente, aspecto considerado positivo neste estudo, uma vez que sugere que o modelo proposto tem aplicação prática na sala de aula, bem como potencial de promover a aprendizagem na visão do educador.

4. Considerações finais

A lista de habilidades e conhecimentos necessários para o pleno exercício da cidadania no século XXI é extensa, incluindo o pensamento computacional. Ele é necessário por permitir que os estudantes melhor conceituem, analisem e resolvam problemas complexos e pode ser aplicado às diversas áreas do saber. Por outro lado, a educação em Computação é considerada um desafio, especialmente o ensino introdutório de programação. Considerando o exposto, esta pesquisa investigou como o ensino do pensamento

computacional, em especial a programação, pode ser promovido no ensino médio pela prática de autorregulação apoiada por tecnologia.

Do ponto de vista da Computação, este trabalho contribui para um dos desafios científicos apontados pela SBC para o decênio 2006-2016, pela concepção de um modelo para aprendizagem do pensamento computacional. Com a proposta apresentada neste trabalho, espera-se criar condições adequadas para promover a aprendizagem do pensamento computacional na educação básica brasileira, uma realidade em países desenvolvidos. Sob a ótica da Educação, espera-se ter corroborado com o alcance de metas traçadas no PNE para o decênio 2014-2024 no que se refere, especialmente, ao desenvolvimento de tecnologias que favoreçam a aprendizagem de estudantes da educação básica.

Como trabalhos futuros pretende-se replicar o experimento realizado em outras turmas de lógica de programação, com um número maior de iterações e participantes e envolvendo vários conteúdos da disciplina. Ainda almeja-se avaliar a usabilidade do penC+, protótipo que implementa o penC, e corrigir possíveis erros identificados. Também, planeja-se analisar longitudinalmente *logs* de interação de estudantes com o penC+ visando-se ter uma melhor compreensão do desenvolvimento do pensamento computacional e da autorregulação da aprendizagem pelo envolvimento desses educandos nas atividades de autoavaliação e avaliação por pares.

5. Agradecimentos

Rozelma Soares de França agradece à CAPES pela concessão de bolsa de mestrado no período de realização desta pesquisa.

Publicações desta pesquisa

FRANÇA, R. S.; AURELIANO, V. O.; TEDESCO, P. C. A. R. Autorregulação da aprendizagem em tecnologias educacionais: tendências e oportunidades. No prelo.

FRANÇA, R. S.; TEDESCO, P. C. A. R. Avaliação por pares na aprendizagem de programação para iniciantes: possibilidades e desafios. No prelo.

FRANÇA, R. S.; TEDESCO, P. C. A. R. Caracterizando a pesquisa sobre autoavaliação na aprendizagem de programação para iniciantes. In: **Anais do XXVI Simpósio Brasileiro de Informática na Educação**. 2015.

FRANÇA, R. S.; TEDESCO, P. C. A. R. Collaborative learning model to support the self-regulated learning of computational thinking. No prelo.

FRANÇA, R. S.; TEDESCO, P. C. A. R. Explorando o pensamento computacional no ensino médio: do design à avaliação de jogos digitais. In: **Anais do XXIII Workshop sobre Educação em Computação**. 2015.

FRANÇA, R. S.; TEDESCO, P. C. A. R. Percepções de professores sobre um modelo para a aprendizagem do pensamento computacional. No prelo.

FRANÇA, R. S.; TEDESCO, P. C. A. R. Um modelo colaborativo para a aprendizagem do pensamento computacional aliado à autorregulação. In: **Anais do XXV Simpósio Brasileiro de Informática na Educação**. 2014. p. 1133-1142.

Referências

ANDRADE, Daiane *et al*. Proposta de Atividades para o Desenvolvimento do Pensamento Computacional no Ensino Fundamental. In: **Anais do XVI Workshop de Informática na Escola**. 2013.

BARCELOS, Thiago Schumacher *et al*. Análise comparativa de heurísticas para avaliação de jogos digitais. In: **Proceedings of the 10th Brazilian Symposium on Human Factors in Computing Systems and the 5th Latin American Conference on Human-Computer Interaction**. 2011. p. 187-196.

BRASIL. MEC - Ministério da Educação. **Plano Nacional de Educação**. Disponível em: <http://bit.ly/1q8Cjlj>. Acesso em: 10 de nov. 2014.

BUNDY, Alan. Computational thinking is pervasive. **Journal of Scientific and Practical Computing**, v. 1, n. 2, p. 67-69, 2007.

CARVALHO, M. L. B.; CHAIMOWICZ, L.; MORO, M.. M. Pensamento Computacional no Ensino Médio Mineiro. In: **Anais do XXI Workshop sobre Educação em Computação**. 2013.p. 640-649.

CEZAR, E. S.; SANTOS, N.; PRATES, R. O. O Uso de Scaffolds no Projeto de Software Educacional. **Cadernos do IME - Série Informática**, v. 22, p. 26-32, 2006.

COUNCIL, E. U. Council resolution of 27 June 2002 on lifelong learning.**Official Journal of the European Communities**, v. 9, 2002.

CSTA - Computer Science Teacher Association. **CSTA K-12 Computer Science Standards**. CSTA Standards Task Force. ACM - Association for Computing Machinery, 2011.

CSTA - Computer Science Teacher Association. **The New Educational Imperative: Improving High School Computer Science Education. Final Report of the CSTA**. Curriculum Improvement Task Force. ACM - Association for Computing Machinery, 2005.

DE SOUZA, Clarisse S. *et al*. Cultural appropriation of computational thinking acquisition research: seeding fields of diversity. In: **Proceedings of the 19th Conference on Innovation & Technology in Computer Science Education**. ACM, 2014. p. 117-122.

DELORS, J. *et al.* Educação: um tesouro a descobrir. **Relatório para a UNESCO da Comissão Internacional sobre Educação para o século XXI.** São Paulo: Cortez, 1996.

FRANÇA, R. S; SILVA, W. C.; AMARAL, H. J. C. do. Ensino de ciência da computação na educação básica: Experiências, desafios e possibilidades. In: **XX Workshop sobre Educação em Computação**. 2012.

GAMA, C. A. . **Integrating metacognition instruction in interactive learning environments**. 2004. Tese de Doutorado. University of Sussex.

KITCHENHAM, B. A.; BUDGEN, D.; BRERETON, P. The value of mapping studies: a participantobserver case study. In: **Proceedings of the 14th International Conference on Evaluation and Assessment in Software Engineering**. British Computer Society, 2010. p. 25-33.

LAHTINEN, E.; ALA-MUTKA, K.; JÄRVINEN, H.. A study of the difficulties of novice programmers. In: **Proceedings of the 10th Conference on Innovation & Technology in Computer Science Education**. 2005. p. 14-18.

LONG, Y.; ALEVEN, V.. Active Learners: Redesigning an Intelligent Tutoring System to Support Self-regulated Learning. In: **Scaling up Learning for Sustained Impact**. Springer Berlin Heidelberg, 2013. p. 490-495.

LYE, S. Y.; KOH, J. H. L.. Review on teaching and learning of computational thinking through programming: What is next for K-12?.**Computers in Human Behavior**, v. 41, p. 51-61, 2014.

PINTRICH, Paul R. *et al*. **A manual for the use of the Motivated Strategies for Learning Questionnaire (MSLQ)**. 1991.

SBC – Sociedade Brasileira de Computação. **Grandes Desafios da Pesquisa em Computação no Brasil – 2006 – 2016.** Disponível em: <http://bit.ly/1Ev25Ya>. Acesso em: 10 de nov. 2014.

SCAICO, P. D. *et al*. Ensino de Programação no Ensino Médio: Uma Abordagem Orientada ao Design com a linguagem Scratch. **Revista Brasileira de Informática na Educação**, v. 21, n. 02, p. 92, 2013.

SHADISH , W. R.; COOK, T. D.; CAMPBELL, D. T. **Experimental and quasi-experimental designs for generalized causal inference**. Boston: Houghton Mifflin Company, 2002.

SITTHIWORACHART, J.; JOY, M.. Effective peer assessment for learning computer programming. In: **Proceedings of the ACM Technical Symposium on Computer Science Education**. ACM, 2004. p. 122-126.

SOUSA, R. V.; *et al*.. Ensinando e aprendendo conceitos sobre a ciência da computação sem o uso do computador: Computação Unplugged!. **Práticas em Informática na Educação: Minicursos do CBIE**, v.1, n.1, 2010.

TOBIAS, S.; EVERSON, H. T. **Knowing what you know and what you don't: Further research on metacognitive knowledge monitoring**. 2002.

VAN SOLINGEN, R.; BERGHOUT, E.. **The Goal/Question/Metric Method: a practical guide for quality improvement of software development**. London: McGraw-Hill, 1999.

VIEIRA, A.; PASSOS, O.; BARRETO, R.. Um Relato de Experiência do Uso da Técnica Computação Desplugada. In: Anais **do XXI Workshop sobre Educação em Computação**. 2013. p. 670-679.

WING, Jeannette M. Computational thinking. **Communications of the ACM**, v. 49, n. 3, p. 33-35, 2006.

ZIMMERMAN, B. J.. Becoming a self - regulated learner: An overview. **Theory into practice**, v. 41, n. 2, p. 64-70, 2002.