Ambiente de programação para a introdução da lógica de programação

Mauro M. Mattos, Luciana P. de Araújo Kohler, Fabrícia D. Zucco, Leonardo Fronza, Artur Bizon, Heitor Ugarte, Bruno F. F. Santos, Gian C. Giovanella

[1]Laboratório de Desenvolvimento e Transferência de Tecnologias (LDTT)
Universidade Regional de Blumenau (FURB)
Blumenau – SC – Brasil.

{mattos, lpa, fabricia, leofronza}@furb.br

{abizon, hucsilveira, bffsantos, gianggiovanella}@furb.br

Abstract. *SBC with the support of other institutions such as the Centro de Inovação para a Educação Brasileira has been creating guidelines and curriculum that include teaching programming in elementary and secondary schools. It occurs that several of the tools available for teaching programming are complex or lack the features of a programming environment, omitting some learning issues. This paper presents a programming environment that can be used to introduce the programming logic, having the features so that the student understands what a programming language is. The paper demonstrates the environment containing these resources, exemplifying their use from a study with elementary school children.*

Resumo. *A SBC com apoio de outras instituições como o Centro de Inovação para a Educação Brasileira, vem criando diretrizes e currículos que incluem o ensino da programação no ensino fundamental e médio. Ocorre que, várias das ferramentas disponíveis para o ensino de programação são complexas ou não possuem as características de um ambiente de programação, omitindo algumas questões de aprendizagem. Apresenta-se um ambiente de programação que pode ser utilizado para a introdução da lógica de programação, tendo as características necessárias para que o aluno compreenda o que é uma linguagem de programação. O artigo demonstra o ambiente contendo esses recursos, exemplifica seu uso a partir de um estudo com crianças de ensino fundamental.*

1. Introdução

Nos últimos anos, a Sociedade Brasileira de Computação (SBC) definiu diretrizes para o ensino da computação na Educação Básica [SBC 2019]. A partir dessas diretrizes, o Centro de Inovação para a Educação Brasileira (CIEB) formalizou currículos que podem ser aplicados na educação infantil, ensino fundamental e ensino médio [CIEB 2018].

Dentre um dos conceitos se destaca a construção de algoritmos e a identificação e categorização de elementos que compõem a interface de um ambiente de programação [CIEB 2018]. Para a aplicação desses conteúdos em sala de aula é necessária a existência de ferramentas que suportam esses conceitos e que sejam voltados ao público alvo em questão. Com essa finalidade, adaptou-se uma plataforma com aspectos de gamificação desenvolvida para introduzir o pensamento computacional em crianças de ensino

DOI: 10.5753/cbie.wie.2019.1259

fundamental para atender também os conceitos que compõe a interface de um ambiente de desenvolvimento integrado (IDE).

Considera-se que uma IDE deve conter no mínimo um editor de texto no qual é inserido o código-fonte, e, por meio de seu próprio ambiente, este código é executado. Ainda, uma IDE possui um depurador de código, indicação de erros sintáticos e semânticos e um interpretador ou compilador de código integrado, responsáveis por traduzir o texto escrito e permitir a execução do programa. Dessa forma, tem-se como objetivo disponibilizar um ambiente que ensine a programação para crianças, caracterizando a interface de programação com os elementos de uma IDE.

2. Ferramenta Desenvolvida

Com o objetivo de disponibilizar um ambiente que ensine a programação para crianças, caracterizando a interface de programação com os elementos de uma IDE, adaptou-se uma ferramenta já desenvolvida e testada com crianças [Araújo et al. 2018] para contemplar as características de uma IDE, sendo: compilador próprio para permitir a execução no ambiente; analisador léxico, sintático e semântico de modo a identificar e apresentar ao usuário os erros de codificação cometidos; e um protótipo de um depurador, de modo a apresentar a linha que está em execução no determinado momento.

O conceito da ferramenta em questão existe há 10 anos, sendo um *framework* utilizado para introduzir conceitos de programação para os alunos das fases iniciais dos cursos de Ciência da Computação e Sistemas de Informação. A partir do *framework*, nos últimos dois anos foi desenvolvida uma plataforma para o ensino do pensamento computacional para ser aplicada com crianças das escolas de ensino fundamental entre 1º e 5º ano. A partir desta plataforma e testes previamente realizados, foram desenvolvidos os ajustes relatados neste artigo.

A ferramenta apresenta um robô denominado Furbot como personagem principal da história o qual possui missões diferenciadas durante cada cena do jogo. O jogo possui um enredo para contextualizar o jogador. O enredo conta que a Terra foi invadida por alienígenas sendo que estes deixaram pistas no planeta. A missão do robô é salvar o planeta da invasão alienígena, coletando as pistas deixadas por eles pelo caminho.

Para que o Furbot se movimente pelo mapa, ele deve ser programado a partir dos comandos de programação "`andar(DIRECAO);`", sendo que a direção pode ser direita, esquerda, acima ou abaixo. Os comandos são executados de forma sequencial, sendo adicionados um abaixo do outro em um editor de texto disponível. Ainda, pode-se utilizar o laço de repetição enquanto, com condicionais que indicam fatores identificados no caminho, como o tipo do caminho ou se está vazio em alguma das direções.

A ferramenta desenvolvida atualmente em Unity na linguagem C#, já teve uma versão em Java com as mesmas funcionalidades. Para a análise do código, a ferramenta adota os princípios de um compilador (analisador léxico, analisador sintático, analisador semântico e gerador de código intermediário) para então realizar a execução do código-fonte no próprio ambiente. Nessa análise são identificados se todos os comandos escritos no editor são comandos válidos para o programa, sendo esta a análise léxica. Em seguida, analisa-se se os comandos estão descritos na ordem correta seguindo a sintaxe definida pela linguagem, sendo esta a análise sintática. Por fim, em caso de existir comando de

repetição, analisa-se se a condição codificada representa uma condição *booleana* sendo necessária para a execução do laço. Caso todas as análises retornem sucesso, o código-fonte é traduzido para a linguagem da ferramenta, neste caso C#, e então, o Furbot realiza o percurso no mapa, conforme codificado pelo programa. Na Figura 1 tem-se a visão geral do jogo.

Figura 1. Visão geral do jogo

Como pode-se observar na Figura 1, o jogo possui um cenário composto inicialmente por um robô (o Furbot), um drone (a S-223) e um caminho a ser percorrido. Em cada uma das fases, o jogador precisa programar o robô para percorrer o caminho, desviando de obstáculos (como árvores, arbustos e pedras) e capturar os objetivos do mapa (como tesouros, marcas deixadas pelos alienígenas, vidas, energias, entre outros). Neste mapa, o robô deve sair de onde se encontra e atingir a última posição do caminho localizado na direita do mapa, no qual possui uma marca deixada por um alienígena. Ainda, o jogador deve cuidar para andar somente por cima do caminho, pois se andar pelo gramado ele perderá mais energia, podendo não conseguir completar a fase. A S-223 é a ajudante do Furbot e fornece dicas para ele durante todo o jogo.

Neste exemplo da Figura 1, tem-se no lado direito o editor de texto com um exemplo de código-fonte produzido pelo jogador. Na parte inferior, tem-se a S-223 que apresenta a mensagem de erro conforme o compilador identifica. São apresentados erros tanto léxicos quanto sintáticos identificados pelo compilador.

Com o código-fonte correto, ao ser executado pela ferramenta o Furbot começa a se movimentar no cenário. Para que o jogador visualize qual linha de código é executada para cada movimento, a mesma fica em negrito conforme o Furbot se movimenta. Essa situação pode ser observada na Linha 6 da Figura 2. Dessa forma, tem-se um princípio de um depurador de código-fonte.

Quando o Furbot atingir o objetivo final da fase, que neste caso é coletar a pista deixada pelo alienígena, a fase é encerrada, mostra-se a pontuação adquirida e o jogador é encaminhado para a próxima fase. Todas as fases são constituídas por cenários que contém objetivos a serem desviados e coletados e devem ser programados para que o robô realize a ação.

3. Testes, Resultados e Discussões

Para testar o ambiente desenvolvido, foram realizadas oficinas em escolas de ensino fundamental do 1º ao 5º ano e em grupos com crianças entre 7 e 12 anos durante os últimos 2 anos (2017 e 2018). Ao todo, aproximadamente 205 crianças já utilizaram a plataforma

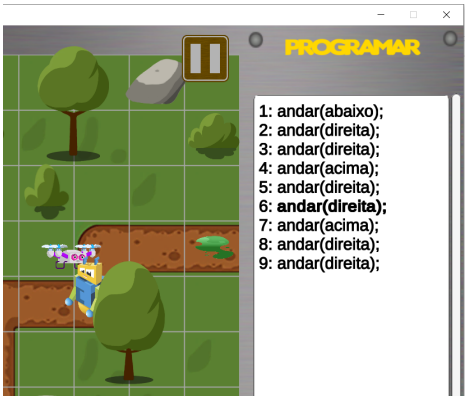

Figura 2. Princípio de depurador de código-fonte

em mais de 90 oficinas realizadas e alguns dos estudos de caso realizados são relatados em [Araújo et al. 2018].

As oficinas aconteceram nas escolas estaduais Vitor Hering e Pedro II tendo duração de duas horas aula cada e aconteceram uma vez por semana, durante os últimos dois anos, totalizando as 90 oficinas. Elas ocorriam durante o período da aula, sendo substituídas por alguma matéria letiva. Desse modo, a participação dos alunos era obrigatória nas oficinas, contando como carga horária curricular. Ainda, a professora pedagoga responsável pela turma acompanhava os instrutores da oficina, de modo a unir conteúdos de sala de aula com os conteúdos relacionados à oficina. Realizou-se oficinas com turmas de 1º ao 5º ano do ensino fundamental, sendo que cada turma recebia uma oficina individualmente.

Para avaliar se as características de uma IDE facilitaram a programação dessas crianças, utilizou-se da observação direta e de anotações realizadas ao longo dessas oficinas. Nos primeiros contatos com a ferramenta, as crianças demonstraram dificuldades em conseguir arrumar seu código-fonte quando o mesmo não executava. Dessa forma, elas eram indicadas a ler a mensagem de erro apresentada na interface gráfica e então auxiliadas a arrumar o erro em questão. Após duas ou três oficinas, as crianças já conseguiam sozinhas arrumar seus erros quando esses se referiam a erros léxicos, sintáticos ou semânticos. Elas só precisavam ser auxiliadas em relação a erros de lógica de programação, conceito este que estava sendo desenvolvido durante as oficinas. Ainda, quando uma criança chamava para questionar a respeito de um erro sintático, ouvia-se outra criança ajudando e comentando "é só ler ali embaixo". Esse comentário era frequente, demonstrando que a mensagem surtiu efeito e ajudou as crianças a identificarem sozinhas seus erros de programação.

Em relação ao princípio do depurador apresentado, considera-se que ele ajudou a identificar os erros lógicos gerados. Por exemplo, as vezes a criança colocava duas vezes o comando `andar(direita);` e então o Furbot batia na parede. Ao ver a execução sequencial e em negrito, ela conseguia perceber qual linha estava a mais. O mesmo aconteceu para linhas ausentes, de modo a facilitar a manutenção do código escrito.

Em relação aos currículos propostos pelo CIEB, a ferramenta em questão auxilia na aquisição das habilidades relacionadas ao pensamento computacional a ser trabalho com o 1º ano do ensino fundamental. Sabendo-se que o pensamento computacional

dividi-se em abstração, algoritmos, decomposição e reconhecimento de padrões utilizou-se dessas habilidades indicadas para realizar a correlação com a ferramenta apresentada.

A primeira habilidade diz respeito a abstração sendo "compreender que os computadores não têm inteligência e apenas realizam o que é programado". A partir do código-fonte que o jogador precisa produzir para movimentar o robô, ele tem a noção de que o robô se movimentará somente para os lados que indicar na programação. Dessa forma, compreende-se que ele perceberá que o computador só realizará o que for programado.

Em relação a segunda habilidade referente a algoritmos, tem-se "compreender o conceito de algoritmo como uma sequência de passos ou instruções, por meio de símbolos, sinais ou imagens, que pode ser executada e verificada por meio da depuração". A partir das atualizações dessa ferramenta, com o depurador acoplado, esta habilidade também é atendida. Pode-se afirmar isso, pois com a demonstração da execução sequencial (sendo que a linha em execução fica em negrito) o aluno percebe os passos que são executados pelo robô e identifica quando um trecho de código-fonte está equivocado.

Em relação ao terceiro fator, sendo a decomposição, tem-se a habilidade de "exercitar a decomposição, por meio da quebra de atividades rotineiras em diversos passos ou instruções". Essa habilidade é atendida pelos comandos `andar(direção);` que quebram a rotina de andar em diversos passos sequenciais. Ao invés do aluno programar que o robô deve andar até o final do caminho, ele precisa programar a quantidade de instruções.

Por fim, em relação ao reconhecimento de padrões tem-se a habilidade de "identificar que todos os softwares são programados". Novamente, a partir da programação sequencial que permite o robô andar pelo cenário, o aluno tem esta compreensão.

Como a ferramenta ainda está em desenvolvimento, as habilidades relacionadas entre o 2º e 5º ano do ensino fundamental ainda não foram abordadas. Um dos motivos é que, embora as crianças de 2º ao 5º ano já trabalharam com a ferramenta, como elas não tiveram o contato anterior, elas precisam primeiro desenvolver as habilidades do 1º ano para então poder evoluir para as próximas habilidades. Dessa forma, pretende-se incluir as novas habilidades nas próximas versões da plataforma.

Referências

Araújo, L., da Silveira, H. U. C., e Mattos, M. (2018). Ensino do pensamento computacional em escola pública por meio de uma plataforma lúdica. In *Anais dos Workshops do VII Congresso Brasileiro de Informática na Educação (CBIE 2018)*. Brazilian Computer Society (Sociedade Brasileira de Computação - SBC).

CIEB (2018). Currículo de referência em tecnologia e computação. http://curriculo.cieb.net.br/curriculo, Julho.

SBC (2019). Diretrizes para ensino de computação na educação básica. http://www.sbc.org.br/educacao/diretrizes-para-ensino-de-computacao-na-educacao-basica, Julho.