

URI Online Judge Blocks: Construindo Soluções em uma Plataforma Online de Programação

Jessica Dagostini¹, Marcos Lima¹, Jean Luca Bez², Neilor Tonin¹

¹Universidade Regional Integrada do Alto Uruguai e das Missões - URI Erechim

²Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)

{jessica, marcos, bez, neilor}@urionlinejudge.com.br

Abstract. *One of the main challenges while teaching algorithms and data structures is the transition from the abstract logic of the algorithm, which a student understands, to a programming language, that a computer can understand. This change in paradigm can be troublesome when coding an algorithm for the first time. Many alternatives provide different ways to smooth this process. However, most of these tools focus mainly on the logic of the program and the concept of the algorithm. The goal of this paper is to present a work that aims at providing a tool to translate the logic of an algorithm into an implementation, allowing a smooth transition between paradigms. To evaluate our proposal, we integrated the Blockly API from Google to the URI Online Judge platform. Therefore, we provide a valuable asset to the student, by making the learning to code into a more dynamic, visual, and interactive process.*

Resumo. *Uma das principais dificuldades no ensino de algoritmos e estrutura de dados é a tradução da lógica abstrata do algoritmo, que o estudante compreende, para uma linguagem de programação, que o computador possa entender. Essa mudança de paradigma pode trazer certas dificuldades durante as primeiras implementações. Existem diversas alternativas para facilitar e intermediar esse processo. No entanto, a maioria delas foca na lógica e na concepção do algoritmo. O objetivo deste trabalho é apresentar uma nova opção para esse intermédio, através de uma ferramenta que traduz essa lógica em código, tornando essa mudança de paradigma mais gradual. Para isso, nós integramos a API Blockly do Google à plataforma URI Online Judge. Desta forma propomos uma valiosa ferramenta para o aluno, ajudando a tornar o processo de aprendizagem de programação mais dinâmico, gráfico e intuitivo.*

1. Introdução

Dentro dos cursos que possuem em seus currículos as disciplinas de Algoritmos e Lógica de Programação, alguns alunos apresentam dificuldades em compreender esses tópicos [Hoed 2017]. Um dos pontos que podem estar relacionados à origem dessas dificuldades é a metodologia de ensino empregada pelo professor. De acordo com [Koliver et al. 2004], não existe uma metodologia única e consolidada para o ensino de algoritmos, mas diversas abordagens que possuem pontos fortes e fracos.

É comum que seja necessário praticar e aplicar os conteúdos vistos na sala de aula para uma compreensão completa. “Programação é uma habilidade adquirida através da

prática” [Kurnia et al. 2001]. Muitos professores adotam essa abordagem, orientando aos estudantes fazerem exercícios de programação dentro de um ambiente de juiz online. Estas plataformas normalmente fornecem um repositório de exercícios, onde o aluno poderá resolver problemas em uma linguagem de programação suportada pela ferramenta. Quando o aluno envia sua solução, a ferramenta automaticamente retorna um *feedback*, informando se a solução do estudante está correta ou incorreta [Kurnia et al. 2001].

Embora esta metodologia seja eficaz com alguns estudantes, ainda é possível detectar dificuldades. Como as soluções precisam estar em uma linguagem de programação, o estudante não precisa apenas aprender os conceitos de algoritmos e a lógica de programação, mas também uma nova linguagem e sua estrutura. Um grande número de estudantes apresentam dificuldades ao transpor uma ideia na forma de um programa. Essa súbita mudança de modelo pode levar à frustração na compreensão do que deve ser feito, e em como isso deve ser traduzido em uma linguagem entendida pela máquina. [Winslow 1996]

Nesse sentido, o objetivo deste trabalho é apresentar uma ferramenta que contribua para o aprendizado de algoritmos, fazendo uso de um ambiente de programação visual com o intuito de atenuar a transição entre modelos. Para isso, integramos a biblioteca do *Google Blockly* na plataforma URI Online Judge. Dessa forma, esperamos fornecer um primeiro contato mais simples e intuitivo com a lógica de programação e codificação, permitindo que o estudante inicialmente concentre seus esforços na lógica de programação, mas mantendo contato com o código, mudando gradualmente de modelo.

Este trabalho encontra-se dividido em seis seções. A Seção 2 detalha o fundamento teórico do trabalho proposto, discutindo alguns conceitos relacionados ao aprendizado e algumas ferramentas empregadas em nossa solução proposta. A Seção 3 descreve os trabalhos relacionados comparando com a nossa abordagem. A Seção 4 apresenta o URI Online Judge Blocks, com algumas especificações técnicas e sua integração à plataforma principal. A Seção 5 discute alguns dos resultados coletados até o momento da escrita deste artigo e nossas primeiras impressões. Finalmente, a Seção 6 conclui este artigo e apresenta os trabalhos futuros.

2. Fundamentação Teórica

Em linguagens de programação, assim como em linguagens naturais, significados são associados a um universo de signos pelos quais nós podemos agir. A compreensão de uma linguagem é um dos mais significantes momentos no curso do desenvolvimento cognitivo [de La Taille et al. 1992]. Para estudantes da área de Ciência da Computação, essa compreensão é essencial, uma vez que é a única forma de “comunicação” entre o programador e a máquina. A incompreensão da lógica de programação, por muitas vezes, acaba gerando a incapacidade de transcrever um algoritmo para uma linguagem computacionalmente compreensível [Iepsen et al. 2010]. Essas dificuldades podem ser consideradas como fatores contribuintes nas desistências dentro de cursos relacionados a área da computação [Hoed 2017]. Lidar com esses fatores e facilitar a compreensão destas linguagens é um dos maiores desafios encontrados pelos professores.

Conforme comentado na Seção 1, muitos professores de Ciência da Computação utilizam juízes online em suas aulas. Um dos principais benefícios ao utilizar estas plataformas é que as mesmas já possuem um vasto repositório de exercícios, além de

permitirem uma correção em tempo real, dando ao estudante um *feedback* rápido sobre sua solução e tornando o aprendizado mais responsivo. Baseados neste fato, propomos a adição, dentro de um juiz online, de um ambiente que possa auxiliar na introdução de conceitos de programação. Para isso, ao pesquisar por soluções que pudessem se adequar as nossas necessidades, nos deparamos com uma biblioteca que permite programação visual ao mesmo tempo que traduz essa lógica em código compreensível pela máquina. Sendo assim, poderíamos disponibilizar uma ferramenta simples que permita ao estudante focar na compreensão dos conceitos iniciais de lógica de programação, sem ignorar a codificação da solução, possibilitando uma transição gradual entre modelos.

A seguir, descreveremos as duas ferramentas que utilizamos neste trabalho. No item 2.1 o juiz online escolhido será apresentado, sendo mostrado alguns conceitos importantes relacionados ao seu funcionamento. No item 2.2, iremos rapidamente introduzir a biblioteca Google Blockly e nosso motivo para usá-la.

2.1. URI Online Judge

É muito importante que o estudante tenha um *feedback* rápido e correto quanto a abordagem por ele utilizada para montar a solução de um problema. Todavia, por possuir um número considerável de estudantes em uma sala de aula, é difícil para o professor dedicar uma atenção especial a cada um deles. Realizar uma correção adequada para cada uma das soluções criadas por cada um de seus estudantes torna o processo ainda mais difícil e trabalhoso.

Com o intuito de conseguir auxílio em relação a estes pontos apresentados, os professores buscam por uma forma automatizada de corrigir as soluções de seus estudantes. Um juiz online é um sistema que realiza essa correção. Estes normalmente possuem um repositório de problemas e, para cada problema, uma série de casos de testes que determinarão a corretude do programa submetido para julgamento [Kurnia et al. 2001].

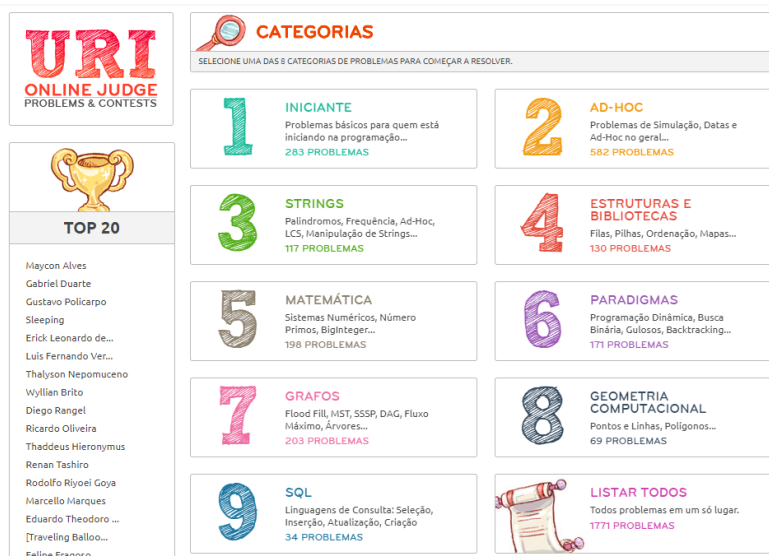


Figura 1. As principais categorias de problemas do URI

O URI Online Judge é uma plataforma web de programação que contém problemas no estilo da *Association for Computing Machinery - International Collegiate Programming*

Contest (ACM-ICPC) [ACM 2018]. A plataforma está disponível gratuitamente em urionlinejudge.com.br. Ela aceita submissões em diversas linguagens de programação, como C, C#, C++, Java, JavaScript, Kotlin, Lua, Python, Ruby e Scala. A ferramenta oferece um serviço gratuito de correção de códigos-fonte de forma automática e em tempo real. Seu repositório de problemas está dividido em nove categorias, cobrindo uma diversidade de subtópicos, conforme ilustrado pela Figura 1.

A primeira categoria, denominada “Iniciante”, contém problemas para aqueles que estão iniciando na programação, abrangendo a maioria dos conceitos básicos relacionados a algoritmos e estrutura de dados (como condicionais e laços de repetição). Neste trabalho, a API Blockly (detalhada na Seção 2.2) foi integrada a um grande número de problemas desta categoria para promover uma forma mais interativa de construir as soluções.

O URI possui uma diversidade de módulos integrados ao seu sistema principal de julgamento, e um destes é o módulo Academic . O URI Online Judge Academic é uma ferramenta especialmente desenvolvida para professores e treinadores de equipes – no contexto do ICPC – que permite a eles facilmente criar disciplinas e listas de exercícios a partir do repositório de problemas do URI. Esta ferramenta também torna o acompanhamento do progresso de cada estudante mais fácil, permitindo que o professor atue como um mediador ativo do conhecimento.

2.2. Google Blockly

O Google Blockly é uma biblioteca de programação de código aberto que usa programação visual, permitindo que seus usuários construam códigos através da conexão de blocos, sem a necessidade de uma sintaxe de linguagem. Esta ferramenta também possui uma bem documentada API (*Application Programming Interface*, Interface de Programação de Aplicativos) que permite aos desenvolvedores facilmente integrá-la a diversas aplicações.

O Blockly é utilizado em algumas aplicações educacionais, sendo sua maioria focada para o ensino de Ciência da Computação para a Educação Básica (como descrito na Seção 3). Seu principal objetivo nesse cenário é auxiliar no processo de ensino-aprendizagem de lógica de programação, diferenciando-se de outras aplicações por promover uma interface intuitiva, além de transformar os blocos conectados em códigos-fonte funcionais nas linguagens Dart, JavaScript, Lua, PHP e Python. Dessa forma, essa ferramenta auxilia novos programadores e também aqueles que procuram por uma implementação mais visual [Marron et al. 2012]. Entretanto, existem outros trabalhos que já implementam suporte a outras linguagens de programação, como C/C++, por exemplo em [Carlos et al. 2016].

A biblioteca Blockly foi desenvolvida para ser facilmente integrada do mais simples ao mais complexo sistema e para ser estendida e customizada de acordo com as particularidades de cada aplicação. Nós nos beneficiamos desses dois pontos ao integrar ela ao URI Online Judge.

3. Trabalhos Relacionados

O Google Blockly é integrado com algumas outras ferramentas, como o Blockly Games [Google 2015], App Inventor [Wolber 2011] e o Block.ino [Carlos et al. 2016]. Estas tem como objetivo geral auxiliar na construção de aplicações de uma forma mais fácil e intuitiva. O Blockly Games tem como público principal crianças que nunca tiveram

contato com programação. Essa ferramenta integra os blocos em forma de jogos, divididos em atividades que simulam e estimulam o pensamento lógico. Esta abordagem é útil quando aplicada no ensino primário, mas acaba não sendo a mais indicada para a graduação. Nosso trabalho busca integrar os blocos em uma aplicação de mais alto nível ao gerar e mostrar os códigos-fonte, assim melhor auxiliando novos estudantes de graduação no entendimento inicial da lógica de programação e como isso se traduz em código-fonte.

O APP Inventor tem como público principal aqueles que estão iniciando na programação, mas focados no desenvolvimento de aplicativos *mobile*. Ele integra o Blockly ao permitir a criação da lógica da aplicação através da conexão de blocos. Adicionado a isso, o APP Inventor possui um ambiente de discussão e tutoriais para os usuários aprenderem a utilizar os blocos na construção de seus aplicativos. A diferença entre a ferramenta apresentada neste trabalho e o APP Inventor é a forma como a API Blockly é utilizada. Como o foco principal daquela ferramenta é o desenvolvimento de aplicativos, o seu uso em sala de aula é bem específico uma vez que nem todos os cursos possuem uma disciplina especialmente focada em aplicativos *mobile*.

Outro trabalho que também utiliza o Google Blockly é o Block.ino. Seu foco é auxiliar professores e estudantes de educação básica no processo de ensino-aprendizagem de forma multidisciplinar. “O protótipo chamado de block.ino consiste em uma interface adaptada para desenvolvimento de programas computacionais para uma placa Arduino controlando sensores e atuadores por meio de acesso remoto ao recurso” [Carlos et al. 2016]. A nossa abordagem é semelhante, entretanto nosso foco é voltado à mudança de modelos, em transpor a lógica criada de forma mais fácil com os blocos para código-fonte compreendido pela máquina. Além disso, nossa ferramenta é voltada à estudantes que estão iniciando nos cursos de graduação em computação, ou que possuem matérias relacionadas à área.

Além da biblioteca Blockly, existe outra conhecida ferramenta que utiliza programação visual para criar lógica de aplicações. Esta é o Scratch [Liang et al. 2007], desenvolvido pelo Instituto de Tecnologia de Massachusetts (MIT), que permite a construção de jogos, histórias e animações. Seu propósito é de auxiliar no desenvolvimento cognitivo ao mesmo tempo que ensina lógica de programação e encoraja a criatividade. É um ambiente ideal para ser utilizado com crianças e durante o ensino fundamental. Todavia, em cursos de graduação relacionados a área de computação a ferramenta pode ser pouco usada, uma vez que ela não foca na geração e na exibição do código manipulado pelos blocos.


Apesar da ferramenta que aqui propomos utilizar a mesma linguagem visual já empregada por estas outras aplicações, o URI Online Judge Blocks (ou URI Blocks) tem um foco diferente. Através dele, nós pretendemos fornecer um ambiente que possa ser aplicado na introdução à programação também para jovens adultos dentro de cursos de graduação, de modo que não exija um contexto específico de aplicação. Ao integrar essa biblioteca a um juiz online, disponibilizamos um ambiente que pode auxiliar no desenvolvimento da lógica de programação, permitindo que o estudante primeiro entenda essa parte tão necessária, mas ao mesmo tempo não o afastando da codificação em si, além de tornar o ambiente do juiz um local completo para prática e aprendizagem de programação.

4. URI Online Judge Blocks

Com o URI Blocks, o estudante pode fazer uso de uma linguagem visual para construir soluções para problemas da vida real e questões mais complexas, como as utilizadas em competições de programação. Estudantes podem resolver até 89 problemas do repositório do URI utilizando blocos. Estes problemas são de baixa complexidade, onde os usuários devem utilizar algoritmos sequenciais (manipulação de variáveis), seleção (uso de condicionais) e laços de repetição para resolvê-los. O primeiro e mais simples problema disponível no repositório pode ser resolvido com blocos, o URI 1001 (Figura 2).

URI Online Judge | 1001

Extremamente Básico

Adaptado por Neilor Tonin, URI  Brasil

Timelimit: 1

Leia 2 valores inteiros e armazene-os nas variáveis **A** e **B**. Efetue a soma de **A** e **B** atribuindo o seu resultado na variável **X**. Imprima **X** conforme exemplo apresentado abaixo. Não apresente mensagem alguma além daquilo que está sendo especificado e não esqueça de imprimir o fim de linha após o resultado, caso contrário, você receberá *"Presentation Error"*.

Entrada

A entrada contém 2 valores inteiros.

Saída

Imprima a mensagem "X = " (letra X maiúscula) seguido pelo valor da variável **X** e pelo final de linha. Cuide para que tenha um espaço antes e depois do sinal de igualdade, conforme o exemplo abaixo.

Exemplos de Entrada	Exemplos de Saída
10 9	X = 19
-10 4	X = -6
15 -7	X = 8

Figura 2. Primeiro problema do URI Online Judge, o URI 1001

Todos os problemas do repositório do URI seguem uma mesma estrutura: a descrição pode conter uma pequena história, com alguns personagens ficticiais, para contextualizar o problema, modelar a ideia e tornar claro como o algoritmo (a solução codificada) deverá funcionar; uma área contendo informações relacionadas as especificações de entrada e os limites de cada variável (Entrada); e uma descrição de como os dados gerados como solução devem ser mostrados (Saída).

Na Figura 3 apresentamos o ambiente para solução de problemas com blocos. O número de identificação do problema que está sendo resolvido e a linguagem em que o código-fonte construído através dos blocos será gerado é representado pela área (1) da Figura 3. Inicialmente as linguagens disponíveis no URI Blocks são Python 2.7.6 e Python 3.4.0. Existe a possibilidade de gerar soluções criadas dentro do ambiente *Blocks* também em outras linguagens, como JavaScript e Lua. Estas serão futuramente integradas na plataforma, após um período de avaliação inicial da nossa proposta.

Como este trabalho integra a API Blockly com um juiz online que disponibiliza problemas no formato ACM-ICPC, o ambiente foi estruturado seguindo esta lógica de problemas. Desta forma, conforme ilustrado pela área (2) da Figura 3, agrupamos os blocos em três subáreas: aqueles que lidam com a entrada de dados, blocos responsáveis pelo processamento, e blocos para formatar e exibir a saída.

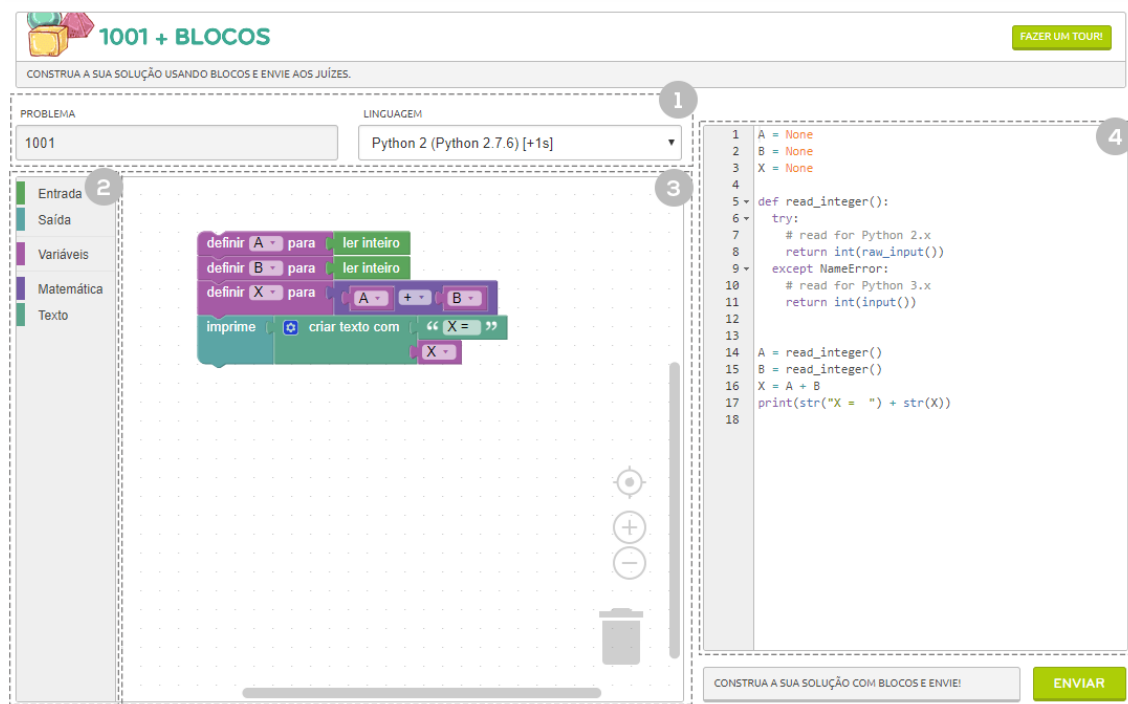


Figura 3. Interface do URI Blocks para o problema 1001

Adicionado a isso, cada problema possui um conjunto pré-determinado de blocos que estão de acordo com as necessidades do exercício em questão. No caso do problema URI 1001, nenhum bloco de repetição ou de vetores foi incluso como opção de seleção, uma vez que essas estruturas não são necessárias para a solução do problema. Ambas características foram pensadas com o intuito de facilitar o entendimento e a solução do problema através do ambiente URI Blocks.

Com o Google Blockly, há a possibilidade de definir e criar novos blocos para realização de determinadas funções. Utilizamos esta funcionalidade para permitir que o usuário possa resolver os problemas do contexto de um juiz online. Nesse sentido, foram criados alguns blocos para a leitura de dados, formatação numérica e para a apresentação das saídas de acordo com as especificações do problema. Além disso, a API já disponibiliza traduções para diversos idiomas, entre eles Português e Espanhol, e também permite realizar a tradução de blocos customizados. Como o URI Online Judge está disponível nos idiomas Português, Inglês e Espanhol, o URI Blocks também conta com tradução para estes idiomas.

O editor onde os blocos deverão ser conectados, a fim de montar a solução do problema, é mostrado pela área (3) da Figura 3. Por fim, a caixa a direita, área (4) corresponde ao código gerado pela construção da solução em *blocks* que deve ser submetido para avaliação (*i.e.* testado pelo juiz) quando o botão de enviar é pressionado. A geração do código-fonte é feita em tempo real, refletindo as ações realizadas dentro da área de edição dos blocos.

Como forma de ilustrar a interação de um estudante resolvendo o problema 1001, descreveremos o passo a passo dessa solução que também é ilustrado na Figura 3.

Conforme a descrição do URI 1001, nós precisamos criar duas variáveis A e B e então ler seus respectivos valores a partir de uma entrada de dados. Depois, precisamos criar a variável X que receberá a soma das variáveis A e B. Como mostrado na descrição de saída do problema, precisamos mostrar juntamente ao resultado processado o texto "X = ". Para isso, nós adicionaremos o bloco de texto que é responsável por criar o código que mostrará o texto na tela. Adicionando todos esses blocos no editor, o respectivo código-fonte será automaticamente gerado na área de código, em Python.

Concluída a construção da solução do problema, precisamos enviar essa solução para ser avaliada pelo juiz. Assim que a solução for avaliada, a resposta é retornada para o estudante. As possíveis respostas¹:

- **Accepted:** submissão retorna o resultado correto para todos os testes;
- **Wrong Answer:** código não gerou o resultado esperado para todos os testes;
- **Presentation Error:** a solução retorna os valores corretos, porém a apresentação destes não está conforme a especificação do problema;
- **Runtime Error:** código apresentou erros/exceções em tempo de execução.

5. Primeiras Impressões

A versão beta do URI Blocks foi lançada em 2016, durante a *Week of Code* (WoC) [Programaê! 2018]. WoC é uma iniciativa ligada a Hora do Código [Code.org 2018b], promovida pela Code.org [Code.org 2018a], que tem como objetivo iniciar qualquer pessoa que tenha interesse em aprender a programar. Uma vez que o objetivo do nosso trabalho foi semelhante à proposta do evento, aproveitamos a oportunidade para receber alguns comentários iniciais sobre a aceitação da ferramenta pela comunidade. Naquele momento, apenas os primeiros cinco problemas do repositório do URI (1001 à 1005) estavam disponíveis para serem resolvidos utilizando blocos. Durante aquela semana, estes problemas receberam um total de 40.330 submissões, dos quais 38.898 foram envios regulares (codificados manualmente pelos usuários) e 1.432 foram utilizando os blocos.

A grande aceitação do URI Blocks durante a WoC motivou a continuação do desenvolvimento da ferramenta e, em 25 de setembro de 2017, o URI Blocks foi oficialmente lançado. Até o momento, são cerca de 89 problemas do repositório do URI que possuem essa ferramenta disponível. Já se somam 29.326 submissões utilizando os blocos, feitas por mais de 15.673 usuários, e suas páginas foram visualizadas mais de 100.000 vezes.

Dentro destes 89 problemas que contam com a ferramenta, os que possuem maior número de submissões são o 1001, 1002, 1003, 1005 e 1008. Somados, eles são responsáveis por 41.5% das submissões que utilizam blocos. A Tabela 1 mostra a distribuição dessas submissões, por resposta.

Percebe-se que o problema 1001 é o maior responsável por submissões, totalizando 5.116 (42% do total de submissões para estes 5 problemas). Além de ser o primeiro e mais simples problema do repositório, ao qual a maioria dos usuários inicia por sua resolução, o 1001 é utilizado como base para o tutorial de introdução ao URI Blocks. Nesse tutorial,

¹ Além das respostas apresentadas, o juiz também pode retornar *Time Limit Exceeded* quando o código demora mais que o tempo esperado para executar, e *Compilation Error*, quando ocorre algum erro de compilação.

Problema	AC	RE	PE	WA	Total
1001	1.887	1.026	676	1.527	5.116
1002	564	994	270	787	2.615
1003	545	222	232	311	1.310
1005	444	608	95	773	1.920
1008	196	467	67	479	1.209
Total	3.636	3.317	1.340	3.877	12.170

Tabela 1. Distribuição, por resposta, das submissões em com URI Blocks.

a solução para o mesmo é desenvolvida passo a passo, ficando disponível para o usuário apenas submetê-la para julgamento ao final. O problema 1003 é o que apresenta o maior percentual de *Accepted* (AC), onde 41.6% das submissões para o problema estão corretas. O que apresenta maior percentual de erro é o problema 1008, onde a maioria dos envios feitos recebeu *Runtime Error* (RE) ou *Wrong Answer* (WA) como resposta.

De maneira geral, o número de respostas que foram aceitas pelo juiz corresponde a 29.9% do total de submissões desta amostra em análise. Uma vez que essa ferramenta tem o intuito de auxiliar estudantes novatos na introdução de conceitos de programação, é possível considerar esse percentual bastante promissor.

6. Conclusão

Neste artigo, apresentamos uma ferramenta que tem como objetivo auxiliar o processo de aprendizagem de algoritmos e estrutura de dados, fornecendo uma maneira visual, intuitiva e dinâmica de construir códigos. Para fazer isso, integramos a API do Google Blockly aos problemas da categoria “Iniciante” do URI Online Judge.

O URI Blocks é uma ferramenta que, ao mesmo tempo, facilita a compreensão da lógica de programação de forma visual, e mantém o estudante em contato com uma linguagem de programação. Assim, este trabalho surge como uma alternativa para uma transição mais gradual entre modelos, uma vez que uma das principais dificuldades dos estudantes de Ciência da Computação e Engenharia é transpor a lógica de uma solução para um código em linguagem de programação. Com os blocos, os estudantes podem entender de forma mais fácil e visual os conceitos relacionados a algoritmos.

Lançado oficialmente em setembro de 2017, disponibilizando 89 problemas, a ferramenta teve mais de 29.000 envios de cerca de 15.000 usuários. Desde que foi lançado, em fase de testes, a ferramenta demonstrou ser muito promissora. Para aumentar o uso dessa ferramenta nas salas de aula, planejamos adicionar suporte para o URI Blocks ao módulo URI Academic. Com a integração a este módulo os professores terão acesso à lógica criada pelos seus estudantes para os problemas propostos com os Blocos, permitindo-lhes ter uma melhor visão geral do grupo e tomar ações pontuais ou globais para resolver questões relacionadas à compreensão do conteúdo teórico apresentado em aula.

Como trabalhos futuros, planeja-se possibilitar o envio de soluções criadas com o URI Blocks em Lua e JavaScript, ambas linguagens já suportadas pelo URI. Ainda, planejamos estender a biblioteca e dar suporte a linguagem C/C++, já que esta é bastante utilizada pelos usuários do URI Online Judge.

7. Agradecimentos

Agradecemos a Fundação Lemann, através da iniciativa Programaê!, pelo apoio financeiro com o projeto; a equipe do portal URI Online Judge que auxiliou no desenvolvimento da ferramenta; e a Gabriela Binder, acadêmica de Psicologia, pela colaboração com a fundamentação teórica relacionada às teorias de aprendizagem.

Referências

- ACM (2018). The ACM-ICPC International Collegiate Programming Contest.
- Carlos, L. M., de Lima, J. P., Simão, J. P. S., and Silva, J. (2016). block. ino: Um experimento remoto para ensino de lógica de programação, robótica e eletrônica básica. In *Anais dos Workshops do Congresso Brasileiro de Informática na Educação*, volume 5, page 151.
- Code.org (2018a). Anybody can learn — Code.org.
- Code.org (2018b). Hour of code. <https://hourofcode.com>. Acessado em: 2018 January 14.
- de La Taille, Y., de Oliveira, M. K., and Dantas, H. (1992). *Piaget, Vygotsky, Wallon: teorias psicogenéticas em discussão*. Summus editorial.
- Google (2015). Blockly about. <https://blockly-games.appspot.com/about>. Acessado em: 2018 January 13.
- Hoed, R. M. (2017). Análise da evasão em cursos superiores: o caso da evasão em cursos superiores da área de computação.
- Iepsen, E. F., Bercht, M., and Reategui, E. (2010). Persona-algo: personalização dos exercícios de algoritmos auxiliados por um agente afetivo. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 1.
- Koliver, C., Dorneles, R. V., and Casa, M. E. (2004). Das (muitas) dúvidas e (poucas) certezas do ensino de algoritmos. In *XII Workshop de Educação em Computação*.
- Kurnia, A., Lim, A., and Cheang, B. (2001). Online judge. *Computers & Education*, 36(4):299–315.
- Liang, C.-C., Park, A. Y., and Guan, J.-L. (2007). In vitro scratch assay: a convenient and inexpensive method for analysis of cell migration in vitro. *Nature protocols*, 2(2):329–333.
- Marron, A., Weiss, G., and Wiener, G. (2012). A decentralized approach for programming interactive applications with javascript and blockly. In *Proceedings of the 2Nd Edition on Programming Systems, Languages and Applications Based on Actors, Agents, and Decentralized Control Abstractions*, AGERE! 2012, pages 59–70, New York, NY, USA. ACM.
- Programaê! (2018). Hora do Código: Programaê!
- Winslow, L. E. (1996). Programming pedagogy – a psychological overview. *SIGCSE Bull.*, 28(3):17–22.
- Wolber, D. (2011). App inventor and real-world motivation. In *Proceedings of the 42nd ACM technical symposium on Computer science education*, pages 601–606. ACM.