

Integração do ambiente BOCA com o ambiente Moodle para avaliação automática de algoritmos

Rafael Hernandez Galasso¹, Benjamin Grando Moreira²

¹ Centro de Ciências Tecnológicas da Terra e do Mar (CTTMar)
Universidade do Vale do Itajaí (UNIVALI) – Itajaí, SC – Brasil

² Centro de Engenharias da Mobilidade
Universidade Federal de Santa Catarina (UFSC) – Joinville, SC – Brasil

druidas@gmail.com, benjamin.grando@ufsc.br

Abstract. *BOCA Online Contest Administrator is an environment used to manage competitions of computer programming, like the brazilian “Maratona de Programação”. Besides controlling the competition, the environment has a feature known as “online judge” that conducts a automatic evaluation of algorithms submitted, classifying these as correct or incorrect. This article presents the integration of this environment with Moodle, allowing to create questions that the answer is an algorithm and that is evaluated automatically, returning to the student the avaluation.*

Resumo. *O BOCA Online Contest Administrator é um ambiente utilizado para gerenciar competições que envolvem programação de computadores, como a Maratona de Programação. Além de controlar a competição, o ambiente possui um recurso conhecido como juiz online que realiza a avaliação automática dos algoritmos submetidos, classificando esses como corretos ou incorretos. Esse artigo apresenta a integração desse ambiente com o Moodle, possibilitando que sejam criadas questões que a resposta é um algoritmo e que são avaliadas automaticamente, gerando um retorno imediato para o aluno a cada submissão da questão.*

1. Introdução

Por definição, algoritmo é uma sequência de passos bem definida em uma ordem específica transformando uma entrada fornecida em uma saída. Também pode-se dizer que algoritmo é uma ferramenta para resolver problemas bem especificados (CORMEN, 2002).

Algoritmos é uma disciplina fundamental para o ensino de programação e se faz presente em diversos cursos, como os de engenharia. O uso de algoritmos pode ser constatado em competições, como a olimpíada de informática e a maratona de programação, exigindo dos alunos algumas aptidões, como os princípios de lógica e a resolução de problemas através da interpretação dos algoritmos descritos. Segundo Rapkiewicz (2006), a interpretação da solução aparece como um dos principais fatores para a desmotivação na disciplina, culminando assim em altos índices de reprovação e evasão por parte dos alunos.

Muitas instituições de ensino que procuram modernizar seu sistema e facilitar o acesso ao conhecimento estão optando pelo uso do ambiente Moodle, que surge usando o conceito construcionista social de educação. O Moodle é composto por pacotes de software, abordando o ensino on-line em produções de cursos e sites na internet (MOODLE, 2012).

Sendo assim, tendo em vista que o ambiente acadêmico está cada vez mais dinâmico, o trabalho aqui apresentado criou um tipo de questão para o módulo de questionário no Moodle, com a finalidade de professores criarem problemas para seus alunos que envolvam a resolução de algoritmos.

No envio da resolução, por parte do aluno, um juiz online compila o código com um conjunto de casos de testes, sendo que cada caso de teste contém as entradas esperadas pelo programa que são cadastrados pelo professor, e compara com um conjunto de saída padrão, também cadastrado pelo mesmo. Estando o conjunto de saída padrão idêntico ao conjunto de saída gerado pelo algoritmo do aluno, a resolução é considerada correta, permitindo que haja um retorno imediato sobre a solução enviada pelo estudante.

O trabalho desenvolvido integra o ambiente Moodle ao ambiente BOCA Online Contest Administrator (BOCA) para realizar a avaliação automática da questão. A solução apresentada buscou não alterar características do próprio Moodle, permitindo uma implantação mais simples. A solução também foi desenvolvida para que os ambientes fossem implementados em servidores independentes já que a utilização do BOCA durante a avaliação dos algoritmos não deve comprometer o desempenho do ambiente Moodle.

2. Dificuldades no ensino de algoritmos

Com um índice elevado de reprovação e desistência em instituições de ensino no Brasil, a disciplina de algoritmos tem sido foco dos professores, preocupados com a melhoria no processo de ensino e a qualidade oferecida para os alunos, mostrando assim necessidade de algumas alterações tanto na parte didática como na metodológica da apresentação (RODRIGUES JUNIOR, 2004).

Outro problema encontrado no método tradicional de ensino na disciplina é o de avaliar. As avaliações ocorrem como uma ação isolada, especial, com data marcada, sendo o seu objetivo único de aprovação. Isto desenvolve no aluno um receio, prejudicando assim seu aprendizado na disciplina (RODRIGUES JUNIOR, 2004).

O problema por parte do aluno, segundo Rapkiewicz (2006), é com relação a verificação de se o algoritmo que foi construído realmente funciona como esperado. Testes de mesa costumam ser empregados nessa etapa da disciplina para a verificação de que o algoritmo desenvolvido é eficaz. Um passo a passo deve ser executado pelo aluno para verificar a funcionalidade de sua solução. Isso além de desmotivar o aluno traz um novo problema, que seria o ato de seguir o passo a passo.

Algumas dificuldades são da natureza da disciplina de algoritmos, não sendo do aluno ou professor. De acordo com Siebra (2009), algumas destas dificuldades são:

- Grande numero de alunos: por ser normalmente oferecida em períodos iniciais dos cursos, as disciplinas possuem grande numero de alunos matriculados,

dificultando a individualização de sua avaliação e o numero de avaliações propostos pelo professor;

- Dificuldade de o professor compreender a logica do aluno: é difícil mudar um raciocínio logico depois de construído e tentar construir outra forma de resolução de problemas. Isso acarreta em uma difícil tarefa para o professor de compreender a lógica específica de cada aluno;
- Turmas heterogêneas: turmas onde existem diferentes níveis de conhecimento entre os alunos e ritmo de aprendizagem. Alguns alunos possuem algum contato prévio de programação antes de ingressar no curso ou até mesmo trabalham na área, enquanto outros não tiveram contato com algoritmos e programação; e
- Motivação dos alunos: fica a cargo do professor fazer com que o interesse pela matéria seja mantido, não acarretando em desistência da disciplina. Muitos meios podem ser usados para que o professor consiga deixar o aluno interessado na matéria, como instigar o aluno para que busque outras formas de soluções de problemas ou técnicas novas.

Com o sistema desenvolvido acredita-se que se o aluno tiver um retorno imediato em exercícios práticos seu interesse irá crescer, o que pode influenciar de forma positiva na disciplina já que o aluno não precisa esperar por dias para saber se sua solução está correta ou não e pode buscar corrigi-la logo em seguida da submissão.

3. Juiz online

Juiz online é um recurso disponível em diversos ambientes que gerenciam competições de programação de computadores e tem por objetivo realizar, de maneira automatizada, a avaliação dos algoritmos submetidos. É de responsabilidade do juiz online fazer: a compilação do código submetido; execução do programa gerado; especificação de dados de entrada para a execução; e por fim a comparação da saída da execução com uma saída padrão esperada.

Tanto os dados de entrada quanto os dados de saída, são dados padrão da questão e são utilizados considerando que, se o algoritmo está correto, ele gerará os dados de saída a partir da entrada definida. Esses dados não são de conhecimento dos usuários que submetem o algoritmo.

Como o juiz online precisa executar o código submetido, o sistema operacional fica suscetível a falhas do código e também a códigos maliciosos e o juiz online precisa lidar com essas situações. Sendo assim é preferível que este juiz execute em um servidor próprio, não comprometendo as aplicações principais.

Foram pesquisados diferentes juizes online a fim de proporcionar a realização do projeto proposto, com destaque para o PC2¹, Mooshak² e o BOCA³. Dentre os pesquisados houve uma seleção, eliminando de antemão juizes que não fossem de código aberto, como o PC2. Também levou-se em consideração que sua documentação fosse disponível e de fácil acesso, sendo assim o BOCA selecionado. Também foi

¹ Site: <http://www.ecs.csus.edu/pc2>

² Site: <http://mooshak.dcc.fc.up.pt/~zp/mooshak/>

³ Site: <http://www.ime.usp.br/~cassio/boca/>

considerado que o BOCA é o ambiente utilizado na Maratona de Programação, principal evento de programação e realizado nacionalmente.

O ambiente BOCA é desenvolvido em linguagem PHP que tem interação com as equipes via navegador e foi desenvolvido com o objetivo de controlar competições no modelo da Maratona de Programação da Sociedade Brasileira de Computação. O sistema conta com um juiz online que ajuda no processo de julgar uma submissão. Uma vez submetido o problema pelas equipes o script é capaz de executar comandos de compilar, executar e comparar a saída gerada pelos times com a saída correta (CAMPOS, 2011).

4. Trabalhos similares

Na busca por trabalhos desenvolvidos utilizando o ambiente BOCA e Moodle sendo aplicados para correção automática de exercícios de algoritmos de programação, dois trabalhos se destacaram, sendo o primeiro proposto por Souza (2009) que realizou uma adaptação do BOCA para uma aplicação chamada BOCA-LAB. O que se fez no BOCA-LAB foram algumas alterações na interface do ambiente original para atender melhor as questões pedagógicas.

O resultado da aplicação do BOCA-LAB foi positivo. Os participantes foram ativos no processo das numerosas correções, ficando comprovado que a ferramenta contribuiu de forma positiva no processo de aprendizagem, gerando uma métrica de desempenho da turma (SOUZA, 2009).

O segundo trabalho foi proposta por Cunha *et al.* (2006) e desenvolveu um juiz online integrado ao Moodle. O sistema funciona a partir de um compilador GCC e um testador que verifica se dada uma entrada padrão sua saída está de acordo com a saída esperada em um arquivo XML. Esse corretor gera relatórios de problemas encontrados através dos resultados obtidos nos testes e na análise do código.

O diferencial do trabalho proposto com o primeiro apresentado está que no primeiro trabalho existe a necessidade de interação com mais um ambiente para a prática dos exercícios, enquanto na solução proposta o aluno mantém-se no Moodle. Além disso, um questionário do Moodle poderá conter tanto exercícios em que a resposta é um algoritmo, quanto outros tipos de questão do Moodle.

Com relação ao segundo trabalho, este desenvolveu um juiz online que é bastante simples, trabalhando apenas com as linguagens C e C++ e com pouca verificação na parte de segurança. Destaca-se no trabalho dois recursos interessantes: ele mostra as mensagens de erros da compilação (na solução proposta apenas informa que existiu algum erro) e também é preparado para utilizar vários casos de teste (na solução proposta apenas um caso de teste é aplicado).

5. Solução proposta

A solução proposta adiciona um tipo de questão (denominada “Juiz online”) para ser utilizada em questionários do ambiente Moodle e o professor deve adicionar uma questão para cada problema que pretende aplicar. Por recurso do próprio Moodle, o professor poderá utilizar várias dessas questões compondo um questionário.

Para a construção dos problemas o professor preenche os campos necessários no ambiente Moodle, sendo eles, o título do problema, corpo do problema (enunciado do problema) e um conjunto de casos de teste. Cada caso de teste contém entradas para o algoritmo e saída esperada para a entrada fornecida. O cadastro também exige que seja informada a linguagem de programação para a resolução do problema. A instalação padrão do BOCA trabalha com as linguagens C, C++ e Java e ele compilará o algoritmo com o compilador apropriado para a linguagem indicada no momento do cadastro. A Figura 1 é uma edição da tela de cadastro da questão que busca ilustrar esta etapa do processo.

Adding a Juiz Online question?

Geral

Categoria: Padrão para M.Programação (25)

Nome da pergunta*: Tobogan de bolinhas

Texto da questão*

Família de fontes: Tamanho da fonte: Parágrafo

A primeira linha de um caso de teste contém um inteiro N indicando o número de aletas do brinquedo. A segunda linha contém dois inteiros L e H, indicando mais alta (a segunda a ser descrita) tem a extremidade ligada à haste direita, assim alternadamente.

Caminho: p

Input.*

Output.*

Para a resposta do aluno : Você deve definir uma linguagem de programação, para que o mesmo possa submeter seus algoritmos na lingu

Linguagem de Programação:* C++

Figura 1. Edição da tela de cadastro de questão.

Os alunos devem submeter seus algoritmos em forma de código-fonte na linguagem especificada pelo professor e recebendo em seguida o retorno da avaliação da questão. O algoritmo é enviado para o ambiente BOCA que avaliará a solução, onde ele é compilado e executado com o conjunto de entrada cadastrados pelo professor e comparando com a saída padrão definida.

Estando avaliado, o aluno terá como resposta que a questão está correta ou incorreta, sem avaliação parcial (a Figura 2 ilustra a avaliação de uma questão como correta). No caso da questão incorreta, também é mostrado no comentário da avaliação os possíveis retornos:

- Erro de compilação: significa que o algoritmo não chegou a compilar;
- Erro de formatação: representando que a saída do programa não segue a especificação exigida. A saída tem que ser exatamente como solicita. Erros de formatação geralmente significam algum espaço em branco, ou nova linha, a mais ou a menos; e

- Tempo limite excedido: representando que o algoritmo excedeu o tempo estipulado. Na maratona de programação esse tempo é bastante reduzido, mas para esse trabalho foi especificado 30 segundos na execução, o que provavelmente somente algoritmos com laço de repetição infinito alcançarão.

Tentativa	Estado	Notas / 1,00	Nota / 100,00	Revisão	Comentários
Visualização prévia	Finalizadas <small>Enviada(s) quarta, 17 julho 2013, 17:15</small>	1,00	100,00	Revisão	Correto Algoritmo

Figura 2. Parte da tela onde o aluno visualiza o retorno da avaliação da questão.

Por recurso fornecido pelo Moodle, é possível habilitar que uma questão possa ser respondida com várias tentativas (quantidade configurável). Sendo assim, ao receber um retorno de algoritmo incorreto o aluno poderia tentar outras vezes até conseguir alcançar o sucesso na elaboração de sua solução.

Também é importante destacar que, mesmo a avaliação automática julgando uma solução como incorreta, o professor pode alterar o resultado manualmente, ajustando a nota para uma solução parcial. Nesse caso, o professor poderia confiar nos resultados corretos, dispensando sua avaliação e avaliando apenas os resultados incorretos, contribuindo com uma redução na carga de correções.

5.1 questões técnicas do banco de dados

Um objetivo do trabalho foi trazer uma solução independente entre o ambiente Moodle e o BOCA, permitindo que esses sejam executados em servidores distintos: um com o Moodle e outro com o BOCA. Essa distribuição permite que problemas com o BOCA (como a sobrecarga com o julgamento de muitos algoritmos) não afetem a utilização do Moodle (questões de segurança e de desempenho da aplicação) e também permitindo a hipótese de um servidor BOCA atendendo a várias instituições.

Como o BOCA foi desenvolvido para o Sistema Gerenciador de Banco de Dados (SGBD) PostgreSQL, o Moodle precisa estar instalado utilizando o PostgreSQL. Além disso, para permitir a comunicação entre bancos de dados em locais diferentes, o PostgreSQL necessita da utilização do Dblink.

Dblink é um conjunto de funções, que possibilita a conexão entre banco de dados PostgreSQL com o intuito de ter acesso a dados externos. Sendo assim o Dblink permite o acesso remoto a uma tabela de uma determinada base de dados, e a realizar a execução de consultas (DEVMEDIA,2007). Destaca-se que existem soluções que também permitem conectar o PostgreSQL com outros bancos de dados e poderiam ser utilizadas para uma instalação do Moodle em outro SGBD.

O banco de dados do sistema BOCA possui doze tabelas. Dentre elas quatro são utilizadas pelo juiz online do BOCA (mostradas no diagrama da Figura 3), sendo necessárias para o funcionamento da solução apresentada. Também foi preciso criar duas tabelas (*tempaluno* e *tempprofessor*), para que houvesse a manipulação dos dados entre as tabelas já existentes. Com relação as tabelas do BOCA de interesse da solução apresentada estão:

- Problemtable: utilizada para o cadastramento das questões para a competição. Além de diversos atributos da questão, o BOCA espera por um arquivo compactado com extensão “.zip” com as entradas e saídas cadastradas pelo professor, bem como informações do problema. Este arquivo compactado é requisitado pelo juiz online no momento de execução do algoritmo sendo julgado. O arquivo compactado é gerado automaticamente pelo trabalho.
- Runtable: a medida que o aluno submete os seus algoritmos o atributo *rundata* é alimentado com um arquivo com a extensão necessária de cada linguagem de programação;
- Answertable: após a execução do algoritmo o mesmo irá alimentar o atributo *runanswer* da tabela, armazenando se o algoritmo está correto ou possíveis problemas na compilação ou na execução; e
- Langtable: essa tabela armazena as linguagens de programação que serão utilizadas. Por padrão o BOCA adiciona as linguagens de programação C, C++ e Java. Podem ser adicionadas outras linguagens, mas não terá efeito na solução proposta. Uma restrição é que o identificador dessas linguagens não pode ser alterado.

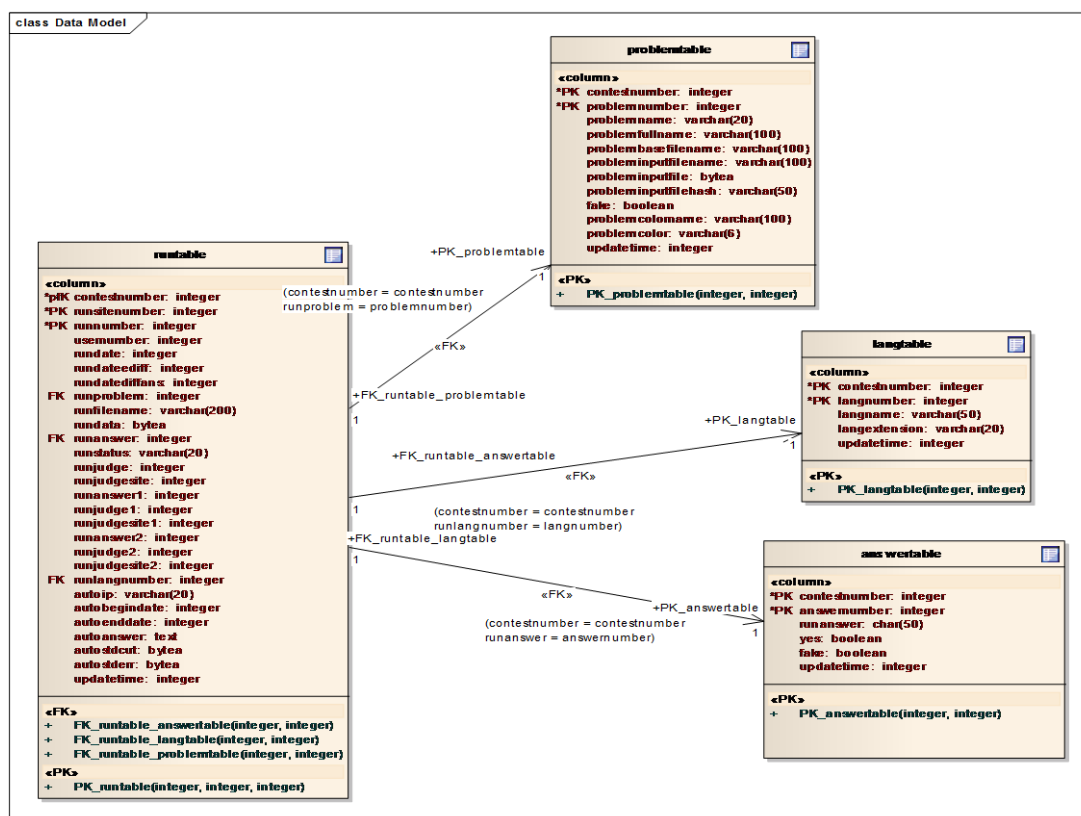


Figura 3. Tabelas utilizadas pelo juiz online do BOCA.

5.2 Integração do juiz online do BOCA

A integração do BOCA com o ambiente Moodle foi realizada através da criação de gatilhos (*triggers*), que é um recurso existente em diversos SGBD, dispensando

alterações no código do Moodle e do BOCA. A Figura 4 ilustra o fluxo entre as bases de dados gerado pelos gatilhos. O elemento “juiz online” é um processo do BOCA que faz a avaliação das questões.

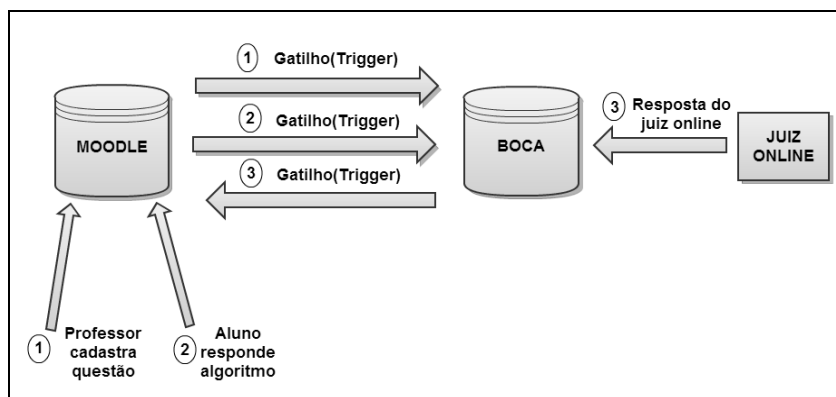


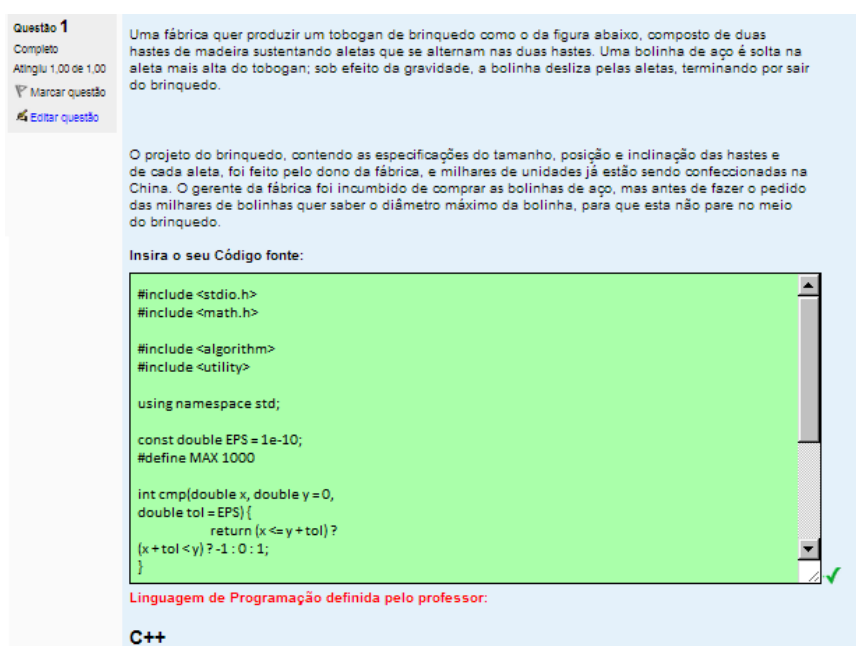
Figura 4. Fluxo da interação das bases de dados dos Moodle e do BOCA.

Quando um professor cadastra uma questão no ambiente Moodle esta é gravada no banco de dados do Moodle e essa inserção ativa um primeiro gatilho, que grava os dados da questão na tabela *tempprofessor* adicionada na base de dados do ambiente BOCA. Quando dados são gravados na tabela *tempprofessor* outro gatilho é acionado carregando os dados relacionados ao problema criado pelo professor e o arquivo compactado contendo as configurações da questão esperada pelo BOCA. Esses dados são adicionados na tabela *problemtable*, registrando o problema no BOCA.

Conforme o aluno responde as questões no ambiente Moodle, o envio é registrado nas tabelas do banco do ambiente e é acionado outro gatilho para a tabela *tempaluno*. A seguir outro gatilho registrado no BOCA é acionado pegando o arquivo com a resposta do aluno e as informações da questão e cadastra na tabela *runtable*, do BOCA.

Para a verificação se um problema submetido está correto faz-se uso de um processo do BOCA chamado *autojuding* (juiz online – é um script em linguagem PHP que deve ficar em execução permanente no servidor), que compila o algoritmo enviado pelo aluno, gerando assim uma saída, que é comparada com a saída padrão definida pelo professor na criação do problema.

O processo de avaliação dos algoritmos é executado a cada minuto, sempre verificando se há algum problema para ser julgado existente nas tabelas do BOCA. Havendo um novo problema para ser julgado, ele julga e o resultado será gravado na tabela *runtable* do BOCA. Ao realizar essa gravação outro gatilho é acionado, pegando o resultado na tabela do BOCA e alterando as tabelas do ambiente Moodle, possibilitando que o aluno possa visualizar o retorno da avaliação de sua solução (conforme ilustrado na Figura 5).



Questão 1
Completo
Atingiu 1,00 de 1,00
▼ Marcar questão
✎ Editar questão

Uma fábrica quer produzir um tobogan de brinquedo como o da figura abaixo, composto de duas hastes de madeira sustentando aletas que se alternam nas duas hastes. Uma bolinha de aço é solta na aleta mais alta do tobogan; sob efeito da gravidade, a bolinha desliza pelas aletas, terminando por sair do brinquedo.

O projeto do brinquedo, contendo as especificações do tamanho, posição e inclinação das hastes e de cada aleta, foi feito pelo dono da fábrica, e milhares de unidades já estão sendo confeccionadas na China. O gerente da fábrica foi incumbido de comprar as bolinhas de aço, mas antes de fazer o pedido das milhares de bolinhas quer saber o diâmetro máximo da bolinha, para que esta não pare no meio do brinquedo.

Insira o seu Código fonte:

```
#include <stdio.h>
#include <math.h>

#include <algorithm>
#include <utility>

using namespace std;

const double EPS = 1e-10;
#define MAX 1000

int cmp(double x, double y = 0,
double tol = EPS) {
    return (x <= y + tol) ?
(x + tol < y) ? -1 : 0 : 1;
}
```

Linguagem de Programação definida pelo professor:

C++

Figura 5. Retorno da avaliação da questão como correta.

Ao submeter a questão é mostrada uma imagem de carregamento solicitando um período de espera para o julgamento da questão. Quando o processo de julgamento se encerra, a área onde se encontra o algoritmo altera sua cor para verde se a resposta está correta, ou vermelho para resposta incorreta.

Destaca-se um problema encontrado com relação ao arquivo “zip”. Inicialmente foi implementado o envio do arquivo diretamente para o banco de dados, mas não foi possível, assim como também não foi possível enviar os arquivos de entrada e saída diretamente pelo banco de dados pois esses ficavam com problema de codificação das tabelas e identificadores de nova linha. A forma encontrada para o envio desse arquivo foi via FTP, fazendo necessário instalar um servidor FTP no ambiente de execução do BOCA.

6. Considerações finais e conclusão

O trabalho teve como principal objetivo a integração de um juiz online já consolidado ao Moodle, proporcionando aos alunos um ambiente para exercitar seus conhecimentos e obter um retorno imediato de sua resposta, podendo posteriormente corrigir e reenviar para avaliação.

Com o sistema, espera-se que haja uma contribuição ao ensino de algoritmos nas disciplinas de programação, e que o retorno imediato possa proporcionar uma maior interação ao aluno, possibilitando melhoras na prática de algoritmos. Permitir que vários casos de teste sejam utilizados pelo sistema é um recurso faltante para alcançar a pretendida melhora e deverá ser implementada no futuro, embora o BOCA não permita esse tipo de possibilidade.

A solução proposta também exige que sejam criados os gatilhos no banco de dados do Moodle e do BOCA de maneira manual. É importante uma instalação

facilitada para efetiva utilização, assim como uma distribuição do BOCA já preparada para ser integrado.

Referências

- Campos, C. P. (2011) “Autojudging”, <http://www.ime.usp.br/~cassio/boca/boca/doc/AUTOJUDGING.txt>, outubro.
- Cormen, T. H., et. al. (2002) “Algoritmos: teoria e prática”, Elsevier, Rio de Janeiro.
- Cunha, et. Al. (2006) “Corretor automático de exercícios programa para auxílio no ensino à distância”, <http://bcc.ime.usp.br/principal/tccs/2006/matuki/Monografia%20Daniel%20Matuki.pdf>, outubro.
- DEVMEDIA (2007) “Consultas remotas no PostgreSQL”, SQL Magazine, volume 34.
- MOODLE (2012) “Moodle: sobre o moodle”, http://docs.moodle.org/all/pt_br/Sobre_o_Moodle, agosto.
- Rapkiewicz, C. E., et al. (2006) “Estratégias pedagógicas no ensino de algoritmos e Programação associadas ao uso de jogos educacionais”, CINTED-UFRGS: Novas Tecnologias na Educação, <http://seer.ufrgs.br/renote/article/view/14284/8203>, outubro.
- Rodrigues júnior, M. C. (2004) “Experiências Positivas para o Ensino de Algoritmos”, II Workshop de Educação em Computação e Informática Bahia-Sergipe, <http://www.uefs.br/erbase2004/documentos/weibase/Weibase2004Artigo001.pdf>, outubro.
- Siebra, S. A. e Silva, D. R. D. (2009) “Prática de Ensino de algoritmos”, Volume 2, Universidade Federal Rural de Pernambuco, <http://www.slideshare.net/aej400aej/prtica-de-ensino-de-algoritmo-volume-1-e-2>, outubro.
- Souza, A. (2009). “Adaptação de um corretor automático de código para auxílio da atividade de exercício e prática no ensino de programação”, trabalho de conclusão de curso da Universidade do Estado de Santan Catarina para obtenção do grau de Bacharel em Ciência da Computação, <http://www.pergamum.udesc.br/dados-bu/000000/000000000000D/00000D51.pdf>, outubro.