LabPy: Laboratório virtual de ensino em python

Silvério Sirotheau[12], Renan Filip Balieiro[1], Eloi Favero[12], João Carlos dos Santos[1]

[1]Universidade Federal do Pará (UFPA) - Instituto de Ciências Exatas e Naturais
Rua Augusto Corrêa, 01 – CEP 66075-110. Caixa postal 479 – Belém – PA – Brasil
[2]Programa de Pós-graduação em Ciência da Computação – PPGCC – UFPA
{silverio,renanfilip,favero,jcas}@ufpa.br

Abstract. *This article presents the LabPy - Phyton Programming Laboratory. It's a web environment with new specific features: i) dynamic content - web page content with in-line interpreter for program fragments; ii) programming exercises with immediate and automatic answer evaluation; iii) exercises with objective and subjective (discursive) questions, with instant automatic feedback as well; iv) multiple submissions for subjective exercises; v) elaboration of tests with automatic evaluation from the problem set. Labpy assists the student's practical learning and releases the professor from the most part of the activities' monitoring and correction.*

Resumo. *Este artigo apresenta o LabPy – laboratório de programação em Python. É um ambiente Web com características específicas e novas: i) conteúdo dinâmico – conteúdo em páginas web com interpretador "in-line" para fragmentos de programas; ii) exercícios de programação com avaliação imediata e automática das respostas; iii)exercícios com questões objetivas e subjetivas (questões discursivas) também com feedback automático imediato; iv) múltiplas submissões para exercícios subjetivos; v)elaboração de provas com avaliação automática a partir das listas de questões. LabPy auxilia o aprendizado prático do aluno e libera da maior parte do trabalho professor no acompanhamento e correção das atividades.*

1. Introdução

As disciplinas de algoritmos e linguagens de programação são imprescindíveis para os estudantes dos cursos de computação. Isso se deve muito ao fato de estabelecer os fundamentos para diversos campos nos quais a informática se aplica, proporcionando uma melhor compreensão e utilização de tecnologias computacionais atuais, incentivando o desenvolvimento do raciocínio lógico e capacidade na resolução de problemas (Moreira e Favero, 2009).

O ensino e aprendizagem de programação ainda constituem um grande desafio. Dificuldades no aprendizado de programação refletem em altos índices de reprovação de alunos em disciplinas que têm a programação como base, sendo alvo de estudos (Bosse

DOI: 10.5753/cbie.wcbie.2018.749

e Gerosa, 2015),(Cukierman, 2015); além disso, o mau desempenho leva ao desinteresse por parte do aluno na disciplina e em alguns casos, transforma-se em uma aversão (Amaral Júnior et al., 2018). Uma série de problemas estão associados a estas dificuldades (Moreira e Favero, 2009), (Mota et al. 2009), (Sirotheau et al. 2011). Nestes casos pode-se citar a dificuldade de entender a abstração exigida no desenvolvimento de algoritmo; a sobrecarga de trabalho dos professores; a dificuldade criada por turmas heterogêneas e; a limitação de estudo teórico em detrimento do lado prática. Vargas e Martins (2005) resumem a problemática de ensino de programação em dois desafios: (1) despertar no aluno a criatividade necessária para o desenvolvimento de soluções computacionais e (2) desenvolver a habilidade de representar a solução do problema usando lógica de programação.

Diante dos problemas expostos, ainda há muito a se fazer para melhorar o ensino de programação, auxiliando o trabalho dos professores e o progresso de seus alunos. Paes et al (2013) relatam que o aprendizado de programação baseia-se na prática e repetição regular de exercícios. Enquanto Prior (2003) diz que habilidade de programação não pode ser adquirida sem um significante esforço em atividades praticadas em laboratório.

Na literatura encontram-se vários ambientes que visam auxiliar o processo de ensino-aprendizagem de programação e algoritmo, tais como: JavaTool (Mota, 2008), Online *Python* Tutor (Guo, 2013), Pythy (Edwards et al., 2014) e entre outros. Apesar da rica história de trabalhos relacionados a este campo, sempre se pode acrescentar novas funcionalidades e aperfeiçoamentos buscando proporcionar aos alunos experiências e práticas significativas no processo do ensino e aprendizagem.

Esta pesquisa apresenta um ambiente virtual de aprendizagem de programação para a linguagem *Python*, que de acordo com Fangohr (2004) é indicado para disciplinas introdutórias por ser uma linguagem intuitiva e de fácil uso. O ambiente auxilia o aprendizado prático do aluno, seja na modalidade à distância ou em aulas de laboratório, e libera a maior parte do trabalho professor no acompanhamento e correção das atividades.

Este artigo está organizado em seis seções incluindo esta. A seção 2 apresenta trabalhos relacionados. A seção 3 apresenta o Labpy. A seção 4 apresenta aspecto de implementação. A seção 5 apresenta *interface* e funcionalidades. A seção 6 apresenta a conclusão e trabalhos futuros.

2. Trabalhos relacionados

O tema sobre o ensino de algoritmo e programação nos cursos de computação contínua em alta. Na literatura encontram-se vários ambientes de ensino-aprendizagem de algoritmo e programação, sendo alguns revisados abaixo (ver tabela 1).

Mota et al.,(2008) apresentam o JavaTool, uma ferramenta desenvolvida para

auxílio ao ensino de programação, sendo utilizada nas disciplinas iniciais de programação e nas aulas práticas de laboratório. Está centrado num interpretador para linguagem Java, que possibilita a animação e visualização detalhada da execução do algoritmo. As principais funcionalidades do JavaTool são: edição de código Java, compilação, depuração, visualização da animação do código, exibição dos dados em vários formatos (binário, hexadecimal e octal) e a visualização do histórico de execução da animação tanto gráfico como textual.

Guo (2013) apresenta o Online *Python* Tutor, uma ferramenta de visualização de programas para educação em ciência da computação. A ferramenta possui as seguintes características: (1) *python* como linguagem principal; (2) é executada em um navegador *web* sem qualquer instalação de *software* e/ou *plugin* e; (3) pode ser embutido em páginas de *web*, para geração de conteúdo educacional.

Edwards et al., (2014) apresentam o Pythy, um ambiente de programação baseado na *web* para *python.* O ambiente possui como características uma *interface* amigável; suporte de gerenciamento de tarefas; suporte interativo com tutores em tempo real; controle de versão e salvamento automático (*cloud storage*); suporte a exercício e manipulação de mídia (imagem e som).

Tabela 1. Comparação entre os trabalhos relacionados a esta proposta.

	Javatool	**Python Tutor**	**Pythy**	**LabPy**
Ano	Mota et al.,(2008)	Guo(2013)	Edwards et al.,(2014)	2018
Linguagem	Java	Python	Python	Python
Avaliação Automática	N	N	N	S
Refatoração de código	N	N	N	S
Múltiplas submissões	N	N	S	S
Provas	N	N	S	S
Competições entre grupos	N	N	N	S
Lista de Exercícios	S	N	S	S
Conteúdo para Estudo	N	N	S	S
Ambiente Web Próprio	N	S	S	S
Área do Professor	N	N	N	S
Área do Aluno	S	N	S	S

Diante desse contexto, a principal motivação deste trabalho é criação de uma

ferramenta para ensino de conceitos introdutórios de algoritmo e programação. O ambiente permite criar conteúdo com exemplo de códigos executáveis *in-loco* nos módulos; criar listas de questões associadas a módulos de conteúdo; gerar listas de exercícios com níveis de dificuldades; compor provas e lista de exercícios a partir de um banco de questões; avaliar de forma automática o *feedback* das respostas dos alunos; permite refatoração do código e; gerar notas para verificar o desempenho dos alunos de uma turma ao longo da disciplina tornando possível uma aprendizagem mais dinâmica aos alunos seja em atividades *web* ou em laboratório.

3. O ambiente Labpy

Uma das maiores dificuldades de um aluno de programação é entender e visualizar abstratamente cada passo da execução do programa. Aprender a gramática de uma linguagem de programação, consertar erros de sintaxe, desenvolver novos algoritmos, escrever um novo programa, depurar e consertar erros são os principais desafios de um aluno que inicia um curso de programação, em ordem crescente de dificuldade (Miyadera, 1999).

O LabPy é um ambiente interativo com o intuito de auxiliar os alunos no aprendizado de programação, sendo utilizado como uma ferramenta de auxílio ao professor para realizar atividades laboratoriais de programação. Embora existam várias ferramentas de ensino em *python* baseados na *web,* LabPy difere nos aspectos: (i) consultar o **conteúdo didático interativo**, segmentado em módulos baseados em assuntos específicos da linguagem e lógica de programação – conteúdo em páginas *web* com interpretador para fragmentos de programas – além da leitura estática o usuário testa "in-line" os fragmentos de código; (ii) visualizar e realizar análise acerca do funcionamento dos programas por meio da **programação interativa**; (iii) acompanhar o **conteúdo não-linear** sem ter a obrigatoriedade de passar por várias sessões anteriores para chegar a que pretende; (iv) submeter respostas a exercícios de programação, num ambiente com ***feedback* automático e** imediato com **múltiplas submissões** para uma mesma pergunta; v) **trabalhar iterativamente** com avaliação de programas não dual, pela refatoração de respostas, embora muitos sistemas atribuem um escore de 100% correto quando o resultado do programa submetido for igual ao da resposta de referência, no LabPy a avaliação tende a se aproximar incrementalmente do 100% correto. Um programa pode retornar o resultado correto, mas pode não ter sido bem escrito, o aluno tem oportunidade de refatorar a solução em direção da solução ideal, aprimorando e ressubmetendo; (vi) idem (v) para provas; (vii) responder **questões objetivas e subjetivas** (questões discursivas) também com *feedback* automático imediato, múltiplas submissões para exercícios subjetivos e/ou provas com avaliação automática a partir das listas de exercícios e questões; (viii) ser **acompanhado pelo professor**, possibilitando o professor acompanhar o desempenho individual do aluno ou da turma inteira.

Uma visão geral da arquitetura do ambiente é apresentada na Figura 1, a qual o professor a partir de uma *interface*, define as avaliações e exercícios, assim como as suas respostas. As questões podem ser apenas para o treino dos alunos ou utilizadas como parte de uma avaliação.

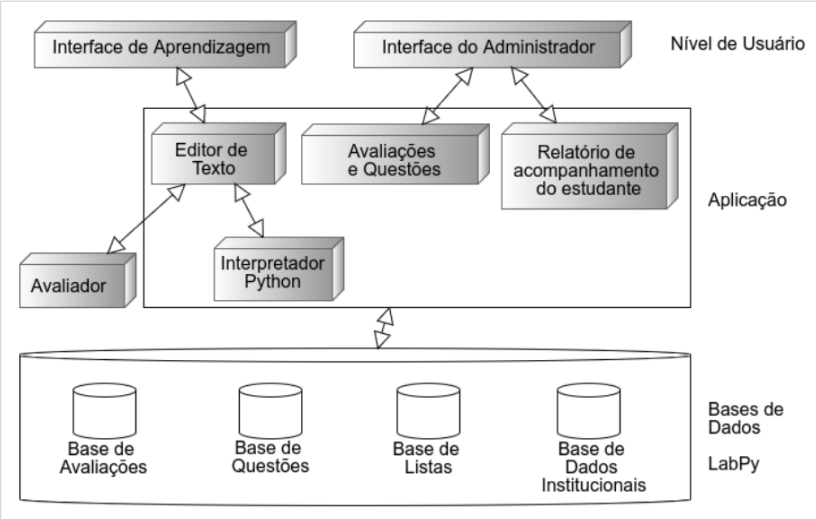

Figura 1. Arquitetura do ambiente de programação.

4. Aspectos de Implementação

O LabPy é um ambiente *web* desenvolvido usando tecnologias padrões. Para *frontend* foram utilizadas tecnologias como PHP, JavaScript, HTML5, CSS3 e o *framework* Bootstrap. Já no *backend* foi utilizado o *Skulpt*, um interpretador de linguagem *Python* desenvolvido em *Javascript* com o objetivo de executar os programas no próprio navegador, ou seja, não necessita de instalação de *softwares*, *plugins* ou suporte do lado do servidor, possui também um módulo de avaliação automática baseado em *n*-gramas para *feedback* imediato.

A Figura 2 mostra as etapas do que acontece durante a execução da ferramenta no *backend*. Após a entrada, o código passa pelo *Skulpt*, sendo compilado para *bytecode* e, em seguida para *javascript* sendo exibido na saída. Nesse momento o interpretador realiza as operações e atribuições especificadas no código por meio de um avaliador automático que utiliza um algoritmo baseado em *n*-gramas comparando a proximidade do código junto a resposta de referência do professor.

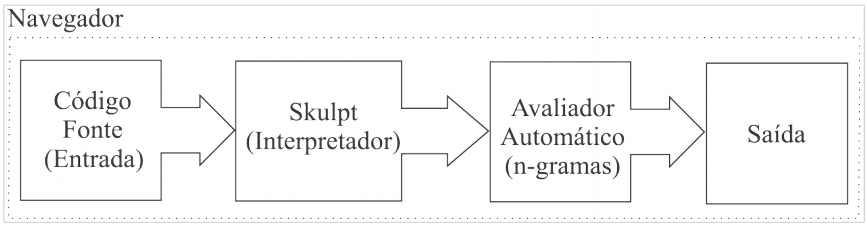

Figure 2. Etapas de processamento *backend*.

Um programa em *Python* pode ser avaliado diante de diversas métricas, de

acordo com o nível de *feedback*: 1) **Sintaxe:** o resultado emitido pode ser correto ou incorreto, caso o resultado seja incorreto, uma mensagem de erro é emitida por meio do interpretador presente no *browser*. 2) **Resultado da Execução:** após ser compilado, o resultado do programa é avaliado pelo sistema, caso esteja correto com a resposta de referência atribui-se uma nota final mínima, por exemplo 7,0. Após o resultado da saída da execução do programa ser correto, transpassa pela avaliação de qualidade da codificação da resposta para dar possibilidade para o aluno refatorar seu código buscando uma aproximação com a resposta de referência; esta medida é baseada em métricas de qualidade de software (Moreira e Favero 2009).

5. *Interface* e Funcionalidades

O conteúdo do LabPy é segmentado em módulos, baseados em assuntos específicos da linguagem. O conteúdo é editado pelo professor e compreende em um guia completo sobre a linguagem (Iniciante, intermediário e avançado). Em cada um destes há um determinado número de tópicos no formato de *link*, no qual o aluno tem acesso.

Cada módulo presente no sistema possui diversas questões para fixação de conceitos acerca do que foi estudado. Além disso, ao final de cada assunto é disponibilizado exercícios práticos referentes ao que foi aprendido, e assim o aluno possa ser avaliado corretamente.

5.1. Exercícios

O LabPy possui inúmeros exercícios que compõem uma base de questões. Tanto as perguntas quanto as respostas são previamente cadastradas pelo professor, por meio da área administrativa do sistema. As respostas de referência precisam ser cadastradas para permitir a avaliação automaticamente pelo sistema.

Na inserção do exercício o professor escolhe um dos módulos e o grau de dificuldade (fácil, médio e difícil); para os exercícios de programação são registrados dados para testar os códigos. Por exemplo, para o programa do fatorial, pode-se testar com -1, 0, 1, 5, 10. Forçando o aluno a fazer uma solução bem abrangente. Caso o código só funciona para valores maiores que zero o sistema informa o estudante que a solução está parcialmente correta, mas não cobre todos os testes.

Estando cadastrada, o aluno tem acesso à questão via um editor de texto com a função *syntax highlighting*, permite que os principais comandos em *Python* fiquem em evidência. A Figura 3 apresenta um exemplo de acesso a um exercício e sua execução.

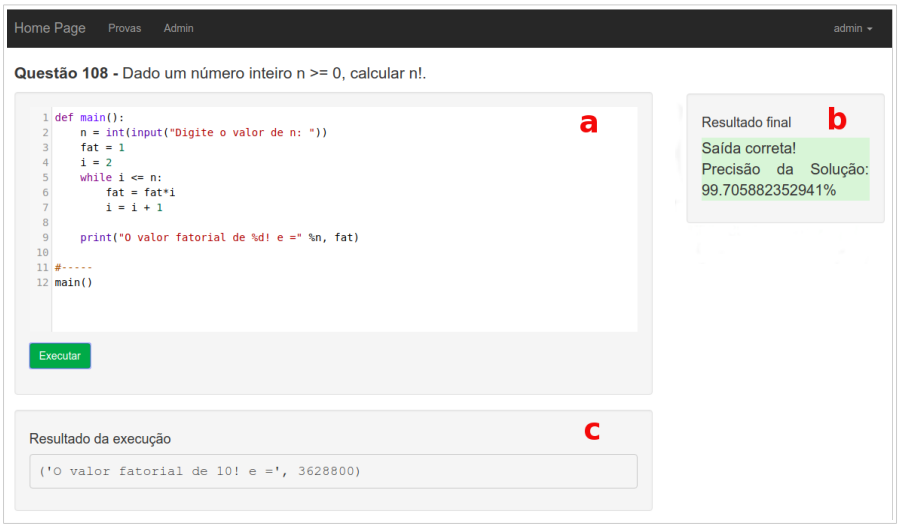

Figure 3. Exemplo de um exercício. a) A execução da questão é feita pelo editor; b) o *feedback* após a execução; c) saída gerada pelo exercício.

5.2. Área do Professor

O perfil do professor possui o gerenciamento das questões, das avaliações, do acompanhamento da turma por meio de relatórios de desempenho e a situação de cada aluno. O perfil permite cadastrar novos alunos no sistema por meio da inserção de uma turma completa (arquivo no formato .xml) ou de um usuário específico.

O perfil tem a possibilidade de visualizar e acessar duas áreas distintas, cada uma referente a uma funcionalidade específica:

1. **Área do aluno:** área principal do ambiente que contém os módulos, no qual os alunos também possuem acesso.
2. **Área principal do administrador:** acesso global a outras seções presentes no sistema (6), possibilitando a visualização rápida de turmas, usuários, exercícios, módulos e provas, ver Figura 4.

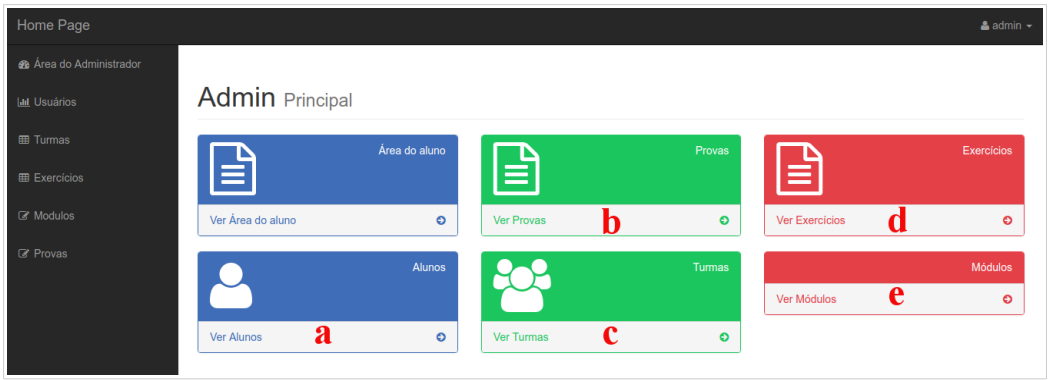

Figure 4. Área principal do administrador.

Permite ao professor **(a)** gerenciar seus alunos, podendo cadastrar, editar ou

excluir; **(b)** gerenciar as avaliações, criando, editando ou excluindo uma avaliação por completo, ou também pode realizar estas operações para questões individualmente. **(c)** gerenciar suas turmas, podendo cadastrar, editar ou excluir alguma turma. **(d)** gerenciar as questões presentes no sistema cadastrando, editando ou excluindo. Além disso, permite gerenciar as respostas referentes a cada questão. **(e)** gerenciar os módulos presentes no sistema, permitindo a criação, edição e exclusão de um determinado módulo.

5.3. Área do aluno

O ambiente apresenta uma área de avaliações visualizada pelo aluno ao selecionar a opção Provas. Quando uma avaliação é disponibilizada no sistema, o LabPy apresenta cada tarefa na tela no formato de *link*, onde o aluno possa ter uma flexibilidade na escolha das atividades que acharem mais importantes, sem a necessidade de seguir uma ordem específica.

Na área de estudo, o aluno visualiza o conteúdo disponibilizado e *links* que permitem o acesso a outras funcionalidades, sendo estes: (1) **Provas:** área destinada a todas as provas cadastradas pelo professor, com opção de redirecionamento para área de avaliação. (2) **Exercícios:** o aluno tem acesso aos exercícios no final de cada módulo. (3) **Vídeos:** disponível aos estudantes alguns *links* contendo vídeos sobre o assunto, para um aprendizado mais interativa. (4) **Referências Externas:** assim como vídeos e fontes de materiais externos, a sessão de conteúdo também conta com referências externas de artigos ou livros. Além disso, ao longo de cada módulo existe uma série de exemplos que podem ser executados na sessão. Quando o aluno executar o exemplo visualizará a saída gerada pelo código em questão. A Figura 5 representa um exemplo de uma saída gerada.

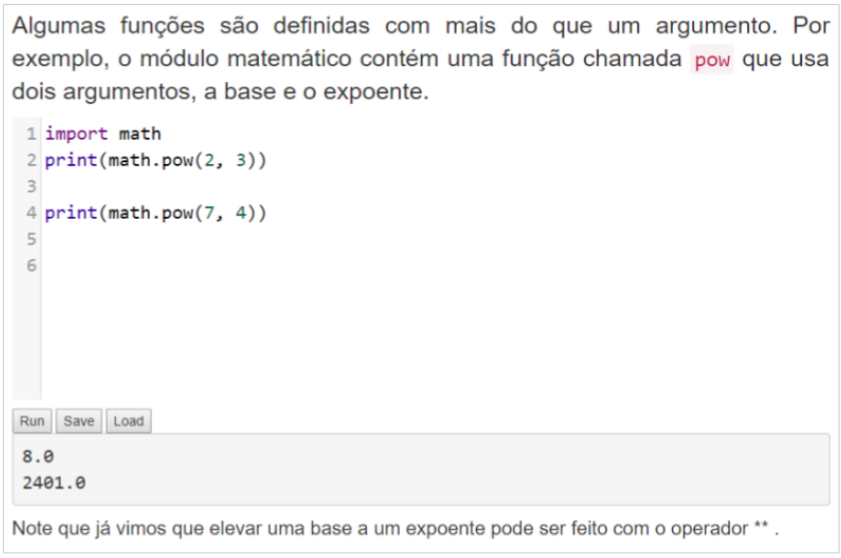

Figure 5. Saída gerada pela execução de um programa.

6. Conclusão e trabalhos futuros

Ao longo dos anos os ambientes virtuais de ensino ganharam relevância e seu uso se intensificou em universidades, isto se deve ao fato do avanço de novas tecnologias, como a *Web*, que permitiram a criação de sistemas interativos e eficazes, para auxiliar estudantes e professores no processo de ensino – aprendizagem. Além disso, a demanda por formatos inovadores de conteúdo no campo da educação aumentou consideravelmente, em especial em cursos de computação com a programação que realmente necessita de novos métodos de estudo e de práticas de laboratório.

De maneira geral, o LabPy fornece um ambiente com conteúdo focado no ensino de programação por meio da linguagem *Python*: (i) consultar o **conteúdo didático interativo:** segmentado em módulos baseados em assuntos específicos da linguagem e sobre lógica de programação; (ii) **praticar a programação interativa:** submeter respostas a exercícios de programação, num ambiente com *feedback* automático e imediato; (iii) **trabalhar iterativamente:** refatorar o código e submeter novamente; (iv) **responder questões objetivas e subjetivas** das listas de exercícios e/ou provas.

Dessa forma, um sistema deste tipo também apresenta um ganho significativo para o professor, liberando-o da atividade mais pesada de corrigir exercícios e acompanhar de perto cada estudante. Por outro lado como todo exercício receberá um *feedback* imediato; o estudante ganha tempo; não existe a necessidade de uma espera para saber se ele está indo na direção certa.

Como trabalhos futuros, consideramos aplicar o ambiente em uma turma real para analisar os dados. Além disso, realizar estudos para aprimorar as métricas da avaliação automática e elaboração de um estudo de usabilidade no ambiente.

7. Referências

Amaral Júnior, A. E. do; Sena, L. A.; Soares, L. O. A.; Ferreira, W. M.; e Maia, E. H. B. (2018). Lolgica: Uma Plataforma de Ensino Voltada para Linguagens de Programação. Revista da META.

Cunha, L. S. da; Tonetti, P.; Sanavria, C. Z. (2017). O Ensino de Informática no Brasil: Uma Análise da Produção Científica em Eventos da SBC (2010–2014). Anais do Computer on the Beach, p. 031-040.

Edwards, S. H.; Tilden, D. S.; Allevato, A. (2014). Pythy: improving the introductory python programming experience. In:Proceedings of the 45th ACM technical symposium on Computer science education. ACM, p. 641-646.

Fangohr, Hans. (2004) A comparison of C, MATLAB, and Python as teaching languages in engineering. In: International Conference on Computational Science. Springer, Berlin, Heidelberg. p. 1210-1217.

Guo, P. J. (2013). Online python tutor: embeddable web-based program visualization for cs education. In: Proceeding of the 44th ACM technical symposium on Computer science education. ACM. p. 579-584.

Limperos, A. M. et al. (2015). Online teaching and technological affordances: An experimental investigation into the impact of modality and clarity on perceived and actual learning. Computers & Education, v. 83, p. 1-9, 2015

Mota, M. P.; Pereira, L. W. K.; Favero, E. L. (2008). Javatool: Uma ferramenta para o ensino de programaçao. In: Congresso da Sociedade Brasileira de Computação. Belém. XXVIII Congresso da Sociedade Brasileira de Computação. p. 127-136.

Miyadera, Y.(2000).A programming language education system based on program animation. Proc. of ICEUT2000.

Paes, R.B.; Malaquias, R.; Guimarães, M.; Almeida, H. (2013). Ferramenta para a Avaliação de Aprendizado de Alunos em Programação de Computadores. In Anais dos Workshops do Congresso Brasileiro de Informática na Educação (Vol. 2, No. 1).

Prior, J. C. (2003).“Online assessment of SQL query formulation skills”. In Proceedings of the Fifth Australasian Conference on Computing Education. Adelaide, Australia.

Sirotheau, S. et al. (2001). Aprendizagem de iniciantes em algoritmos e programação: foco nas competências de autoavaliação. In: Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE).

Vargas, K. S.; Martins, J. (2005). Ferramenta para apoio ao ensino de Introdução à Programação. XIV Seminário de Computação, Universidade Regional de Blumenau, Santa Catarina.