Um estudo sobre as dificuldades no processo de aprendizagem de programação no Curso de Análise e Desenvolvimento de Sistemas na FAFICA – Faculdade de Filosofia, Ciências e Letras de Caruaru-PE

Márcia Valéria Rocha de Souza[1], **A. César C. França**[1]

[1]Faculdade de Filosofia Ciências e Letras de Caruaru (FAFICA)
R. Azevedo Coutinho – Petrópolis – Caruaru – PE - CEP 55030-240

{marciavr.souza,cesarfranca}@gmail.com

Resumo. *Para formar bons profissionais, uma instituição de nível superior deve oferecer cursos de qualidade para os seus alunos incentivando-o de forma lúdica para o aprendizado. O presente artigo trata de um estudo realizado na Faculdade de Filosofia Ciências e Letra de Caruaru sobre as dificuldades dos alunos do curso de Análise e Desenvolvimento de Sistemas do primeiro ao quarto período no processo aprendizagem. Foram apontadas as principais dificuldades e a sugestão de algumas ferramentas de apoio ao ensino dos principais conceitos de programação.*

Abstract. *In order to develop high-level professionals, a higher education course should offer good quality courses for their students, encouraging them to learn. This article reports a study conducted at the Faculty of Philosophy, Sciences and Letter of Caruaru (FAFICA) on the difficulties of students of Analysis and Systems Development from the first to the fourth period in programming learning. The survey data pointed out the major difficulties faced by the students, and we conclude suggesting potential tools to support the teaching of the key programming concepts.*

1. Introdução

Um bom profissional em programação deve possuir habilidades para codificar instruções executáveis por um computador. Portanto, durante os estágios iniciais de seu aprendizado, adquirir uma boa base dos conhecimentos básicos em linguagem de programação assim como o desenvolvimento em lógica facilita na construção de algoritmos executáveis.

O currículo de referência da Sociedade Brasileira de Computação (SBC) prevê que os cursos que tenham sistemas computacionais e suas aplicações como atividadefim, atividade-meio ou cursos de Licenciatura em Computação, deverão possuir em sua grade curricular a disciplina de linguagem de programação, sendo esta a que possui um acentuado grau de dificuldade por parte dos alunos [Mota et al. 2003].

O objetivo deste artigo é apontar as principais dificuldades dos alunos na aprendizagem de programação do curso de Análise e Desenvolvimento de Sistemas desta instituição. Espera-se que após a mensuração dos dados fiquem evidentes os pontos de dificuldade dos alunos podendo contribuir também para a tomada de decisões e ações de

professores e coordenadores para a melhoria do curso e consequentemente o aprendizado em sala de aula.

O artigo está organizado da seguinte forma: a Seção 2 apresenta a contextualização referente ao primeiro trabalho onde foi feito um estudo comparativo com ferramentas de auxílio ao aprendizado em programação; a Seção 3 trata-se da forma como foram coletados os dados para a pesquisa; na Seção 4 são apresentados os resultados da coleta dos dados e a discussão destes. Por fim, na Seção 5 estão as conclusões deste trabalho.

2. Contextualização

Em um trabalho realizado anteriormente foi apontado através da literatura, algumas dificuldades dos alunos iniciantes em entender a lógica de programação e desenvolver meios necessários para chegar à resolução de um problema. Elaborou-se uma pesquisa de algumas ferramentas disponíveis na internet, gratuitas, que podem auxiliar na compreensão. As ferramentas foram classificadas de acordo com os conceitos básicos como é apresentado na Tabela 1(vide Souza e França [2013]). Compararam-se as competências como sugere Gomes, Henriques e Mendes [2008] em um modelo construtivista de aprendizagem em três fases: na resolução de problemas diversos, reconhecimento da utilidade da programação e a capacidade de construção de algoritmos. Após esta etapa as ferramentas foram catalogadas em três níveis: básico, intermediário e avançado. A finalidade deste agrupamento é uma possível aplicação em sala de aula, para isso é necessário identificar quais principais dificuldades dos alunos do primeiro ao quarto período o qual se refere o presente artigo e, por fim, utilizar a ferramenta adequada aos problemas identificados.

Tabela 1. Conceitos trabalhados nas ferramentas avaliadas (vide Souza e França [2013])

	Compilação	Alocação de memória	Alocação vetorial	Tipos de dados	Operações Aritméticas	Operações Lógicas	Estr. de Controle de	Funções	Recursão	Depuração passo-a-passo
CodeMonster	*Não*	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	*Não*
FutCode	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	*Não*
Guido VanRobot	*Não*	*Não*	*Não*	*Não*	*Não*	**SIM**	**SIM**	**SIM**	*Não*	**SIM**
KidsRuby	*Não*	**SIM**	*Não*	*Não*	**SIM**	**SIM**	**SIM**	*Não*	*Não*	*Não*
RobotProg	*Não*	**SIM**	*Não*	*Não*	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**
TBC-AED	*Não*	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	*Não*	**SIM**
Visual Alg	*Não*	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**	**SIM**
Web Portugol	*Não*	**SIM**	**SIM**	*Não*	**SIM**	**SIM**	**SIM**	*Não*	*Não*	**SIM**

3. Método

Com o objetivo de analisar as dificuldades dos estudantes na aprendizagem em programação, no fim do primeiro semestre acadêmico de 2013, foi realizada uma pesquisa com os alunos do primeiro ao quarto período do curso de Análise e Desenvolvimento de Sistemas desta instituição. Aplicou-se um questionário presencial constituído de três partes: (i) dados demográficos, (ii) estudo de programação e (iii) auto avaliação. A primeira parte consiste em coletar informações referentes aos dados populacionais. A segunda parte visa avaliar o interesse do aluno em relação aos estudos

e o conhecimento das principais dificuldades no aprendizado em programação. A terceira parte do questionário propõe uma autoavaliação do aluno sobre alguns conceitos iniciais em linguagem de programação como compilação, alocação de variáveis e vetores, tipo abstrato de dados, operações de aritmética e lógica, estrutura de controle de luxo, funções e métodos, ponteiros e referências, recursão e depuração.

A coleta dos dados foi realizada entre os dias 06 e 10 do mês de maio. Foram aplicados, respectivamente, questionários para duas turmas do 1° e do 3° Período e uma turma do 2° e 4° Período. O número de matriculados nas disciplinas Programação I, Programação II, Programação Orientada a Objetos II e Programação Web, na devida ordem, nos 1°, 2°, 3° e 4° períodos foram de 198 (cento e noventa e oito) alunos, porém na aplicação do questionário, devido à ausência de uns, contabilizou-se 137 (cento e trinta e sete) respondidos totalizando, aproximadamente, 70% do total de alunos nestes períodos letivos.

4. Resultados e Discussão

O curso oferecido pela instituição pesquisada é de nível tecnólogo e oferece ao aluno o contato com linguagens de programação de alto nível largamente utilizadas no mercado profissional. Dado o foco da instituição em preparar rapidamente os alunos para o mercado de trabalho, as disciplinas de programação focam no aprendizado da linguagem Java® e no paradigma orientado a objetos.

Inicialmente observou-se uma redução gradual no número de alunos por período letivo. Gomes, Henriques e Mendes [2008] comentam que o desinteresse pode acontecer por dificuldades de interpretação ou devido os alunos se sentirem demasiados ansiosos para começar a codificar sem antes compreender os dados do problema, dificultando a fase seguinte, a construção do algoritmo. A dificuldade de empregar o raciocínio lógico, afirmam Barros, Delgado e Machion [2004], também gera um ambiente desmotivante, justificando esta redução de alunos através de uma possível evasão ou reprovação na disciplina.

Dificuldades pedagógicas e metodológicas

Jenkins [2002] cita as várias causas do insucesso generalizado em disciplinas de programação, como: o baixo nível de abstração, a falta de competências de resolução de problemas, a inadequação dos métodos pedagógicos aos estilos de aprendizagem dos alunos, entre outros; e afirma que as linguagens de programação possuem sintaxes adequadas para profissionais, mas não para aprendizes inexperientes.

Dentre os entrevistados, apenas 7 (sete) estudantes são atuantes no mercado profissional, e apenas quatro destes estão acima dos 26 anos. Entre as principais dificuldades apontadas deste grupo estão o método de ensino, tempo para dedicar aos estudos e apenas um deles registrou dificuldades no aprendizado com relação à insuficiência de material e compreensão de linguagens de baixo nível. Pouco mais de 80% dos alunos com até 18 anos estão no 1° período, e as principais dificuldades estão relacionadas com o aprendizado da lógica de programação seguido da metodologia de ensino dos professores. A lógica de programação e a metodologia aplicada pelos professores também estão entre os principais problemas citados pelos alunos da faixa etária dos 19 aos 25 anos, seguido do tempo para a dedicação aos estudos e o conhecimento na língua inglesa. Souza e França [2013] comentam que as dificuldades

podem estar associadas tanto à forma de ensino, como também, na familiaridade prévia com noções básicas de criação de algoritmos.

Familiaridade prévia e disponibilidade de tempo para dedicação aos estudos

Com o propósito de identificar a familiaridade dos alunos com a programação antes da graduação, foi questionada com qual idade e qual foi a primeira linguagem de contato. Aproximadamente 44% dos estudantes responderam que tiveram seu primeiro contato com programação dos 8 aos 17 anos e 81 alunos tiveram como primeira linguagem o Java, seguido da linguagem C. Em relação aos alunos que já trabalham na área e o tempo de dedicação por semana, 4 praticam de 1 a 3h, dois praticam de 7 a 9h e apenas um dedica 10h ou mais de práticas em programação. Dos 137 alunos que responderam a pesquisa, 85 deles praticam programação entre zero a três horas por semana e um grupo de 52 alunos praticam mais de 4 horas. De acordo com o ranking do site Tiobe [2013] a linguagem Java aparece em 2° lugar por dois anos consecutivos entre as mais utilizadas no mercado.

A compreensão de outras linguagens já serve como um diferencial no mercado de trabalho. Dentre elas, as mais citadas são o HTML/CSS, o C/C++ e o Javascript. Das linguagens que receberam menos de dez votos estão o Ruby, ActionScript, VBNet e o Cobol/Fortran, Objective C e o Prolog. As linguagens Perl e Lisp/Haskel não receberam nenhum voto. Entre as citadas pelos alunos que não continham na lista do questionário estão o Portugol, ShellScript, Visualg, ADVPL, Progress 4gl, Basic, Natural Clipper, Lua, Assembly e Grails.

Lógica de programação e metodologia de ensino

A literatura concorda que a metodologia de ensino é um desafio relevante para o ensino básico de lógica de programação. Segundo Gomes, Henriques e Mendes [2008], a um nível mais básico, o ensino das linguagens de programação tem como propósito permitir que os alunos desenvolvam as suas capacidades, adquirindo os conhecimentos básicos necessários para conceber programas capazes de resolver problemas reais simples. Sendo assim, somente após o aprendizado destes conceitos, o estudante pode travar contato com uma linguagem de programação concreta [Santos e Costa 2006]. Almeida, Castro e Castro [2006] afirmam que um dos principais componentes na aprendizagem de programação é a organização das habilidades para a resolução de problemas e como tais habilidades são construídas, envolve identificar as dificuldades encontradas pelos alunos. Porém, em uma sala de aula, o nível de conhecimento e assimilação dos conteúdos passados é diferente de aluno para aluno o que sugere um ensino individualizado.

No entanto, Falckembach e Araújo [2005], discutem que esse tipo de ensino, personalizado e individualizado, entra em choque com o pressuposto básico da educação tradicional que é a padronização. Segundo Falckembach e Araújo [2005] a forma de ensino dentro da sala de aula é a mesma para todos os alunos, pois, é extremamente difícil para um professor levar em consideração o perfil, as metas, as necessidades e o nível de conhecimento de cada aluno, de modo a proporcionar a cada um, um ensino adaptado.

Disponibilidade de material de apoio ao aprendizado

A biblioteca da instituição possui um acervo vasto para o estudo em linguagens de programação e é eminente a amplitude de materiais relativos ao tema encontrados na internet que se tornou um grande aliado ao aprendizado. Por esse motivo, dificuldades registradas pelos alunos relativas à falta de material para estudo não são necessariamente justificáveis. Por outro lado, estes dados podem revelar uma ineficácia do material disponível às necessidades de aprendizado dos alunos. As ferramentas apresentadas na Tabela 1, por exemplo, oferecem estratégias interativas que se propõem a ser mais eficientes do ponto de vista da velocidade e da consistência do aprendizado. No entanto, não encontramos dados disponíveis que possam reforçar esta proposta.

Dificuldades relacionadas à complexidade dos conceitos de programação

Na terceira etapa do questionário, auto avaliação, os estudantes responderam sobre o grau de conhecimento em noções básicas de programação. As análises dos dados foram feitas por período já que alguns dos conceitos ainda não foram passados para alunos do primeiro período como Vetores, Tipo Abstrato de Dados, Funções e Métodos, Ponteiros, Referências e Recursão. Estão apresentados, neste tópico, apenas os conceitos que foram registrados foram do domínio dos alunos.

A ideia de abstração de dados refere-se a uma modelagem de uma estrutura de dados de acordo com seu funcionamento. No segundo período, 68,42% dos estudantes conhecem o conceito, mas desconhece a forma de aplicação. Menos da metade dos alunos do terceiro período dominam o conceito. No quarto período houve um empate entre os que também sabem do conceito, mas não sabem da aplicação e dos que dominam o assunto.

Tabela 2. Porcentagem dos estudantes por período letivo que dominam os conceitos citados

Período	N° Alunos	Compilação	Alocação de variáveis	Alocação vetorial	Tipos dados abstratos	Operações Aritméticas	Operações Lógicas	Estr. Controle de -	Funções e Métodos	Ponteiros e Referências	Recursão	Depuração
1°	62	54,8%	45,1%	1,6%	6,4%	66,1%	30,6%	35,4%	9,6%	6,4%	1,6%	20,9%
2°	19	84,2%	94,7%	78,9%	15,7%	84,2%	47,3%	100%	52,6%	21,5%	5,2%	10,5%
3°	28	71,4%	82,1%	60,7%	46,4%	96,4%	67,8%	92,8%	64,2%	25%	10,7%	35,7%
4°	28	82,1%	78,5%	46,4%	39,2%	100%	78,5%	85,7%	71,4%	32,1%	21,4%	39,2%

Ponteiros, Referências, Recursão e Depuração são os conceitos de mais dificuldades entre os alunos. A depuração, apesar de ter sido discutida em sala de aula no primeiro período, 38,70% dos alunos nunca ouviu falar. No segundo período, a maioria, 47,36%, sabe do que se trata, porém não sabem usar. No terceiro período, houve um empate entre os que conhecem, mas não sabem da funcionalidade e os que dominam o conceito. O mais alarmante é o conceito de recursão, a grande maioria em todos os períodos nunca ouviram falar. O conceito de ponteiros e referências, do segundo ao quarto período, mais de 50% respondeu que sabem do que se trata, mas não sabem usar.

Problemas no aprendizado de conceitos de programação x Ferramentas de apoio

Na última parte do questionário os alunos avaliam sua compreensão dos conceitos citados em cinco opções, os extremos correspondem ou a total abstração ou o domínio. O domínio seria a compreensão e os demais apontam um déficit a ser resolvido. Analisando as dificuldades dos alunos com a referenciada Tabela 1, podemos identificar quais das ferramentas, estudadas anteriormente, poderão ser aplicadas. O primeiro dos conceitos trabalhados segundo as competências é o de compilação, apesar de mais de 50% da turma do primeiro período dominar o conceito, uma parcela considerável de 28 alunos sente dificuldades entre a usabilidade e a utilidade na aplicação. Das ferramentas citadas na Tabela 1, apenas o FutCode trabalha este conceito, porém como no FutCode também é aplicado conceitos ainda não estudados pelas turmas do primeiro período, convém analisar uma forma desta ferramenta contribuir na aprendizagem.

Menos da metade dos alunos dominam o conceito de alocação de varáveis no primeiro período, 34 dos 62 alunos responderam que possuem algumas dificuldades neste conceito. Todas as ferramentas citadas na Tabela 1, com exceção do Guido VanRobot, trabalham o conceito de alocação de variáveis, porém o TBC-AED trabalha conceitos mais avançados e não possuem uma versão para iniciantes, o CodeMonster e o Kids Ruby utilizam as linguagens javascript e Ruby, respectivamente, o que seria desaconselhável para alunos do primeiro período que ainda não tiveram nenhum contato com linguagens de programação em sala de aula. As demais turmas apresentaram um nível satisfatório de compreensão do conceito de alocação de variáveis.

Nos terceiro e quarto período, onze e quinze alunos, respectivamente, dos 28 entrevistados em cada turma, responderam que sentem alguma dificuldade em alocação vetorial. Entre as ferramentas que podem ser aplicadas para este conceito segundo a Tabela 1, estão: CodeMonster, FutCode, TBC-AED, VisuAlg e Web Portugol. No conceito de tipo abstrato de dados, do segundo ao quarto período, mais da metade dos que responderam o questionário afirmaram que sentem algum bloqueio. As ferramentas que podem ser indicadas para servir como apoio são: CodeMonster, FutCode, TBCAED e VisuAlg.

Sobre o conceito de operações aritméticas, há uma distribuição ínfima do segundo ao quarto período, de alunos com bloqueio neste conceito. Já no primeiro período, 21 alunos dos 62 entrevistados possuem dificuldades. Em operações lógicas, mais da metade no primeiro e segundo períodos possuem dificuldades, no terceiro e quarto períodos somam 15 alunos. Já no conceito de controle de fluxo, o primeiro período tem mais dificuldades, 40 alunos dos 62 entrevistados. Nos terceiro e quarto períodos somam 06 alunos que sentem algum bloqueio. Todas as ferramentas citadas na Tabela 1 trabalham estes três conceitos, exceto o Guido VanRobot.

Os conceitos de funções, métodos, ponteiros, referências e recursão, são trabalhados inicialmente no segundo período, onde 9 dos 19 alunos apontaram dificuldades em funções e métodos. No terceiro e quarto períodos, respectivamente 10 e 8 sentem dificuldades. De acordo com a tabela 3 deste artigo, observa-se que os conceitos mais críticos são os de ponteiros, referências e recursão. O conceito de recursão chega a ser o mais grave onde mais de 50% de todas as turmas respondeu que nunca ouviu falar. Infelizmente a nossa tabela referenciada de um estudo anterior não avalia as ferramentas com o conceito de ponteiros e referências. As ferramentas

CodeMonster, FutCode, RobotProg e VisuAlg são as únicas que trabalham Funções, Métodos e Recursão.

A depuração é um processo de análise de código passo a passo e aplicável desde o primeiro período. Porém, mais de 50% em todas as turmas relataram algum tipo de dificuldade neste conceito. E das ferramentas que podem se aplicadas estão o Guido VanRobot, RobotProg, TBC-AED, VisuAlg e Web Portugol.

Segundo as competências trabalhadas sugeridas por Gomes, Henriques e Mendes [2008], aponta-se o FutCode como uma ferramenta educacional que possibilita o aluno reconhecer alguns dos principais conceitos em programação. Na Tabela 3, estão as análises das habilidades dos alunos, onde eles avaliaram suas competências em programação ficando notável a dificuldade dos alunos em resolver problemas reais. Sugerindo, então, o FutCode ou uma adaptação desta ferramenta para auxiliar na compreensão dos alunos.

Tabela 3. Análise das Habilidades

Períodos	1° *(n=62)*	2° *(n=19)*	3° *(n=28)*	4° *(n=28)*
Dificuldade na construção de algoritmos simples	14%	5,26%	7,14%	7,14%
Constrói algoritmos mas não visualiza a utilidade	26%	26,31%	25%	32,14%
Programa bem e visualiza a utilidade mas tem limitações para resolver problemas reais	50%	57,89%	50%	42,85%
Programa bem e visualiza a utilidade e é capaz de resolver problemas reais	8%	10,52%	17,85%	17,85%

No trabalho anterior, as ferramentas foram categorizadas em três níveis de complexidade. Iniciantes, onde são trabalhados os conceitos básicos como operações aritméticas, lógicas e estrutura de controle, focando a construção de algoritmos, podendo ser indicada para turmas iniciantes, como o primeiro período. Entre as ferramentas citadas para o nível iniciante estão o RobotProg e o Guido VanRobot. As ferramentas de nível intermediário dedicam-se aos conceitos de alocação de memória em variáveis e vetores, tipo abstrato de dados e funções. Com o propósito de trabalhar com linguagens de programação reais, passaria a transparecer para o aluno a utilidade da programação. Seriam as ferramentas de nível intermediário o CodeMonster e o KidsRuby. Para o nível avançado, a proposta é fazer com que os alunos visualizem a solução de problemas em domínios reais. O FutCode atende o requisito onde são trabalhados conceitos avançados, como compilação e recursão, este último com o pior índice registrado na instituição.

5. Conclusão

De acordo com os dados apresentados, constata-se que há um déficit no aprendizado dos alunos da Faculdade de Filosofia Ciências e Letras de Caruaru. A proposta desde artigo é contribuir com a instituição em oferecer para os seus alunos um curso satisfatório. A procura pela graduação em Análise e Desenvolvimento de Sistemas é grande, mas por se tratar do aprendizado de conceitos complexos, é significativo o número de alunos que desistem ao longo do curso, como pudemos observar nos números dos matriculados do primeiro ao quarto período. Outro agravante é como um aluno chega ao quarto período sem saber dos conceitos básicos de programação como alocação de uma variável, por

exemplo. Neste artigo, estão em evidência as principais dificuldades dos alunos e espera-se que com esses dados, coordenadores e professores possam tomar medidas para reduzir a problemática. A contribuição central deste estudo consiste na avaliação da adequação de meios alternativos de ensino de programação, evidenciando que estes podem trazer resultados positivos tanto para os alunos quanto para a instituição. Ressaltando a queixa da maioria dos entrevistados, é necessário que os docentes possam acompanhar os alunos no processo aprendizagem, identificar as dificuldades e dispor de estratégicas de ensino simples, como as ferramentas apresentadas, que de forma lúdica acabam contribuindo em sala de aula.

Embora este artigo relate um passo fundamental para a adoção de ferramentas de apoio ao aprendizado em cursos de programação, os dados sobre a influência de tais ferramentas sobre o real aprendizado dos alunos ainda são insipientes. Pesquisas futuras devem investigar a adoção prática de uma ou mais destas ferramentas em sala de aula, e avaliar se os benefícios propostos se confirmam ou não.

Referências

Almeida, N. F. A., Castro, T., Castro, A. N. (2006). "Utilizando o Método Clínico Piagetiano para Acompanhar a Aprendizagem de Programação". http://www.lbd.dcc.ufmg.br/colecoes/sbie/2006/019.pdf, Junho 2013.

Barros, L. N., Delgado, K. V., Machion, A. C. G. (2004). "An ITS for programming to explore practical reasoning". http://goo.gl/z4f50, Junho 2013.

Deitel, H. M.; Deitel, P. J. (2010). "Java Como Programar". Ed. Pearson Prentice Hall, Junho 2013.

Falckembach, G. A. M., Araújo, F. V. (2005). "Aprendizagem de algoritmos: dificuldades na resolução de problemas". http://www.fabricioviero.com.br/artigos/a4_siie.pdf, Junho 2013.

Gomes, A., Henriques, J., Mendes, A. J. (2008). "Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores", http://goo.gl/DGLXh, Junho 2013.

Jenkins, T. (2002). "On the difficulty of learning to program", http://www.ics.heacademy.ac.uk/Events/conf2002/tjenkins.pdf, Junho 2013.

Mota, M. P., Brito, S. R., Moreira, M. P. and Favero, E. L. (2009) "Ambiente Integrado à Plataforma Moodle para Apoio ao Desenvolvimento das Habilidades Iniciais de Programação", http://goo.gl/dFHKS. Junho 2013.

Santos, R. P. ; Costa, H. A. X. (2006). "Análise de Metodologias e Ambientes de Ensino para Algoritmos, Estruturas de Dados e Programação aos Iniciantes em Computação e Informática." http://goo.gl/RcCjk, Junho 2013.

SBC - Sociedade Brasileira de Computação (2003). Currículo de Referência da Sociedade Brasileira de Computação para Cursos de Graduação em Computação e Informática". http://www.sbc.org.br. Junho 2013.

Souza, M. V. R., França, A. C. C. (2013). "Ferramentas de Auxílio ao Aprendizado de

Programação: Um Estudo Comparativo". http://www.erbase2013.itatechjr.com.br/index.php/programacao/weibase. Junho 2013.

Tiobe. (2013). "TIOBE Programming Community Index for july 2013". Disponível em <http://www.tiobe.com/>. Acessado em Julho 2013.