Aprendizagem da Programação: Problemas e Soluções

Learning Programming: Problems and Solutions

Martinha Piteira, Instituto Universitário de Lisboa (ISCTE-IUL) / Instituto Politécnico de Setúbal - ESTSetúbal, Portugal, martinha.piteira@estsetubal.ips.pt

Carlos J. Costa, Instituto Universitário de Lisboa (ISCTE-IUL), Portugal, carlos.costa@iscte.pt

Resumo

A aprendizagem da programação por alunos iniciantes é uma tarefa difícil devido à natureza da matéria, sendo muitas vezes uma causa de desmotivação que leva à desistência dos cursos de programação. Neste contexto este artigo identifica os principais problemas e soluções propostas, referidas na literatura. É igualmente apresentado um estudo empírico para identificação das dificuldades percecionadas por alunos e professores relativamente a conceitos de programação, contextos de aprendizagem, situações de aprendizagem e materiais/recursos. Com base na revisão da literatura e estudo empírico conduzido é proposta uma solução de aprendizagem suportada por um ambiente tangível. Através desta experiência pretende-se identificar em que medida a solução proposta influencia a perceção das dificuldades na aprendizagem da programação.

Palavras-chave: Aprendizagem, Programação, Ambientes Tangíveis

Abstract

Programming learning for beginners is a difficult task due to the nature of the subject, often leading to demotivation and withdrawal of programming courses. In this context, this article identifies the main problems and proposed solutions that are reported in the literature. It also presents an empirical study to identify the difficulties perceived by students and teachers in relation to programming concepts , learning contexts , situations and learning materials / resources. Based on the literature review and empirical study conducted, a learning solution supported by a tangible environment is proposed. Through this experience we intend to identify to what extent the proposed solution influences the perception of difficulties in programming learning.

Keywords: *Learning , Programming , Environments Tangibles*

1. INTRODUÇÃO

Programar é uma das competências especialmente requeridas nos nossos dias. É também uma das principais áreas no ensino da informática, contribuindo para a aquisição das competências requeridas, quer ao nível prático quer conceptual. Contudo, a literatura reporta que os alunos, que iniciam a aprendizagem nessa área enfrentam dificuldades de natureza diversa. Essas

ISSN 2183-489X
DOI http://dx.doi.org/10.18803/capsi.v14.46-67

dificuldades levam com frequência à desistência da disciplina ou cursos, com as naturais implicações para alunos e instituições [Denning et al. 2005], [Kinnunem et al. 2006].

Diversos trabalhos de investigação têm sido conduzidos com o objetivo de estudar as dificuldades associadas ao ensino da informática, nomeadamente as dificuldades na aprendizagem da programação, ex: [Milne et al. 2002], [Lahtinen, et al. 2005], [Robins et al. 2003], [Winslow 1996], [Eckerdal et al. 2005], [Lister et al. 2004], [Ma 2007].

Na literatura os estudos conduzidos reportam-se a diferentes áreas do ensino da informática. Pears e colegas [Pears et al. 2007] identificaram que as principais áreas estudadas são: currículo, pedagogia, escolha da linguagem e ferramentas de suporte à aprendizagem. Conduziram igualmente um estudo de investigação no qual identificam a necessidade de aprofundar mais a investigação: estudos no ensino; aprendizagem e avaliação; estratégias educacionais e instituições; problemas e soluções.

> O presente trabalho de investigação foca-se em duas das áreas sugeridas por Pears e colegas [Pears et al. 2007], problemas e soluções.

A organização do presente trabalho de investigação é a seguinte: na secção 2 é feita uma breve revisão dos problemas e soluções na aprendizagem da programação, na secção 3 é descrito um estudo empírico que tem o propósito de identificar as dificuldades percecionadas dos alunos no primeiro ano do ensino superior, na secção 4 é proposta uma solução concetual e descrita a respetiva implementação. Por último são apresentadas as considerações finais e trabalho futuro.

2. A Aprendizagem da Programação: Breve Revisão dos Problemas e Soluções

2.1. Problemas na Aprendizagem da Programação

O ensino da informática é uma das áreas que requer, pela constante atualização tecnológica e pelas características atuais dos alunos, um esforço permanente de acompanhamento por parte dos educadores. Trabalhos de investigação focados na identificação dos vários fatores envolvidos na aprendizagem são um contributo relevante para a área. Diversos estudos têm sido conduzidos com os mais variados objetivos: estudar as dificuldades na aprendizagem, (ex: [Milne et al. 2002], [Lahtinen et al. 2005], [Robins e al. 2003], [Winslow 1996] e [Schulte et al. 2010]; identificar modelos mentais dos alunos iniciantes, [Ma 2007]; propor soluções baseadas em ferramentas

educacionais, [Eckerdal et al. 2005], [Lister et al. 2004]; identificar o impacto da linguagem de programação que é ensinada, [Winslow 1996], entre outros.

Milne e Rowe [Milne et al. 2002] realizaram um trabalho de investigação com o objetivo de identificar as dificuldades percecionadas por alunos e professores na aprendizagem da programação. Os autores focaram o seu estudo na identificação das dificuldades relativas aos conceitos de programação. Os conceitos identificados como os mais difíceis foram ponteiros e alocação dinâmica da memória. A deficiente compreensão dos alunos, de como a memória do computador funciona após a execução dos programas é apontado como um dos fatores para os alunos bloquearem na compreensão total dos conceitos. Esse bloqueio apenas poderá ser ultrapassado quando adquirirem um modelo mental adequado de como os programas funcionam, do que é guardado na memória e de como os objetos se relacionam entre eles na memória. Também Lahtinem e colegas [Lahtinem et al. 2005] conduziram um trabalho de investigação similar concluindo que os conceitos que requerem uma maior abstração são aqueles que apresentam uma maior dificuldade na sua aprendizagem.

Um dos fatores que poderá contribuir para uma melhor compreensão, entre outros, é a prática. [Lahtinem et al. 2005]. Alunos e professores consideram que as aprendizagens práticas da programação são as mais úteis. Argumentam os autores que quanto mais concretas as situações de aprendizagem forem, mais a aprendizagem se faz. Aprender a fazer deveria ser uma prática em todas as áreas. Os alunos até podem compreender os conceitos individualmente mas apresentam dificuldades quando têm que os aplicar em conjunto. Dado que, muitas das dificuldades estão relacionadas com questões mais avançadas e não propriamente com os conceitos em si. Lahtinem, Ala-Mutka e Järvinen [Lahtinem et al. 2005], referem igualmente que materiais/recursos e situações de aprendizagem deveriam ser mais orientados para as competências de desenvolvimento do programa, modificação e depuração.

Robins e colegas [Robins et al. 2003] conduziram um trabalho de revisão da literatura e discussão sobre o ensino e a aprendizagem da programação onde sugerem que o foco não deve ser no professor, mas sim na aprendizagem do aluno através de uma efetiva comunicação entre aluno e professor, com o objetivo de aprofundar os princípios da aprendizagem e as competências, criando assim, alunos independentes e refletivos, proporcionando-lhe objetivos claros, estimulando-lhes o interesse, envolvendo-os na aprendizagem através do material/recurso do curso, e com avaliações apropriadas e retorno das suas ações. Continuam os autores referindo Winslow [Winslow 1996], que a boa pedagogia requer que o professor mantenha os factos iniciais, modelos e regras simples, e apenas expandir e refiná-las à medida que o aluno ganhar experiência. Schulte

e Busjahn, [Schulte et al. 2010], referem no seu trabalho de investigação, que tarefas que suportem o desenvolvimento de um modelo de domínio são necessários. Atividades de aprendizagem que envolvam um simples ler exemplos de código para recordação ou para resumir ou documentar não ajudam os alunos a desenvolver modelos de domínio. Em contrapartida, os autores referem que as atividades que envolvam modificar ou reutilizar o exemplo para atender a diferentes necessidades de aprendizagem podem serem particularmente úteis para os alunos desenvolverem esse modelo de domínio.

Costa e Aparício [Costa et al. 2014] referem que existem um conjunto de outras razões que contribuem para as dificuldades da aprendizagem da programação como originalidade, competências requeridas, estruturar / desenhar o programa, a escolha da linguagem, a duração do curso e as características individuais dos alunos. Outros aspetos referidos são os modelos mentais ineficientes que os alunos iniciantes na aprendizagem da programação apresentam com os consequentes fatores motivacionais e de autoestima.

Na tabela 1 são apresentadas as principais dificuldades referidas pelos diversos autores.

	Menor Experiência	Novidade	Muitas Competências	Desenhar o Programa	Compreensão do Programa	Linguagem de Programação	Duração do curso – tempo	Diferentes Capacidades	Atitudes Face à Programação	Fatores Cognitivos	Autoconfiança	Motivação	Modelos Mentais Ineficientes
Wilson, 2001				√				√					√
Jenkins, 2002		√	√	√	√	√			√	√			
Milne et al. 2002			√		√		√			√			

Robins et al. 2003	√		√	√	√		√	√	√	√			
Lahtinen et al. 2005			√	√	√		√	√		√			
Ala-Mutka et al. 2005			√	√	√	√		√		√			
Sajaniem et al. 2006			√	√	√		√	√		√			
Gomes et al. 2007			√	√	√	√	√	√	√		√	√	√
Mow 2008			√	√	√		√	√	√	√			√
Radosevic et al., 2009			√	√	√		√	√	√	√			
Renumol et al. 2009			√	√	√		√	√	√	√			
Cummings 2010		√	√	√	√	√	√	√					
Kordaki, 2010	√		√	√	√	√	√	√					
Pat et al. 2001	√								√		√		

Rahmat, et al., 2011	√		√									√	√
Ma et al, 2011	√		√									√	√

Tabela 1 - Dificuldades na Aprendizagem da Programação

2.2. Soluções Propostas para a Aprendizagem da Programação

Variadas são as abordagens na literatura relacionada com o ensino da informática e que pretendem contribuir para a resolução das dificuldades na aprendizagem da programação. A utilização de ferramentas, ambientes gráficos, e linguagens de programação alternativas são algumas das abordagens que têm sido utilizadas para incentivar o interesse e melhorar o desempenho da aprendizagem da programação.

A maioria dessas abordagens podem ser categorizadas em ferramentas narrativas, ferramentas gráficas para treino de código, abordagens que utilizam linguagens alternativas, ou exercícios mais apelativos que permitem ao aluno envolver-se na sua resolução e consequentemente gerar um maior interesse pela aprendizagem [McWhorter 2008].

Concretamente as abordagens que pretendem contribuir para um maior envolvimento e aprendizagem centrada no aluno são as abordagens que integram artefactos tangíveis como forma de estender a interação dos alunos, não os limitando à utilização de um ecrã de computador, [McWhorter 2008], [Brauner et al. 2010].

A utilização de artefactos tangíveis na aprendizagem de programação oferece significantes vantagens. Por exemplo, a visualização permite especificar o problema mais facilmente. Observar o artefacto, como por exemplo, um robot num ambiente físico torna a correção de erros mais fácil. A visualização realça um dos processos do cérebro humano mais importantes que é a entrada visual. Contribui também como facilidade na programação, é divertido e um dos maiores contributos é na pedagogia [McWhorter 2008].

Neste contexto descrevemos algumas das soluções, focando-nos nas ferramentas. Variadas soluções são referidas na literatura e podemos categorizá-las em ambientes de visualização e ambientes baseados em artefactos tangíveis.

Nas ferramentas de visualização são diversos os estudos empíricos referidos na literatura. Por exemplo, A*lice*, [Cooper et al. 2000] e *Object Karel*, [Xinogalos et al. 2006], são duas das ferramentas que utilizam uma abordagem baseada em metáforas (micromundos). Permitem ao aluno através de um ambiente gráfico criar, definir comportamentos e interagir com esses objetos. Raptor [Carlisle et al. 2005], Jeliot [Moreno et al. 2007], Verificator [Radosevic et al.

2009], Jype [Helminem et al. 2010)],Ville [Rajala et al. 2007] e UUhistle [Sorva et al. 2010], são igualmente ferramentas referidas na literatura que foram propostas como soluções para contribuir para a resolução de problemas na aprendizagem.

As soluções descritas anteriormente têm como objetivo providenciar aos alunos iniciantes na aprendizagem da programação ambientes amigáveis e efetivos na aprendizagem, contudo esses ambientes apresentam limitações ao confinar os alunos aos constrangimentos de um ecrã [Helminem et al. 2010]. Atendendo a essa limitação outras abordagens que ensinam conceitos de programação utilizando sistemas físicos do mundo real têm sido introduzidas e com um crescente interesse na sua utilização, (ex: [Kynigos 2008] e [Arlegui et al. 2008]. Exemplos dessas soluções são a utilização dos micros- robots como suporte à aprendizagem.

Por exemplo McGill (McGill 2012) conduziu um estudo que teve como objetivo estudar os efeitos motivacionais gerados pela utilização de robots físicos na aprendizagem da programação. Também William McWhorten [McWhorter 2008] conduziu um estudo com o objetivo de identificar a efetividade da utilização de atividades robóticas através dos Lego Mindstorms, e em que medida essas atividades influenciavam a aprendizagem autorregulada na disciplina de introdução à programação na University of North Texas. Brauner e colegas [Brauner et. 2010] conduziram também um estudo que teve como objetivo comparar o efeito da utilização de robots tangíveis com a utilização de ferramentas de visualização numa disciplina de introdução à programação. Os resultados mostraram que a utilização dos robots é benéfica para a aprendizagem.

Apesar dos benefícios da utilização dos robots na aprendizagem da programação um dos aspetos referidos na revisão da literatura como menos positivo é a necessidade na compreensão de aspetos técnicos relacionados com a robótica, desviando a atenção do aluno da aprendizagem principal para pormenores secundários. Com a imposição dessa compreensão prévia os alunos que per si, já não se sentem muito motivados, podem perder o interesse pela aprendizagem numa fase inicial do processo, [Braught 2012], [Blank 2005].

Braught [Braught 2012], propõe no seu trabalho uma biblioteca para múltiplas plataformas como forma de permitir acesso e manipulação ao hardware do robot, como sensores e atuadores,

tornando assim mais fácil a interação com o curso e diminuindo o tempo e a necessidade de compreensão dos detalhes robóticos, não criando obstáculos que prejudiquem o curso. Contudo esta biblioteca destina-se a ser utilizada em conceitos avançados de programação e não por alunos iniciantes na aprendizagem da programação.

Com base nos problemas e soluções anteriormente identificados foi criada uma matriz de relação, ver tabela 2, onde para cada problema se identifica a solução ou soluções que podem contribuir na sua resolução.

Problemas	Desenhar o Programa	Compreensão do Programa	Linguagem de Programação	Duração do Curso – Tempo	Diferentes Capacidades	Atitudes Face à Programação	Fatores Cognitivos	Autoconfiança	Motivação	Modelos Mentais Ineficientes
Raptor [Carlisle et al. 2005]	√					√	√			
Jeliot [Moreno, et al. 2007]	√	√				√	√		√	√
Verificator [Radosevic et al. 2009]	√	√			√		√			√
Jype [Helminem et al. 2010]		√						√	√	√
Ville [Rajala et al. 2007]	√		√			√	√		√	
Alice [Cooper et al. 2000]	√	√			√					√

[M. McGill, 2012]		√			√	√	√	√	√	√
[McWhorter 2008]	√	√				√		√	√	

Tabela 2 - Soluções Propostas

Na secção 2 foi realizada uma breve revisão da literatura relativa aos problemas e soluções na aprendizagem da programação. Variados problemas na aprendizagem foram identificados. Por exemplo, problemas de raciocínio na compreensão dos conceitos mais abstratos, dificuldade na definição e criação de modelos mentais adequados, dificuldades na utilização em conjunto dos diferentes conceitos para a resolução de um determinado problema, dificuldades na compreensão entre o que é ensinado e ligação com o "mundo real". Como soluções autores argumentam que as situações de aprendizagem devem ser as mais concretas possíveis, privilegiando a aprendizagem prática, como também os materiais/recursos serem o mais orientados possíveis para as competências de programação: desenho do programa, modificação e depuração. Nesse sentido diversos ambientes de suporte à aprendizagem têm sido desenvolvidos.

3. DIFICULDADES PERCECIONADAS NA APRENDIZAGEM DA PROGRAMAÇÃO – ESTUDO EMPÍRICO

3.1. Introdução

Na seção anterior abordámos algumas das problemáticas relacionadas com o ensino da programação. Descrevemos também algumas das propostas para solução dessas problemáticas.

Fundamentados na revisão da literatura efetuada conduzimos um estudo empírico que teve como objetivo identificar as dificuldades na aprendizagem da programação dos alunos de uma instituição de ensino superior politécnico. Após essa identificação apresentamos e discutimos os resultados.

3.2. Objetivos do Estudo

Este estudo pretende identificar as perceções dos alunos relativas às dificuldades na aprendizagem da programação, como também as perceções que os professores têm sobre as dificuldades dos seus alunos.

Como objetivo geral para o presente estudo foi definido o seguinte: Identificar as dificuldades percebidas pelos alunos iniciantes na aprendizagem de programação.

Com base no objetivo geral foram definidas as seguintes questões: Q1) Quais as dificuldades percebidas nas situações de aprendizagem?; Q2) Quais as dificuldades percebidas nos conceitos de programação?; e, Q3) Qual a utilidade percebida nos contextos de aprendizagem e materiais/recursos?.

3.3. Abordagem Metodológica

A abordagem metodológica adotada seguiu os seguintes procedimentos: (i) identificação das principais dificuldades referidas na revisão da literatura, desenvolvimento de instrumentos de recolha de dados (questionário), lançamento de questionários a amostra de alunos e professores e respetiva análise (ii) recolha, tratamento e análise aos resultados das provas de avaliação.

O questionário utilizado no presente estudo, foi criado e adaptado com base na revisão da literatura efetuada. Foram definidas questões fechadas e a sua classificação feita com base na escala de valores likert, entre um a sete. Foram obtidas um total de cento e quatro respostas de alunos e dez de professores.

As questões no questionário foram agrupadas em quatro grupos: situações de aprendizagem, conceitos de programação, contextos de aprendizagem e materiais/recursos. Para cada grupo foram identificadas variadas dimensões, que se encontram detalhadas e analisadas na secção 3 deste artigo.

O presente estudo pretende analisar as dificuldades na aprendizagem da programação dos alunos iniciantes. Assim sendo, o questionário foi apresentado a uma amostra de alunos do primeiro ano que já tinham tido pelo menos uma unidade curricular de introdução à programação. No total foram obtidas centro e quatro respostas, sendo 24 do género feminino e 80 do género masculino.

A média de idades da amostra é de 22 anos.

3.4. Apresentação de Dados e Discussão de Resultados

Da análise aos dados observamos que os alunos se mostram bastante mais confiantes na perceção das suas dificuldades comparativamente à perceção que os professores têm dos seus alunos. Esta diferença na perceção das dificuldades pode resultar da ineficiente compreensão que os alunos têm sobre o que sabem e o que deveriam na realidade saber, e consequentemente não têm a noção

exata das dificuldades. Em contrapartida, os professores têm uma noção mais próxima das dificuldades, que resulta por um lado, dos resultados que os alunos obtêm nas provas de avaliação, por outro pela sua experiência no contacto direto com os alunos em sala de aula. Esta análise é suportada através da análise ao gráfico 1. Observamos que as provas de avaliação em todas as dimensões apresentam valores mais próximos de muito difícil. Aproxima- se desses valores as perceções dos professores, ficando mais distantes as perceções dos alunos.

Estes resultados estão em consonância com as observações feitas por Lahtinen, Mutka e Jarvinen [Lahtinen et al. 2005] e Milne and Rowe [Milne et al. 2002], os alunos acreditam perceber sobre um dado tópico, mas quando analisadas as questões dos exames ou inqueridos pelos professores verificamos que estavam errados na sua confiança.

Esta perceção pode ser um problema para o sucesso do aluno na aprendizagem da programação, dado que cria autoconfiança sobre o conhecimento adquirido e quando confrontado com os exames e exercícios nos quais precisam de os aplicar falham. Uma quebra na autoconfiança poderá contribuir para o aluno abandonar a disciplina, desmotivar e dedicar menos tempo de estudo à programação de computadores.

Observamos nas situações de aprendizagem, ver gráfico 2, e nos conceitos de programação, ver gráfico 1 que as dimensões percecionadas como mais difíceis são aquelas que necessitam de uma compreensão mais abstrata. Estes resultados têm suporte nos trabalhos de Lahtinen, Mutka e Jarvinen [Lahtinen et al. 2005] e Milne e Rowe [Milne et al. 2002].

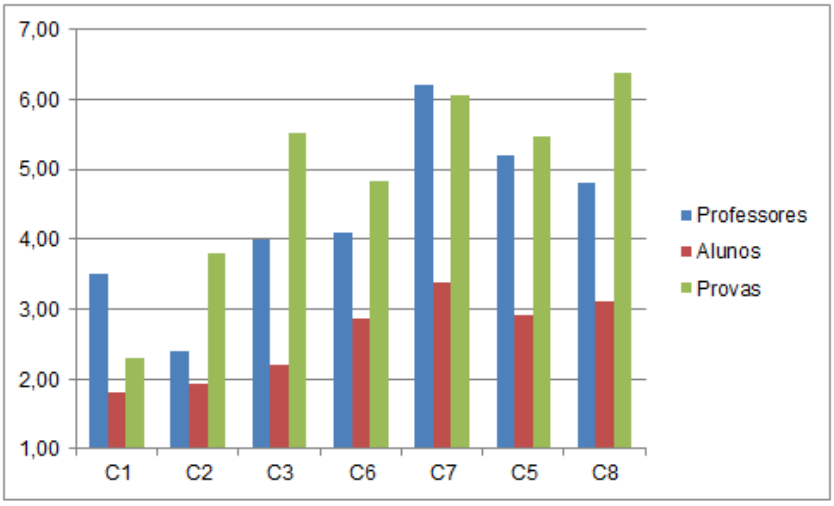

Gráfico 1 – Conceitos de Programação versus Provas de Avaliação

(0-não sabe), (1-fácil de aprender), (7-difícil de aprender)

(C1) – Variáveis; **(C2)** – Estruturas de seleção; **(C3)** – Ciclos de repetição; **(C5)** – Parâmetros; **(C6)** – Tabelas; **(C7)** – Ponteiros e referências; **(C8)** – Tipos de dados estruturados;

Conceitos como ponteiros e referências, tipos abstratos de dados, tipos estruturados de dados, desenhar um programa para a resolução de uma determinada tarefa ou dividir o programa em funcionalidades são situações de aprendizagem e conceitos de programação que requerem um raciocínio abstrato, como também uma compreensão e manipulação de várias "entidades" para os quais os alunos ainda não desenvolveram capacidades e treino específico. Milne e Rowe [Milne et al. 2002] referem que conceitos como os ponteiros são usualmente fáceis de explicar, no entanto é a sua implementação e consequente comportamento durante a execução do programa que causa os maiores problemas aos alunos na sua compreensão.

Uma outra questão que surge ao analisarmos os dados é a perceção de dificuldade quando os alunos devem utilizar os diversos conceitos em conjunto. Os alunos parecem que compreendem os conceitos individualmente, mas quando os devem utilizar em conjunto manifestam dificuldade na sua utilização e na compreensão no modo de os utilizar.

Em resposta à questão de investigação Q1 baseados na análise dos dados podemos afirmar que as maiores dificuldades percecionadas pelos alunos nas situações de aprendizagem foram compreensão na estruturação do programa (I2), aprendizagem da sintaxe da linguagem (I3) e desenhar o programa para a resolução de uma determinada tarefa (I4).

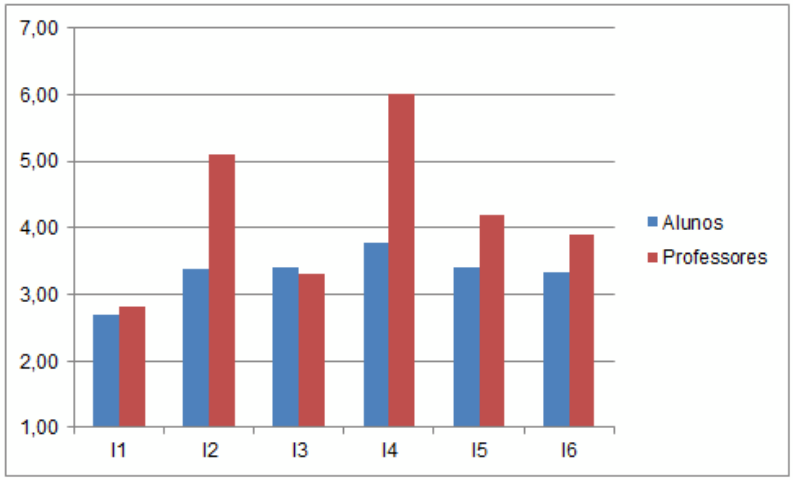

Gráfico 2 – Situações de Aprendizagem

(0-não sabe), (1-fácil de aprender), (7-difícil de aprender)

(I1) – utilização do ambiente de desenvolvimento; **(I2)** – compreensão na estruturação do programa; **(I3)** – aprendizagem da sintaxe da linguagem; **(I4)** – desenhar o programa para resolução de uma determinada tarefa; **(I5)** – dividir as funcionalidades em procedimentos; **(I6)** – interpretar e resolver erros no programa.

Na resposta à questão de investigação Q2 baseados na análise dos dados podemos afirmar que as maiores dificuldades percecionadas pelos alunos nos conceitos de programação foram ponteiros e referências (C7), tipos estruturados de dados (C8) e tabelas (C6).

Na análise aos resultados obtidos nos contextos de aprendizagem ver gráfico 3, verificamos que os alunos preferem sessões práticas em alternativa às sessões puramente teóricas. Inclusive os alunos mostram-se mais confiantes na aprendizagem que fazem quando estudam sozinhos, ao invés da aprendizagem realizada em aulas teóricas. Podemos afirmar baseados nos resultados que as aulas meramente expositivas não contribuem para uma efetiva aprendizagem da programação. Quando relacionamos os resultados obtidos nos contextos com os materiais observamos precisamente que os alunos preferem ter programas completos (código + executável) como material de estudo. Pode-se afirmar baseados nestas observações que os alunos preferem aprender através do fazer. Estas observações têm suporte nos trabalhos de Robins e colegas [Robins et al. 2003] e tanto alunos como professores estão em concordância neste aspeto.

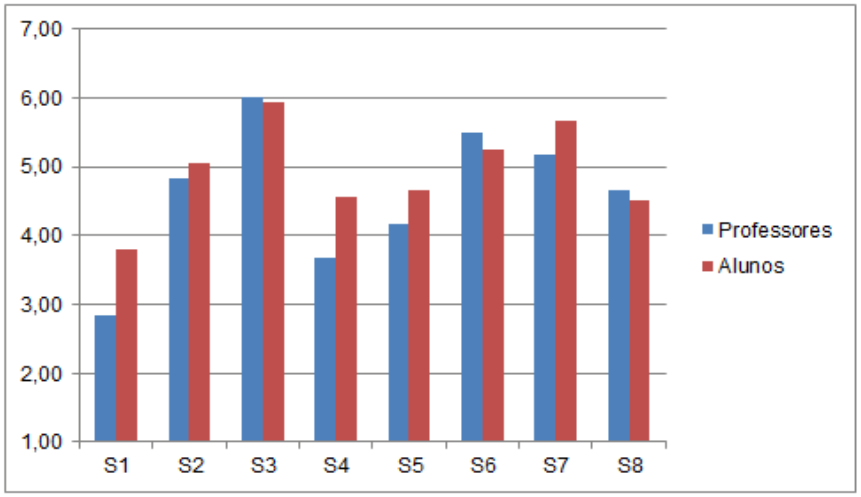

Gráfico 3 - Contextos de Aprendizagem

(0-não sabe), (1-pouco útil), (7-muito útil)

(S1) – Nas aulas teóricas; **(S2)** – Em sessões de resolução de exercícios com os colegas; **(S3)** – Em sessões práticas de resolução de exercícios (aulas de laboratório); **(S4)** – Quando estudo sozinho; **(S5)** – Quando estudo / trabalho sozinho na resolução de exercícios da disciplina; **(S6)** – Em sessões de esclarecimento de dúvidas com o docente (horário de dúvidas); **(S7)** – Em sessões de resolução de exercícios na aula teórica; **(S8)** – Em fóruns de discussão na Internet;

Tendo em consideração que um dos aspetos que contribui para as dificuldades na aprendizagem é a crescente complexidade que alguns dos conceitos de programação apresentam, como também a utilização conjunta de diferentes conceitos. Nesse sentido os materiais/recursos disponibilizados aos alunos podem contribuir para ultrapassar essas dificuldades se tiverem em conta a visualização e a interatividade no sentido de permitir ao aluno a criação de um modelo mental adequado e correto na sua compreensão e ajudar na resolução de problemas.

Ao analisarmos a perceção da utilidade dos materiais/recursos, observamos que os alunos preferem como materiais/recursos programas executáveis, tutoriais e vídeos educativos na Internet, ver gráfico 4. Esta perceção confirma que suporte audiovisual e interatividade nos materiais influenciam positivamente a aprendizagem.

Por último na análise efetuada aos dados nos contextos de aprendizagem e materiais/recursos podemos responder questão de investigação Q3 identificando que os contextos percecionados com maior utilidade são sessões práticas de resolução de exercícios-laboratório (S3), sessões de resolução de exercícios-aula teórica (S7), sessões de esclarecimento de dúvidas com o docente (S6) e sessões de resolução de exercícios com os colegas (S2). Nos materiais/recursos os percecionados como mais

úteis foram os exemplos de programas completos-código + executáveis (M3), tutorias disponíveis na Internet (M6), vídeos educacionais disponibilizados no youtube (M7) e conteúdos disponibilizados na plataforma Moodle (M11).

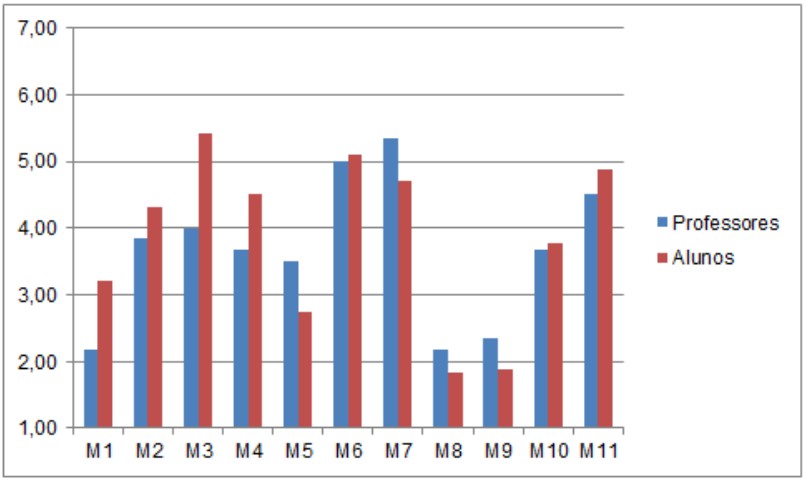

Gráfico 4 - Materiais / Recursos

(0-não sabe), (1-pouco útil), (7-muito útil)

(M1) – Livro recomendado para a disciplina; **(M2)** – Apontamentos tirados nas aulas; **(M3)** – Exemplos de programas completos (código + executável); **(M4)** – Imagens estáticas de programas estruturados; **(M5)** – Visualizações interactivas (com ferramentas tipo Alice, BlueJ, Robocode ou outras similares); **(M6)** – Tutoriais disponíveis na Internet; **(M7)** – Vídeos educacionais disponibilizados no youtube; **(M8)** – Twitter – Páginas relacionadas com a aprendizagem da programação; **(M9)** – Facebook – Páginas relacionadas com a aprendizagem da programação; **(M10)** – Cópias das transparências / Powerpoint; **(M11)** – Conteúdos disponibilizados na plataforma;

Na secção 3 foram analisados e discutidos os resultados do estudo empírico relativo à perceção das dificuldades na aprendizagem da programação e respondidas as questões de investigação previamente identificadas. Seguidamente apresentamos a proposta de solução tangível de suporte à aprendizagem, baseada na revisão da literatura e no estudo empírico realizado.

4. PROPOSTA DE SOLUÇÃO DE SUPORTE À APRENDIZAGEM DA PROGRAMAÇÃO

4.1. Introdução

Com base na revisão da literatura descrita nas secções 1 e 2, e estudo empírico descrito na secção 3 foram identificados os principais problemas que ocorrem na aprendizagem da programação. Dificuldades na compreensão de conceitos de programação que requerem um nível de abstração maior, desmotivação e ineficiente contextualização, contribuem para as dificuldades sentidas pelos alunos na correta compreensão da estruturação e da finalidade do programa. Estes problemas têm consequentes efeitos no desempenho e consequente a desistência dos alunos do curso ou a sua retenção no mesmo ano letivo.

4.2. Objetivos

Com base no problema anteriormente descrito e fundamentados na revisão da literatura, o presente trabalho tem como objetivo a proposta de uma solução tangível de suporte à aprendizagem, através da qual pretendemos mitigar as dificuldades ao nível do raciocínio abstrato, da motivação e da eficácia na programação.

4.3. Descrição da Proposta de Solução de Suporte à Aprendizagem

Para a solução proposta foram identificadas três estratégias complementares: personificação [Halkyard 2012], personalização [Halkyard 2012] e estratégias motivacionais. Foram igualmente identificados os mecanismos para implementação das estratégias: tangibilidade [Marshall, 2007], valor da tarefa [Wigfield e Eccles 2000], [Pintrich 1999], diversão [Malone et al 1987], competição [Burguillo 2010] e medida da performance, ver tabela 3.

Através da personificação é possível utilizar aspetos sociais para promover a aprendizagem. A personificação acontece quando um objeto inanimado é descrito usando uma linguagem associada com as pessoas, como ele e ela, para indicar que ele/ela é um agente social e que pode gerar uma resposta social e um aumento na aprendizagem. Na solução proposta a personificação será implementada através do mecanismo tangibilidade. Através da tangibilidade é possível realizar atividades “hands-on” e manipulação física de um objeto. A tangibilidade na solução proposta foi implementada através do recurso a robots físicos e seus componentes. A personalização é uma estratégia centrada no aluno. O aluno deverá durante a sua aprendizagem criar a sua própria visão do problema e estabelecer mecanismos e estratégias para alcançar os objetivos definidos. O mecanismo de suporte a esta estratégia será a medida de performance obtida pelo aluno na sua autoavaliação e autoconhecimento. Na solução proposta a estratégia de personalização será

conseguida através dos desafios/objetivos de aprendizagem presentes nos exercícios de programação criados para serem utilizados neste contexto. Esses desafios serão realizados com recurso a uma biblioteca educacional, desenvolvida para facilitar o processo de aprendizagem e a interação com o artefacto tangível (robot).

Problemas	**Estratégias**	**Mecanismos**
Raciocínio Abstrato	Personificação [Halkyard 2012]	Tangibilidade [Marshall, 2007]
Motivação	Estratégias Motivacionais	Valor da Tarefa [Wigfield e Eccles 2000], [Pintrich 1999], Diversão [Malone et al 1987], Competição [Burguillo 2010]
Eficácia na Programação	Personalização da Aprendizagem [Halkyard 2012]	Medida de Performance

Tabela 3 – Problemas, Estratégias e Mecanismos de Aprendizagem

Os exercícios presentes nas bibliotecas foram desenhados de forma incremental permitindo guiar o aluno na utilização dos principais mecanismos de programação: sequência, condicional, repetição e gestão da persistência ou estado, como também permite desafiar o aluno a introduzir novos conceitos ou mecanismos à medida que são requeridos pelos problemas (problemas tangíveis reais) que necessitam de resolver, ao longo de todo o processo de aprendizagem.

Por último as estratégias motivacionais implementadas através do mecanismo de valorização social, diversão e competitividade lúdica resultam da utilização, no seu todo, da solução tangível proposta: robot, biblioteca educacional, exercícios de programação e plataforma física.

4.4. Implementação da Solução Ambiente Tangível de Suporte à Aprendizagem

A solução implementada é composta por hardware e software. Os componentes de hardware são: o robot e a base onde o robot executa as ações, ver figura 1 e software: biblioteca de abstração e bibliotecas educacionais.

Figura 1 – Ambiente de testes e artefacto tangível

O robot representa o artefacto tangível. A plataforma física onde o robot realiza as ações foi construída de modo a permitir configurar diferentes desafios, com maior ou menor grau de complexidade, ver figura 1.

Por últimos os componentes de software que integram o ambiente: biblioteca de abstração e bibliotecas educacionais. Estas bibliotecas foram criadas para a plataforma robótica open-source Arduíno[1]. Ambas as bibliotecas foram desenvolvidas com recurso à linguagem de programação JAVA.

Através da biblioteca de abstração pretende-se encapsular os detalhes técnicos relacionados com a robótica permitindo aos alunos a utilização de instruções de "alto nível" na manipulação do ambiente tangível. Pretende-se assim, focar atenção nas tarefas de programação, reduzindo as dificuldades técnicas ou tecnológicas e as barreiras associadas à utilização do artefacto, contribuindo assim para simplificar o processo de aprendizagem. As bibliotecas educacionais estão acessíveis ao aluno, através de um ambiente de desenvolvimento Java e suportam a interação com objetos reais à volta do robot permitindo uma perceção mais clara do código que está a ser executado.

[1] Arduino é uma plataforma eletrónica open-source

4.5. Descrição do Curso de Aprendizagem da Introdução à Programação

Para este estudo foram planificados e implementados dois cursos: Curso I – utiliza o ambiente tangível, Curso II – não utiliza o ambiente tangível, com um total de 35 horas divididas em 5 dias por semana e com uma carga diária de 6 horas.

O curso I com suporte do ATPSAP foi estruturado em sessões e os objetivos para cada sessão correspondem à exposição e aplicação dos conceitos definidos para cada uma dessas sessões. No curso II sem suporte do ATSAP foram apenas feitas ligeiras alterações nos exercícios, contudo os objetivos educacionais mantiveram-se.

5. CONSIDERAÇÕES FINAIS E TRABALHO FUTURO

O propósito do artigo é apresentar e discutir os problemas e soluções na aprendizagem da programação. Nesse sentido foi efetuada uma revisão da literatura e conduzido um estudo empírico com o objetivo de identificar as perceções das dificuldades da aprendizagem da programação nos conceitos de programação, situações de aprendizagem, materiais/recursos e contextos de aprendizagem. Com base na literatura e estudo empírico concluímos que os problemas de aprendizagem na programação se manifestam principalmente em três aspetos: raciocínio abstrato, motivação, eficácia na programação e ineficiente contextualização. Também ao nível dos contextos de aprendizagem e materiais/recursos observámos que os alunos consideram de grande utilidade para a sua aprendizagem os contextos que permitam uma aprendizagem prática e colaborativa em contrapartida de aulas meramente expositivas. Por último os materiais/recursos de visualização multimédia/interativa e materiais que permitam interação, visualização da execução e resultado final são também de percecionados de grande utilidade.

Com base na revisão da literatura e estudo empírico propomos uma solução de aprendizagem baseada em três estratégias: Personificação, Personalização e Motivação. Para suporte à implementação das estratégias é proposto um ambiente tangível de aprendizagem.

Como trabalho futuro será efetuado um trabalho experimental de validação da solução proposta.

REFERÊNCIAS

Arlegui, J., Menegatti, E., Moro, M., Pina, A. Robotics, Computer Science Curricula and Interdisciplinary Activities. Workshop Proceedings of SIMPAR, pp. 10-21. International Conference on Simulation, Modeling and Programming for Autonomous Robots. Venice, November, 3-4, 2008.

Blank, D. et al. Pyro: A python-based versatile programming environment for teaching robotics. J.Educ. Resour. Comput. 3(4). Article 1, 2005.

Braught, G. "dLife: A Java Library for Multiplatform Robotics, AI and Vision in Undergraduate CS and Research". SIG9CSE'12, February 29-March 3, 2012, North Carolina, USA.

Brauner, P., Leonhardt, T., Ziefle, M., Schroeder, U. The Effect of Tangible Artifacts, Gender and Subjective Technical Competence on Teaching Programming to Seventh Graders. ISSEP, LNCS 5941, (2010) pp. 61-71. Springer-Verlag Berlin

Burguillo, J. C. "Using Game Theory and Competition-based Learning to stimulate student motivation and performance". Computers & Education, vol. 55, (2010) no. 2, pp. 566–575.

Carlisle, M., Wilson, T., Humphries, J. and Hadfield, S. RAPTOR: a visual programming environment for teaching algorithmic problem solving. In Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education, St. Louis, Missouri, ACM Press, 176-180, 2005.

Cooper, S., Dann, W., Paush, R. Alice: a 3-D tool for introductory programming concepts, Journal of Computing in Small Colleges, (2000), 15(5), 108-117.

Costa, C, & Aparício, M. Evaluating Success of a Programming Learning Tool. Proceedings of the International Conference on Information Systems and Design of Communication. ISDOC,2014.

Denning, P. J. & McGettrick. A Recentering Computer Science. Communnications. ACM, 48(11), pp.15-19, 2005.

Eckerdal, A. e Thuné, M. Novice Java Programmers' conceptions of "object" and "class", and variation theory. SIGCSE Bulletin, 37(3), 2005.

Eccles, J. S.; Wingfield, A. Motivational beliefs, values and goals. Annual Review of Psychology, (2002) v. 53, p. 109-132.

Gomes, A. & Mendes, A.J. Learning to Program – difficulties and solutions,. International Conference on Engineering Education, 2007 – ICEE, Coimbra, Portugal.

Halkyard, J.S. The Separate and Collective effects of Personalization, Personification, and Gender on Learning With Multimedia Chemistry Instructional Materials. 2012. (Doctoral Dissertation). University of San Francisco.

Helminem, J., Malmi,L. Jype – A Program Visualization and Programming Exercise Tool for Phyton. Softvis'(2010), October 25-26, Salt Lake City, Utah, USA.

Kinnunem, P., Malmi, L. Why Students Drop Out CS1 Course? ICER' 2006, September, Canterbury, United Kingdom.

Kynigos, C. Black-and-White-Box Perspectives do Distributed Control and Constructionism in Learning with Robots. Workshop Proceedings of SIMPAR, pp. 1-9. International Conference on Simulation, Modeling and Programming for Autonomous Robots. Venice, November, 3-4, 2008.

Lahtinen, E., et al. Study of the Difficulties of Novice Programmers. ITiCSE – June 27-29 – 2005, Monte da Caparica, Portugal.

Lister, R., Adams, E., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M., McCartney, R., Moström, J., Sanders, K., Seppälä, O., Simon, B., and Thomas, L. A multi-national study of reading and tracking skills in novice programmers. 2004, ACM SIGCSE Bulletin, 36(4).

Marshall, P. Do tangible interfaces enhance learning? In Proceedings of the 1st international Conference on Tangible and Embedded interaction, ACM Press (2007), 163-170.

Ma, L. Investigating and Improving Novice Programmers' Mental Models of Programming Concepts. 2007. (Doctoral Dissertation). University of Strathclyde, Department of Computer & Information Sciences.

McWhorter, W. The Effectiveness of Using Lego Mindstorms Robotics Activities to Influence Self-Regulated Learning in a University Introductory Computer Programming Course. 2008. (Doctoral Dissertation). University of North Texas.

Malone, T. W., & Lepper, M. R. Making learning fun: A taxonomy of intrinsic motivation for learning. In R. E. Snow & M. G. Farr (Eds.), Aptitudes, learning, and instruction, Volume 3, Cognitive and affective process analyses, 1987 (pp. 223-254). Hillsdale, NJ: Erlbaum.

Milne, I., Rowe, G. Difficulties in Learning and Teaching Programming – Views of Students and Tutors. Journal of Education and Information Technologies 7:1, 55-66, 2002.

McGill, M. Learning to Program with Personal Robots: In uences on Student Motivation. ACM Transactions on Computing Education, Vol. 12, No. 1, Article 4, 2012. Publication date: March.

Moreno, A., Joy, M. Jeliot 3 in Demanding Educational Setting, 2007. Available from: http://www.sciencedirect.com [doi:10.1016/j.entcs.2007.01.033]

Mow, I.T. Issues and Difficulties in Teaching Novice Computer Programming. Innovative Techniques in Instruction. Springer Netherlands, 2008. Pages 199-204.

Pat, B. & Lyons, G. The Effect of Student Attributes on Success in Programming. Proceedings of the 6th Annual Conference on Innovation and Technology in Computer Science Education. Pages 49-52, 2001.

Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J., Devlin, M., and Paterson, J. A survey of literature on the teaching of introductory programming. In ITiCSE- WGR'07: Working group reports on ITiCSE on Innovation and technology in computer science education, (2007) pages 204-223, New York, NY, USA. ACM.

Pintrich, P. R. The role of motivation in promoting and sustaining self-regulated learning. International Journal of Educational Research, v. 31, p. 459-470, 1999.

Radosevic, D., Orehovacki, T., Lovrencic, A. Verificator:Educational Tool for Learning Programming. Informatics in Education, 2009. Vol. 8, No. 2, 261-280.

Rahmat et al. Major Problems in Basic Programming that Influence Student Performance. Journal Procedia – Social and Behavioral Sciences, 2012, vol.59 pag. 287-296

Rajala, T., Laakso, M.-J., Kaila, E. & Salakoski, T. VILLE - A Language-Independent Program Visualization Tool. In proceedings of the Seventh Baltic Sea Conference on Computing Education Research, Koli National Park, Finland, November 15-18, 2007. Conferences in Research and Practice in Information Technology, Vol. 88, Australian Computer Society. Raymond Lister and Simon, Eds.

Robins, A., Rountree, J., Rountree, N. Learning and Teaching Programming: A Review and Discussion. Computer Science Education. Vol.13, pp. 137-172, 2003.

Schulte, C., Busjahn, T. "An Introduction to Program Comprehension for Computer Science Educators". ITiCSE, June 26-30, 2010. Bilkent, Ankara, Turkey.

Sorva, J., Sirkiä, T. Uuhistle – A Software Tool for Visual Program Simulation. Koli Calling, ctober 38-21, 2010. Koli, Finland.

Winslow, L.E. Programming Pedagogy – A psychological overview. SIGCSE Bulletin, 28, 17- 22, 1996.

Xinogalos, S., Satratzemi, M., & Vassilios Dagdilelis, V. An Introduction to Object-Oriented Programming with a Didactic MicroWorld: Object Karel. Computers & Education, 47(2), 148- 171, 2006.