risti
Revista Ibérica de Sistemas e Tecnologias de Informação
Revista Ibérica de Sistemas y Tecnologías de Información

Recebido/Submission: 10/06/2017
Aceitação/Acceptance: 01/09/2017

Desenvolvimento e validação de uma prova de avaliação das competências iniciais de programação

Joana Martinho Costa[1], Guilhermina Lobato Miranda[1]

joana.martinho.costa@campus.ul.pt, gmiranda@ie.ulisboa.pt

[1] Instituto de Educação da Universidade de Lisboa, Alameda da Universidade, 1649-013 Lisboa, Portugal

DOI: 10.17013/risti.25.66-81

Resumo: Atualmente verifica-se a insuficiência de instrumentos de avaliação validados que possam mensurar as diferentes etapas de aprendizagem inicial de programação. Por este motivo desenvolvemos uma prova a partir da Taxonomia de Bloom que permite comparar o desempenho dos alunos independentemente da linguagem de programação com que iniciaram a sua aprendizagem. A prova apresenta questões de escolha múltipla para os quatro níveis iniciais e questões de desenvolvimento para os dois últimos níveis. Após a sua construção, foi validada por 12 especialistas e submetida a um grupo-piloto (N=29) para analisar a fiabilidade e dificuldade dos itens. Obtivemos um alfa de 0,837 e correlação positiva entre todos os itens da prova e o total dos itens com nível de significância $p<0,05$. As diferenças entre a frequência de respostas corretas nos itens são estatisticamente significativas (Q(12)=76,767; n=29; p=0,000). Estes resultados demonstram que a prova é válida, fiável e discriminativa.

Palavras-chave: Programação Inicial; Taxonomia de Bloom; Desempenho; Validação; Fiabilidade

Development and Validation of an Assessment Instrument of the Initial Programming Skills

Abstract: Nowadays there is a noticeable lack of validated evaluation instruments that can measure the different stages of initial programming learning. For this reason, we developed an instrument from Bloom's Taxonomy that allows to compare the students' performance regardless of the programming language with which they started their learning. The instrument presents multiple choice questions for the first four levels and open-ended questions for the last two levels. After its construction, it was validated by 12 specialists and submitted to a pilot-group (N=29) to analyze the items' reliability and difficulty. We obtained an alpha of 0,837 and positive correlation between al the test's items and the total of items with a significance level of $p<0,05$. The differences between the frequency of correct answers in the items are statistically significative (Q(12)=76,767; n=29; p=0,000). These results demonstrate that the test is valid, reliable and discriminative.

Keywords*:* Initial Programming; Bloom's Taxonomy; Performance; Validation; Reliability

1. Introdução

A comunidade científica tem desenvolvido diversas propostas que visam ultrapassar as dificuldades inerentes à aprendizagem inicial da programação (e.g. Moons & Backer, 2013; Liu, Cheng & Huang, 2011; Johnsgard & McDonald, 2008; entre outros). No entanto, apesar de existirem estudos que apresentam resultados promissores nesta área (cf. Johnsgard & McDonald, 2008; Sykes, 2007; Wang, Mei, Lin, Chiu, & Lin, 2009; Zhang, Liu, &, Pablos, 2014) os instrumentos de recolha de dados utilizados variam significativamente, dificultando a possibilidade de retirar conclusões. Por exemplo, no estudo meta-analítico realizado por Costa e Miranda (2017), foi difícil comparar os resultados das investigações que visaram melhorar as competências de programação dos estudantes que se encontravam numa fase inicial de aprendizagem, não só porque foram usadas diferentes linguagens de programação, mas ainda porque a avaliação feita usou diferentes instrumentos de medida. É o caso dos estudos de Wang, Mei, Lin, Chiu and Lin (2009) e de Sykes (2007), que utilizaram questionários, sendo o primeiro composto por 20 questões de escolha múltipla sobre conceitos de programação e o segundo por uma combinação de questões relacionadas com problemas de programação, umas fazendo apelo a conhecimentos, outras a competências e outras a resolução de problemas. Outras investigações usaram os resultados de provas académicas como medida de avaliação da aprendizagem inicial da programação (cf. Cooper, Dann, & Paush, 2003; Moskal, Lurie, & Cooper, 2004; Johnsgard, & McDonald, 2008; entre outros). Costa e Miranda (2017) referem que estas diferenças são um constrangimento à análise de resultados e que a existência de instrumentos validados para avaliar a aprendizagem da programação seria uma solução para melhorar a comparação entre diferentes estudos.

Füller et al. (2007) e Lister (2003) desenvolveram procedimentos genéricos a partir da Taxonomia de Bloom para o desenvolvimento de instrumentos de avaliação e de cursos que tenham como objetivo a aprendizagem da programação. No entanto, mantém-se a limitação da inexistência de um instrumento genérico validado que possa ser aplicado aos alunos na fase inicial desta aprendizagem .

Para colmatar a ausência deste tipo de instrumentos, propomos a criação de uma prova de programação em *pseudocódigo*. Esta prova deverá abranger os conceitos iniciais de programação e permitir compreender em que medida os estudantes aprendem a programar independentemente da linguagem de programação com que iniciaram a sua aprendizagem.

2. Metodologia de Desenvolvimento da Prova

Construímos a prova a partir das orientações de Tuckman (2012) para o desenvolvimento de testes de desempenho, nomeadamente: a) Desenvolvimento do plano geral do conteúdo; b) Desenvolvimento dos itens da prova; c) Demonstração da validade através da validação do conteúdo e da fiabilidade, os dois indicadores que demonstram a validade de provas de medição ou de avaliação de conhecimentos (Tuckman, 2012); e d) Análise dos itens da prova previamente validados.

2.1. Plano Geral do Conteúdo

Esta prova incide sobre os conceitos estruturantes de programação incluídos no currículo de disciplinas da área da Informática do ensino secundário em cursos científico-

humanísticos e profissionalizantes, nomeadamente o desenvolvimento de algoritmos, variáveis, tipos de dados, operadores e estruturas de controlo (e.g. Pinto, Dias & João, 2009; Direção-Geral de Formação Vocacional, 2005). Para compreender até que ponto o aluno aprende estes conceitos considerámos pertinente a aplicação de um esquema de classificação que permitisse desenvolver esta análise.

Analisámos a Taxonomia de Bloom, uma taxonomia de objetivos educacionais que tem como principal característica a categorização de questões de acordo com o nível de complexidade de domínio cognitivo necessário para a sua resolução (Bloom, Engelhart, Furst, Hill & Krathwohl, 1976). Posteriormente foi apresentada uma proposta com a Taxonomia de Bloom revista (Krathwohl, 2002). Nesta versão as alterações consistem essencialmente na organização do conhecimento em matriz ao invés de uma apresentação unidimensional para permitir a distinção entre o tipo de conhecimento e o processo cognitivo e na substituição dos nomes de cada categoria por verbos que se aproximam dos nomes dos objetivos de aprendizagem. Também Füller et al. (2007) desenvolveram uma taxonomia em matriz que separa as capacidades de produzir e de interpretar código.

A construção da prova poderia ser suportada a partir de qualquer uma destas taxonomias mas considerando que este estudo incide sobre uma fase tão inicial da aprendizagem, a quantidade restrita de conteúdos programáticos não será suficiente para beneficiar da complexidade destas taxonomias. No entanto, consideramos que serão uma solução eficaz para a análise das competências de programação numa fase mais avançada. Como a nossa proposta abrange apenas os conceitos iniciais, entendemos que a utilização da Taxonomia de Bloom original seria a mais adequada.

2.1.1. Taxonomia de Bloom original

A taxonomia de Bloom completa é representada em três domínios: cognitivo, afetivo e psicomotor (Bloom et al., 1976). O domínio cognitivo está relacionado com a memória e o desenvolvimento de capacidades intelectuais; o domínio afetivo centra-se em objetivos relacionados com interesses, atitudes e valores; e o domínio psicomotor centra-se em habilidades motoras e de manipulação . Considerando os objetivos desta prova antes descritos entendemos que a sua construção deveria ser sustentada no domínio cognitivo. Bloom et al. (1976) consideram que o domínio psicomotor tem pouca aplicação no ensino secundário e universitário. O domínio afetivo, apesar de ser objeto de interesse no campo da aprendizagem da programação, não faz parte dos objetivos do desenvolvimento desta prova, pelo que também foi excluído.

O domínio cognitivo da taxonomia é dividido em seis níveis: Conhecimento, Compreensão, Aplicação, Análise, Síntese e Avaliação. A Tabela 1 sintetiza os objetivos de todos os níveis da Taxonomia de Bloom (Bloom et al., 1976; Pinto, 2001). O primeiro nível, o Conhecimento, é apresentado como o mais simples e concreto enquanto que o último, o nível referente à Avaliação, é considerado o mais complexo e abstrato (Krathwohl, 2002; Lister, 2003). Bloom et al. (1976) referem que é esperado mais sucesso nos primeiros níveis da taxonomia e um decréscimo ao longo da hierarquia. Os autores apresentaram ainda um estudo em que verificaram que é mais frequente os indivíduos com baixos resultados escolares atingirem também baixos resultados nos níveis mais elevados da taxonomia e altos resultados nos níveis inferiores do que a situação inversa.

Nível da Taxonomia de Bloom	Objetivos
Conhecimento	Relembrar fórmulas, processos ou informação específica da atividade que aprendeu
Compreensão	Explicar determinada informação sem associar relações com outros conteúdos
Aplicação	Aplicar fórmulas, processos ou informação simples numa nova situação
Análise	Categorizar e distinguir informação
Síntese	Projetar uma solução a partir da decomposição de um problema
Avaliação	Analisar uma solução, otimizá-la e identificar e corrigir eventuais erros

Tabela 1 – Objetivos da Taxonomia de Bloom

Bloom et al. (1976) referem ainda que é importante ter em conta as experiências anteriores dos participantes, porque um problema simples poderá exigir que o aluno apresente um raciocínio complexo ou, caso já tenha experienciado problemas semelhantes em situações de aprendizagem anteriores, poderá apenas requerer que o aluno recorde ou aplique uma solução que já conhece.

2.2. Itens da Prova

Na fase de estruturação dos itens da prova definimos o formato de questões para cada nível da Taxonomia de Bloom. De acordo com Gronlund (1988, citado por Pinto, 2001), os quatro níveis iniciais poderão ser avaliados com Perguntas de Escolha Múltipla (PEM) e os dois últimos com Perguntas de Desenvolvimento (PD), já que requerem que o aluno demonstre um conhecimento mais profundo e não limitam a sua resposta. Pinto (2001) considera ainda que os dois primeiros níveis apresentam maior facilidade na aplicação de PEM.

Em seguida, ponderámos o número de questões que deveria constar na prova. Considerando que estamos a construir uma prova para ser aplicada em 90 minutos, a nossa proposta é constituída por oito a 12 PEM e duas PD. Optámos por este intervalo de PEM para que cada nível da Taxonomia tivesse duas a três questões e cada questão demorasse aproximadamente dois minutos a ser respondida já que, sendo uma prova para avaliar as competências iniciais de programação, entendemos que o aluno precise de tempo suficiente para a leitura e compreensão dos enunciados. Nas PD o tempo estimado foi consideravelmente maior para que o aluno tenha tempo suficiente para planear e desenvolver a sua resposta (Tuckman, 2012). Estimámos que o aluno utilize cerca de 20 minutos para responder a cada questão. A nossa proposta está de acordo com os valores apresentados por Pinto (2001), onde refere que deverão ser utilizadas 40 a 50 PEM para uma prova de 60 a 90 minutos e cinco a sete PD para uma prova de 120 minutos.

Após definirmos o formato e o número de questões que deveriam constar na prova, construímos um número superior ao proposto para que após o processo de validação e de fiabilidade pudéssemos optar pelos itens mais rigorosos. No total foram construídos 26 itens para serem submetidos ao processo de validação (Tabela 2).

Nível da Taxonomia de Bloom	Tipo de itens	Número de itens
Conhecimento	PEM	6
Compreensão	PEM	8
Aplicação	PEM	5
Análise	PEM	3
Síntese	PD	2
Avaliação	PD	2

Tabela 2 – Estrutura da prova submetida ao processo de validação

2.3. Análise Estatística

Para determinar a validade de conteúdo, calculámos o CVR (Content Validity Ratio) para cada item, utilizando a fórmula proposta por Lawshe (1975). O cálculo foi efetuado a partir da análise dos itens por um painel de especialistas constituído por uma amostra não-probabilista e de conveniência (Almeida e Freire, 2016) definida com base nos seguintes critérios: os participantes teriam de ser professores de informática ou especialistas em informática, com prática de ensino de programação e com prática de programação, disponíveis para responder e participar na validação da prova. A solicitação à participação foi efetivada recorrendo ao contacto direto por correio eletrónico e a grupos de professores de informática em redes sociais. Após a recolha de participantes, o painel ficou composto por 12 especialistas: sete professores da área do ensino da Informática e cinco programadores de *software*.

Para cada item da prova, cada especialista teria de indicar se considerava o item adequado ou não adequado para o nível de ensino e conteúdo curricular apresentado. De acordo com Lawshe (1975) para um item ser considerado válido num painel constituído por 12 especialistas, o CVR associado deverá atingir pelo menos o valor de 0,56.

Em seguida analisámos a fiabilidade dos itens validados pelo painel de especialistas. Para efetuarmos esta análise, aplicámos a prova a um grupo-piloto composto por 29 alunos de cursos profissionais que iniciaram a aprendizagem da programação no ano letivo de 2016/2017.

Posteriormente, codificámos as respostas dos itens de escolha múltipla com o valor *0* caso a resposta estivesse incorreta e com o valor *1* caso estivesse correta. No caso do itens de desenvolvimento utilizámos o mesmo procedimento. Contudo, a resposta só era considerada correta conferindo uma lista de verificação dos passos do algoritmo desenvolvida para cada um dos itens.

As respostas codificadas dos alunos foram correlacionadas com a pontuação total e selecionámos os itens que obtiveram uma correlação ao nível de significância de 5%. Para o cálculo da correlação utilizámos o coeficiente de correlação de Spearman por se tratar da análise de itens dicotómicos com a pontuação total (Marôco, 2014; Eisinga, Grotenhuis & Pelzer, 2013).

Após selecionarmos os itens válidos e fiáveis, procedemos à análise da dificuldade de cada item, verificando as diferenças significativas entre a frequência de respostas corretas.

Efetuámos esta análise a partir do teste Q de Cochran com comparações múltiplas (Marôco, 2014), utilizando o software SPSS Statistics (v. 24). Posteriormente, comparámos estes resultados com o nível hierárquico da Taxonomia de Bloom correspondente a cada item.

3. Resultados

3.1. Validação do Conteúdo

O resultado do CVR de cada item é apresentado na Tabela 3.

Nível da Taxonomia de Bloom	Item	CVR	Item Válido
Conhecimento	1	0,67	Sim
	2	0,67	Sim
	3	1	Sim
	4	1	Sim
	5	1	Sim
	6	1	Sim
Compreensão	7	0,5	Não
	8	1	Sim
	9	0,67	Sim
	10	0,33	Não
	11	0,83	Sim
	12	0,33	Não
	13	0,67	Sim
	14	0,5	Não
Aplicação	15	0,67	Sim
	16	0,67	Sim
	17	1	Sim
	18	1	Sim
	19	0,33	Não
Análise	20	1	Sim
	21	1	Sim
	22	0,83	Sim
Síntese	23	1	Sim
	24	1	Sim
Avaliação	25	1	Sim
	26	1	Sim

Tabela 3 – Validação do conteúdo por painel de especialistas

No total foram validados 21 itens, distribuídos por todos os níveis da Taxonomia de Bloom: seis do nível Conhecimento; quatro dos níveis Compreensão e Aplicação; três do nível Análise; e dois dos níveis Síntese e Avaliação. Seguidamente os itens validados pelos especialistas foram submetidos ao processo de fiabilidade descrito na secção seguinte.

3.2. Fiabilidade

Na Tabela 4 apresentamos o resultado do cálculo do Coeficiente de Correlação de Spearman entre a pontuação de cada item validado e a pontuação total.

Nível da Taxonomia de Bloom	**Item**	**R_S**
Conhecimento	1	0.196
	2	0.493*
	3	0.211
	4	0.636*
	5	0.496*
	6	0.271
Compreensão	8	0.011
	9	0.657*
	11	-0.162
	13	0.685*
Aplicação	15	0.197
	16	0.403*
	17	0.542*
	18	0.580*
Análise	20	0.344
	21	0.576*
	22	0.509*
Síntese	23	---
	24	0.732*
Avaliação	25	0.462*
	26	0.546*

Tabela 4 – Coeficiente de Correlação de Spearman

Após o cálculo da correlação entre cada item e o total, selecionámos os que atingiram um nível de significância de pelo menos 0,05. Os itens selecionados que correspondem

a este nível de significância estão assinalados com asterisco. O item Q10 foi excluído porque não teve nenhuma resposta correta logo não foi possível calcular a correlação com a pontuação total. Optámos por selecionar apenas os itens com uma correlação significativa porque existem itens com esta característica em todos os níveis da Taxonomia. No total foram selecionados 13 itens: três do nível Conhecimento; dois do nível Compreensão; três do nível Aplicação; dois do nível Análise; um do nível Síntese; e dois do nível Avaliação.

Em seguida, calculámos o índice de fiabilidade através do modelo alfa de Cronbach para os itens selecionados. A prova atingiu um alfa de 0,837 o que, segundo Peterson (1994), é considerado um bom valor de fiabilidade.

3.3. Análise do Grau de Dificuldade dos Itens

Para analisar as diferenças entre a frequência de respostas corretas em cada item validado, considerámos como hipótese nula todos os itens terem o mesmo grau de dificuldade.

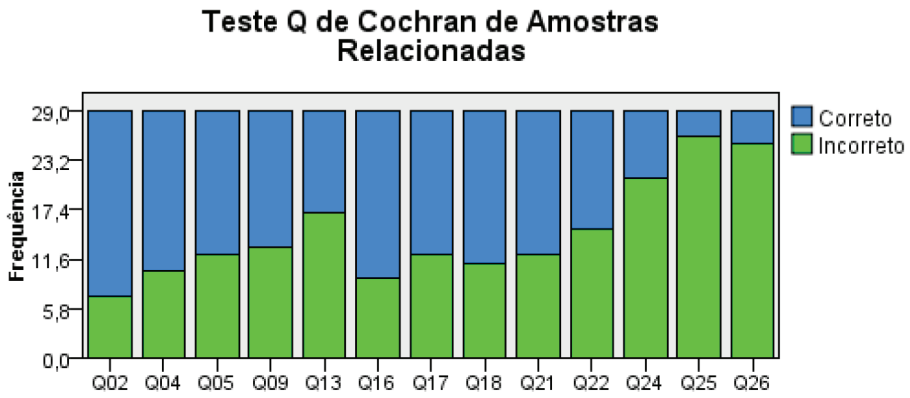

Figura 1 – Frequência de respostas corretas por item

A Figura 1 apresenta a frequência de respostas corretas em cada um dos itens. As diferenças entre as proporções são estatisticamente significativas: Q(12) = 76,767; n=29; p=0,000 o que indica que os itens não têm todos o mesmo grau de dificuldade. Podemos observar uma tendência para a diminuição de respostas corretas a partir do nível Aplicação.

A análise de comparações múltiplas permitiu observar diferenças significativas entre as proporções de respostas corretas entre as primeiras questões da prova e as últimas. Na Tabela 5 apresentamos os resultados que obtiveram o p-value menor que 0,001, ou seja, as comparações duas-a-duas entre os itens cuja diferença entre a proporção de respostas correta é maior (método *Pairwise*).

Amostra 1 – Amostra 2	Teste T	Erro-padrão	Teste Z	p-value	p-value ajustado
Q25-Q02	0,655	0,116	5,670	0,000	0,000
Q26-Q02	0,621	0,116	5,371	0,000	0,000
Q25-Q16	0,586	0,116	5,073	0,000	0,000
Q25-Q04	0,552	0,116	4,774	0,000	0,000
Q26-Q16	0,552	0,116	4,774	0,000	0,000
Q25-Q18	0,517	0,116	4,476	0,000	0,001
Q26-Q04	0,517	0,116	4,476	0,000	0,001
Q25-Q21	0,483	0,116	4,178	0,000	0,002
Q25-Q05	0,483	0,116	4,178	0,000	0,002
Q25-Q17	0,483	0,116	4,178	0,000	0,002
Q26-Q18	0,483	0,116	4,178	0,000	0,002
Q24-Q02	0,483	0,116	4,178	0,000	0,002
Q25-Q09	0,448	0,116	3,879	0,000	0,008
Q26-Q05	0,448	0,116	3,879	0,000	0,008
Q26-Q17	0,448	0,116	3,879	0,000	0,008
Q26-Q21	0,448	0,116	3,879	0,000	0,008
Q26-Q09	0,414	0,116	3,581	0,000	0,027
Q24-Q16	0,414	0,116	3,581	0,000	0,027

Tabela 5 – Comparação pelo método *Pairwise*

As comparações com as diferenças estatisticamente mais significativas ocorreram entre os itens Q25 e Q02, Q26 e Q02, Q25 e Q16, Q25 e Q04 e Q26 e Q16 (p-value < 0,001 e $p\text{-value}_{aj}$ < 0,001) o que indica que os itens Q02 (nível Conhecimento), Q04 (nível Conhecimento) e Q16 (nível Aplicação) correspondem às questões com menor grau de dificuldade e as questões Q25 (nível Avaliação) e Q26 (nível Avaliação) correspondem às questões com maior grau de dificuldade.

Analisando estes resultados em conjunto com a frequência de respostas apresentada na Figura 1, podemos afirmar que, tal como era esperado, os primeiros níveis da Taxonomia de Bloom têm, no geral, uma maior proporção de respostas corretas do que os últimos níveis. Este resultado permite-nos concluir que a prova tem itens pouco discriminativos pertencentes aos primeiros níveis da taxonomia mas de elevada importância pelo seu carácter motivacional e itens mais discriminativos nos últimos níveis que aferem conhecimento de programação mais profundo.

4. Conclusões

Neste artigo apresentámos o processo de criação e validação de uma prova de programação, desenvolvida para contribuir para a uniformização dos instrumentos de recolha de dados na aprendizagem inicial da programação. No processo de elaboração

seguimos as orientações de Tuckman (2012) para o desenvolvimento de testes de desempenho e considerámos os tipos de dados pretendidos para análise, a flexibilidade de resposta, o tempo de preenchimento para cada questão e o grau de dificuldade.

Na estrutura da prova utilizámos questões de escolha múltipla que permitem reduzir a subjetividade da correção e questões de desenvolvimento que fornecem mais detalhes sobre o raciocínio do aluno. Após excluirmos os itens menos rigorosos consideramos que o número total (10 PEM e três PD) é adequado para 90 minutos. Os resultados obtidos na validação do conteúdo e fiabilidade da prova demonstram que a prova é constituída por itens válidos.

Consideramos que a utilização de uma taxonomia constituiu uma mais-valia no desenvolvimento da prova porque permitiu apoiar a construção e análise das questões de acordo com o grau de dificuldade pretendido. Os resultados da análise do grau de dificuldade revelaram que a prova tem itens mais discriminativos correspondentes aos últimos níveis da Taxonomia de Bloom e menos discriminativos mas mais motivantes nos primeiros níveis da taxonomia. Estes resultados correspondem às expectativas de Bloom et al. (1976) quando referem que é esperado mais sucesso nos primeiros níveis da taxonomia.

No Anexo A disponibilizamos a prova na versão final. No entanto, entendemos que deverá ser validada com novas amostras porque a fiabilidade foi analisada a partir de um grupo composto por apenas 29 alunos da mesma escola. Seria interessante verificar as diferenças estatísticas entre as respostas da prova utilizando grupos de diferentes cursos profissionais e com diferentes características.

Como trabalho posterior iremos desenvolver um estudo que utilizará esta prova para aferir os conhecimentos iniciais de programação adquiridos pelos alunos através de diferentes linguagens de programação. Este estudo será implementado no ano letivo 2017/2018 e permitirá verificar as diferenças entre grupos relativamente à aprendizagem com um ambiente de software ou com uma linguagem de programação tradicional. A utilização de um grupo-piloto que iniciou a aprendizagem em 2016/2017 para verificar a fiabilidade da prova permite antever com alguma confiança que os resultados serão fiáveis na instrumentação utilizada uma vez que o grupo-piloto tem características próximas dos grupos que participarão no trabalho experimental (Tuckman, 2012). Esperamos ainda que esta prova possa vir a ser utilizada em estudos desenvolvidos por outros investigadores, pois a credibilidade da investigação reside também no uso de instrumentos válidos e fiáveis (cf. Almeida e Freire, 2016; Runa e Miranda, 2015; Tuckman, 2012; entre outros).

Referências

Almeida, L., & Freire, T. (2016). *Metodologia de investigação em psicologia e educação* (5.ª Ed.. Revista). Braga: Psiquilíbrios.

Bloom, B. S., Engelhart, M. D., Furst, E. J., Hill, W. H., & Krathwohl, D. R. (1976). *Taxionomia de objetivos educacionais: domínio cognitivo*. Porto Alegre: Editora Globo.

Cooper, S. Dann, W., & Paush, R. (2003). Teaching objects first in introductory computer science. In *Proceeding of the 34th Technical Symposium on Computer Science Education* (pp. 191-195). New York: ACM.

Costa, J. M., & Miranda, G. L. (2017). Relation between Alice software and programming learning: A systematic review of the literature with meta-analysis. *British Journal of Education Technology*, 48(6), 1464–1474. doi: 10.1111/bjet.12496

Direção-Geral de Formação Vocacional (2005). Programa componente de formação técnica da disciplina de Linguagens de Programação.

Eisinga, R., Grotenhuis, M., & Pelzer, B. (2013). The reliability of a two-item scale: Pearson, Cronbach, or Spearman-Brown?. *International Journal of Public Health*, 58(4), pp. 637–642. doi: 10.1007/s00038-012-0416-3

Fuller, U., Johnson, C. G., Ahoniemi, T., Cukierman, D., Hernán, Losada, I., & Jackova, J., et al. (2007). Developing a computer science specific learning taxonomy. In *Proceedings of the 7th Annual Conference on Innovation and Technology in Computer Science Education* (pp. 152-170). United Kingdom: Association for Computing Machinery.

Johnsgard, K., & McDonald, J. (2008). Using Alice in Overview Courses to Improve Success Rates in Programming I. In *Proceedings of the 21st Conference on Software Enginering Education and Training* (pp. 129-136). Charleston, S.C.: IEEE. doi: 10.1109/CSEET.2008.35

Krathwohl, D. R. (2002). A Revision of Bloom's Taxonomy: An Overview. *Theory into Practice, 41(4), 212–218.*

Lawshe, C. H. (1975). A Quantitative Approach to Content Validity. P*ersonnel Psychology*, 28 (4), 563–575.

Lister, R. (2003). First year programming: let all the flowers bloom. In *Proceedings of the* F*ifth Australian Conference on Computing Education* (pp. 221-230). Australia: Association for Computing Machinery.

Liu, C., Cheng, Y., & Huang, C. (2011). The effect of simulation games on the learning of computational problem solving. *Computers & Education, 57*, 1907–1918.

Marôco, J. (2014). *Análise Estatística com o SPSS Statistics* (6ª ed.). Pêro Pinheiro: Report Number.

Moons, J., & Backer, C. (2013). The design and pilot evaluation of an interactive learning environment for introductory programming influenced by cognitive load theory and constructivism. *Computers & Education, 60*, 368–384.

Moskal, B., Lurie, D., & Cooper, S. (2004). Evaluating the effectiveness of a new instructional approach. In *Proceedings of the 35th Technical Symposium on Computer Science Education* (pp. 75-79). New York: ACM.

Peterson, R.A. (1994). A meta-analysis of Cronbach´s coefficient alpha. *Journal of Consumer Research*, 21(2), 381–391.

Pinto, A. C. (2001). Factores relevantes na avaliação escolar por perguntas de escolha múltipla. *Psicologia, Educação e Cultura*, 5 (1), 23–44.

Pinto, M., Dias, P., & João, S. (2009). Programa de Aplicações Informáticas B. Lisboa: Direção-Geral de Inovação e Desenvolvimento Curricular.

Runa, A. , & Miranda, G. L. (2015). Validação Portuguesa das Escalas de Bem-estar e Mal-estar Emocional. *RISTI - Revista Ibérica de Sistemas e Tecnologias de Informação,* (16), 129–144. DOI: 10.17013/risti.16.129-144

Sykes, E. R. (2007). Determining the effectiveness of the 3D Alice programming environment at the computer science I level. *Journal od Educational Computing Research, 36*(2), 223–244.

Tuckman, B. (2012). *Manual de Investigação em Educação* (4ª ed.). Lisboa: Fundação Calouste Gulbenkian.

Wang, T., Mei, W., Lin, S., Chiu, S., & Lin, J. (2009). Teaching programming concepts to high schoool students with Alice. In *Proceedings of the 39th ASEE/IEEE Frontiers in Education Conference* (pp. 1-6). St Antonio, TX: IEEE.

Zhang, J., Liu, L., &, Pablos, P. (2014). The auxiliary role of information technology in teaching enhancing programming course using Alice. International Journal of Engineering Education, 30(3), 1–6.

Anexo A

Prova De Avaliação Das Competências Iniciais De Programação

Esta prova está inserida no âmbito de um estudo sobre a aprendizagem inicial da programação. A sua aplicação pretende aferir o conhecimento de conceitos estruturantes de programação, nomeadamente o desenvolvimento de algoritmos, variáveis, tipos de dados, operadores e estruturas de controlo.

A prova está dividida em dois grupos. O primeiro grupo é constituído por perguntas de escolha múltipla com quatro opções de resposta. Cada questão tem uma única resposta correta que deverá assinalar colocando uma cruz na opção que considera correta. O segundo grupo é constituído por perguntas de desenvolvimento. Deverá desenvolver a sua resposta no espaço disponibilizado abaixo de cada questão.

Grupo I

Perguntas de Escolha Múltipla

1. Caso pretenda criar um valor que não irá ser alterado ao longo do meu algoritmo, devo inicializar:
   - □ Uma constante
   - □ Uma variável
   - □ Uma estrutura de repetição
   - □ Uma estrutura de decisão
2. Em programação, a condição *Se.../Senão...* é uma estrutura de:
   - □ Constantes
   - □ Variáveis
   - □ Repetição
   - □ Decisão
3. Em programação, a condição *Enquanto...* é uma estrutura de:
   - □ Constantes
   - □ Variáveis
   - □ Repetição
   - □ Decisão
4. Considere o seguinte algoritmo:

```
1.  Algoritmo Contas
2.  Var n1, n2: inteiro
3.  Início
4.      Ler n1, n2
5.      Escrever n1 + n2
6.  Fim
```

4.1. A variável n1 é declarada na linha:
- □ 2
- □ 3

- □ 4
- □ 5

4.2. É atribuído um valor à variável n1 na linha:

- □ 2
- □ 3
- □ 4
- □ 5

5. Considere o seguinte algoritmo:

1.	*Algoritmo Contas*
2.	*Var n1, n2: real*
3.	*Início*
4.	*Ler n1*
5.	*Se n1 > 0*
6.	*n1 ← n1 / 2*
7.	*Senão*
8.	*Escrever "Número inválido"*
9.	*Fim*

5.1. Caso pretenda guardar o resultado do cálculo *n1 / 2* na variável *n1* apenas caso *n1* seja maior que 3, devo modificar a instrução da linha:

- □ 5 para a instrução *Se n1 > 2*
- □ 5 para a instrução *Se n1 > 3*
- □ 4 para a instrução *n1 ← n1 / 2*
- □ 4 para a instrução *n1 ← 3*

5.2. Caso a variável *n1* recebida na linha 4 seja o valor 3, o *output* do programa será:

- □ Número inválido
- □ 0
- □ 1.5
- □ O programa não escreve nenhum resultado

5.3. Caso a variável *n1* recebida na linha 4 seja o valor 0, o *output* do programa será:

- □ Número inválido
- □ 0
- □ 1.5
- □ O programa não escreve nenhum resultado

6. Considere o seguinte algoritmo:

```
1.  Algoritmo ContasMais
2.  Var n1, n2, n3: inteiro
3.  Início
4.  Escrever "Insira um número real"
5.      Ler n1
6.  Escrever "Insira um número real menor que o anterior"
7.  Ler n2
8.  Escrever "Insira um número real menor que o primeiro"
9.  Ler n3
10. Se n1> n2 e n1> n3
11. Escrever n1 + n2 + n3
12. Senão
13. Escrever 0
14. Fim
```

6.1. Indique a linha de código onde é utilizado um tipo de operador aritmético:

- □ Linha 2
- □ Linha 5
- □ Linha 10
- □ Linha 11

6.2. O operador utilizado na linha 11 é do tipo:

- □ Relacional
- □ Lógico
- □ Aritmético
- □ Variável

Grupo II

Perguntas de Desenvolvimento

1. Crie um programa que peça ao utilizador o comprimento do lado de um quadrado e que devolva o resultado do cálculo da área.

2. Crie um programa com o nome *Calculadora* que devolva o resultado da soma, multiplicação, divisão ou subtração de dois números decimais recebidos pelo utilizador. No final deverá escrever o resultado da operação escolhida pelo utilizador.

Exemplo 1
Input: 3+6
Output: 9

Exemplo 2
*Input: 4*2.5*
Output: 10

3. Considere o seguinte programa:

```
1.    programa Subtrair
2.        Var n1, resultado: inteiro
3.        Var n2: decimal
4.        Constante n3: 9
          Início
              Ler n1
              Ler n2
              resultado ← n1 – n2 - n3
              Escrever resultado
      Fim
```

3.1. O tipo da variável *resultado* está incorreto. Justifique esta afirmação.