Learning Analytics no Ensino de Introdução à Programação de Computadores

Elaine H. T. Oliveira[1], **David B. F. de Oliveira**[1],
Leandro S. G. Carvalho[1], **Filipe D. Pereira**[2]

[1]Instituto de Computação – Universidade Federal do Amazonas (UFAM)
[2]Departamento de Ciência da Computação – Universidade Federal de Roraima (UFRR)
{elaine,david,galvao}@icomp.ufam.edu.br, filipe.dwan@ufrr.br

Resumo. *Disciplinas de programação introdutória apresentam uma alta taxa de reprovação no mundo todo, e na Universidade Federal do Amazonas, isso também acontece. Desde que um grupo de professores dessa instituição resolveu reformular a disciplina, algumas iniciativas de* Learning Analytics *vêm sendo adotadas. O objetivo deste artigo é apresentá-las de maneira abrangente, além de alguns resultados obtidos durante esses últimos anos de pesquisa.*

Abstract. *Introductory programming classes have a high failure rate worldwide, and at Federal University of Amazonas, this is also the case. Since a group of professors from this institution decided to reformulate the discipline, some Learning Analytics initiatives have been adopted. The objective of this article is to present them in a comprehensive way, in addition to some results obtained during these last years of research.*

1. Introdução

Introdução à Programação de Computadores (IPC), mais conhecida no exterior como *Computer Science* 1 (CS1), é uma disciplina que inicia estudantes de graduação, principalmente da área de Ciências Exatas e Engenharia, em lógica de programação. Geralmente, também é nessa disciplina que eles têm o primeiro contato com uma linguagem de programação e com estruturas de dados básicas.

Há tempos, pesquisas apontam para índices de reprovação e evasão bastante elevados em disciplinas de IPC [Bennedsen and Caspersen 2019, Blikstein 2011, Castro-Wunsch et al. 2017, Ihantola et al. 2015, Lacave et al. 2018, Watson and Li 2014]. Alguns pesquisadores argumentam que parte desse problema se deve ao fato de que alunos que não são de computação (*non-majors*) podem sentir mais dificuldade por não possuírem a motivação intrínseca de alunos que sabem que vão programar durante todo o seu curso e vida profissional [Echeverría et al. 2017, Norman and Adams 2015, Santana and Bittencourt 2018].

No entanto, vivemos em uma época em que as tecnologias digitais permeiam e influenciam toda a sociedade, seja através dos onipresentes *smartphones*, dos vários dispositivos móveis embarcados ou da Inteligência Artificial que está nos sistemas que utilizamos no dia-a-dia. Por isso, saber programar ou, ao menos, ter noções de lógica de programação é uma habilidade importante, senão essencial para o desenvolvimento de um cidadão consciente da realidade que o cerca.

DOI: 10.5753/wapla.2022.226870

Preocupados com a situação da disciplina em sua instituição que, em 2016, estava com uma taxa de reprovação de 60%, professores e professoras do Instituto de Computação da Universidade Federal do Amazonas resolveram repensar e reformular a disciplina [Carvalho et al. 2016]. Esse processo consistiu em adotar um conjunto coordenado de estratégias de ensino-aprendizagem, sendo algumas delas: uma metodologia mais voltada à prática, com o apoio de um juiz on-line; avaliações mais frequentes; material didático totalmente reformulado e padronizado às diversas turmas; e apoio de tutores (alunos de pós-graduação em estágio em docência).

Desde então, o grupo vem se dedicando em acompanhar os vários aspectos relacionados ao processo de ensino e aprendizagem em programação. Para que isso aconteça de forma efetiva, o mecanismo de ação conta com basicamente quatro elementos: adoção de ferramentas on-line, coleta de dados educacionais, *Learning Analytics* e gamificação.

O objetivo deste artigo é apresentar algumas iniciativas e resultados obtidos relacionados a *Learning Analytics* em uma instituição pública de ensino superior, com dados de disciplinas introdutórias de programação.

A seguir, será apresentado o contexto das pesquisas relacionadas à disciplina IPC na Universidade Federal do Amazonas. Na Seção 2, é descrito o contexto educacional da instituição. Na Seção 3, as quatro engrenagens mencionadas são detalhadas, com exemplos de publicações e resultados obtidos. Na Seção 4, são apresentados iniciativas e resultados. Por fim, na Seção 5, são apresentadas as considerações finais com as perspectivas de trabalhos futuros.

2. Contexto educacional

As turmas de IPC da Universidade Federal do Amazonas são ministradas pelo Instituto de Computação. Ao todo, a unidade atende 17 cursos de graduação nas áreas de engenharia e ciências exatas. O conteúdo programático da disciplina é dividido em sete módulos: (1) estrutura sequencial; (2) estrutura condicional simples (*if-then-else*); (3) estrutura condicional aninhada (*if-then-else* aninhada); (4) estrutura de repetição por condição (laços *while*); (5) vetores e *strings*; (6) estrutura de repetição por contagem (laços *for*); e (7) matrizes. Cada módulo possui um material didático correspondente e uma lista de exercícios práticos, disponibilizados por meio do juiz on-line CodeBench[1]. Os exercícios devem ser solucionados usando a linguagem Python.

O CodeBench tem sido uma ferramenta extremamente útil para o processo tanto de ensino quanto o de aprendizagem. Ele possui um Ambiente de Desenvolvimento Integrado (*Integrated Development Environment* – IDE), usado pelos alunos para desenvolver as soluções dos exercícios propostos. Atualmente, esse ambiente suporta as principais funcionalidades de um IDE típico, tais como: *Autocompletion*, *Autosave*, *Syntax Highlighting*, busca e substituição de *strings*, etc.

O código desenvolvido por um aluno para um dado exercício fica disponível em um diretório que pode ser acessado através de um terminal *shell* disponível na própria IDE. Tais terminais dão acesso a um ambiente virtual Linux criado através de Docker Containers. O CodeBench cria um Docker Container independente para cada aluno, de tal forma que um aluno não pode acessar os códigos dos demais. Além disso, todos os

[1] `https://codebench.icomp.ufam.edu.br/`

códigos são executados nos *containers* de seus respectivos alunos. Esse isolamento das execuções provê alta segurança e impede qualquer tipo de acesso ao *host* principal do sistema.

O CodeBench conta com um banco de 5244 exercícios de programação já cadastrados, que podem ser usados pelos professores para compor os trabalhos de suas turmas. Ao todo, são 6252 usuários, entre alunos e professores da Universidade Federal do Amazonas e de outras cinco instituições parceiras.

Como mencionado previamente, para que todo o processo de ensino e aprendizagem ocorra de forma efetiva, o mecanismo de ação do grupo de pesquisa é apoiado em quatro elementos (adoção de ferramentas on-line, coleta de dados educacionais, *Learning Analytics* e gamificação), explicitados na seção a seguir.

3. Mecanismo de ação

Para que houvesse uma mudança real no processo de ensino e aprendizagem da disciplina, a equipe responsável por repensar a metodologia resolveu se apoiar em alguns elementos importantes que visassem algumas atividades automáticas e decisões baseadas em evidências.

3.1. Adoção de ferramentas on-line

Um dos pontos primordiais foi a adoção de ferramentas on-line, sendo uma delas um juiz on-line. Em cursos de computação, os juízes on-line são ferramentas essenciais para facilitar o ensino e a aprendizagem. Através dos sistemas juízes on-line, os professores podem elaborar e disponibilizar exercícios de programação para seus alunos. Os alunos, por sua vez, devem codificar soluções para os exercícios propostos através de IDEs on-line disponibilizadas por tais sistemas. Quando o aluno submete a solução de um dado exercício, o sistema informa instantaneamente se sua solução está correta ou errada.

Na Universidade Federal do Amazonas, foi adotado o juiz on-line CodeBench, criado por um dos autores deste trabalho. Ainda que de propósito geral, a decisão de desenvolver um juiz on-line próprio foi importante para personalizar o ensino e entender os requisitos necessários de um sistema como esse. Sendo o CodeBench uma ferramenta ‘da casa’, ele pode ser adaptado às necessidades dos docentes, discentes e das pesquisas em andamento. Ele pode ser utilizado como ferramenta de apoio em qualquer modalidade de ensino, seja presencial, remoto ou híbrido. Além disso, o sistema se mostrou ainda mais útil quando a pandemia veio e o ensino remoto foi necessário.

3.2. Gamificação

Uma das estratégias adotadas foi a integração de uma plataforma de gamificação baseada em um jogo RPG multiplayer com o CodeBench. O jogo contém personagens, edificações e seres mágicos próprios. Ao solucionar um exercício de programação no juiz on-line, uma carta é sorteada para o aluno, definindo sua sorte e permitindo que ele avance na história do jogo. Dentro do ambiente (mundo do jogo), os alunos podem interagir entre si (aspecto multiplayer), coletar recompensas, realizar missões, vencer inimigos e explorar vários elementos existentes no mapa. Assim como na maioria dos jogos RPG, os alunos podem escolher seus próprios avatares (femininos ou masculinos) e acumular pontos de experiência e moedas à medida que evoluem no jogo [Pessoa et al. 2019].

A versão atual do jogo é composta por sete fases e em todas elas têm edificações bloqueadas, cartas de experiência e moedas, missões coletivas, presentes na primeira versão do jogo, e são propostos novos elementos de jogos, como missões secundárias, passagens secretas, locais de descanso para recuperar a vida e batalha final de cada fase, havendo alteração dos objetivos, enredo, cenários e descrições dos eventos.

3.3. Coleta de dados educacionais

Os dados gerados em juízes on-line diferem bastante dos dados gerados por outros tipos de aplicações Web. Para o juiz on-line CodeBench, a coleta de dados abrange os testes e submissões feitos pelos alunos, as mensagens de erro gerados pelo compilador/interpretador e todo o processo de desenvolvimento dos códigos.

O processo de desenvolvimento de código pode ser coletado em diferentes níveis de granularidade, podendo ser de: códigos dos testes e submissões, *snapshots*, edições em linha ou o mais granular, o de *keystrokes*. Cada nível está associado a um tamanho de dado e à frequência da coleta. No CodeBench, essa coleta é feita com a granularidade *keystroke*.

Além das atividades no IDE, são coletados: movimentos do mouse e cliques na interface, recursos didáticos acessados, *logins* e *logouts* e atividades na plataforma de gamificação. A coleta dos dados da gamificação também é bem granular. [Ihantola et al. 2015] explicam que essa é uma maneira direta de configurar uma coleta de dados para realizar uma análise refinada dos comportamentos de programação dos alunos. Isso permite que várias pesquisas possam ser realizadas e diferentes estratégias adotadas. Devido a essa coleta, as oportunidades de pesquisa sobre o uso de *Learning Analytics* em sistemas juízes on-line são amplas e diferenciadas.

O dataset[2] do CodeBench contém os *logs* coletados de todos os alunos matriculados em IPC nos anos acadêmicos de 2016 a 2021 e está disponível ao público para análises e colaborações em pesquisa.

3.4. Learning analytics

O alto índice de reprovações e desistências em disciplinas de programação, notadamente para cursos STEM (Science, Technology, Engineering and Mathematics), é um problema grave e recorrente. Uma forma de lidar com esse problema é tentar entender as causas e trabalhar em possíveis soluções, provendo informações e recomendações que ajudem os professores a mitigar as dificuldades dos alunos. Algumas perguntas que recorrentemente procuramos responder são:

- Quais são os alunos que estão com dificuldade na disciplina? Em que assuntos ou exercícios eles têm mais dificuldade?
- Quais são os exercícios mais indicados para cada aluno?
- Que aspectos o aluno precisa melhorar para ter um bom desempenho na disciplina?
- Os alunos estão desenvolvendo práticas inaceitáveis (‘cola’) na resolução das listas de exercícios?

[2] `https://codebench.icomp.ufam.edu.br/dataset`

Para responder essas e outras questões, o grupo de pesquisa da instituição recorre a técnicas de Learning Analytics como predições, descrições, intervenções, recomendações e personalização, que serão exploradas na Seção 4. A seguir, são apresentados alguns resultados e iniciativas relacionados a *Learning Analytics* obtidos até o momento.

4. Algumas iniciativas e resultados

Para responder às questões listadas na Seção 3.4, foi gerado um **perfil de programação** para cada aluno a partir de atributos extraídos dos *logs* do juiz on-line [Dwan et al. 2017, Pereira et al. 2019]. São mais de 30 atributos, sendo estes alguns deles:

- propPaste – Proporção entre caracteres colados e caracteres digitados;
- changes – Mudanças no código entre submissões;
- qtdEvents – Quantidade de eventos durante o desenvolvimento;
- procrastination – Tempo entre o início da codificação e o término do trabalho;
- attempts – Número médio de submissões por exercício;
- syntaxError – Proporção de submissões com erros.

Uma das iniciativas de pesquisa conduzidas foi utilizar técnicas de clusterização sobre os perfis de programação. Nessa iniciativa, os alunos são agrupados em três grandes grupos: alunos efetivos, medianos e não efetivos [Pereira et al. 2020b]. Chamamos de "alunos efetivos" aqueles que progridem no aprendizado da programação, normalmente levando a resultados bem-sucedidos [Robins 2019]. Por outro lado, "alunos inefetivos" são aqueles que não progridem ou aplicam esforço excessivo, geralmente levando a resultados malsucedidos [Robins 2019]. E, por último, os medianos são os que não se encaixam em nenhuma das duas categorias anteriores. A Figura 1 mostra que o desempenho do aluno está diretamente relacionado com o seu comportamento no juiz on-line. Neste caso, o Cluster A contém os alunos efetivos, o Cluster B se refere aos medianos e o Cluster C, aos inefetivos.

Para lidar com as altas taxas de reprovação e desistência, é essencial identificar, com antecedência, alunos com dificuldades [Dwan et al. 2017, Fonseca et al. 2019, Pereira et al. 2019]. Usando *Deep Learning* com base nos perfis de programação, foi possível prever o desempenho dos alunos com uma acurácia de 82,5% ainda nas duas primeiras semanas de aula [Pereira et al. 2020a]. Com a previsão de desempenho antecipada, professores podem prover exercícios extras e mais desafiadores para alunos que têm um bom desempenho na disciplina e suporte personalizado para alunos com dificuldade.

Um dos resultados mais atuais sobre a predição de desempenho é a explicabilidade do modelo de aprendizagem de máquina [Pereira et al. 2021a]. O grupo de pesquisa entende que uma resposta numérica obtida a partir de métodos caixa preta não atende aos anseios do professor e do aluno. A informação deve ser acompanhada de sentido e significado. Se alguma atitude deve ser tomada, os atores do processo devem estar cientes das causas que levaram a tal resultado. Então, além de prever o desempenho, métodos de predição com o SHAP (SHapley Additive exPlanations) [Lundberg and Lee 2017], baseado em teoria dos jogos, são capazes de explicar os resultados de suas predições. Essa predição antecipada empoderada por sua explicação pode favorecer ainda mais as intervenções.

Já que conhecemos quais são os comportamentos que levam a um bom desempenho nas disciplinas de programação, uma questão de pesquisa que tem direcionado

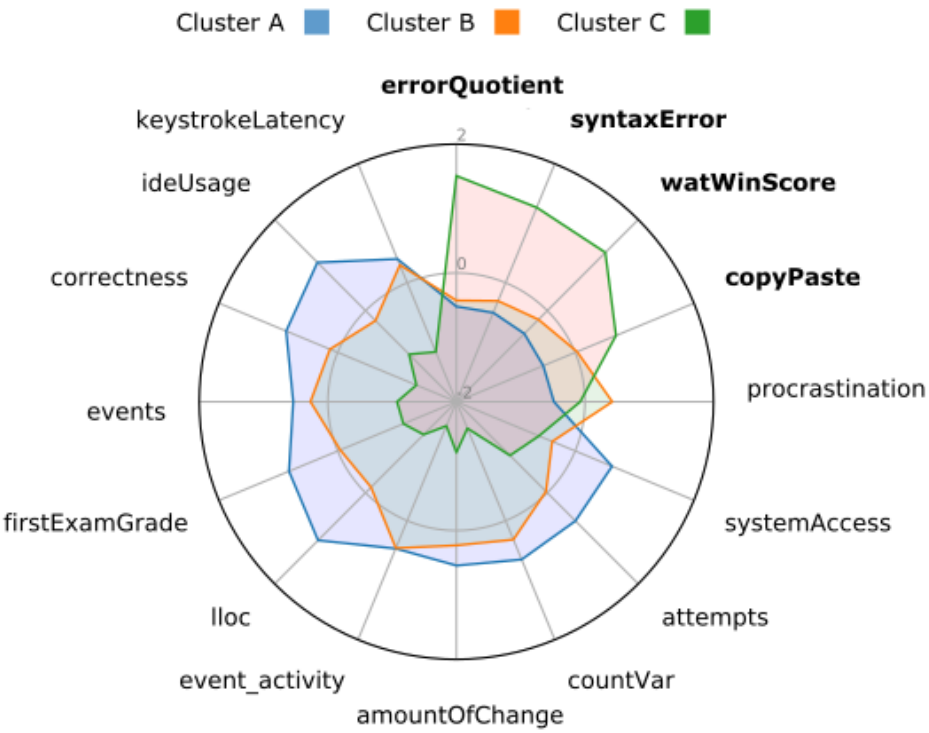

Figura 1. Perfil de programação do aluno em cada cluster (Cluster A - alunos efetivos, Cluster B - medianos e Cluster C - inefetivos).

alguns esforços atuais do grupo é: seria possível usar gamificação customizada para estimular cada aluno a se comportar de uma forma mais efetiva? Uma das estratégias usadas para esse fim envolve a análise dos mais variados dados gerados pela gamificação, os tipos de jogadores, as preferências dos usuários para, assim, prover um ambiente mais estimulante para o aluno se engajar na disciplina [Pessoa et al. 2019, Ribeiro et al. 2020, Pessoa et al. 2021].

Outra frente de análise é a detecção de plágio dos códigos [Lavareda Filho et al. 2020, Oliveira et al. 2021]. Nas turmas iniciais de programação, os códigos desenvolvidos pelos alunos tendem a ser simples e pequenos, dificultando o uso de técnicas convencionais de detecção de plágio. Além disso, nessas turmas, os alunos tendem a desenvolver códigos parecidos, pois assimilam o estilo de codificação do professor. Dessa forma, em vez de usar técnicas tradicionais de detecção de plágio, o grupo de pesquisa optou por analisar o comportamento dos alunos durante a criação dos códigos no IDE do juiz on-line. Partindo da premissa de que o comportamento de quem está colando é diferente de quem está realmente resolvendo os exercícios, chegamos aos resultados apresentados na Tabela 1.

Além dos exemplos citados, outras frentes de pesquisas relacionadas a *Learning Analytics* e ao comportamento dos alunos no juiz on-line estão em desenvolvimento, sendo elas:

Tabela 1. Resultados dos classificadores de plágio

	Precisão	Revocação	Medida-F
Plágio	0,753	0,906	0,823
Não-plágio	0,891	0,721	0,797
Média	0,824	0,811	0,809

- **Classificação de dificuldade de exercícios de programação**: o comportamento dos alunos ao tentarem resolver uma questão permite inferir a dificuldade dessa questão [Lima et al. 2021, Lima et al. 2020].
- **Autenticação contínua**: o padrão de comportamento do aluno permite identificar se o usuário logado é de fato quem ele diz ser [Lavareda Filho et al. 2020].
- **Detecção automática de alunos se sentindo confusos**: através do comportamento dos alunos no sistema, é possível identificar alunos que estão se sentindo confusos em um dado exercício.
- **Recomendação de problemas baseados em comportamentos efetivos e inefetivos**: com base no comportamento dos alunos, é possível recomendar ações de intervenção e de mudança de atitude para cada aluno [Pereira et al. 2021b, Freitas Júnior et al. 2020].

5. Conclusões

Neste artigo, foram apresentadas várias iniciativas de um grupo de pesquisa da Universidade Federal do Amazonas, que adota uma metodologia baseada na prática de exercícios e na correção automática de códigos. Os resultados obtidos até o momento foram apresentados de maneira bastante geral, com várias referências para trabalhos já publicados.

O objetivo deste artigo foi apresentar a grande variedade e as múltiplas possibilidades de pesquisa em *Learning Analytics* advindas da adoção de um juiz on-line, da coleta de dados educacionais e gamificação. Com a visão ampla apresentada neste artigo, espera-se contribuir para uma rica troca de experiências entre pesquisadores e profissionais sobre a adoção de LA, fornecer *insights* práticos sobre a adoção e a implementação de LA e apresentar um caso de uma ferramenta real em utilização e que contribui para adoção de LA no Brasil.

Agradecimentos

O presente trabalho é decorrente do projeto de Pesquisa e Desenvolvimento (PD) 001/2020, firmado entre a Fundação da Universidade do Amazonas e FAEPI, que conta com financiamento da Samsung, usando recursos da Lei de Informática para a Amazônia Ocidental (Lei Federal nº 8.387/1991), estando sua divulgação de acordo com o previsto no artigo 39.º do Decreto nº 10.521/2020. Além disso, Elaine Oliveira recebeu apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) (Processo 308513/2020-7).

Referências

Bennedsen, J. and Caspersen, M. E. (2019). Failure rates in introductory programming: 12 years later. *ACM Inroads*, 10(2):30–36.

Blikstein, P. (2011). Using learning analytics to assess students behavior in open-ended programming tasks. *Proceedings of the 1st International Conference on Learning Analytics and Knowledge*, pages 110–116.

Carvalho, L., Fernandes, D., and Gadelha, B. (2016). Juiz online como ferramenta de apoio a uma metodologia de ensino híbrido em programação. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 27, page 140.

Castro-Wunsch, K., Ahadi, A., and Petersen, A. (2017). Evaluating neural networks as a method for identifying students in need of assistance. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, pages 111–116. ACM.

Dwan, F., Oliveira, E., and Fernandes, D. (2017). Predição de zona de aprendizagem de alunos de introdução à programação em ambientes de correção automática de código. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 28, page 1507.

Echeverría, L., Cobos, R., Machuca, L., and Claros, I. (2017). Using collaborative learning scenarios to teach programming to non-cs majors. *Computer applications in engineering education*, 25(5):719–731.

Fonseca, S., Oliveira, E., Pereira, F., Fernandes, D., and de Carvalho, L. S. G. (2019). Adaptação de um método preditivo para inferir o desempenho de alunos de programação. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 30, page 1651.

Freitas Júnior, H. B., Pereira, F. D., de Oliveira, E. H. T., de Oliveira, D. B. F., and de Carvalho, L. S. G. (2020). Recomendação automática de problemas em juízes online usando processamento de linguagem natural e análise dirigida aos dados. In *Anais do XXXI Simpósio Brasileiro de Informática na Educação*, pages 1152–1161. SBC.

Ihantola, P., Vihavainen, A., Ahadi, A., Butler, M., Börstler, J., Edwards, S. H., Isohanni, E., Korhonen, A., Petersen, A., Rivers, K., Rubio, M. A., Sheard, J., Skupas, B., Spacco, J., Szabo, C., and Toll, D. (2015). Educational data mining and learning analytics in programming: Literature review and case studies. *ACM. Proceedings of the 2015 ITiCSE on Working Group Reports*, pages 41–63.

Lacave, C., Molina, A. I., and Cruz-Lemus, J. A. (2018). Learning analytics to identify dropout factors of computer science studies through bayesian networks. *Behaviour & Information Technology*, 37(10-11):993–1007.

Lavareda Filho, R. M., Colonna, J. G., and Oliveira, D. B. F. (2020). Autenticação contínua de alunos utilizando biometria comportamental em ambiente juiz on-line. In *Anais do XXXI Simpósio Brasileiro de Informática na Educação*, pages 1193–1202. SBC.

Lima, M., de Carvalho, L. S. G., de Oliveira, E. H. T., Oliveira, D. B. F., and Pereira, F. D. (2020). Classificação de dificuldade de questões de programação com base em métricas de código. In *Anais do XXXI Simpósio Brasileiro de Informática na Educação*, pages 1323–1332. SBC.

Lima, M. A., Carvalho, L. S., de Oliveira, E. H., de Oliveira, D. B., and Pereira, F. D. (2021). Uso de atributos de código para classificar a dificuldade de questões de programação em juízes online. *Revista Brasileira de Informática na Educação*, 29:1137–1157.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in neural information processing systems*, pages 4765–4774.

Norman, V. T. and Adams, J. C. (2015). Improving non-CS major performance in cs1. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, pages 558–562.

Oliveira, D. B., Lavareda Filho, R. M., Oliveira, E. H., Carvalho, L. S., Pereira, F. D., Colonna, J. G., and Menezes, A. (2021). Um método de detecção de plágio para sistemas juiz on-line baseado no comportamento dos alunos. In *Anais do XXXII Simpósio Brasileiro de Informática na Educação*, pages 836–848. SBC.

Pereira, F. D., Fonseca, S. C., Oliveira, E. H., Cristea, A. I., Bellhäuser, H., Rodrigues, L., Oliveira, D. B., Isotani, S., and Carvalho, L. S. (2021a). Explaining individual and collective programming students' behavior by interpreting a black-box predictive model. *IEEE Access*, 9:117097–117119.

Pereira, F. D., Fonseca, S. C., Oliveira, E. H., Oliveira, D. B., Cristea, A. I., and Carvalho, L. S. (2020a). Deep learning for early performance prediction of introductory programming students: a comparative and explanatory study. *Brazilian journal of computers in education.*, 28:723–749.

Pereira, F. D., Junior, H. B., Rodriguez, L., Toda, A., Oliveira, E. H., Cristea, A. I., Oliveira, D. B., Carvalho, L. S., Fonseca, S. C., Alamri, A., et al. (2021b). A recommender system based on effort: Towards minimising negative affects and maximising achievement in cs1 learning. In *International Conference on Intelligent Tutoring Systems*, pages 466–480. Springer.

Pereira, F. D., Oliveira, E. H., Fernandes, D., and Cristea, A. (2019). Early performance prediction for CS1 course students using a combination of machine learning and an evolutionary algorithm. In *2019 IEEE 19th International Conference on Advanced Learning Technologies (ICALT)*, volume 2161, pages 183–184. IEEE.

Pereira, F. D., Oliveira, E. H., Oliveira, D. B., Cristea, A. I., Carvalho, L. S., Fonseca, S. C., Toda, A., and Isotani, S. (2020b). Using learning analytics in the amazonas: understanding students' behaviour in introductory programming. *British journal of educational technology*, 51(4):955–972.

Pessoa, M., Fernandes, D., de Carvalho, L. S. G., Oliveira, E., Nakamura, W., and Conte, T. (2019). Codeplay: Uma plataforma de gamificação baseada em jogos de rpg multiplayer. In *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 30, page 843.

Pessoa, M., Melo, R., Haydar, G., Oliveira, D. B., Carvalho, L. S., Oliveira, E. H., Conte, T., Pereira, F. D., Rodrigues, L., and Isotani, S. (2021). Uma análise dos tipos de jogadores em uma plataforma de gamificação incorporada a um sistema juiz on-line. In *Anais do XXXII Simpósio Brasileiro de Informática na Educação*, pages 474–486. SBC.

Ribeiro, R. B. S., de Carvalho, L. S. G., Oliveira, E. H. T., Oliveira, D. B. F., and Pessoa, M. S. P. (2020). Investigação empírica sobre os efeitos da gamificação de um juiz online em uma disciplina de introdução à programação. *Revista Brasileira de Informática na Educação*, 28:461–490.

Robins, A. V. (2019). *Novice Programmers and Introductory Programming*, page 327–376. Cambridge Handbooks in Psychology. Cambridge University Press.

Santana, B. L. and Bittencourt, R. A. (2018). Increasing motivation of CS1 non-majors through an approach contextualized by games and media. In *2018 IEEE Frontiers in Education Conference (FIE)*, pages 1–9. IEEE.

Watson, C. and Li, F. W. (2014). Failure rates in introductory programming revisited. In *Proceedings of the 2014 conference on Innovation & technology in computer science education*, pages 39–44. ACM.