

Francisco de Assis Zampirolli

MCTest

COMO CRIAR E CORRIGIR EXAMES
PARAMETRIZADOS AUTOMATICAMENTE

2	1	6	6	0	7	7	6	6	9	6	6	4	9	9
2	5	5	3	8	6	0	6	7	7	9	5	9	0	9
1	1	9	2	2	5	7	4	1	3	9	8	7	9	8
8	6	2	8	7	7	4	8	1	5	4	1	7	0	1
9	0	7	5	5	1	6	2	2	5	8	2	6	7	7
1	7	9	4	6	6	5	0	9	0	0	4	2	4	4
2	8	5	8	8	1	4	6	1	9	4	5	3	6	3

2a Edição

Em Construção

MCTest:

Como Criar e Corrigir Exames Parametrizados Automaticamente

Francisco de Assis Zampirolli

21 de janeiro de 2026

© 2024 Francisco de Assis Zampirolli da Universidade Federal do ABC (UFABC).
Todos os direitos reservados.

Este livro está sob a Licença:

Creative Commons Attribution-ShareAlike 4.0 International License

Detalhes no endereço: creativecommons.org/licenses/by-sa/4.0

Projeto gráfico: Francisco de Assis Zampirolli

CATALOGAÇÃO NA FONTE

SISTEMA DE BIBLIOTECAS DA UNIVERSIDADE FEDERAL DO ABC

Z26m Zampirolli, Francisco de Assis

MCTest : como criar e corrigir exames parametrizados automaticamente / Francisco de Assis Zampirolli – Santo André, SP : Edição do Autor, 2024.

xxiv, 244 p. : il.

ISBN: 978-65-00-79086-3 (1a Edição)

1. Avaliação da Aprendizagem. 2. Correção Automática. 3. Questão Paramétrica. 4. Programação – Problemas e Exercícios. 5. Moodle. Título.

CDD 22 ed. — 005.4

Dedico este trabalho com imenso amor à minha esposa Cristina e aos meus filhos, Rafael e Eduardo. Agradeço de coração por estarem sempre ao meu lado em todas as conquistas e desafios da vida.

Epígrafe

“Não basta saber, é preciso aplicar. Não basta querer, é preciso também fazer.”
(Johann Wolfgang von Goethe)

Prefáceis

Prefácio da primeira edição

A avaliação dos estudantes é uma atividade crucial para o trabalho de um professor, ao permitir avaliar de maneira precisa o desempenho dos estudantes ao longo do curso, fornecendo *feedback* importante para o seu desenvolvimento acadêmico. Além disso, a avaliação também é um meio importante para aprimorar a prática educativa, permitindo que os professores identifiquem pontos fortes e fracos de sua metodologia e aprimorem sua abordagem pedagógica para proporcionar um melhor aprendizado aos estudantes.

Avaliar inúmeros estudantes individualizadamente é uma tarefa desafiadora para professores em todos os níveis de ensino, especialmente em disciplinas que exigem habilidades e competências específicas. Esse desafio é ainda maior em exames que envolvem exercícios de programação (EP), uma vez que exigem uma avaliação minuciosa dos processos de resolução de problemas, além da compreensão do código e da lógica utilizados pelos estudantes.

Este livro visa prover uma solução abrangente e eficaz para a avaliação de estudantes. A abordagem colaborativa adotada por docentes da Universidade Federal do ABC (UFABC), empregando o sistema de código aberto MCTest, viabiliza a reutilização de conjuntos de questões previamente utilizados por outros colegas que ministram a mesma disciplina.

O MCTest permite a criação e correção de exames de múltipla escolha e dissertativa, incluindo EP no *plugin* VPL (*Virtual Programming Lab*) do ambiente Moodle. Ele oferece questões parametrizadas e exames individualizados que podem ser usados por várias turmas simultaneamente. A principal contribuição desse sistema reside na capacidade de compartilhar questões parametrizadas, que incorporam enunciados em *LATEX* intercalados com códigos em Python.

Este livro consolida mais de uma década de experiência na avaliação anual de milhares de estudantes, a partir de 2012, quando se deu início à automatização do processo seletivo da Especialização em Tecnologias e Sistemas de Informação (TSI) da UFABC, com a criação da primeira versão do MCTest, desenvolvida em Matlab. Desde então, o sistema passou por um substancial processo de evolução, beneficiando-se da colaboração de diversos colegas da UFABC, cujas contribuições permitiram identificar novas necessidades e aprimorar suas funcionalidades. A versão web mais recente abordada neste livro é a 5.2 e encontrando-se disponível para instalação no endereço github.com/fzampirolli/mctest. Duas implantações ativas do sistema estão em execução na UFABC: a versão de produção, hospedada em mctest.ufabc.edu.br; e a versão de desenvolvimento, *backup* e divulgação em vision.ufabc.edu.br.

É de extrema importância enfatizar que o MCTest não tem um caráter comercial e não segue rigidamente o ciclo de vida convencional de desenvolvimento de software. Sua abordagem é fundamentada na prototipagem, seguindo os princípios da Engenharia de Software, em suma, novos requisitos e protótipo rápido para validação de conceitos. No entanto, até o momento, o sistema ainda não atingiu sua versão final, desenvolvido predominantemente por um único programador (com exceção da adaptação realizada no VPL, pelo Heitor Rodrigues Savegnago e seu orientador Prof. Dr. Paulo Henrique Pisani, a quem expresso minha sincera gratidão). Para poder progredir em direção a um produto comercial, seriam necessários processos de engenharia reversa e contínuos aprimoramentos.

Apesar das limitações acima mencionadas, o MCTest é utilizado por alguns docentes e gestores da UFABC, desempenhando um papel de significativa relevância no âmbito da avaliação discente. Essa utilização resulta não apenas em uma notável redução da carga de trabalho repetitivo, mas também contribui para a mitigação de possíveis ocorrências de falhas.

A implantação do sistema é agilizada por meio da adoção do ambiente *VirtualBox* (virtualbox.org), que permite uma implantação local para fins de teste em uma máquina com bom desempenho de processamento, armazenamento (5 GB livre) e memória RAM (≥ 8 GB). Ao optar pela instalação dos sistemas operacionais Ubuntu ou Mint, seguida pela subsequente execução do arquivo `setup-all.sh` disponibilizado no repositório GitHub (github.com/fzampirolli/mctest), mediante privilégios de administrador, o MCTest é instalado de forma contínua, estando acompanhado por um conjunto inicial de dados de teste presente no arquivo `mctest.sql`. Os exemplos ilustrativos apresentados neste trabalho encontram-se no arquivo `book/1ed-br/mctestLivro.sql`, estando também acessíveis no repositório GitHub. A fim de empregar esses exemplos no MCTest, é necessário observar as instruções fornecidas no arquivo `setup-all.sh`.

A obra foi elaborada em 14 capítulos, divididos em quatro partes principais. A Parte I tem como principal objetivo introduzir os leitores ao sistema, oferecendo detalhes sobre seus componentes e funcionalidades essenciais. A Parte II explora a variedade de tipos de questões disponíveis. A Parte III abrange a criação e administração de exames. Por fim, a Parte IV proporciona estudos de caso e exemplos práticos, sendo estes últimos divulgados em artigos científicos. Estes experimentos foram realizados em modalidades híbridas, totalmente remotas e totalmente presenciais. Para aqueles sem interesse na parte paramétrica, que requer conhecimentos de programação, ou sem usar o banco de questões, há a possibilidade de pular diversos capítulos, conforme explicado no final do Capítulo 1.

Ao usar o MCTest, os professores têm à disposição uma ferramenta para aprimorar a avaliação das habilidades e competências dos estudantes. Assim, os docentes conseguem reduzir o esforço exigido na criação e correção de exames e atividades práticas. No entanto, o ponto crucial desse processo reside na ampla coleção de questões elaboradas e disponibilizadas pelos professores da UFABC, que atualmente totalizam 2.273 questões, embora limitadas aos docentes de cada disciplina.

O MCTest permanece em constante processo de evolução e aprimoramento, incentivando os usuários a contribuírem com suas experiências para torná-lo ainda mais eficiente e alinhado com as demandas educacionais em constante transformação.

Ao longo desta obra, as funcionalidades do MCTest serão exploradas, permitindo aos leitores

compreenderem suas características e recursos. Espera-se que esse trabalho proveja um guia completo sobre como empregar o MCTest em ambientes educacionais.

A partir de 2023, estudantes de Trabalho de Conclusão de Curso e Mestrado em Ciência da Computação na UFABC desempenham um papel ativo no processo de implementação de melhorias no sistema. Há expectativas de que essas melhorias sejam incorporadas à versão atual disponibilizada no GitHub. É importante ressaltar, contudo, que ainda existem áreas que demandam aprimoramento. Isso inclui a navegação entre as interfaces e a avaliação das competências e habilidades dos estudantes. Por exemplo, em um EP submetido por meio do VPL, quando se requer a utilização de estruturas de repetição com o comando *while*, a avaliação deve ser condizente com o cumprimento ou descumprimento dessa especificação.

Com o intuito de aprimorar a compreensão durante a leitura, é importante salientar que os seguintes termos serão amplamente utilizados ao longo deste trabalho: exercício de programação (EP), questão de múltipla escolha (QM), questão dissertativa ou de texto (QT) e quadro de respostas (QR). Nesse contexto, os estudantes efetuam as marcações das QMs no QR.

Leitores interessados em adquirir versões impressas deste livro podem encontrar mais informações na pasta `book` disponível no seguinte endereço: github.com/fzampirolli/mctest.

Este livro foi concebido e elaborado exclusivamente pelo autor. No entanto, lamentavelmente, não passou por um processo de revisão formal. Os textos apresentados foram predominantemente avaliados por meio de consultas a redes geratativas, pertencentes a uma categoria de modelos de aprendizado de máquina. A compreensão dos colegas interessados é confiada para o aprimoramento de edições futuras por meio de seus valiosos *feedbacks* construtivos.

Finalmente, é de suma importância enfatizar que as opiniões e declarações apresentadas nesta obra são devidamente embasadas por referências sempre que possível, ou por meio de experimentos realizados e devidamente citados no próprio livro. Nos casos em que a obtenção de referências se torna inviável, tais posicionamentos refletem a perspectiva pessoal do autor e não devem ser considerados uma representação oficial da instituição à qual o autor está afiliado.

Francisco de Assis Zampirolli

7 de setembro de 2023

Prefácio da segunda edição – em construção

Aprimoramentos e correções realizados no texto da primeira edição deste livro ([ZAMPIROLI, 2023](#)), utilizando o MCTest versão 5.2, serão descritos nesta segunda edição e implementados na versão 5.3 do MCTest. No arquivo [_setup-all.sh](#), estão detalhadas as instruções para a instalação dessas versões. Agradeço aos colegas que destacaram informações não ressaltadas no texto original, as quais foram incorporadas nesta nova versão, conforme formato descrito a seguir:

Destaque:

Este é um exemplo de destaque para chamar a atenção do leitor em algum ponto.

Na tela de atualização de questões, ver Seção [4.2 – Questão de múltipla escolha \(QM\)](#), foram adicionados cinco novos campos: contador de correções, contador de acertos e, para a Teoria de Resposta ao Item, os parâmetros *a*-discriminação, *b*-habilidade e *c*-chute. Esses parâmetros são fundamentais para a criação de exames adaptativos, ver Seção [8.6 – Exames adaptativos](#).

Nesta mesma tela de atualização de questões, o campo de “Retorno desta alternativa” (*feedback*) foi alterado para também aceitar a parte paramétrica, entre `[[code: e]]`, já incluída em “Descrição” e “Alternativas” da questão.

A Seção [5.3 – QM paramétrica com múltiplas variações: um estudo de caso](#) apresenta uma nova questão paramétrica altamente flexível, proporcionando uma extensa variedade de configurações e variações. Esta questão automatiza o processo de geração de itens, incorporando operadores relacionais e não relacionais, e oferece controle sobre parâmetros essenciais, como o número de itens, alternativas, operadores e itens corretos. Este tipo de questão, caracterizado por sua adaptabilidade através de parâmetros configuráveis, revela-se facilmente extensível para uma variedade de conjuntos, não se restringindo exclusivamente aos conjuntos de operadores relacionais e não relacionais. Para tal adaptação, é suficiente que o primeiro conjunto contenha elementos com premissas verdadeiras, enquanto o segundo apresente premissas falsas. Pode-se considerar, por exemplo, conjuntos como caracteres alfanuméricos permitidos em um nome de arquivo/variável versus caracteres proibidos, entre outros.

Na tela de atualizar exame, foi adicionado um novo campo para a seleção dos tópicos da disciplina que serão considerados na avaliação, ver Seção [6.4 – Recorte III da tela de exame – tópicos](#). Após escolher o tópico desejado e clicar no botão “Salvar”, apenas as questões relacionadas a esse tópico serão apresentadas. Este novo recurso é crucial para otimizar o tempo de carregamento da tela, especialmente em situações em que a disciplina possui inúmeras questões, resultando em uma melhoria significativa na eficiência do processo.

A opção pelo termo “tela” em detrimento de “formulário” é respaldada pela natureza das interfaces gráficas presentes no MCTest, as quais exibem informações provenientes de registros de banco de dados. Em geral, essas páginas proporcionam uma visão mais abrangente dos atributos desses registros, enriquecendo a compreensão do conteúdo. Apesar da constante necessidade de aprimoramento na Interface Humano-Computador, a escolha por “tela” é orientada pela dinâmica

e interatividade inerentes às informações apresentadas. Além disso, as figuras deste livro, que são representações visuais desses recortes de tela, buscam ilustrar a experiência do usuário, contribuindo para a compreensão e usabilidade do sistema.

A Seção 8.6 – [Exames adaptativos](#) foi criada para descrever o funcionamento de uma nova funcionalidade opcional que permite a geração de exames adaptativos com base no desempenho do estudante em avaliações anteriores. Para o correto funcionamento dessa funcionalidade, as questões de múltipla escolha devem ser classificadas segundo a taxonomia de Bloom, e múltiplas variações do exame devem ser geradas. Assim, estudantes com baixo desempenho em avaliações anteriores receberão exames personalizados, contendo questões dos primeiros níveis dessa taxonomia.

No QRcode apresentado no PDF de um exame, foi incluída também a variação do exame, criada após clicar no botão “Criar-Variações”. Esse recurso é importante para vincular o PDF digitalizado com as respostas dos estudantes à página do exame que será utilizada para fazer as correções, clicando no botão “Upload-PDF”. Além disso, com essa informação no QRCode, não é mais necessário criar arquivo criptografado e compactado com o gabarito do exame, anteriormente armazenado no servidor. Agora, o gabarito de cada exame é acessado diretamente no banco de dados, no registro contendo as variações do exame.

Foi inserida a Seção 8.7 – [Recomendações para realização tranquila de exames presenciais](#), para destacar algumas recomendações essenciais para garantir a realização de um exame impresso com questões de múltipla escolha. Por exemplo, o método de correção automática **NÃO** funcionará se os quatro discos pretos não estiverem intactos na página digitalizada. Além disso, ao utilizar questões disponíveis no banco de dados do MCTest, é necessário clicar no botão “Criar-Variações” antes de clicar no botão “Criar-PDF”. Adicionalmente, o número ID exibido ao lado do botão “Criar-Variações” **DEVE** ser o mesmo número exibido em vermelho na folha do exame impresso, abaixo do cabeçalho. Caso esses números difiram das páginas digitalizadas, a correção automática **NÃO** funcionará. O código foi adaptado para incluir a variação do exame também no QRCode. No entanto, para aceitar as variações anteriores, seria necessário armazenar todas as variações geradas no banco de dados, o que consumiria muito espaço em disco, sendo inviável no momento.

É altamente aconselhável seguir a prática de, antes de administrar um exame para uma ou diversas turmas com inúmeros estudantes, imprimir uma cópia, preenchê-la e submetê-la a uma correção automatizada. Caso essa abordagem demonstre eficácia durante essa simulação preliminar, é razoável esperar que funcione de maneira adequada durante a situação real de aplicação, correção e fornecimento de *feedback* do exame.

A Seção 9.1.4 – [QT para a integração MCTest+Moodle+CodeRunner](#) apresenta brevemente a possibilidade de reutilizar questões já criadas no formato Moodle+VPL. Ao exportar essas questões para o formato XML e importá-las para o Moodle com o plugin [CodeRunner](#) instalado, é possível criar avaliações com mais esse recurso no Moodle. Mais exemplos de uso serão incluídos neste livro assim que o plugin [CodeRunner](#) for incluído no Moodle da UFABC.

Foi incluído na Seção 13.3 – [Aplicação do MCTest-5 com Moodle+VPL no Ensino de PDI](#) um resumo que descreve a implementação do MCTest com Moodle+VPL no curso de Processamento Digital de Imagens (PDI). Enquanto os estudos anteriores abordaram cursos introdutórios de lógica

de programação, este relato destaca a aplicação dessa integração em um contexto mais específico. O trabalho “*A Practical Digital Image Processing Course with `morph.py`*”, de Zampirolli, Josko, Teubl e Kurashima (2024), foi premiado no Simpósio Brasileiro de Educação em Computação, consolidando uma jornada iniciada em 2012, quando o PDI foi utilizado na primeira versão do MCTest. Esse curso prático utiliza a biblioteca Python `morph.py` para auxiliar no ensino de PDI, abordando conceitos desde os fundamentos até tópicos avançados, com suporte de exemplos e exercícios práticos. Um estudo de caso com 15 participantes validou a eficácia do método, destacando desafios e soluções no ensino de PDI para iniciantes.

Uma versão estendida desse artigo foi publicada em Zampirolli, Josko, Teubl, Kurashima e Lopes (2025), detalhando a versão anterior e incluindo mais exemplos de uso da biblioteca Python `morph.py`.

Foi incluído na Seção 13.4 – MCTest-5 com Feedback Inteligente baseado em LLMs um resumo que descreve a evolução do ecossistema MCTest-5 através da integração de *Large Language Models* (LLMs) para o fornecimento de feedback inteligente. O estudo “*Intelligent Feedback for Individualized Introductory Programming Exercises*”, publicado por Zampirolli, Teubl, Pisani e Silva (2026), apresenta uma arquitetura inovadora que combina exercícios parametrizados com uma camada de processamento via IA (utilizando modelos disponíveis no <https://groq.com>). Ao contrário de sistemas de avaliação tradicionais que oferecem apenas vereditos do compilador, esta implementação utiliza engenharia de *prompts* para atuar como um tutor autônomo, fornecendo dicas qualitativas sem entregar a solução pronta. Experimentos realizados em cursos de introdução à programação demonstraram, por meio de métricas estatísticas como o *d* de Cohen, que a ferramenta reduziu significativamente a lacuna de feedback e foi percebida pelos estudantes como um suporte significativo para a correção autônoma de erros de lógica.

O Apêndice A – Acompanhamento de acesso de estudantes no Moodle descreve uma experiência realizada na disciplina de Processamento da Informação na UFABC no primeiro quadrimestre de 2024, em duas turmas, com o objetivo de substituir a lista de presença por um processo automático de verificação de acesso dos estudantes no Moodle durante as aulas. Foi desenvolvido o serviço LabMoodle, que acompanha as atividades dos estudantes cadastrados em uma disciplina no Moodle. Além disso, foram oferecidos 80 Exercícios de Programação (EPs) com correção automática pelo VPL. A disciplina abordou os conceitos fundamentais de Lógica de Programação e foi ministrada em um laboratório de informática. A estratégia de ensino incluiu aulas conceituais no Google Colab seguidas da resolução dos EPs pelos estudantes, com ênfase em vetores e matrizes. Os resultados foram avaliados por meio de exames elaborados no MCTest e compostos por questões com correção automática utilizando o plugin VPL do Moodle. O serviço LabMoodle permitiu o acompanhamento das atividades dos estudantes, destacando a participação e os acessos durante as aulas. Os dados coletados proporcionaram uma análise comparativa entre as duas turmas, evidenciando diferenças no desempenho e na participação dos estudantes antes e após a prova de recuperação.

O Apêndice B – Exames Adaptativos na disciplina CS1 continua o relato do apêndice anterior, porém focado nos testes adaptativos. Foram aplicados 6 testes, sendo apenas o primeiro não adaptativo. Esses testes foram utilizados como avaliação formativa com objetivo motivacional.

Foram implementados 3 formas diferentes de criar esses testes adaptativos: SAT (*Semi-Adaptive Testing*), WPC (*Weighted Probability of Correctness*) e MLE (*Maximum Likelihood Estimation*). A implementação desses testes foi realizada durante o andamento do curso, sendo o último MLE auxiliado pelo estudante Lucas Montagnani Calil Elias, como Trabalho de Conclusão de Curso do Bacharelado em Ciência da Computação na Universidade Federal do ABC. No final do curso, foi aplicado um questionário com 17 respondentes e realizada uma análise da significância estatística.

O Apêndice C – [Sobre o SEB](#) mostra como o SEB (*Safe Exam Browser*) pode ser utilizado para restringir o acesso a uma atividade VPL no Moodle. O arquivo de configuração do SEB foi aplicado à atividade VPL, em várias avaliações na UFABC, garantindo que os estudantes accessem apenas por meio do ambiente seguro do SEB. Essa solução robusta e segura, que também restringe o acesso pelos IP's do laboratório, ajuda a evitar o acesso não autorizado e garante a integridade do processo de avaliação.

Finalmente, o Apêndice D – [Sobre validação de código](#) detalha a implementação de um sistema de validação de código Python para o MCTest. A ferramenta Bandit é utilizada para identificar vulnerabilidades em código Python inserido em questões. Um servidor em Go foi criado para receber o código Python de uma questão paramétrica, executar o Bandit e retornar o resultado ao MCTest antes de salvar a questão. A estrutura e o funcionamento do sistema, incluindo a comunicação entre os componentes, são apresentados em detalhes. Este conteúdo é uma adaptação do trabalho de conclusão de curso desenvolvido no curso de Ciência da Computação da Universidade Federal do ABC, no período de 2023-2024, pelo estudante Gabriel Tavares Frota de Azevedo.

Francisco de Assis Zampirolli
21 de janeiro de 2026

Conteúdo

Conteúdo	xi
Lista de Figuras	xvii
Lista de Códigos	xxiii
Lista de Abreviaturas e Siglas	xxv
1 Introdução	1
1.1 Apresentação	1
1.1.1 Objetivo	2
1.1.2 Método	2
1.1.3 Resultados	3
1.2 Breve histórico do MCTest	3
1.2.1 MCTest no TSI	4
1.2.2 Versões do MCTest	4
1.2.3 MCTest nos processos seletivos e exames de disciplinas do TSI	6
1.2.4 MCTest na Escola Preparatória	6
1.3 Direitos autorais	7
1.3.1 Direitos autorais do software	7
1.3.2 Direitos autorais das questões disponíveis no BD	8
1.4 Instalando o MCTest	9
1.5 Organização do livro	10
I Fundamentos do MCTest	13
2 Visão geral do MCTest	15
2.1 Navegação geral e usuários	16
2.2 Criar entidades	19
2.2.1 Instituto	20
2.2.2 Curso	20
2.2.3 Disciplina	24
2.2.4 Turma	26

2.2.5	Tópico	28
2.3	Considerações finais	29
3	Recursos avançados	39
3.1	Administrador	40
3.1.1	Criar um curso relacionado a dois institutos	40
3.1.2	Criar uma disciplina relacionada a dois cursos	40
3.1.3	Acesso ao banco de dados	43
3.2	Coordenador	43
3.2.1	Criar um tópico relacionado a duas ou mais disciplinas	43
3.2.2	Criar várias turmas com arquivo CSV	47
3.3	Professor	49
3.3.1	Criar turma com arquivo CSV	50
3.3.2	Criar turma com arquivo CSV – restrições	51
3.4	Considerações finais	52
II	Questões no MCTest	55
4	Questões estáticas	57
4.1	Tutoriais gerais sobre a navegação de questões	57
4.2	Questão de múltipla escolha (QM)	62
4.3	Questão dissertativa (QT)	66
4.4	Considerações finais	68
5	Questões paramétricas	71
5.1	QM paramétrica	72
5.1.1	Um exemplo	73
5.1.2	QM paramétrica no MCTest	74
5.2	QT paramétrica	78
5.2.1	Exemplo 1 – Teste de mesa	78
5.2.2	Exemplo 2 – Teste de mesa e gabarito	79
5.2.3	Exemplo 3 – Teste de mesa e gabarito com <code>settrace</code>	80
5.3	QM paramétrica com múltiplas variações: um estudo de caso	85
5.3.1	Explicação do método <code>gerar_QM_itens</code>	86
5.3.2	Explicação do método <code>gerar_QM_itens</code> : Verifica parâmetros	87
5.3.3	Explicação do método <code>gerar_QM_itens</code> : Outros métodos	90
5.3.4	Explicação do método <code>gerar_QM_itens</code> : Laço Principal	93
5.3.5	Aplicação do método <code>gerar_QM_itens</code>	93
5.4	QT paramétrica, com código	98
5.4.1	Introdução ao Moodle e ao VPL	98
5.4.2	Questão com exercício de programação no Moodle+VPL	99

5.4.3 Questão com exercício de programação no MCTest+Moodle+VPL	101
5.5 Considerações finais	109
III Exames no MCTest	111
6 Visão geral dos exames	113
6.1 Acesso à tela de exames	114
6.2 Recorte I da tela de exame	116
6.2.1 Área à esquerda	116
6.2.2 Área central	119
6.2.3 Área à direita	120
6.3 Recorte II da tela de exame – turmas	121
6.4 Recorte III da tela de exame – tópicos	121
6.5 Recorte IV da tela de exame – questões	122
6.6 Recorte V da tela de exame – configurando os detalhes	122
6.7 Considerações finais	125
7 Exames com Quadro de Respostas (QR)	127
7.1 Utilizando exames exclusivamente com QR	128
7.1.1 Estilo vertical	128
7.1.2 Estilo horizontal	130
7.2 Restrições nos QRs	130
7.3 Corrigindo exames com QR	132
7.3.1 Arquivo CSV com as correções	135
7.3.2 Teoria de Resposta ao Item	136
7.3.3 <i>Feedback</i> ao estudante	138
7.4 Considerações finais	138
8 Exames com QR+QM e/ou QT	141
8.1 Desenvolvendo QMs e uma QT para inclusão em um exame	142
8.1.1 QT com valor exato	144
8.1.2 Elaborar exame com questões formuladas	144
8.2 Criando as variações	144
8.3 Detalhando os gabaritos no arquivo CSV	146
8.4 Criando o PDF do exame com QR+QM+QT	148
8.5 Corrigindo exames com QR+QM	151
8.6 Exames adaptativos	154
8.6.1 Exemplificando o uso do método SAT	156
8.6.2 Estimativa dos parâmetros do modelo TRI	159
8.7 Recomendações para realização tranquila de exames presenciais	160
8.8 Considerações finais	162

9 Variações de exames com QM+QT para o Moodle	163
9.1 Criando exames e exportando arquivos Aiken e XML	164
9.1.1 QM – equação de primeiro grau – idade	164
9.1.2 QM – equação da reta	165
9.1.3 QT com resposta exata – ordena parte do vetor	166
9.1.4 QT para a integração MCTest+Moodle+CodeRunner	169
9.1.5 Criando exame e exportando as variações	170
9.2 Importando arquivo XML no Moodle	179
9.3 Considerações finais	180
10 Exames com MCTest+Moodle+VPL	183
10.1 Criando um exame com QT integrado ao VPL	184
10.1.1 Criando uma nova questão com matriz integrada ao VPL	184
10.1.2 Criando as variações e imprimindo o PDF do exame	188
10.2 Criando e configurando a atividade VPL para o exame	189
10.3 Soluções da atividade VPL para o exame	191
10.4 Detalhando o arquivo <code>linker.json</code>	193
10.5 Considerações finais	196
IV Experimentos	199
11 MCTest versões 1, 2 e 3 - Quadro de Respostas (QR)	201
11.1 MCTest-1: versão no Matlab	202
11.2 MCTest-2: versão no Android	204
11.3 Considerações finais	206
12 MCTest versões 4, 4.G e 5	209
12.1 MCTest-4: gerando e corrigindo questões criadas no LATEX	210
12.1.1 Artigo descrevendo as versões do MCTest 4 e 4.G	210
12.1.2 Artigo comparando as modalidades semipresencial e presencial CS1 com base na utilidade do MCTest-4	212
12.1.3 Artigo definindo a parametrização do MCTest após a versão 4	214
12.2 MCTest-5: a versão web	218
12.2.1 Artigo apresentando a primeira versão web do MCTest	218
12.2.2 Artigo descrevendo o MCTest em conjunto com <i>Google Forms</i> e <i>Google Sheets</i>	227
12.2.3 Artigo descrevendo a adoção de QMs com pesos diferentes nas alternativas	231
12.3 Considerações finais	234
13 MCTest+Moodle+VPL	237
13.1 MCTest-5, com Moodle	238
13.2 MCTest-5, com Moodle+VPL	238

13.2.1	Artigo com a primeira integração entre MCTest+Moodle+VPL	238
13.2.2	Artigo aplicando a integração MCTest+Moodle+VPL em CS0	240
13.2.3	Artigo sobre o uso de múltiplas linguagens de programação em CS1	242
13.2.4	Artigo aplicando a integração MCTest+Moodle+VPL em CS1	243
13.2.5	Artigo explorando a integração de jogos no ensino de CS0	244
13.3	Aplicação do MCTest-5 com Moodle+VPL no Ensino de PDI	245
13.3.1	Método	246
13.3.2	Experimentos	247
13.4	MCTest-5 com Feedback Inteligente baseado em LLMs	248
13.4.1	Método	248
13.4.2	Experimentos e Resultados	249
13.5	Considerações finais	249
14	Conclusões	251
14.1	Novas possibilidades de aprimoramento	252
14.2	Novas publicações	253
14.3	Novas edições deste livro	254
A	Acompanhamento de acesso de estudantes no Moodle	255
A.1	Contexto da oferta	256
A.2	Intervenção pedagógica	256
A.3	O Serviço LabMoodle	257
A.4	Resultados e discussões	259
A.4.1	Arquivo <code>data.json</code>	261
A.4.2	Arquivo <code>dias_notas_*.csv</code>	262
A.4.3	Arquivo <code>faltas_notas_*.csv</code>	262
A.4.4	Arquivo <code>presenca_notas_*.csv</code>	263
A.4.5	Acessos no Moodle	263
A.5	Acessos às atividades do Moodle	263
A.6	Tabelas comparativas - turma A vs. turma B	264
A.6.1	Comparações antes da prova de recuperação	264
A.6.2	Comparações após a prova de recuperação - resultado final	264
A.7	Histórico de CS0 e CS1 na UFABC	265
A.8	Considerações finais	267
B	Exames Adaptativos na disciplina CS1	275
B.1	Contextualização dos exames adaptativos	276
B.2	Teste 1: Sequencial – Aleatório	276
B.2.1	Criação dos testes adaptativos	277
B.2.2	Correção dos testes adaptativos	277
B.2.3	Descrição das questões paramétricas utilizadas	278

B.3	Teste 2: Método – SAT	279
B.3.1	Criação dos testes adaptativos	280
B.3.2	Correção dos testes adaptativos	281
B.4	Teste 3: Condicional – WPC	282
B.4.1	Criação dos testes adaptativos	282
B.4.2	Correção dos testes adaptativos	282
B.5	Teste 4: Repetição – WPC	283
B.6	Teste 5: Vetor – MLE-v0	283
B.7	Teste 6: Matriz – MLE-v1	283
B.7.1	Descrição das questões utilizadas	284
B.7.2	Calibração das questões	286
B.8	Questionário avaliativo	293
B.8.1	Questões aplicadas	293
B.8.2	Análise estatística dos resultados	297
B.9	Considerações finais	298
C	Sobre o SEB	301
C.1	Configurando uma atividade no SEB	301
C.2	Considerações finais	302
D	Sobre validação de código	305
D.1	Instalações e bibliotecas usadas	306
D.1.1	Instalação do Bandit	306
D.1.2	Instalação do Go	306
D.2	Criar um projeto simples para troca de mensagens	306
D.3	Exemplo simples de uso do Bandit	310
D.3.1	Criação do arquivo <code>example.py</code>	310
D.3.2	Instalação do Bandit	310
D.3.3	Análise do arquivo com o Bandit	310
D.3.4	Interpretação do relatório	310
D.4	Projeto servidor <code>mctest-validator</code>	312
D.4.1	Ajuste no MCTest	312
D.4.2	Comunicação entre MCTest e <code>mctest-validator</code>	314
D.5	Considerações finais	316
Bibliografia		319

Lista de Figuras

1.1	Organização do livro.	10
2.1	Tela principal do MCTest.	17
2.2	Recorte do menu de navegação principal do MCTest.	18
2.3	Tela para a inscrição do usuário.	19
2.4	Tela para administrador atualizar, apagar ou criar um instituto.	21
2.5	Tela para o administrador criar um instituto.	21
2.6	Tela apresentando os detalhes de um instituto.	22
2.7	Tela para confirmar a exclusão do instituto.	22
2.8	Tela que apresenta a lista de cursos, na qual o administrador pode atualizar, apagar ou criar um novo curso.	23
2.9	Tela para administrador criar um curso novo.	24
2.10	Tela apresentando os detalhes de um curso.	25
2.11	Tela que apresenta a lista de disciplinas, na qual o administrador pode atualizar, apagar ou criar uma nova disciplina.	26
2.12	Tela para o administrador criar uma disciplina. Tela semelhante à primeira parte da tela de atualizar uma disciplina.	27
2.13	Segunda parte da tela para o administrador atualizar uma disciplina.	28
2.14	(Parte 1) Tela apresentando os detalhes de uma disciplina.	29
2.15	(Parte 2) Continuação da tela apresentando os detalhes de uma disciplina.	30
2.16	Tela que apresenta a lista de turmas, na qual um professor de uma disciplina pode criar uma nova turma, além de atualizar ou apagar turmas existentes.	31
2.17	Tela utilizada pelo professor para criar uma nova turma após clicar no botão “Criar uma nova Turma”, na tela da Figura 2.16. É importante destacar que o nome da turma não pode conter acentos ou caracteres especiais, pois esse nome fará parte do cabeçalho do exame.	32
2.18	Tela para um professor de uma disciplina poder atualizar uma turma, inserindo estudantes com a importação de um arquivo no formato CSV.	33
2.19	Tela para um professor de uma disciplina poder atualizar uma turma, atualizando ou apagando estudantes da turma. Além disso, é possível incluir um novo estudante na turma.	34

2.20 Tela para um professor de uma disciplina poder atualizar os dados de um estudante da turma.	34
2.21 Tela com a lista de tópicos para o coordenador (ou administrador) criar, atualizar ou apagar um tópico de disciplina(s).	35
2.22 Tela para o coordenador (ou administrador) atualizar um tópico de uma disciplina. . .	36
2.23 Tela apresentando os detalhes de um tópico.	37
2.24 Tela apresentando as três primeiras questões de um tópico.	38
3.1 Tela com a lista de institutos vista pelo administrador, que pode atualizar, apagar ou criar um instituto para simular um curso pertencer aos institutos CECS e CMCC. . .	41
3.2 Tela com a lista de curso, vista pelo administrador.	42
3.3 Tela para o administrador criar um curso novo relacionado a dois institutos. Manter a tecla “Ctrl” pressionada para escolher mais de uma opção.	43
3.4 Tela com a lista de curso, vista pelo administrador, após ter criado o curso Engenharia da Computação na Figura 3.3.	44
3.5 Tela com a lista de disciplinas, vista pelo administrador, com destaque para a disciplina PDI, que está relacionada aos cursos de BCC e POSCOMP.	45
3.6 Tela com a lista de entidades que o administrador pode fazer manutenção diretamente no BD.	46
3.7 Tela com a lista de tópicos, vista pelo coordenador. É possível observar o compartilhamento de tópicos entre disciplinas.	48
3.8 Tela com detalhes da disciplina de Processamento da Informação, vista pelo coordenador, com os tópicos compartilhados também apresentados na Figura 3.7.	49
3.9 (Parte 1) Tela de criação da disciplina de Processamento da Informação pelo administrador.	50
3.10 (Parte 2) Tela de criação da disciplina de Processamento da Informação pelo administrador. É possível incluir professores, turmas e estudantes pela importação de dois arquivos no formato CSV.	51
3.11 Tela mostrando o resultado do cadastro de estudantes utilizando a importação de um arquivo no formato CSV.	52
4.1 Tela para o professor visualizar todas as questões de disciplinas que está cadastrado. .	58
4.2 Exibição de erro ao tentar modificar questão criada por outro autor.	59
4.3 Exibição da questão QE1 criada por outro autor.	59
4.4 Tela para o professor visualizar suas questões e criar novas, incluindo importação de formato TXT.	61
4.5 Tela para um professor criar uma nova questão.	62
4.6 Tela com valores preenchidos para criar uma nova questão, antes de clicar no botão “Salvar”.	63
4.7 Curva Característica do Item (CCI) de 3-Modelo Logístico de Parâmetros, ou 3PLM. Adaptado de Zampirolli, Batista, Josko, Steil e Trevisan (2021).	65

LISTA DE FIGURAS

4.8 (Parte 1) Tela de atualização de questão sem alternativas e <i>feedback</i> , previamente criada pelo professor.	66
4.9 (Parte 2) Continuação da tela de atualização de questão, com estatísticas e ajuda para incluir figura.	67
4.10 Recorte do PDF gerado após clicar no botão “Criar-PDF”, na Figura 4.8.	67
4.11 Recorte do PDF gerado após clicar no botão “Criar-PDF”, na Figura 4.8, para a questão atualizada com cinco alternativas.	68
4.12 Recorte do PDF gerado após clicar no botão “Criar-PDF”, na Figura 4.8, para uma questão dissertativa (“Type: QT”).	68
4.13 Recorte da “Descrição” de uma QT de teste de mesa.	69
4.14 Recorte do PDF gerado da QT de teste de mesa.	70
 5.1 Recorte do PDF gerado para a questão definida no Código 5.3.	77
5.2 Recorte do PDF gerado para a questão definida no Código 5.4.	78
5.3 Recorte do PDF gerado para a questão definida nos Códigos 5.5 e 5.6.	83
5.4 Recorte do PDF gerado para a questão definida nos Códigos 5.8 e 5.9.	88
5.5 Recorte do PDF gerado para a QM com operadores relacionais.	88
5.6 Recorte do PDF gerado para a QM com os parâmetros inválidos da primeira chamada do Código 5.14.	92
5.7 Frenquênciados acertos dos 273 candidatos no processo seletivo do TSI.	97
5.8 Recorte da tela do Moodle de uma atividade VPL.	102
5.9 Recorte do PDF gerado para a QT definida nos Códigos 5.21 e 5.22.	106
 6.1 Tela de exames do professor.	114
6.2 Tela para criar um novo exame.	115
6.3 Recorte do PDF gerado com as configurações padrão, contendo apenas o cabeçalho e o QR.	116
6.4 Recorte I da tela de exame – ver explicações no texto sobre os botões.	117
6.5 Recorte II da tela de exame, com as turmas do professor.	121
6.6 Recorte III da tela de exame, com os tópicos da disciplina.	122
6.7 Recorte IV da tela de exame, com as questões da disciplina, do(s) tópico(s) selecionado(s).	123
6.8 Recorte V da tela de exame, configurando os detalhes do exame.	124
 7.1 Recorte de um exame digitalizado em formato PDF no estilo vertical.	129
7.2 Recorte de um exame em formato PDF no estilo vertical, com as instruções.	129
7.3 Recorte de um exame digitalizado em formato PDF no estilo horizontal.	130
7.4 Exemplos de PDFs de exames com 245 questões no estilo horizontal e 200 questões no estilo vertical.	131
7.5 Exemplo de PDF de um exame com 18 questões, cinco questões por bloco e dois blocos por linha.	132

7.6	Exemplos de exames digitalizados incluem: (a) o gabarito obtido a partir do PDF apresentado na Figura 7.5; (b) e (c) representam dois exemplos adicionais de respostas de estudantes.	134
7.7	À esquerda, são apresentados quatro recortes automáticos das questões erroneamente marcadas no terceiro bloco, na Figura 7.6-(b). No centro, encontra-se outro recorte automático referente ao quarto bloco, em que a alternativa D foi considerada correta. À direita, também no quarto bloco, a imagem é apresentada de forma ampliada para destacar múltiplas marcações em vermelho.	136
7.8	PDF com o <i>feedback</i> das correções enviado para o e-mail do estudante.	139
8.1	Recorte da tela com as questões escolhidas do exame.	145
8.2	Recorte da tela de configuração do exame.	145
8.3	Recorte da tela de configuração do exame, detalhando o botão “Criar-Variações”.	146
8.4	Mensagem após clicar no botão “Criar-Variações”.	146
8.5	E-mail enviado ao professor após clicar no botão “Criar-Variações”.	147
8.6	Recorte do PDF do exame gerado após a ação do botão “Criar-PDF”.	149
8.7	E-mail enviado ao professor após clicar no botão “Criar-PDF”.	150
8.8	Recorte do e-mail com parte do PDF do exame enviado como <i>feedback</i> ao estudante, após a ação do botão “Upload-PDF”. Neste PDF, o enunciado da QT também é apresentado após as QMs.	152
8.9	Tipos de exames adaptativos.	154
8.10	Seleção das questões na tela do exame para sorteio.	156
9.1	Recorte do PDF gerado para a questão das idades dos irmãos, referente ao Código 9.1.	164
9.2	Recorte do PDF gerado para a questão da equação da reta definida no Código 9.2.	166
9.3	Recorte do PDF gerado para a questão com resposta exata definida pelos Códigos 9.3 e 9.4.	169
9.4	Recorte da tela do exame com as quatro questões marcadas.	170
9.5	Recorte da tela do exame com os detalhes da configuração.	170
9.6	Recorte da tela do exame com os detalhes da configuração.	171
9.7	Recorte do e-mail recebido após clicar em “Criar-Variações”.	171
9.8	Exemplo de QM no Moodle.	179
9.9	Exemplo da questão exata do Moodle.	180
9.10	Exemplo exame gerado pelo MCTest.	180
10.1	Recorte do PDF gerado para a questão de matriz definida nos Códigos 10.1 e 10.2.	187
10.2	Recorte da tela de exame, configurando os detalhes do exame com integração VPL.	188
10.3	Recorte da tela do exame para criar as variações e enviar o arquivo no formato JSON.	188
10.4	Recorte do e-mail recebido após clicar em “Criar-Variações”.	189
10.5	Recorte da tela do PDF do exame com integração VPL.	190
11.1	Recorte do PDF com o QR utilizado no MCTest-1.	202

LISTA DE FIGURAS

11.2 Demonstra�o da captura a partir de um gabarito impresso.	205
11.3 As quest�es marcadas na parte inferior indicam um valor ponderado atribu�do a quest�o, as maras � direita representam diferentes gabaritos e a identifica�o do gabarito � a marca � direita e abaixo. Fonte: China, Zampirolli, Neves e Quilici-Gonzalez (2016).	205
11.4 Estat�sticas sobre o n�mero de respostas corretas em uma turma. Fonte: China, Zampirolli, Neves e Quilici-Gonzalez (2016).	206
12.1 Estrutura de pastas do MCTest-4. Fonte: Zampirolli, Batista e Quilici-Gonzalez (2016).	211
12.2 Recorte de um exame gerado com MCTest-4.G.	214
12.3 Recorte do PDF gerado para a quest�o de MRU do C�digo 12.1.	216
12.4 Recorte do PDF gerado para a quest�o para criar uma matriz do C�digo 12.2.	217
12.5 Recorte do PDF gerado para a quest�o param�trica com equa�o e figura referente ao C�digo 12.3.	221
12.6 Recorte do PDF gerado para a quest�o param�trica com grafo e figura referente ao C�digo 12.4.	223
12.7 Recorte do PDF gerado para a quest�o param�trica com matriz criada pelos C�digos 12.6 e 12.7.	227
12.8 Planilha mostrando o question�rio preenchido de um estudante. Adaptado de Zampirolli, Batista, Iriarte e Junior (2020).	228
12.9 Exemplo de exame utilizado em C�lculo 1. Adaptado de Zampirolli, Batista, Iriarte e Junior (2020).	230
12.10 Visualiza�o do arquivo CSV com as corre�es. Adaptado de Zampirolli, Batista, Rodriguez, Rocha e Goya (2021).	231
12.11 C�lculo final do exame. Adaptado de Zampirolli, Batista, Rodriguez, Rocha e Goya (2021).	232
13.1 (a) estrutura de arquivos e pastas; (b) recorte da estrutura de arquivos e pastas gerados automaticamente, detalhando a pasta 1.py para Python. Fonte: Zampirolli, Teubl, Kobayashi, Neves, Rozante e Batista (2021).	243
13.2 Vis�o geral da interven�o pedag�gica empregada no curso de PDI. Fonte: (ZAMPIROLLI, 2024).	247
A.1 Tela do servi�o - Parte 1 - informa�es gerais.	258
A.2 Tela do servi�o - Parte 2: Configura�es dos relat�rios e dos dados da disciplina.	258
A.3 Tela do servi�o - Parte 3: envio dos arquivos e gera�o do relat�rio.	259
A.4 Tela do servi�o - Parte inicial do relat�rio gerado.	260
A.5 Gr�fico contendo os acessos totais dos estudantes da turma A, com destaque para o baixo acesso fora da aula.	269

A.6	Gráfico contendo os acessos totais dos estudantes da turma B, com destaque para o baixo acesso fora da aula.	270
A.7	Quantidade de estudantes que tentaram resolver os EP (verde para EPs sobre vetores e salmão para matrizes).	271
B.1	Questão da taxonomia Lembrar, com chave 2637.	278
B.2	Questão da taxonomia Entender (Grupo 2), com chave 89.	278
B.3	Questão da taxonomia Entender (Grupo 2), com chave 103.	278
B.4	Questão da taxonomia Entender (Grupo 3), com chave 2639.	279
B.5	Questão da taxonomia Entender (Grupo 3), com chave 2641.	279
B.6	Questão da taxonomia Aplicar, com chave 87.	279
B.7	Questão da taxonomia Aplicar, com chave 2783.	279
B.8	Questão da taxonomia Lembrar, com chave 264.	285
B.9	Questão da taxonomia Lembrar, com chave 3078.	285
B.10	Questão da taxonomia Lembrar, com chave 3079.	285
B.11	Questão da taxonomia Entender, com chave 3081.	286
B.12	Questão da taxonomia Entender, com chave 3083.	286
B.13	Questão da taxonomia Entender, com chave 3084.	286
B.14	Questão da taxonomia Entender, com chave 3085.	286
B.15	Questão da taxonomia Aplicar, com chave 3086.	286
B.16	Questão da taxonomia Aplicar, com chave 3087.	286
B.17	Curvas Características dos Itens nos modelos 1PL (esquerda), 2PL (centro) e 3PL (direita).	292
B.18	Curvas de Informação ao Item nos modelos 1PL (esquerda), 2PL (centro) e 3PL (direita).	292
B.19	Curvas de Informação do Teste nos modelos 1PL (esquerda), 2PL (centro) e 3PL (direita).	292
B.20	Curvas de Informação vs Erro Padrão do Teste nos modelos 1PL (esquerda), 2PL (centro) e 3PL (direita).	293
B.21	Questionário com questões gerais.	294
B.22	Questionário com questões sobre material de ensino.	295
B.23	Questionário com questões sobre avaliações.	296
B.24	Questionário com questões sobre teste adaptativo.	296
B.25	Destaque para a Q17. Acho importante os Testes Semanal Individual.	297
C.1	Captura de tela do SEB mostrando a aba “Rede” com restrições de acesso	302
C.2	Captura de tela de uma atividade VPL no Moodle mostrando a chave do SEB	303
D.1	Diretório e arquivos de <code>simple-rest-api</code> .	307
D.2	Diagrama de sequência ilustrando a interação cliente-servidor para criar mensagens.	309
D.3	Diretórios e arquivos de <code>mctest-validator</code> .	315
D.4	Mensagens entre os diversos arquivos de <code>mctest-validator</code> .	317

Lista de Códigos

5.1	Exemplo de método para calcular a área de um círculo.	73
5.2	Exemplo de QM paramétrica para calcular a área do círculo.	76
5.3	Exemplo de QM paramétrica para calcular a área do círculo, utilizando o método <code>createWrongAnswers</code>	77
5.4	Exemplo simples de descrição de QT paramétrica para teste de mesa.	79
5.5	Exemplo prático de teste de mesa paramétrico mostrando o gabarito – Parte 1: Descrição de questão.	81
5.6	Exemplo prático de teste de mesa paramétrico mostrando o gabarito – Parte 2: Bloco de código em Python.	82
5.7	Exemplo de uso de <code>sys.settrace</code>	84
5.8	Exemplo prático de teste de mesa paramétrico utilizando <code>sys.settrace</code> – Parte 1: Descrição de questão.	86
5.9	Exemplo prático de teste de mesa paramétrico utilizando <code>sys.settrace</code> – Parte 2: Bloco de código em Python.	87
5.10	Exemplo de QM paramétrica de operadores relacionais – Parte 1: Descrição de questão.	88
5.11	Exemplo de QM paramétrica de operadores relacionais – Parte 2: Bloco de código em Python com as declarações e a chamada do método <code>gerar_QM_itens</code>	89
5.12	Método <code>gerar_QM_itens</code> – Parte 1: Bloco principal.	89
5.13	Método <code>gerar_QM_itens</code> – Parte 2: Método <code>verifica_parametros()</code>	91
5.14	Exemplo de chamadas de <code>gerar_QM_itens</code> com <code>verifica_parametros</code> retornando <code>pode_gerar = False</code>	92
5.15	Método <code>gerar_QM_itens</code> – Parte 3: Outros métodos.	94
5.16	Método <code>gerar_QM_itens</code> – Parte 4: Laço principal.	95
5.17	Trecho de QM paramétrica utilizada no processo seletivo do TSI.	96
5.18	Código para converter arquivo XML do formato L ^A T _E X para HTML.	98
5.19	Programa em Python para cálculo do fatorial.	101
5.20	Programa em Python para cálculo de parcelas.	103
5.21	Exemplo de QT paramétrica utilizando MCTest+Moodle+VPL – Parte 1: Descrição de questão.	105
5.22	Exemplo de QT paramétrica utilizando MCTest+Moodle+VPL – Parte 2: Bloco de código em Python.	107
8.1	Exemplo de QT com resposta exata.	144

9.1	Exemplo de QM paramétrica para calcular as idades dos irmãos.	165
9.2	Exemplo de QM paramétrica para calcular a equação da reta.	166
9.3	Exemplo de questão paramétrica exata – Parte 1: Descrição de questão.	167
9.4	Exemplo de questão paramétrica exata – Parte 2: Bloco de código.	168
10.1	Exemplo de questão com matriz utilizando MCTest+Moodle+VPL – Parte 1: Descrição de questão.	185
10.2	Exemplo de questão com matriz utilizando MCTest+Moodle+VPL – Parte 2: Bloco de código em Python.	186
10.3	Método mais genérico que <code>matriz2texto</code> para formatar uma matriz.	187
10.4	Exemplo de solução para a questão de matriz do exame no VPL.	192
10.5	Exemplo de solução compacta e genérica para a questão de matriz do exame no VPL.	193
12.1	Exemplo de QM paramétrica para calcular MRU.	215
12.2	Exemplo de QM paramétrica para criar uma matriz.	217
12.3	Questão paramétrica com equação e figura.	220
12.4	QM paramétrica com grafo e figura.	222
12.5	Método para desenhar matriz.	224
12.6	Questão paramétrica com matriz – Parte 1: Descrição de questão.	225
12.7	Questão paramétrica com matriz – Parte 2: Bloco de código em Python.	226
B.1	Exemplo de código R em uma célula do Colab para gerar a Tabela B.9.	290
B.2	Exemplo de código R em uma célula do Colab para gerar o gráfico CCI no modelo 3PL, ver Figura B.17.	291
D.1	Exemplo de código Go para <code>main.go</code>	308
D.2	Exemplo de código Python com potencial vulnerabilidade.	310
D.3	Trechos do conteúdo JSON destacando potencial vulnerabilidade no Código D.2.	311
D.4	Trecho de instruções para a instalação do serviço <code>mctest-validator-api</code>	312
D.5	Método que filtra código em Python da questão e valida em <code>mctest-validator</code>	313
D.6	Trecho de código para tratar o conteúdo JSON retornado pelo <code>requests</code> do método <code>generateCode</code>	314
D.7	Trecho de código incluído na <i>view</i> de atualização de questão para chamar o método <code>generateCode</code>	314

Lista de Abreviaturas e Siglas

ACM – *Association for Computing Machinery*

AVA – Ambiente Virtual de Aprendizagem

BCT – Bacharelado em Ciência e Tecnologia

BCC – Bacharelado em Ciência da Computação

CMCC – Centro de Matemática, Computação e Cognição

CS0-1-2-3 *Computer Science 0, 1, 2 and 3* / cursos introdutórios de computação definidos pela ACM

EaD – Ensino a Distância

EP – Exercício de Programação

EPUFABC – Escola Preparatória da UFABC

EdUFABC – Editora da UFABC

FAPESP – Fundação de Amparo à Pesquisa do Estado de São Paulo

FEEC – Faculdade de Engenharia Elétrica e de Computação

FTP – *File Transfer Protocol* / Protocolo de Transferência de Arquivos

IDE – *Integrated Development Environment* / Ambiente de Desenvolvimento Integrado

IES – Instituição de Ensino Superior

IME – Instituto de Matemática e Estatística

INEP – Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira

INPI – Instituto Nacional de Propriedade Intelectual

MCTest – *Multiple Choice Test* / Teste de Múltipla Escolha (nome inicial, mas inclui também QT)

MEC – Ministério da Educação

Moodle – *Modular Object-Oriented Dynamic Learning Environment* / Ambiente de Aprendizagem Dinâmico Modular Orientado a Objeto (um exemplo de AVA)

NETEL – Núcleo Educacional de Tecnologias e Línguas

PROEC – Pró-Reitoria de Extensão e Cultura

PROGRAD – Pró-Reitoria de Graduação

PROPG – Pró-Reitoria de Pós-Graduação

QM – Questão de Múltipla Escolha

QT – Questão de Texto ou Dissertativa

QR – Quadro de Respostas (para as questões de múltipla escolha)

TCI – Teoria Clássica ao Item

TIC – Tecnologia da Informação e Comunicação

TSI – Especialização em Tecnologias e Sistemas de Informação

TRI – Teoria de Resposta ao Item

UFABC – Universidade Federal do ABC

USP – Universidade de São Paulo

VPL – *Virtual Programming Lab*

Capítulo 1

Introdução

Conteúdo

1.1	Apresentação	1
1.1.1	Objetivo	2
1.1.2	Método	2
1.1.3	Resultados	3
1.2	Breve histórico do MCTest	3
1.2.1	MCTest no TSI	4
1.2.2	Versões do MCTest	4
1.2.3	MCTest nos processos seletivos e exames de disciplinas do TSI	6
1.2.4	MCTest na Escola Preparatória	6
1.3	Direitos autorais	7
1.3.1	Direitos autorais do software	7
1.3.2	Direitos autorais das questões disponíveis no BD	8
1.4	Instalando o MCTest	9
1.5	Organização do livro	10

1.1 Apresentação

A avaliação dos estudantes é uma atividade crucial para o trabalho de um professor, ao permitir avaliar de maneira precisa o desempenho dos estudantes ao longo do curso, fornecendo *feedback* importante para o seu desenvolvimento acadêmico. Além disso, a avaliação também é um meio importante para aprimorar a prática educativa, permitindo que os professores identifiquem pontos fortes e fracos de sua metodologia e aprimorem sua abordagem pedagógica para proporcionar um melhor aprendizado aos estudantes.

Avaliar inúmeros estudantes individualizadamente é uma tarefa desafiadora para professores em todos os níveis de ensino, especialmente em disciplinas que exigem habilidades e competências específicas. Esse desafio é ainda maior em exames que envolvem exercícios de programação (EP), uma vez que exigem uma avaliação minuciosa dos processos de resolução de problemas, além da compreensão do código e da lógica utilizados pelos estudantes. Nesse sentido, é fundamental que os professores tenham acesso a ferramentas e técnicas que possam auxiliá-los nessa tarefa, como sistemas de avaliação automatizados e a utilização de exercícios parametrizados. A adoção dessas ferramentas permite uma avaliação mais eficiente das habilidades dos estudantes, garantindo que eles tenham a oportunidade de demonstrar seu conhecimento e habilidades adequadamente.

1.1.1 Objetivo

Este livro visa disponibilizar uma solução abrangente e eficaz para a avaliação de estudantes. A abordagem colaborativa adotada permite que os professores reutilizem bancos de questões já utilizados por outros colegas que lecionam na mesma disciplina, economizando tempo e esforço na criação de avaliações personalizadas.

Essa metodologia pode ser aplicada tanto em exames unificados em várias turmas com milhares de estudantes, quanto em disciplinas com turmas menores ministradas por um único professor.

1.1.2 Método

Para ajudar professores a avaliar os estudantes, este livro apresenta o MCTest, um sistema de código aberto que permite a elaboração e correção de exames com questões de múltipla escolha (QMs) ou dissertativas (QTs), incluindo EP. Esse sistema oferece questões parametrizadas e exames individualizados, que podem ser utilizados por várias turmas simultaneamente. O MCTest foi inicialmente desenvolvido para corrigir exames em processos seletivos para um curso de especialização na Universidade Federal do ABC (UFABC) em 2012, mas evoluiu significativamente para uma versão inteiramente web.

O MCTest pode ser integrado a Ambientes Virtuais de Aprendizagem (AVAs), como o Moodle, o qual é uma plataforma digital que oferece uma variedade de recursos e ferramentas para melhorar o processo de ensino e aprendizagem. Os AVAs podem ser usados como um complemento ou suporte às aulas presenciais, ou como uma opção para o ensino a distância (EaD). Eles oferecem um espaço virtual no qual estudantes e professores podem interagir, acessar materiais didáticos, realizar atividades, participar de fóruns de discussão e avaliações. Esses ambientes proporcionam aos estudantes maior flexibilidade para acessar o conteúdo e realizar suas tarefas em horários mais convenientes, o que também permite uma maior personalização do processo de aprendizagem.

Assim, este livro ensina como criar exames individualizados, com questões parametrizadas, intercalando textos em L^AT_EX e códigos em Python, e correção automática. Essas questões podem incluir EPs utilizando o *plugin* VPL (*Virtual Programming Lab*) no Moodle. No entanto, antes de elaborar esses EPs, é importante discutir como navegar pelo sistema utilizando um dos seus três tipos de usuários: administrador, coordenador de disciplina e professor. Mesmo um professor sem

1.2. BREVE HISTÓRICO DO MCTEST

habilidade de programação pode utilizar o MCTest para criar exames com questões estáticas, em que a única variação é o sorteio das questões e das alternativas. Cada tipo de usuário possui um conjunto de funcionalidades para criar e alterar as diversas entidades do sistema, incluindo Instituto, Curso, Disciplina, Tópico, Questão, Turma, Professor, Estudante e Exame.

O MCTest já foi implantado com sucesso na UFABC e serve como plataforma para os professores e gestores da instituição. O leitor pode examinar a implantação deste sistema em mctest.ufabc.edu.br para entender melhor como o sistema funciona e como pode ser utilizado para aprimorar a avaliação de habilidades dos estudantes.

1.1.3 Resultados

O livro apresenta os resultados de dezenas de experimentos realizados ao longo dos últimos 12 anos em disciplinas na UFABC. Os experimentos foram realizados em disciplinas como Bases Computacionais da Ciência, Processamento da Informação, Programação Estruturada, Processamento Digital de Imagens e Visão Computacional, tanto nos Bacharelados em Ciência e Tecnologia (BCT) e Ciência da Computação (BCC), quanto na pós-graduação em Ciência da Computação. Além disso, o livro também relata um experimento realizado na disciplina de Funções de Uma Variável (Cálculo) no BCT, utilizando a biblioteca sympy.org para questões algébricas.

Este livro apresenta os resultados de experimentos conduzidos em três modalidades: híbrida, totalmente remota e totalmente presencial. Foram selecionados os melhores resultados obtidos por meio de artigos e registros de software, os quais estão disponíveis para consulta em vision.ufabc.edu.br. Os experimentos demonstram a eficácia do método avaliativo utilizando o MCTest na avaliação de disciplinas de computação, bem como em processos seletivos gerais.

Com o sistema MCTest apresentado neste livro, os professores têm à disposição uma ferramenta poderosa para aprimorar a avaliação de habilidades e competências dos estudantes em programação. Ao utilizar o MCTest, os professores podem minimizar o esforço de criar e corrigir exames e EPs, permitindo que se dediquem a atividades de ensino menos repetitivas e mais significativas.

Além disso, o MCTest oferece uma avaliação mais eficiente, tornando o processo de avaliação menos suscetível a erros humanos e mais preciso. Isso pode proporcionar uma experiência de aprendizado mais satisfatória para os estudantes, que se sentirão mais seguros e motivados a progredir em suas habilidades.

1.2 Breve histórico do MCTest

Nesta seção, será apresentado um breve histórico do MCTest. Essa ferramenta surgiu como resposta à necessidade do curso de Especialização em Tecnologias e Sistemas de Informação (TSI) Lato Sensu da UFABC. Este curso foi pioneiramente oferecido pela universidade em 2010 (TSI-1). Desde a sua segunda edição (TSI-2), em 2012, o MCTest tem sido amplamente utilizado em diversos processos seletivos e exames de disciplinas, desempenhando um papel fundamental também na Escola Preparatória da UFABC (EPUFABC).

Ao longo deste livro, os experimentos realizados com o MCTest serão descritos em detalhes, abrangendo diversos aspectos de seu uso e aplicação. Caso haja interesse em conhecer mais sobre o uso do MCTest em processos seletivos, exames de disciplinas ou na Escola Preparatória da UFABC, basta consultar os capítulos específicos dedicados a esses temas.

1.2.1 MCTest no TSI

O curso TSI-1 estabeleceu uma parceria com o programa Universidade Aberta do Brasil, vinculado ao Ministério da Educação (UAB/MEC), e oferecido na modalidade de ensino a distância (EaD), com exames presenciais correspondendo a mais de 50% da nota de aprovação do estudante. Em 2010, houve 1078 candidatos para as 200 vagas disponíveis em quatro polos do estado de São Paulo, com 16 tutores no total.

A segunda edição do curso (TSI-2) iniciou em 2012, com 1171 inscritos distribuídos em 4 polos, contendo 50 estudantes em cada polo. No entanto, apenas 674 candidatos concluíram o processo seletivo ([ZAMPIROLLI, 2013](#)). Nesta edição, foi desenvolvida a primeira versão do MCTest, como será detalhada na próxima seção.

Na terceira oferta, em 2014 (TSI-3), foram 689 candidatos para 6 polos, com 300 estudantes matriculados. No início, houve 11 tutores, porém, na fase do TCC, em 2015, o número total de tutores aumentou para 16.

A quarta edição do curso (TSI-4) ocorreu em 2017, porém sem as bolsas UAB para professores e tutores. Como resultado, foram oferecidas apenas duas turmas, nos polos da UFABC em Santo André (SA) e São Bernardo do Campo (SBC), com 25 estudantes em cada polo e sem a presença de tutores. Apesar disso, a demanda permaneceu alta, com 863 inscritos para apenas 50 vagas disponíveis nessa edição.

Até a quarta edição do TSI, a parte administrativa do curso era conduzida pela Pró-Reitoria de Extensão. Durante a fase de transição dos cursos de especialização para a Pró-Reitoria de Pós-Graduação, houve um intervalo maior entre as edições do curso, que foi agravado pela pandemia de COVID-19. Como resultado, somente no segundo semestre de 2022 foi possível iniciar o TSI-5, também sem as bolsas UAB, com apenas 25 estudantes em cada polo da UFABC em SA e SBC. Nessa edição, talvez devido à pandemia, apenas 63 candidatos participaram do processo seletivo, em contraste com a edição anterior, que teve uma média de 17,26 candidatos por vaga.

No início de 2023, deu-se início à sexta edição do TSI (TSI-6), que agora conta com o retorno das bolsas UAB. Essa edição está sendo conduzida em paralelo com a TSI-5 e conta com a colaboração de 7 tutores. Ao todo, foram registrados 565 candidatos para os 6 polos do estado de São Paulo, resultando em uma média de 35 estudantes por polo.

1.2.2 Versões do MCTest

O MCTest é um projeto que evoluiu ao longo do tempo, e isso pode ser visto nas diferentes versões registradas no Instituto Nacional de Propriedade Intelectual (INPI). Cada versão registra uma evolução do projeto em termos de tecnologia, metodologia e objetivos.

1.2. BREVE HISTÓRICO DO MCTEST

A primeira versão do MCTest (MCTest-1) foi registrada no INPI com o número de registro BR 51 2013 001231 7. Essa versão foi desenvolvida em Matlab para permitir a correção automática dos exames gerados em Word ou BROffice. O MCTest-1 foi utilizado nos processos seletivos do TSI de 2012 até 2017, contribuindo significativamente para a avaliação das competências cognitivas dos candidatos.

Foi desenvolvida uma versão do MCTest denominada MCTest-2 em colaboração com um estudante de Iniciação Científica, e essa versão foi devidamente registrada no INPI sob o número de registro BR 51 2015 001444 7. Essa versão específica, disponibilizada para dispositivos Android, possibilitou a correção automática dos exames gerados em Word ou BROffice ([CHINA; ZAMPIROLI, 2014; ZAMPIROLI, 2015; CHINA, 2016](#)).

A versão MCTest-3 foi desenvolvida em Python e registrada no INPI com o número de registro BR 51 2015 001445 5. Assim como as versões anteriores, essa versão também permitia a correção automática de exames gerados em Word/BROffice.

Já a versão MCTest-4, registrada no INPI com o número de registro BR 51 2016 001344 3, foi desenvolvida em Python e também permitia a correção automática dos exames. No entanto, os exames foram digitalizados e enviados por FTP para o servidor [vision.ufabc.edu.br](#), com os exames gerados pela versão MCTest-4.G ([ZAMPIROLI, 2016](#)). Essa versão representou um avanço significativo em relação às anteriores, ao permitir o envio remoto dos exames digitalizados e a correção automática, simplificando o processo de avaliação dos candidatos. Essa versão está disponibilizada em [github.com/fzampirolli/mctest4](#) ([ZAMPIROLI, 2016](#)).

Posteriormente, entre maio e agosto de 2018, foi desenvolvida e lançada a versão atual do MCTest, a MCTest-5.2. Essa versão apresentou grandes mudanças, sendo disponibilizada em páginas web e desenvolvida em Python com IDE Django, permitindo a geração e correção automática de exames. A MCTest-5.2 está disponível em [github.com/fzampirolli/mctest](#), com os exames gerados em L^AT_EX e usando o banco de dados MySQL. Desde o seu lançamento, essa nova versão gerou diversas publicações, a partir de 2018, que podem ser acessadas através do site [vision.ufabc.edu.br](#).

A grande contribuição do MCTest ao estado da arte em exames avaliativos e formativos está na possibilidade de criar questões parametrizadas, intercalando descrições em L^AT_EX com códigos em Python, disponível a partir da versão MCTest-4.G. Os exames gerados com essas questões podem ter a correção automática com:

1. Exames digitalizados e enviados para o MCTest;
2. Se forem questões de exercícios de programação (EPs), com respostas dadas em código em alguma linguagem de programação, o MCTest gera um arquivo com os casos de teste e outro com a variação de cada estudante. Esses arquivos devem ser inseridos em atividades VPL no Moodle para a correção automática;
3. Outra possibilidade é o MCTest exportar um banco de questões em formato Aiken ou XML, que podem ser importados pelo Moodle para criar exames.

O Moodle também aceita questões paramétricas, mas são limitadas ao conjunto de funções do PHP, como relatado no artigo de [Zampirolli, Batista, Pimentel e Braga \(2021\)](#).

1.2.3 MCTest nos processos seletivos e exames de disciplinas do TSI

Desde 2012, todos os processos seletivos do TSI têm sido realizados com o uso do MCTest. Até a versão MCTest-4, os exames eram criados em editores de texto, como Word ou BROffice. A partir da versão MCTest-4G, no entanto, os exames começaram a ser criados diretamente pelo sistema.

Além disso, os professores da disciplina de Processamento de Informação (CS1) do Bacharelado em Ciência e Tecnologia da UFABC, na modalidade semipresencial, viram a oportunidade de utilizar o sistema em questões contendo EP, como relatado no artigo de [Zampirolli, Goya, Pimentel e Kobayashi \(2018\)](#). Com o passar do tempo, várias outras turmas e professores também começaram a utilizar o sistema e a sugerir melhorias.

Desde 2012, todas as 13 disciplinas do TSI tiveram seus exames presenciais corrigidos com o auxílio do MCTest. A partir de 2017, os exames também passaram a ser gerados pelo sistema, sendo reaproveitados em cada nova oferta, com pequenas modificações. Essa mudança representou uma significativa melhoria na eficiência e rapidez do processo de avaliação, além de permitir a personalização e adaptação dos exames para cada nova oferta.

1.2.4 MCTest na Escola Preparatória

A UFABC oferece gratuitamente à comunidade da região do ABC um curso preparatório para vestibulares chamado Escola Preparatória da UFABC (EPUFABC). Desde 2017, o MCTest tem sido um importante aliado no processo seletivo dos candidatos da EPUFABC. Esses processos atraem milhares de candidatos para as cerca de 300 vagas disponíveis anualmente.

Para viabilizar esse processo, foi empregado o MCTest para gerar o quadro de respostas (QR) dos candidatos, enquanto as QMs foram impressas separadamente. O uso do MCTest mostrou-se fundamental para agilizar e simplificar o processo seletivo, permitindo a correção automática dos exames e a rápida geração do resultado. Além disso, a precisão e confiabilidade do sistema garantiram uma avaliação sem erros dos candidatos, contribuindo para a excelência acadêmica da UFABC e o sucesso dos estudantes matriculados na EPUFABC.

Essa parceria permitiu o desenvolvimento de mais um sistema (ENEM Interativo) e a publicação de dois artigos, os quais serão resumidos a seguir e detalhados na Parte [IV](#) de experimentos.

ENEM Interativo

Em colaboração com a EPUFABC, foi desenvolvido um sistema interativo denominado [ENEM interativo](#), permitindo que os estudantes utilizem exames antigos do ENEM e os microdados disponibilizados pelo INEP para estudo. Nesse sistema, os exames em formato PDF são convertidos para HTML, e botões de resposta são incluídos para cada questão. Ao término de cada exame, o botão de estatísticas abre uma nova página contendo dados do gabarito, respostas dos estudantes, habilidades (valores fornecidos pelo INEP) e gráficos.

Esses gráficos mostram a Curva Característica do Item (CCI) da Teoria de Resposta ao Item (TRI) ([BIRNBAUM, 1968](#)), sendo calculada a partir dos parâmetros da amostra aleatória de 10.000

1.3. DIREITOS AUTORAIS

estudantes, incluindo a discriminação (a), habilidade (b) e chute (c). Os valores médios e o desvio padrão da amostra também são calculados, permitindo uma melhor compreensão da distribuição dos resultados.

Outro gráfico gerado é o BoxPlot ([JW, 1977](#)), juntamente com o ViolinPlot ([HINTZE; NELSON, 1998](#)), para visualizar as frequências de acertos (valor 1) e erros (valor 0) de forma mais clara e precisa.

É importante ressaltar que esses gráficos estão disponíveis apenas para exames com respostas de mais de 10.000 estudantes. Essa ferramenta tem sido uma contribuição para o processo de preparação dos estudantes para o ENEM, fornecendo informações valiosas sobre o desempenho dos estudantes e as tendências de cada questão ao longo do tempo ([ZAMPIROLI, 2021](#)).

Comparações de três métodos de seleção

Foi realizado um estudo comparativo entre três métodos de seleção de candidatos para a EPUFABC: 1) o uso do MCTest com exames presenciais, aplicando a Teoria Clássica ao Item (TCI); 2) o uso do Moodle durante a pandemia de COVID-19, com a realização dos exames a distância, também utilizando a TCI; e 3) o uso do MCTest com exames presenciais, empregando a TRI.

Os resultados obtidos indicaram que, ao adotar a TRI, as chances de empate entre candidatos foram consideravelmente reduzidas. Adicionalmente, foi constatado que a correlação entre as classificações obtidas pelos métodos TCI e TRI é alta, evidenciando coerência entre os dois métodos de pontuação ([ZAMPIROLI, 2021](#)).

1.3 Direitos autorais

Nesta seção, serão abordados os direitos autorais do software e das questões criadas pelos professores e disponibilizadas no banco de dados do MCTest.

1.3.1 Direitos autorais do software

Até a primeira edição deste livro, o MCTest foi desenvolvido integralmente pelo Prof. Francisco de Assis Zampirolli, da Universidade Federal do ABC (UFABC), visando atender às demandas da instituição na avaliação de inúmeros estudantes e candidatos de processos seletivos. O sistema contou com a valiosa colaboração de colegas da UFABC, que compartilharam suas necessidades em relação às avaliações. No entanto, é crucial ressaltar que o MCTest não é um produto comercial e, infelizmente, não segue rigorosamente o ciclo de vida de desenvolvimento de software.

Trata-se de um sistema construído inspirado no modelo de prototipagem de Engenharia de Software. Para vir a se tornar um produto comercial, seria necessário submetê-lo a uma engenharia reversa e aprimorá-lo, sobretudo no que concerne à naveabilidade entre as telas e à adição de mais funcionalidades de gestão de processos.

Mesmo com as limitações mencionadas anteriormente, o MCTest está em produção na UFABC e é utilizado por professores e gestores para avaliar milhares de estudantes e candidatos, reduzindo significativamente a parte repetitiva dos processos de avaliação.

O MCTest possui código aberto e está disponível no [GitHub](#), podendo ser usado e modificado sob a licença AGNU. Isso implica que todos têm a liberdade de utilizar o sistema. Para manter o MCTest atualizado, melhorias feitas por terceiros e implementadas em seus servidores também devem ser compartilhadas com o público. Essa licença pode ser encontrada na implantação na UFABC em mctest.ufabc.edu.br/license e também está apresentada a seguir:

Licença:

Copyright (C) 2018-2023 Francisco de Assis Zampirolli from Federal University of ABC and individual contributors. All rights reserved. This file is part of MCTest 5.2. Languages: Python, Django, and many libraries described at github.com/fzampirolli/mctest.

You should cite some references included in vision.ufabc.edu.br in any publication about it.

MCTest is free software: you can redistribute it and/or modify it under the terms of the GNU Affero General Public License (gnu.org/licenses/agpl-3.0.txt) as published by the Free Software Foundation, either version 3 of the License or (at your option) any later version.

MCTest is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

1.3.2 Direitos autorais das questões disponíveis no BD

Outro ponto muito importante a respeito dos direitos autorais refere-se às questões elaboradas no MCTest e armazenadas em seu banco de dados (BD). A filosofia do MCTest é trabalhar colaborativamente para produzir e utilizar material didático em avaliações. Dessa forma, as questões elaboradas por um professor em uma disciplina da UFABC estão disponíveis para os demais professores desta disciplina, para uso exclusivo em suas avaliações nesta universidade.

Cada instituição pode definir seus próprios critérios de direito autoral para o conteúdo disponibilizado no BD do MCTest. É importante ressaltar que o direito autoral do software apresentado na seção anterior não cobre essa parte. Portanto, cabe a cada instituição definir seus próprios critérios em suas implantações do MCTest.

A seguir, será apresentado o que foi definido para os professores da UFABC que optaram por utilizar o sistema na implantação disponibilizada em mctest.ufabc.edu.br/According.

De Acordo:

O MCTest (disponível gratuitamente no [GitHub](#)) foi implantado no servidor da UFABC: mctest.ufabc.edu.br.

Na UFABC, o MCTest é um serviço colaborativo para geração e correção automática de atividades de formação (como Listas de Exercícios) e de avaliação (como Testes ou Exames).

Esse serviço está disponível gratuitamente para todos os docentes da UFABC.

Para poder utilizar o MCTest, o professor deve inscrever-se na plataforma e ser cadastrado em alguma disciplina pelo seu coordenador.

Todo o material intelectual (aulas, vídeos, questões, etc.) disponível neste site está protegido pelo direito autoral dos seus respectivos autores e, em última instância, é propriedade da UFABC. Os autores proprietários do material disponibilizado no site concordam em liberar o seu uso sem restrições pelos professores da instituição a qual está vinculado somente no âmbito das disciplinas e cursos desta instituição. Além disso, quando me inscrevo neste serviço, nesta instituição, CONCORDO que:

- Posso utilizar livremente em meus exames todas as questões disponíveis já cadastradas na disciplina;
- Não posso publicar questões criadas por outros professores em quaisquer outras mídias, como, por exemplo, livros e artigos;
- Todas as novas questões que eu criar podem ser reaproveitadas em exames elaborados por outros professores da mesma disciplina, sem prévio aviso;
- Só posso mudar as questões dos outros professores SOMENTE SE HOUVER ALGUM ERRO. Porém, sem alterar a autoria da questão.

1.4 Instalando o MCTest

O MCTest pode ser instalado localmente no computador pessoal utilizando a máquina virtual [VirtualBox](#). Em seguida, é necessário baixar uma das seguintes versões do Linux: Ubuntu 20.04 ou 22.04, ou ainda a distribuição Mint 21 Mate. Essas distribuições já foram testadas, mas o MCTest pode funcionar em outras também.

Após instalar o Linux na máquina virtual ou utilizá-lo diretamente em uma instalação Linux, sem a máquina virtual, basta fazer o *download* do arquivo [`setup-all.sh`](#), disponível no endereço github.com/fzampirolli/mctest do GitHub, e executá-lo em um terminal com privilégios

de superusuário (`sudo su`). Em seguida, execute o comando `source _setup-all.sh`. Com isso, após aproximadamente 10 minutos, o MCTest será aberto em uma aba do navegador, instalando um banco de dados com apenas três usuários de teste, disponível no arquivo `mctest.sql`. Se o leitor quiser instalar a versão com todos os exemplos utilizados neste livro, basta instalar o arquivo `book/1ed-br/mctestLivro.sql`, no endereço github.com/fzampirolli/mctest. Para mais detalhes, consulte o conteúdo do arquivo `_setup-all.sh`.

1.5 Organização do livro

Este livro foi elaborado para fornecer um guia completo e abrangente sobre o MCTest, uma plataforma de avaliação e gerenciamento de aprendizado. O conteúdo está organizado em quatro partes, cada uma focada em um aspecto diferente do MCTest, desde os fundamentos até experimentos e aplicações práticas, conforme apresentado na Figura 1.1 e resumido a seguir:

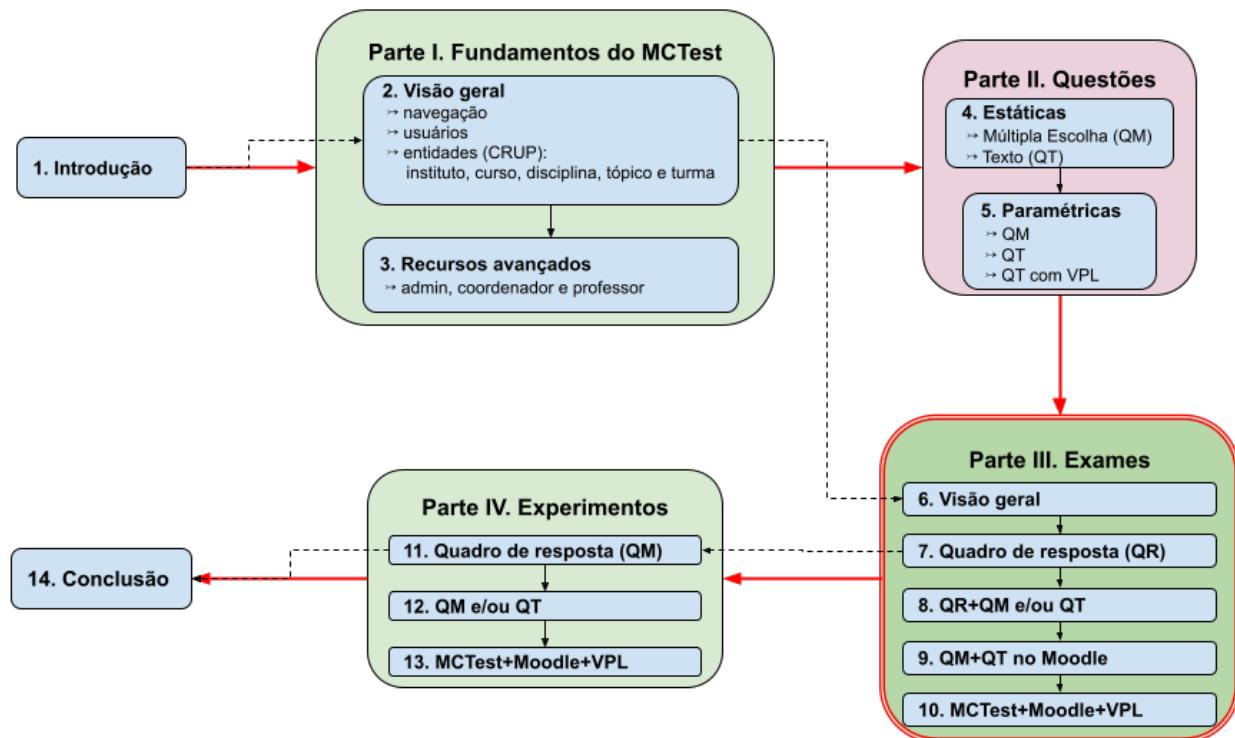


Figura 1.1: Organização do livro.

Parte I – Fundamentos do MCTest: apresenta a introdução ao MCTest, seus componentes básicos e funcionalidades essenciais. Esta parte inclui o Capítulo 2 – Visão geral do MCTest e o Capítulo 3 – Recursos avançados, que abordam a visão geral do sistema, navegação, usuários e os métodos básicos de operação (CRUD – *Create, Read, Update, Delete*). As seções são dedicadas à manutenção de diferentes áreas do sistema por usuários administradores, coordenadores e professores.

Parte II – Questões no MCTest: trata de diferentes tipos de questões disponíveis no MCTest,

1.5. ORGANIZAÇÃO DO LIVRO

como questões estáticas e paramétricas. Os Capítulos [4 – Questões estáticas](#) e [5 – Questões paramétricas](#) discutem questões de múltipla escolha (QMs), questões dissertativas ou de texto (QTs) e QTs com código.

Parte III – Exames no MCTest: aborda o processo de criação e gerenciamento de exames no MCTest, com uma visão geral no Capítulo [6 – Visão geral dos exames](#). Os Capítulos [7 – Exames com Quadro de Respostas \(QR\)](#), [8 – Exames com QR+QM e/ou QT](#), [9 – Variações de exames com QM+QT para o Moodle](#) e [10 – Exames com MCTest+Moodle+VPL](#) exploram o quadro de respostas (QR) e a integração com questões de múltipla escolha (QMs) e/ou questões dissertativas (QTs), além da combinação do MCTest com Moodle e VPL para exames.

Parte IV – Experimentos de Uso do MCTest: apresenta estudos de caso e exemplos práticos do uso do MCTest em ambientes educacionais. Os Capítulos [11 – MCTest versões 1, 2 e 3 - Quadro de Respostas \(QR\)](#), [12 – MCTest versões 4, 4.G e 5](#) e [13 – MCTest+Moodle+VPL](#) apresentam diferentes cenários envolvendo o quadro de respostas (QR), questões de múltipla escolha (QMs) e/ou questões dissertativas (QTs) e a integração do MCTest com Moodle e VPL.

O livro é finalizado com o Capítulo [14 – Conclusões](#), que resume as principais ideias apresentadas ao longo do texto e oferece orientações para futuras pesquisas e desenvolvimentos do MCTest pela comunidade, visto que se trata de um sistema de código aberto. Esse capítulo é especialmente valioso para aqueles que desejam se aprofundar no tema e contribuir para o desenvolvimento contínuo do MCTest, seja por meio de pesquisas adicionais ou pela implementação de novas funcionalidades. Leitores interessados em utilizar o sistema exclusivamente para a elaboração e avaliação de exames que contenham apenas os QRs (as QMs serão apresentadas separadamente) devem seguir os capítulos indicados pelas setas tracejadas na Figura [1.1](#).

Parte I

Fundamentos do MCTest

Conteúdo

2 Visão geral do MCTest	15
3 Recursos avançados	39

Capítulo 2

Visão geral do MCTest

Conteúdo

2.1	Navegação geral e usuários	16
2.2	Criar entidades	19
2.2.1	Instituto	20
2.2.2	Curso	20
2.2.3	Disciplina	24
2.2.4	Turma	26
2.2.5	Tópico	28
2.3	Considerações finais	29

O MCTest é um sistema, gratuito, de código aberto e em desenvolvimento, cujo objetivo é fornecer e gerenciar um banco de dados para avaliações de estudantes em um sistema de ensino. O foco principal do sistema é fornecer questões que possam ser utilizadas em atividades de ensino, além de oferecer geração e correção automática de atividades e exames. Com o MCTest, os educadores terão acesso a uma ferramenta poderosa para avaliar o progresso dos estudantes, aprimorar o processo de ensino e fornecer *feedback* de forma rápida.

O MCTest representa uma importante contribuição ao estado da arte em Tecnologia da Informação e Comunicação (TIC) ao fornecer métodos para a construção e correção automatizada de questões parametrizadas de múltipla escolha (QMs) e dissertativas (QTs). O sistema utiliza diferentes bibliotecas da linguagem de programação Python para calcular parâmetros na descrição e nas alternativas das questões de forma automática. Para mais detalhes, é possível consultar as publicações disponíveis em vision.ufabc.edu.br, algumas delas também serão apresentadas nos capítulos da Parte IV. Além disso, o MCTest também suporta QTs que podem incluir exercícios de programação (EP) ou qualquer outro tipo de resposta.

O MCTest possui um ambiente de produção disponível em mctest.ufabc.edu.br, atualmente

utilizado por diversos professores e setores na Universidade Federal do ABC (UFABC). Por exemplo, a Escola Preparatória (EPUFABC) utiliza o sistema no processo seletivo anual de aproximadamente 3000 candidatos anualmente. Isso demonstra a eficácia e a confiabilidade do MCTest como uma ferramenta de avaliação em larga escala.

O MCTest está disponível gratuitamente no [GitHub](#), permitindo que outras instituições possam copiar, personalizar e instalar o sistema em seus próprios servidores de forma livre e flexível. Além disso, há diversas possibilidades de melhoria e trabalhos futuros para o MCTest, as quais podem ser exploradas seguindo a filosofia de código aberto, como definido no [Copyright © 2018–2023](#) do sistema. Para mais informações sobre as possibilidades de melhoria do MCTest, é possível consultar o *link* mctest.ufabc.edu.br/readme.

2.1 Navegação geral e usuários

A navegação geral do MCTest visa possibilitar a criação e manutenção das entidades principais do sistema, que incluem instituto, curso, disciplina, turma, exame, tópico e questão. Os usuários do sistema são categorizados em três perfis: professor, coordenador e administrador. No entanto, na Figura 2.1, quando nenhum usuário está logado no sistema, um usuário da web pode visualizar os conteúdos públicos de institutos, cursos e disciplinas, bem como se inscrever ou entrar no sistema. Para uma comparação das funcionalidades de cada perfil de usuário, consulte também a Figura 2.2.

Observação:

As figuras apresentadas neste livro correspondem à implantação do MCTest em uma máquina local, utilizando um banco de dados de exemplo disponível no [GitHub](#), com um número limitado de entidades cadastradas. Para ver um exemplo com um banco de dados mais completo, visite o site mctest.ufabc.edu.br. Para adiantar, o leitor pode instalar uma máquina virtual utilizando o [VirtualBox](#) e, em seguida, fazer o *download* de um sistema operacional suportado, conforme explicado no arquivo [_setup-all.sh](#). Basta seguir as instruções fornecidas nesse arquivo para concluir a instalação, incluindo a instalação das bibliotecas necessárias. É importante notar que o processo de instalação do pacote [LATEX](#) pode levar mais de 10 minutos. Para mais detalhes, ver Seção 1.4 – [Instalando o MCTest](#).

Atualmente, além do administrador, o sistema está disponível apenas para usuários do tipo professor, que podem também ser coordenadores de disciplina com permissão para incluir professores e tópicos. O perfil de coordenador é adicionado pelo administrador do sistema ao associá-lo a uma disciplina específica. Cabe ao administrador também a responsabilidade de manter a página do [Django](#).

No MCTest, cada tipo de usuário possui acesso restrito a um conjunto específico de entidades, como ilustrado na Figura 2.2. É importante ressaltar que os itens (c) e (d) são idênticos para professores e coordenadores. Por exemplo, ambos podem visualizar todas as turmas, tópicos e questões das disciplinas em que estão inscritos, mas somente o coordenador pode realizar alterações

2.1. NAVEGAÇÃO GERAL E USUÁRIOS



Institutos Cursos Disciplinas

Entrar Inscrever

MCTest v.5.3 - desenvolvimento

Versão estável, sem previsões de novas melhorias. Use somente a versão em produção: [mctest](#) (BACKUP 08/02/2023).

Atualizar Questão

Criar-PDF	Salvar-Json
Ver esta questão no formato PDF	
Salvar questões de um arquivo (formato Json)	

Escolher Tópico [Ex]<template>

Descrição curta template-equação0

Grupo Somente uma questão por grupo será sorteada para cada exame

Descrição Crie uma matriz 3×5 de inteiros, com elementos $a_{i,j} = i + j$, com índices começando em zero, imprima a soma dos elementos da matriz.

Comando LaTeX

Uma simples questão com comando LaTeX. Após clicar em Criar-PDF, ver próximo slide [ref19a](#); [ref19b](#)

Bem vindo ao MCTest, um portal para a geração e correção de exames.

Serviço oferecido para a comunidade para facilitar o processo de elaboração e correção de exames, com geração e correção de muitos exames.

Conteúdo dinâmico

O MCTest tem os seguintes contadores (ver também [DB1](#); [DB2](#)):

- **Institutos:** 1 [Instituto tem Cursos]
- **Cursos:** 1 [Curso tem Disciplinas]
- **Disciplinas:** 1 [Disciplina tem Tópicos, Turmas e Profs]
- **Turmas:** 1 [Turmas tem Exames, Profs e Estudantes]
- **Exames:** 1 [Exame tem Turmas e Questões] - motivações [[ref18a](#); [ref21g](#); etc.]
- **Tópicos:** 1 [Tópico tem Questões]

- **Questões:** 3 - motivações para o uso da taxonomia de bloom [[ref17cap2](#); [ref18b](#)]
 - **Múltipla Escolha:** 3
 - **Dissertativa:** 0
 - **Paramétrica:** 0 [[ref19a](#); [ref19b](#); [ref20a](#); [ref20b](#); refs21*; refs23*]
- **Estudantes:** 2
- **Usuários:** 4

Você visitou esta página 1 vez .

Enviar sugestões de melhoria para: fzampirolli@ufabc.edu.br



Copyright © 2018-2023 por Francisco de Assis Zampirolli da UFABC e Colaboradores; Quem utiliza; Download; Livro; LEIA-ME.

Apóio: UFABC; #2009/14430-1 e #2018/23561-1 - FAPESP

Figura 2.1: Tela principal do MCTest.

em tópicos de disciplinas que coordena. Além disso, os menus à esquerda são gerais para todo o conteúdo ao qual o usuário tem acesso, enquanto à direita são apresentados apenas os conteúdos de questões, turmas e exames que o usuário criou.

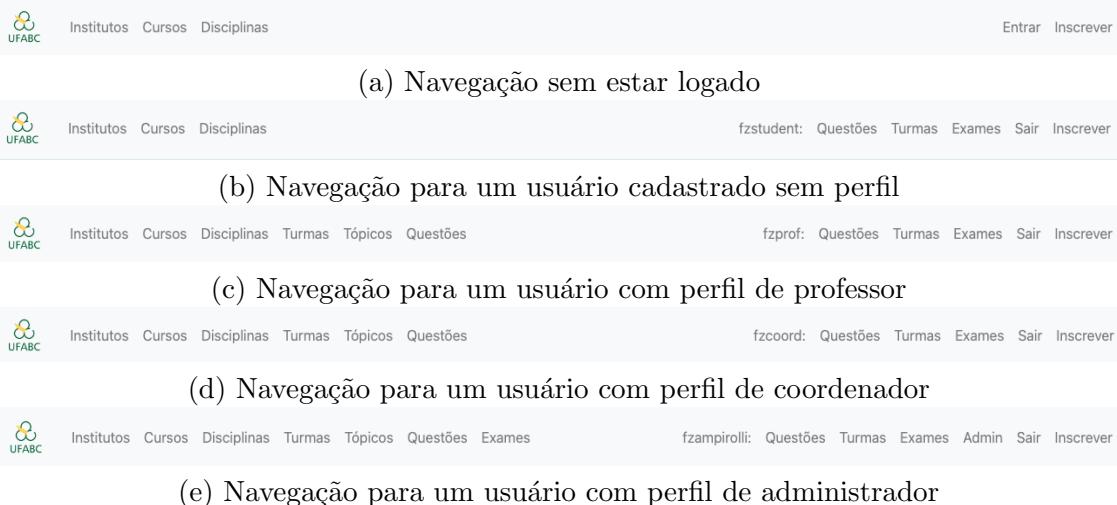


Figura 2.2: Recorte do menu de navegação principal do MCTest.

O MCTest também permite que um professor coordenador de disciplina adicione vários professores a uma disciplina por um arquivo CSV, sendo automaticamente cadastrado no sistema como perfil de professor (esse processo será exemplificado no próximo capítulo). Com isso, os professores incluídos na disciplina têm acesso somente às funcionalidades do sistema que são relevantes ao seu perfil.

Destaque:

Para enfatizar um ponto específico no que se refere ao registro de professores, é de extrema relevância ressaltar que, quando um professor efetua seu cadastro de forma independente, ele não é automaticamente incluído no grupo de professores. Nessa situação, torna-se imperativo que um administrador intervenha a fim de efetuar essa inclusão, por meio do acesso à aba “Admin”, conforme ilustrado na Figura 2.2-(e).

Antes de demonstrar como criar entidades no sistema, é importante cadastrar-se na plataforma, clicando na opção “Inscrever”, conforme indicado na Figura 2.2 e exemplificado na Figura 2.3. É necessário ressaltar que somente os professores podem se inscrever neste serviço e devem utilizar o e-mail institucional fornecido pela instituição durante o cadastro. Por exemplo, o endereço `user@ufabc.edu.br` é válido, enquanto `user@aluno.ufabc.edu.br` é inválido. O e-mail deve conter a URL da instituição (por exemplo, `@ufabc.edu.br`). Tais restrições são cruciais para garantir a validade e a segurança das informações associadas às instituições cadastradas no sistema. Finalmente, o botão “De Acordo”, apresentado na Figura 2.3, foi detalhado na Seção 1.3.2 – Direitos autorais das questões disponíveis no BD.

Melhorias:

- É importante ressaltar que ainda não foi implementada a funcionalidade de segurança para validar o e-mail durante o processo de inscrição. Será necessário implementar essa funcionalidade para garantir a integridade e a segurança dos dados cadastrados no sistema.
- Outra melhoria importante para a segurança do sistema é a utilização de um certificado digital, que permitirá a utilização do protocolo HTTPS em vez de HTTP nos servidores utilizados para a implantação do MCTest. O HTTPS criptografa a comunicação entre o navegador do usuário e o servidor, garantindo a confidencialidade e a integridade dos dados transmitidos.

Inscrever

Inscrever

Somente professores podem se inscrever neste serviço.

É necessário utilizar o e-mail institucional, conforme informado no cadastro da instituição.

Por exemplo, user@ufabc.edu.br é válido, enquanto user@aluno.ufabc.edu.br é inválido.

Email:

Primeiro nome:

Último nome:

Senha:

- Sua senha não pode ser tão parecida com suas outras informações pessoais.
- Sua senha precisa conter pelo menos 8 caracteres.
- Sua senha não pode ser uma senha habitualmente utilizada.
- Sua senha não pode ser inteiramente numérica.

Confirmação de senha:

Informe a mesma senha informada anteriormente, para verificação.

[De Acordo](#)

[Inscriver](#)

Figura 2.3: Tela para a inscrição do usuário.

2.2 Criar entidades

O MCTest é um sistema que gerencia diversas entidades em um banco de dados MySQL, como detalhado neste e no próximo capítulo. Essas entidades são criadas e relacionadas para modelar o domínio educacional com foco na avaliação dos estudantes. Além disso, o sistema oferece funcionalidades para o cadastro, alteração, remoção e consulta dessas entidades e seus relacionamentos.

tos.

Nesta seção, você encontrará a descrição das principais entidades de negócio de um sistema educacional que tem como foco as avaliações. É importante ressaltar que, embora as entidades, questão e exame sejam de extrema importância no contexto do MCTest, elas serão abordadas em capítulos específicos para uma exposição mais detalhada e aprofundada.

2.2.1 Instituto

A entidade instituto representa a instituição cadastrada no sistema e contém atributos como nome, código, logo, site, entre outros. A criação de uma entidade instituto é restrita ao usuário com perfil de administrador.

Na Figura 2.4, ao clicar em “Criar um novo Instituto”, uma tela será aberta para cadastrar informações sobre o instituto, conforme ilustrado na Figura 2.5. Ao clicar em um instituto existente na coluna “Código” ou “Nome” da Figura 2.4, o usuário será direcionado para uma tela para visualizar as informações do instituto, conforme ilustrado na Figura 2.6. É importante observar que nesta tela também é apresentada a lista de cursos, que será detalhada na próxima seção.

Ao clicar no botão “Atualizar” em um instituto existente na coluna “Ações” da Figura 2.4, o usuário será direcionado para uma nova tela para atualizá-lo, semelhante à Figura 2.5. Se o usuário clicar em “Apagar”, o instituto será removido do banco de dados, após confirmação na janela de diálogo apresentada na Figura 2.7.

A Figura 2.5 apresenta o recorte da tela para “Criar de um novo Instituto”, que é similar à de “Atualizar um Instituto”. Os três últimos campos exibidos na tela são utilizados para contabilizar o número de exames e questões gerados e corrigidos pelo sistema desde a criação do instituto. Esses contadores são úteis para monitorar a atividade do sistema em relação a cada instituto. É importante ressaltar que apenas o administrador pode zerar esses contadores a qualquer momento, o que pode ser necessário em algumas circunstâncias, como quando é necessário reiniciar as contagens ou avaliar a atividade recente do sistema.

Observação:

As telas e as ações de criar, ler, atualizar e remover (CRUD – *Create, Read, Update, Delete*) para fazer a manutenção do instituto são semelhantes, em geral, às demais entidades do MCTest. Portanto, a partir deste ponto, serão apresentadas apenas as telas com informações diferentes, para o usuário poder se concentrar nas especificidades da manutenção das entidades do sistema. Além disso, os nomes dessas entidades nessas telas devem seguir o formato LATEX.

2.2.2 Curso

A criação da entidade curso é restrita ao usuário com perfil de administrador, como ilustrado na Figura 2.8, que também pode atualizar ou apagar um curso existente, conforme apresentado na coluna “Ações” (outros tipos de usuários não têm acesso a esses botões). É importante destacar que

2.2. CRIAR ENTIDADES

Listar Institutos

Listar Institutos					
copiar csv excel pdf imprimir colunas visíveis ▾					
Código	↑↓	Nome	↑↓	URL	↑↓
IE		Instituto Exemplo		www.ufabc.edu.br	
Atualizar Apagar					

Showing 1 to 1 of 1 entries

[Criar um novo Instituto](#)

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas

Figura 2.4: Tela para administrador atualizar, apagar ou criar um instituto.

Criar um novo Instituto

Nome	
Código	
Logo	
URL	Todos os professores devem registrar e-mail institucional
Exames gerado	0
Exames corrigidos	0
Questões corrigidas	0
Voltar Salvar	

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas

Figura 2.5: Tela para o administrador criar um instituto.

Detalhar Instituto**Nome:** Instituto Exemplo**Código:** IE**Logo:** logo**URL:** www.ufabc.edu.br**Exames gerado:** 6**Exames corrigidos:** 0**Questões corrigidas:** 0**Listar Cursos****copiar csv excel pdf imprimir colunas visíveis ▾****Search:**

Código	Nome	Ações
CE	IE - Curso Exemplo	Atualizar Apagar

Showing 1 to 1 of 1 entries

Criar um novo Curso

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas

Figura 2.6: Tela apresentando os detalhes de um instituto.

Apagar InstitutoTem certeza que deseja apagar este instituto: **Instituto Exemplo?****Cancelar****Apagar**

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas

Figura 2.7: Tela para confirmar a exclusão do instituto.

um curso pode pertencer a um ou vários institutos, permitindo assim a organização dos cursos de acordo com sua área de atuação ou afinidade temática.

Na UFABC, por exemplo, um instituto como o Centro de Matemática, Computação e Cognição

2.2. CRIAR ENTIDADES

(CMCC) pode conter vários cursos. Além disso, um curso pode pertencer a mais de um instituto, e existem também cursos intercentros, como o Bacharelado em Ciência e Tecnologia (BCT), que pode pertencer ao instituto PROGRAD (Pró-Reitoria de Graduação) e também aos três centros existentes na UFABC.

Outros exemplos de cursos na UFABC incluem a Escola Preparatória, que pode ser um curso do instituto PROEC (Pró-Reitoria de Extensão e Cultura), e o curso de idiomas, que pertence ao instituto NETEL (Núcleo Educacional de Tecnologias e Línguas), entre outros.

Listar Cursos					
copiar csv excel pdf imprimir colunas visíveis ▾					
Código	↑↓	Nome	↑↓	Instituto	↑↓
CE		Curso Exemplo		IE	
Atualizar Apagar					

Showing 1 to 1 of 1 entries

[Criar um novo Curso](#)

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas

Figura 2.8: Tela que apresenta a lista de cursos, na qual o administrador pode atualizar, apagar ou criar um novo curso.

Essa flexibilidade na associação dos cursos aos institutos permite uma maior customização do sistema de avaliação educacional do MCTest, conforme as necessidades específicas de cada instituição ou curso. Dessa forma, é possível adaptar o sistema para atender às demandas e particularidades de diferentes áreas de conhecimento e instituições de ensino.

Ao clicar no botão “Criar um novo Curso” na Figura 2.8, o administrador será direcionado para a tela de criação de um novo curso, apresentada na Figura 2.9. Nesta tela, o administrador deve preencher as informações necessárias sobre o curso, como o nome e código. Além disso, é necessário

selecionar o instituto ao qual o curso pertence, como exemplificado no caso do “Instituto Exemplo”. É importante destacar que é possível associar mais de um instituto a um curso específico, como será exemplificado no próximo capítulo. Após preencher todas as informações, o administrador deve clicar no botão “Salvar” para salvar o curso no banco de dados do sistema.

Criar um novo Curso

Escolha Instituto(s)	
Instituto Exemplo	
Nome	Nome do curso - não incluir o código
Código	Código Curso
<input type="button" value="Voltar"/> <input type="button" value="Salvar"/>	

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas

Figura 2.9: Tela para administrador criar um curso novo.

Melhorias:

É importante destacar que o banco de dados do sistema já suporta o relacionamento entre um conjunto de professores/coordenadores e uma instituição/curso. Portanto, é necessário implementar as funcionalidades que permitam vincular esses profissionais às entidades correspondentes. Isso permitirá que a gestão dos professores/coordenadores seja feita de forma mais organizada e eficiente, facilitando a alocação desses profissionais em diferentes cursos ou instituições.

Ao clicar em um curso nas colunas “Código” ou “Nome” na Figura 2.8, o usuário será direcionado para uma tela detalhando as informações do curso selecionado, conforme ilustrado na Figura 2.10. É importante observar que nesta tela também são inseridas informações sobre as disciplinas do curso selecionado, em forma de uma lista. A manutenção das disciplinas será abordada na próxima seção.

2.2.3 Disciplina

A entidade disciplina é responsável por representar as disciplinas cadastradas no sistema, sendo sua criação restrita ao usuário com perfil de administrador, que também pode atualizar ou apagar uma disciplina existente, como ilustrado na Figura 2.11, conforme apresentado na coluna “Ações” (outros tipos de usuários não têm acesso a esses botões de ação). No entanto, o usuário com perfil de coordenador possui permissão apenas para atualizar a disciplina que coordena, como será detalhado na próxima seção.

2.2. CRIAR ENTIDADES

Detalhar Curso

Curso: Curso Exemplo
Código: CE

Listar Disciplinas

copiar csv excel pdf imprimir colunas visíveis ▾

Search:

Código ↑↓	Nome ↑↓	Ações ↑↓
DE	IE - CE - Disciplina Exemplo	Atualizar Apagar

Showing 1 to 1 of 1 entries

[Criar uma nova Disciplina](#)

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas

Figura 2.10: Tela apresentando os detalhes de um curso.

A entidade disciplina é essencial para a organização das turmas e tópicos, o que permite criar exames específicos para atender a várias turmas simultaneamente, facilitando a gestão e a aplicação de avaliações na plataforma.

A Figura 2.12 mostra a tela para criar uma disciplina. Esta tela é semelhante à primeira parte da tela de atualização de uma disciplina. No caso de uma disciplina com muitas turmas, foram implementadas funcionalidades na segunda parte da tela de atualização de disciplina para o coordenador poder cadastrar todos os professores e estudantes de várias turmas de uma só vez, através da importação de arquivos no formato CSV, conforme ilustrado na Figura 2.13. Essa funcionalidade será detalhada no próximo capítulo, na Seção 3.2.2 – Criar várias turmas com arquivo CSV.

A flexibilidade na associação de cursos permite uma personalização maior do sistema de avaliação educacional do MCTest, conforme as necessidades específicas de cada disciplina. Além

Listar Disciplinas

The screenshot shows a table titled 'Listar Disciplinas'. The table has columns: Código, Nome, Curso, and Ações. The first row contains the values 'DE', 'Disciplina Exemplo', 'CE', and two buttons: 'Atualizar' (blue) and 'Apagar' (red). Above the table is a toolbar with buttons for copiar, csv, excel, pdf, imprimir, and columnas visíveis. There is also a search bar labeled 'Search:'.

Código	Nome	Curso	Ações
DE	Disciplina Exemplo	CE	<button>Atualizar</button> <button>Apagar</button>

Showing 1 to 1 of 1 entries

[Criar uma nova Disciplina](#)

Somente coordenador pode atualizar disciplina

Figura 2.11: Tela que apresenta a lista de disciplinas, na qual o administrador pode atualizar, apagar ou criar uma nova disciplina.

disso, é possível associar professores e coordenadores à disciplina, permitindo uma organização mais eficiente e atribuição de responsabilidades na gestão da disciplina. Para selecionar mais de um professor, basta manter a tecla **Ctrl** pressionada.

É fundamental destacar que a criação e associação de disciplinas aos cursos devem ser realizadas com cautela e atenção pelo coordenador, visando garantir a precisão e efetividade da avaliação educacional. A associação correta de disciplinas aos respectivos cursos, institutos e professores é crucial para a realização de exames precisos e eficazes.

Ao clicar em uma disciplina nas colunas “Código” ou “Nome” da Figura 2.11, será aberta uma tela detalhando as informações da disciplina, conforme ilustrado nas Figuras 2.14 e 2.15.

2.2.4 Turma

A entidade turma é responsável por representar as turmas cadastradas para cada disciplina no MCTest. A criação, atualização ou exclusão de turmas pode ser realizada pelo administrador,

2.2. CRIAR ENTIDADES

Criar uma nova Disciplina

Escolher Curso(s)

IE - Curso Exemplo

Nome	Nome Disciplina
Código	Código da Disciplina
Objetivo	Objetivo Disciplina

Profs

fzampirolli@ufabc.edu.br
fzstudent@ufabc.edu.br
fzprof@ufabc.edu.br
fzcoord@ufabc.edu.br

Coords

fzampirolli@ufabc.edu.br
fzstudent@ufabc.edu.br
fzprof@ufabc.edu.br
fzcoord@ufabc.edu.br

Voltar Salvar

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas

Figura 2.12: Tela para o administrador criar uma disciplina. Tela semelhante à primeira parte da tela de atualizar uma disciplina.

pelo coordenador ou pelo professor da disciplina, conforme apresentado na Figura 2.16. Nesta figura, o professor tem a opção de criar uma nova turma ao clicar em “Criar uma nova Turma”, abrindo a tela apresentada na Figura 2.17. Para isso, é necessário escolher uma disciplina, definir o código da sala da disciplina e especificar se a turma é de teoria ou prática.

Após a criação da turma pelo professor, a tela apresentada na Figura 2.16 deve ser atualizada e é possível clicar em “Atualizar” na coluna “Ações” para inserir estudantes na turma. Essa ação levará à tela de atualização da turma, conforme ilustrado na Figura 2.18. Nessa tela, é possível fazer a importação de um arquivo no formato CSV contendo a lista de estudantes que serão adicionados à turma.

Na Figura 2.16, o professor pode clicar no código da turma (primeira coluna), o que exibe a tela apresentada na Figura 2.19. Nessa tela, é possível atualizar ou apagar estudantes. A Figura 2.20 apresenta a tela utilizada para atualizar os dados de um estudante.

Na Figura 2.18, é possível observar uma lista de todas as disciplinas do professor, se houver alguma, em “Escolher Disciplinas”. Ao clicar no código de uma turma específica (primeira coluna), o professor pode acessar a tela apresentada na Figura 2.19.

Além disso, ao clicar no nome de um estudante específico, o professor acessa a tela mostrada

Incluir vários professores em um arquivo CSV

Nenhum arquivo escolhido

[Ver exemplos \(click com botão direito do mouse e salve o arquivo em seu computador, antes de fazer upload\)](#)

Síntaxe do arquivo CSV - Somente formato UTF-8 - Número máximo de caracteres para estudante é 45: :

João, Silva, joao@ufabc.edu.br
Maria, Campos, maria@ufabc.edu.br
Se necessário, use este conversor de acentos: [conversor](#)
Após importar o arquivo CSV, verificar abaixo se os professores estão marcados em cinza

Criar várias turmas de um arquivo CSV

Nenhum arquivo escolhido

ATENÇÃO: este esquema irá remover todos os estudantes existentes em *TODAS AS TURMAS* desta disciplina

[Ver exemplos \(click com botão direito do mouse e salve o arquivo em seu computador, antes de fazer upload\)](#)

Síntaxe do arquivo CSV - Somente formato UTF-8 - Número máximo de caracteres para estudante é 45: :

1234, João Silva, joao@ufabc.edu.br, 08h-SA-DA1, sala1, PClass, fzprof@ufabc.edu.br
4321, Maria Campos, maria@ufabc.edu.br, 10h-SB-DA2, sala2, TClass, fzprof@ufabc.edu.br
Se necessário, use este conversor de acentos: [conversor](#)

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas
Somente coordenador pode atualizar disciplina

Figura 2.13: Segunda parte da tela para o administrador atualizar uma disciplina.

na Figura 2.20, na qual é possível atualizar dados do estudante, como nome, e-mail e matrícula. Essa funcionalidade permite aos professores manter a lista de estudantes atualizada e corrigir eventuais erros ou inconsistências nas informações.

2.2.5 Tópico

Os tópicos são utilizados para organizar questões dentro de uma disciplina, sendo que cada disciplina possui um conjunto específico de tópicos associados. A criação, atualização ou exclusão de tópicos em disciplinas é permitida apenas para usuários com perfil de coordenador ou administrador, conforme ilustrado na lista de tópicos apresentada na Figura 2.21. Vale destacar que a lista exibida corresponde aos tópicos das disciplinas nas quais o usuário possui participação.

A Figura 2.22 apresenta a tela para atualização de um tópico, que é semelhante à tela de cadastro de um novo tópico. Uma das principais funcionalidades disponíveis nessa tela é a possibilidade de associar mais de uma disciplina ao tópico em questão, mantendo a tecla **Ctrl** pressionada. Essa flexibilidade permite que um mesmo tópico possa ser utilizado em diferentes disciplinas, conforme as necessidades e particularidades de cada uma delas.

Ao clicar em um tópico na primeira coluna da Figura 2.21, será aberta uma tela detalhando as questões relacionadas ao tópico, conforme ilustrado na Figura 2.23 (as outras colunas serão detalhadas nos próximos capítulos). É importante destacar o botão “Criar-PDF”, que permite a criação de um arquivo PDF contendo todas as questões do tópico selecionado, conforme ilustrado na Figura 2.24. Detalhes sobre este arquivo serão abordados em capítulos futuros.

2.3. CONSIDERAÇÕES FINAIS

Detalhar Disciplina

Disciplina: Disciplina Exemplo
Código: DE
Objetivo:

Profs: fzampirolli; fzprof; fzcoord;
Coordenador: fzampirolli; fzcoord;

Listar Tópicos

copiar csv excel pdf imprimir

Search:

Tópico	Ações
Topico Exemplo	Atualizar Apagar

Showing 1 to 1 of 1 entries

Somente coordenandor pode criar/atualizar/apagar tópicos

[Criar um novo Tópico](#)

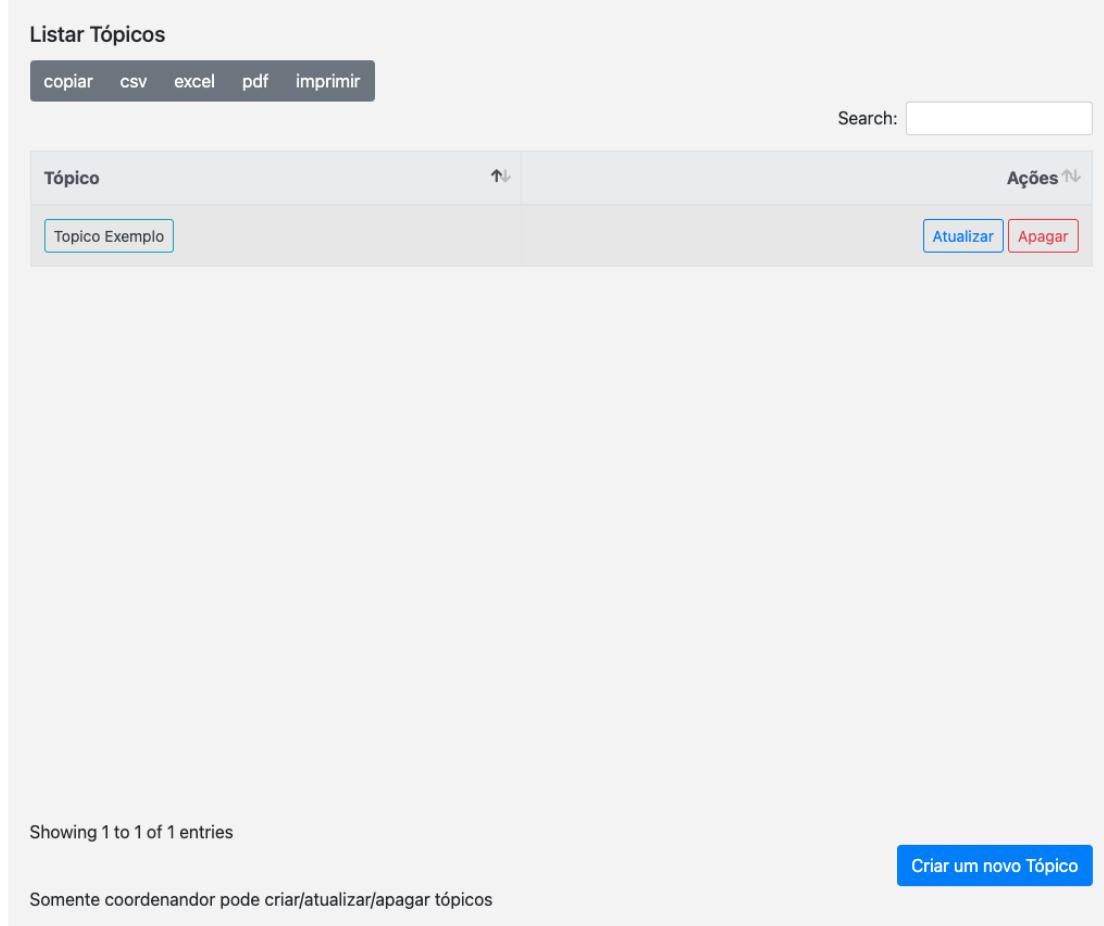
The screenshot shows a web-based application interface for managing disciplines and topics. At the top, there's a header with the title 'Detalhar Disciplina' and some descriptive text about the discipline ('Disciplina: Disciplina Exemplo', 'Código: DE', 'Objetivo:'). Below this, there's a section for 'Profs' and 'Coordenador'. The main area is titled 'Listar Tópicos' and contains a table with one entry: 'Topico Exemplo'. There are buttons for 'copiar', 'csv', 'excel', 'pdf', and 'imprimir' at the top of the table, and a search bar above it. To the right of the table, there are 'Atualizar' and 'Apagar' buttons. At the bottom, there's a message stating 'Somente coordenandor pode criar/atualizar/apagar tópicos' and a blue button labeled 'Criar um novo Tópico'. A footer at the bottom of the page says 'Showing 1 to 1 of 1 entries'.

Figura 2.14: (Parte 1) Tela apresentando os detalhes de uma disciplina.

2.3 Considerações finais

Neste capítulo, foi apresentada uma visão geral do MCTest, enfatizando sua estrutura e funcionalidades essenciais. Foi discutida a navegação geral do sistema, destacando como os usuários podem acessar as diferentes entidades disponíveis, como institutos, cursos, disciplinas, turmas e tópicos.

Os diferentes tipos de usuários com acesso ao MCTest foram apresentados, incluindo administrador, coordenador e professor, juntamente com suas respectivas permissões e responsabilidades específicas.

O administrador foi identificado como o usuário com maior privilégio, possuindo acesso irrestrito a todas as funcionalidades e dados do sistema. Sua responsabilidade abrange a configuração

Listar Turmas						
copiar csv excel pdf imprimir colunas visíveis ▾						
Código	↑↓	Período	↑↓	Profs	↑↓	Ações
Turma Exemplo		2023.1		* fzampirolli@ufabc.edu.br		Atualizar Apagar

Showing 1 to 1 of 1 entries

Somente coordenador pode criar/atualizar/apagar tópicos

[Criar uma nova Turma](#)

Figura 2.15: (Parte 2) Continuação da tela apresentando os detalhes de uma disciplina.

do sistema, criação e gerenciamento de outros usuários, bem como a definição de parâmetros gerais do MCTest.

O coordenador, por sua vez, possui um conjunto de ferramentas específicas para gerenciar a disciplina, permitindo a criação e manutenção de tópicos relevantes para o andamento das aulas, bem como a criação de turmas e seus respectivos estudantes. É importante ressaltar que o acesso às configurações gerais do sistema é restrito ao administrador. Além disso, o coordenador pode cadastrar outros coordenadores e professores para colaborarem na gestão da disciplina. Uma funcionalidade interessante disponível ao coordenador é a criação de turmas por meio da importação de arquivos CSV, prática que favorece a organização e evita possíveis erros durante a inserção manual de dados.

O professor tem acesso mais restrito em comparação com o coordenador e o administrador. Sua responsabilidade abrange a criação e manutenção dos exames, turmas e estudantes atribuídos a ele, assim como a elaboração e utilização de questões da disciplina. No entanto, ele não possui permissão para cadastrar ou editar disciplinas, ou tópicos, reservadas ao coordenador e ao administrador.

2.3. CONSIDERAÇÕES FINAIS

Listar Turmas

Código da Turma ↑↓	Disciplina ↑↓	Período ↑↓	Profs ↑↓	Ações ↑↓
Turma Exemplo	IE - CE - Disciplina Exemplo	2023.1	* fzampirolli@ufabc.edu.br	Atualizar Apagar

Showing 1 to 1 of 1 entries

[Criar uma nova Turma](#)

Somente professores cadastrados em disciplinas
Entre em contato com o coordenador da disciplina

Figura 2.16: Tela que apresenta a lista de turmas, na qual um professor de uma disciplina pode criar uma nova turma, além de atualizar ou apagar turmas existentes.

Os detalhes específicos de cada funcionalidade atribuída a esses usuários serão abordados em capítulos futuros, fornecendo informações detalhadas sobre as ferramentas disponíveis para cada perfil, bem como as ações que podem ser realizadas no sistema.

As entidades questão e exame desempenham papéis fundamentais no MCTest, e suas descrições detalhadas serão realizadas nas Partes II e III, respectivamente.

A elaboração de questões parametrizadas representa o aspecto mais desafiador do sistema, exigindo criatividade, conhecimento da sintaxe L^AT_EX e domínio da linguagem Python. No entanto, após a criação bem-sucedida dessas questões, elas podem ser reutilizadas em diversos exames, desde que adequadamente parametrizadas. Essa característica simplifica a criação de novos exames.

A entidade exame, por sua vez, é considerada o elemento central do sistema, uma vez que todo o processo avaliativo é construído em torno dela. Os exames representam a ferramenta por meio da

Criar uma nova Turma

Escolher Disciplinas

IE - CE - Disciplina Exemplo ▾

Escolher Prof(s)

fzampirolli@ufabc.edu.br
fzcoord@ufabc.edu.br
fzprof@ufabc.edu.br

Se você não é o coordenador da disciplina, para incluir os alunos deve-se escolher pelo menos um professor da turma
O professor que criar uma turma e NÃO marcar o e-mail acima NÃO poderá adicionar alunos à turma
Buscar com CTRL+F - Segure CTRL para marcar mais de um professor

Código Código Turma

Sala Sala da Turma

Período (Formato: 2023.2) 2023.2

Tipo

Practical Class ▾

Voltar Salvar

Importar estudantes por arquivo csv em Minhas Turmas
Somente professores cadastrados em disciplinas
Entre em contato com o coordenador da disciplina

Figura 2.17: Tela utilizada pelo professor para criar uma nova turma após clicar no botão “Criar uma nova Turma”, na tela da Figura 2.16. É importante destacar que o nome da turma não pode conter acentos ou caracteres especiais, pois esse nome fará parte do cabeçalho do exame.

qual os estudantes são avaliados e os resultados obtidos, tornando essa entidade indispensável para o êxito do MCTest como um sistema de avaliação educacional.

2.3. CONSIDERAÇÕES FINAIS

Atualizar Turma

Nenhum arquivo escolhido

[Ver exemplos \(click com botão direito do mouse e salve o arquivo em seu computador, antes de fazer upload\)](#)

Sintaxe do arquivo CSV - Somente formato UTF-8 - Número máximo de caracteres para estudante é 45:
123, joão, joao@gmail.com
987, maria, maria@gmail.com

Se necessário, use este conversor de acentos: [conversor](#)

Após importar o arquivo CSV, verificar abaixo se os estudantes estão marcados em cinza

<input type="button" value="Escolher Disciplinas"/> IE - CE - Disciplina Exemplo ▾	<input type="button" value="Estudantes"/> 123; joão silva; joao@gmail.com 987; maria campos; maria@gmail.com
<input type="button" value="Profs"/> fzampirolli@ufabc.edu.br fzcoord@ufabc.edu.br fzprof@ufabc.edu.br	

Se você não é o coordenador da disciplina, para incluir os alunos deve-se escolher pelo menos um professor da turma
Buscar com CTRL+F - Segure CTRL para marcar mais de um professor

Turma Exemplo

TE

2023.1

Practical Class ▾

Somente professores cadastrados em disciplinas ou coordenador
Entre em contato com o coordenador da disciplina

Figura 2.18: Tela para um professor de uma disciplina poder atualizar uma turma, inserindo estudantes com a importação de um arquivo no formato CSV.

Detalhar Turma

Turma: CE-test
Tipo: PClass
Profs: fzampirolli@ufabc.edu.br;

Listar Estudantes

ID	↑↓	Estudante	↑↓	Email	↑↓	Ações ↑↓
444		aluno teste 4		fzampirolli@gmail.com		Atualizar Apagar
1002		Aluno Zampirolli		lists.zampirolli@gmail.com		Atualizar Apagar

Showing 1 to 2 of 2 entries

ID Nome Email Criar um novo Estudante

Atualizar Turma com CSV

Atenção, um estudante pode ter muitas turmas
Um estudante será removido da turma
Importar estudantes por arquivo CSV em Minhas Turmas
Somente professores cadastrados em disciplinas
Entre em contato com o coordenador da disciplina

Figura 2.19: Tela para um professor de uma disciplina poder atualizar uma turma, atualizando ou apagando estudantes da turma. Além disso, é possível incluir um novo estudante na turma.

Atualizar Estudante

Código	444
Nome	aluno teste 4
Email	fzampirolli@gmail.com

Salvar

Figura 2.20: Tela para um professor de uma disciplina poder atualizar os dados de um estudante da turma.

2.3. CONSIDERAÇÕES FINAIS

Listar Tópicos

Listar Tópicos		
copiar csv excel pdf imprimir colunas visíveis ▾		
Tópico	Disciplina	Ações
Topico Exemplo	IE - CE - Disciplina Exemplo	Atualizar Apagar
Showing 1 to 1 of 1 entries		
Criar um novo Tópico		

Tópicos das disciplinas que eu participo
Somente coordenador pode criar/atualizar tópicos

Figura 2.21: Tela com a lista de tópicos para o coordenador (ou administrador) criar, atualizar ou apagar um tópico de disciplina(s).

Atualizar Tópico

Escolher Disciplinas

IE - CE - Disciplina Exemplo

Tópico Topico Exemplo

Descrição

Voltar Salvar

Importante: usar nome curto que represente o tópico da disciplina

Somente coordenador cadastrado em disciplinas
Entre em contato com o coordenador da disciplina

Figura 2.22: Tela para o coordenador (ou administrador) atualizar um tópico de uma disciplina.

2.3. CONSIDERAÇÕES FINAIS

Detalhar Tópico

Tópico: Topico Exemplo

Descrição:

Criar-PDF

Ver todas as questões deste tópico em formato PDF, classificadas por similaridade

Listar Questões								
copiar csv excel pdf imprimir colunas visíveis ▾								
Tópico	↑↓	Tipo ↑↓	Grupo ↑↓	Dif. ↑↓	Par. ↑↓	ID ↑↓	Descrição curta ↑↓	Ações ↑↓
[DE]<Topico Exemplo>	QM 5	□	1	no	2443	QE1	Atualizar Apagar	
[DE]<Topico Exemplo>	QM 5	□	1	no	2444	QE2	Atualizar Apagar	
[DE]<Topico Exemplo>	QM 5	□	1	no	2445	QE3	Atualizar Apagar	

Showing 1 to 3 of 3 entries

Importante: para cada exame, somente uma questão é escolhida de cada grupo
Somente professores cadastrados em curso pode ver mais detalhes

Figura 2.23: Tela apresentando os detalhes de um tópico.

MCTest

Topic: Topico Exemplo

Count: 1

Short Description: QE2

Group:

Type: QM

Difficulty: 1

Bloom taxonomy: remember

Last update: 2023-04-17

Who created: fzampirolli@ufabc.edu.br

URL: <http://127.0.0.1:8000/topic/question/2444/update/>

Parametric: NO

#2444 1. Segunda Questao

A._{•2}a alternativa criada B._{•4}5a alternativa criada C._{•3}4a alternativa criada D._{•#01}a alternativa criada E._{•3}a alternativa criada

Count: 2

Short Description: QE1

Group:

Type: QM

Difficulty: 1

Bloom taxonomy: remember

Last update: 2023-04-17

Who created: fzampirolli@ufabc.edu.br

URL: <http://127.0.0.1:8000/topic/question/2443/update/>

Parametric: NO

#2443 1. Primeira Questao - Sempre a primeira alternativa criada eh a correta.

A._{•2}a alternativa criada B._{•#01}a alternativa criada C._{•3}4a alternativa criada D._{•4}5a alternativa criada E._{•3}a alternativa criada

Count: 3

Short Description: QE3

Group:

Type: QM

Difficulty: 1

Bloom taxonomy: remember

Last update: 2023-04-17

Who created: fzampirolli@ufabc.edu.br

URL: <http://127.0.0.1:8000/topic/question/2445/update/>

Parametric: NO

#2445 1. Terceira questao

A._{•3}a alternativa criada B._{•3}4a alternativa criada C._{•4}5a alternativa criada D._{•#01}a alternativa criada E._{•2}a alternativa criada

Figura 2.24: Tela apresentando as três primeiras questões de um tópico.

Capítulo 3

Recursos avançados

Conteúdo

3.1 Administrador	40
3.1.1 Criar um curso relacionado a dois institutos	40
3.1.2 Criar uma disciplina relacionada a dois cursos	40
3.1.3 Acesso ao banco de dados	43
3.2 Coordenador	43
3.2.1 Criar um tópico relacionado a duas ou mais disciplinas	43
3.2.2 Criar várias turmas com arquivo CSV	47
3.3 Professor	49
3.3.1 Criar turma com arquivo CSV	50
3.3.2 Criar turma com arquivo CSV – restrições	51
3.4 Considerações finais	52

Este capítulo fornece uma visão mais detalhada do sistema MCTest, apresentado no capítulo anterior, destacando as funcionalidades específicas disponíveis para os três tipos de usuários: administrador, coordenador e professor. Cada um desses usuários tem acesso a um conjunto específico de funcionalidades para criar e gerenciar as diversas entidades do sistema, incluindo instituto, curso, disciplina, tópico e turma.

É relevante enfatizar que as entidades questão e exame serão discutidas em capítulos distintos nas Partes II e III deste livro, respectivamente, devido à sua significativa importância no processo de avaliação. Nesta seção, serão abordadas as funcionalidades mais avançadas do MCTest disponíveis para cada tipo de usuário, visando facilitar a utilização efetiva do sistema.

Para fornecer exemplos práticos, serão criadas novas entidades no banco de dados disponibilizado no [GitHub](#), apresentado na seção anterior. Essas entidades serão criadas de forma gradual e explicativa ao longo deste capítulo.

Cabe destacar que os usuários têm acesso somente às funcionalidades pertinentes ao seu papel no sistema. Por exemplo, um professor pode optar por estudar somente a Seção 3.3 – Professor deste capítulo para obter informações relevantes para as suas responsabilidades no MCTest.

3.1 Administrador

Nesta seção, serão discutidas outras funcionalidades destinadas ao administrador do sistema. O papel do administrador consiste em configurar o sistema conforme as necessidades da instituição, criar institutos, cursos e disciplinas necessários e disponibilizar as disciplinas aos coordenadores para poderem utilizá-las no sistema.

3.1.1 Criar um curso relacionado a dois institutos

No capítulo anterior, foi abordada a navegação individual de cada usuário na plataforma MCTest. Especificamente, nas Seções 2.2.1 – Instituto e 2.2.2 – Curso, foi explorado como o administrador pode criar institutos e cursos, respectivamente. Em situações em que um curso é oferecido por dois institutos distintos, pode ser vantajoso configurar comportamentos específicos para cada um deles. Para ilustrar essa ideia, foi considerado hipoteticamente um curso de Engenharia da Computação na UFABC, dividido entre o Centro de Matemática, Computação e Cognição (CMCC) e o Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas (CECS). Nesse contexto, seria interessante configurar o MCTest para permitir um acesso mais personalizado a cada instituto. Uma possível configuração para essa situação pode ser visualizada na Figura 3.1, com a criação dos institutos CECS e CMCC.

Após o administrador clicar em “Criar um novo Curso” na Figura 3.2, será exibida a tela de criação de um novo curso, apresentada na Figura 3.3. Para associar o curso a dois institutos distintos, basta manter o botão “Ctrl” pressionado e clicar nos institutos desejados. A tela de criação de um novo curso é semelhante à tela de atualização de um curso já existente. Observe na Figura 3.4 a lista de cursos vista pelo administrador após ter criado o curso de Engenharia da Computação na Figura 3.3. Na coluna “Instituto”, o curso de Engenharia da Computação está associado aos centros CECS e CMCC.

3.1.2 Criar uma disciplina relacionada a dois cursos

Nas Seções 2.2.2 – Curso e 2.2.3 – Disciplina, foram abordados os procedimentos para o administrador poder criar cursos e disciplinas, respectivamente. Para ilustrar a relevância da criação de uma disciplina relacionada a dois cursos, será apresentado o seguinte cenário.

A disciplina de Processamento Digital de Imagens (PDI) é uma área da Ciência da Computação que se dedica ao estudo e desenvolvimento de técnicas e algoritmos para manipulação, análise e PDI. Essa disciplina é muito importante tanto para estudantes de graduação em Ciência da Computação quanto para estudantes de mestrado e doutorado, por oferecer uma base sólida de conhecimentos e habilidades para a solução de problemas em diversas áreas de aplicação.

3.1. ADMINISTRADOR

Listar Institutos

Listar Institutos					
Copiar CSV Excel PDF Imprimir Colunas Visíveis					
Código	Nome	URL	Ações		
CECS	Centro de Engenharia, Modelagem e Ciências Sociais	cecs.ufabc.edu.br	Atualizar	Apagar	
CMCC	Centro de Matemática, Computação e Cognição	cmcc.ufabc.edu.br	Atualizar	Apagar	
IE	Instituto Exemplo	www.ufabc.edu.br	Atualizar	Apagar	

Showing 1 to 3 of 3 entries

[Criar um novo Instituto](#)

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas

Figura 3.1: Tela com a lista de institutos vista pelo administrador, que pode atualizar, apagar ou criar um instituto para simular um curso pertencer aos institutos CECS e CMCC.

No curso de graduação em Ciência da Computação, a disciplina de PDI aborda diversas técnicas relevantes para o processamento e análise de imagens digitais. Entre elas, destaca-se a representação de imagens digitais, que envolve o uso de matrizes para representar as cores dos pixels, e técnicas de filtragem, que permitem a melhoria da qualidade da imagem, através da redução de ruídos e realce de contornos de objetos.

Além disso, a disciplina de PDI também aborda técnicas de segmentação de imagens, que permitem a identificação de regiões de interesse na imagem, e técnicas de reconhecimento de padrões em imagens, que permitem a identificação de objetos, faces e outros elementos na imagem.

Durante o curso de mestrado ou doutorado em Ciência da Computação, a disciplina de PDI explora técnicas avançadas, incluindo o uso de Redes Neurais Convolucionais (CNN – do inglês *Convolutional Neural Network*). Essas técnicas têm se mostrado altamente eficazes para a segmentação, reconhecimento e classificação de imagens.

Listar Cursos

Listar Cursos					
copiar csv excel pdf imprimir colunas visíveis ▾					
Código	↑↓	Nome	↑↓	Instituto	↑↓
CE		Curso Exemplo		IE	
Atualizar Apagar					

Showing 1 to 1 of 1 entries

[Criar um novo Curso](#)

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas

Figura 3.2: Tela com a lista de curso, vista pelo administrador.

As CNN são uma classe de Redes Neurais Artificiais que têm sido amplamente utilizadas em tarefas de PDI, como a segmentação de objetos em imagens médicas ou a classificação de imagens em categorias específicas. Essas redes podem extrair características das imagens de forma automática, sem a necessidade de recursos humanos para a extração manual de características.

Com a utilização de CNN, é possível obter excelentes resultados na segmentação, reconhecimento e classificação de imagens. Por exemplo, na área médica, é possível identificar automaticamente regiões de interesse em imagens de ressonância magnética ou tomografia computadorizada, auxiliando no diagnóstico de doenças.

Portanto, a disciplina de PDI é de grande importância tanto para os estudantes de graduação quanto para os estudantes de mestrado e doutorado em Ciência da Computação. Além disso, as questões relacionadas a essa disciplina podem ser compartilhadas entre esses dois cursos.

Dessa forma, é viável adicionar a disciplina de PDI no MCTest e relacioná-la com os cursos de graduação e pós-graduação, conforme ilustrado na Figura 3.5. Esse processo é semelhante ao de

3.2. COORDENADOR

Criar um novo Curso

Escolha Instituto(s)

Centro de Engenharia, Modelagem e Ciências Sociais
Centro de Matemática, Computação e Cognição
Instituto Exemplo

Nome Engenharia da Computação

Código EC

Voltar Salvar

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas

Figura 3.3: Tela para o administrador criar um curso novo relacionado a dois institutos. Manter a tecla “Ctrl” pressionada para escolher mais de uma opção.

associar um curso a dois institutos.

3.1.3 Acesso ao banco de dados

Embora o administrador tenha acesso direto ao banco de dados (BD) ao pressionar o botão “Admin” na Figura 2.2-(e), a melhor opção é evitar alterar o BD usando esse recurso. É recomendado utilizar as telas de navegação do MCTest para fazer as alterações necessárias, pois alterar o BD diretamente pode resultar em relacionamentos incorretos entre as entidades, o que pode tornar o sistema inoperante.

3.2 Coordenador

Nesta seção, serão abordadas as funcionalidades destinadas ao coordenador de disciplina, que envolvem a manutenção de tópicos e disciplinas. O coordenador tem a responsabilidade de criar e gerenciar os tópicos e disciplinas que serão utilizados pelos professores para criar exames e Exercícios de Programação (EPs).

3.2.1 Criar um tópico relacionado a duas ou mais disciplinas

A interdisciplinaridade é uma abordagem essencial na educação moderna, que visa integrar conhecimentos de diferentes áreas para enriquecer o processo de aprendizagem e formação dos estudantes. Nesse sentido, a UFABC se destaca na interdisciplinaridade, conforme apresentado em sua missão:

Listar Cursos

Código	Nome	Instituto	Ações
CE	Curso Exemplo	IE	<button>Atualizar</button> <button>Apagar</button>
EC	Engenharia da Computação	CECS CMCC	<button>Atualizar</button> <button>Apagar</button>

Showing 1 to 2 of 2 entries

[Criar um novo Curso](#)

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas

Figura 3.4: Tela com a lista de curso, vista pelo administrador, após ter criado o curso Engenharia da Computação na Figura 3.3.

Missão da UFABC:

Promover o avanço do conhecimento por ações de ensino, pesquisa e extensão, tendo como fundamentos básicos a interdisciplinaridade, a excelência e a inclusão social.

Nesse sentido, compartilhar um tópico entre duas ou mais disciplinas pode trazer inúmeros benefícios para a educação. Como mencionado na Seção 2.2.5 – Tópico, um tópico só existe no MCTest se pertencer a uma disciplina, mas pode ser compartilhado entre várias delas. Além disso, as questões só existem se estiverem inseridas em um tópico.

Um exemplo prático dessa abordagem pode ser observado na UFABC, na qual a lógica de programação é abordada em diversas disciplinas. Especificamente, esse tópico é tratado no Bacha-

3.2. COORDENADOR

Listar Disciplinas			
copiar csv excel pdf imprimir colunas visíveis ▾			
Código	↑↓	Nome	↑↓
DE		Disciplina Exemplo	CE
PDI		Processamento Digital de Imagens	BCC POSCOMP

Showing 1 to 2 of 2 entries

[Criar uma nova Disciplina](#)

Somente coordenador pode atualizar disciplina

Figura 3.5: Tela com a lista de disciplinas, vista pelo administrador, com destaque para a disciplina PDI, que está relacionada aos cursos de BCC e POSCOMP.

relado em Ciência e Tecnologia (BCT), no segundo exame da disciplina de Bases Computacionais da Ciência (CS0) para os ingressantes e no primeiro exame de Processamento da Informação (PI – CS1) no terceiro quadrimestre dos ingressantes. Atualmente, a maioria das turmas dessas duas disciplinas utiliza a linguagem Python. Além disso, a disciplina de Programação Estruturada (PE – CS2), oferecida no Bacharelado em Ciência da Computação, também trata da lógica de programação no primeiro exame, utilizando a linguagem C.

Ao elaborar questões sobre lógica de programação de forma independente da linguagem utilizada, é possível compartilhar essas questões entre as três disciplinas mencionadas. Essa prática tem várias vantagens:

Otimização de recursos: Ao compartilhar questões e tópicos, é possível economizar tempo e esforço dos professores na criação de materiais didáticos, permitindo que se concentrem em outras tarefas importantes, como atendimento aos estudantes e pesquisas de novas metodologias de

The screenshot shows the Django Admin interface with the following sections:

- ACCOUNT** (blue header):
 - Usuários**: + Adicionar | Modificar
- AUTENTICAÇÃO E AUTORIZAÇÃO** (blue header):
 - Grupos**: + Adicionar | Modificar
- COURSE** (blue header):
 - Classrooms**: + Adicionar | Modificar
 - Courses**: + Adicionar | Modificar
 - Disciplines**: + Adicionar | Modificar
 - Institutes**: + Adicionar | Modificar
- EXAM** (blue header):
 - Classroom exams**: + Adicionar | Modificar
 - Exams**: + Adicionar | Modificar
 - Student exam questions**: + Adicionar | Modificar
 - Student exams**: + Adicionar | Modificar
 - Variation exams**: + Adicionar | Modificar
- STUDENT** (blue header):
 - Students**: + Adicionar | Modificar
- TOPIC** (blue header):
 - Answers**: + Adicionar | Modificar
 - Questions**: + Adicionar | Modificar
 - Topics**: + Adicionar | Modificar

Figura 3.6: Tela com a lista de entidades que o administrador pode fazer manutenção diretamente no BD.

ensino;

Consistência no conteúdo: Ao utilizar tópicos comuns entre diferentes disciplinas, os estudantes são expostos a uma abordagem consistente e coerente, facilitando a compreensão e a conexão

3.2. COORDENADOR

entre os conceitos aprendidos;

Desenvolvimento de habilidades transferíveis: Ao aprender sobre um tópico em várias disciplinas, os estudantes têm a oportunidade de desenvolver habilidades transferíveis, como pensamento crítico, resolução de problemas e comunicação. Essas habilidades são essenciais para o sucesso em diversas áreas profissionais;

Fomento da interdisciplinaridade: Compartilhar tópicos entre disciplinas incentiva os estudantes a estabelecer conexões entre diferentes campos do conhecimento, promovendo uma educação mais integrada e abrangente.

A Figura 3.7 exemplifica uma lista de tópicos, demonstrando que um determinado tópico pode pertencer a três disciplinas diferentes. Adicionalmente, a Figura 3.8 apresenta detalhes específicos da disciplina de Processamento da Informação, com destaque para os tópicos compartilhados. Essa configuração permite que um professor utilize todas as questões desses tópicos ao elaborar um exame.

3.2.2 Criar várias turmas com arquivo CSV

No capítulo anterior, foi abordada a criação de disciplinas na Seção 2.2.3 – [Disciplina](#), e a criação de turmas com estudantes foi apresentada na Seção 2.2.4 – [Turma](#). No entanto, em algumas situações, pode ser necessário criar várias turmas da mesma disciplina simultaneamente. Para atender a essa demanda, foram implementadas funcionalidades que permitem ao coordenador cadastrar todos os professores e estudantes de múltiplas turmas de uma só vez, por meio da importação de arquivos no formato CSV, conforme ilustrado na Figura 3.10.

Melhorias:

É importante ressaltar que esse recurso precisa ser aprimorado e, por enquanto, o botão “Upload-Turma” irá remover todos os estudantes de todas as turmas associadas a essa disciplina, atualizando com os novos dados do arquivo CSV. Portanto, é fundamental ter cuidado ao utilizar essa funcionalidade e garantir que os dados do arquivo CSV estejam corretos e atualizados.

As telas para criação e atualização de disciplinas apresentadas nas Figuras 3.9 e 3.10 são semelhantes e estão disponíveis tanto para o administrador quanto para o coordenador da disciplina. Na Figura 3.10, o primeiro arquivo CSV é destinado ao coordenador para inserir diversos professores na disciplina, com o perfil de professor já configurado, como apresentado a seguir:

Arquivo CSV com dados dos professores:

João, da Silva, joao@ufabc.edu.br
Maria, Gonçalves, maria@ufabc.edu.br

Listar Tópicos

Listar Tópicos		
Tópico	Disciplina	Ações
01-Lógica de Programação: Condicional	CMCC - BCC - Programação Estruturada UFABC - BCT - Bases Computacionais da Ciência UFABC - BCT - Processamento da Informação	Atualizar Apagar
01-Lógica de Programação: Repetição	CMCC - BCC - Programação Estruturada UFABC - BCT - Bases Computacionais da Ciência UFABC - BCT - Processamento da Informação	Atualizar Apagar
01-Lógica de Programação: Sequencial	CMCC - BCC - Programação Estruturada UFABC - BCT - Bases Computacionais da Ciência UFABC - BCT - Processamento da Informação	Atualizar Apagar
Topico Exemplo	IE - CE - Disciplina Exemplo	Atualizar Apagar

Showing 1 to 4 of 4 entries

[Criar um novo Tópico](#)

Tópicos das disciplinas que eu participo
Somente coordenador pode criar/atualizar tópicos

Figura 3.7: Tela com a lista de tópicos, vista pelo coordenador. É possível observar o compartilhamento de tópicos entre disciplinas.

Por sua vez, o segundo arquivo CSV nesta figura é utilizado pelo coordenador para adicionar várias turmas, contendo estudantes e professores, com um formato pré-definido como segue:

Arquivo CSV com dados de estudantes e professores:

```
123, João Silva, js@gmail.com, DA1, sala1, PClass, fzprof@ufabc.edu.br
987, Maria Campos, mc@gmail.com, DA2, sala2, TClass, fzprof@ufabc.edu.br
```

Este último arquivo CSV apresenta as seguintes colunas, em ordem: identificação do estudante, nome completo do estudante, e-mail do estudante, turma, sala, “PClass” (para turma prática) ou “TClass” (para turma teórica) e e-mail do professor.

3.3. PROFESSOR

Detalhar Disciplina

Disciplina: Processamento da Informação

Código: PI

Objetivo:

Ementa: Introdução a algoritmos. Variáveis e tipos de dados. Operadores aritméticos, lógicos e precedência. Métodos/Funções e parâmetros. Estruturas de seleção. Estruturas de repetição. Vetores. Matrizes. Entrada e saída de dados. Depuração. Melhores práticas de programação.

Profs: fzampirolli; fzprof;

Coordenador: fzampirolli; fzcoord;

Listar Tópicos

[copiar](#) [csv](#) [excel](#) [pdf](#) [imprimir](#)

Search:

Tópico	Ações
01-Lógica de Programação: Condicional	Atualizar Apagar
01-Lógica de Programação: Repetição	Atualizar Apagar
01-Lógica de Programação: Sequencial	Atualizar Apagar

Showing 1 to 3 of 3 entries

[Criar um novo Tópico](#)

Somente coordenador pode criar/atualizar/apagar tópicos

Figura 3.8: Tela com detalhes da disciplina de Processamento da Informação, vista pelo coordenador, com os tópicos compartilhados também apresentados na Figura 3.7.

3.3 Professor

Nesta seção será abordada mais funcionalidades destinadas aos professores, que incluem a manutenção de turmas, questões e exames. O professor é responsável por criar as turmas, selecionar as questões e criar os exames para avaliar os estudantes. Com o sistema MCTest, os professores podem criar exames com questões de múltipla escolha (QMs) ou dissertativas (QTs), incluindo EPs

Atualizar Disciplina

The screenshot shows the 'Atualizar Disciplina' (Update Discipline) page in Moodle. The discipline 'Processamento da Informação' is selected. The 'Nome' (Name) field contains 'Processamento da Informação'. The 'Código' (Code) field contains 'PI'. The 'Objetivo' (Objective) field contains the text: 'Ementa: Introdução a algoritmos. Variáveis e tipos de dados. Operadores aritméticos, lógicos e precedência. Métodos/Funções e parâmetros. Estruturas de seleção. Estruturas de repetição. Vetores. Matrizes. Entrada e saída de dados. Depuração. Melhores práticas de programação.' The 'Profs' (Teachers) section lists four email addresses: fzampirolli@ufabc.edu.br, fzstudent@ufabc.edu.br, fzprof@ufabc.edu.br, and fzcoord@ufabc.edu.br. The 'Coord' (Coordinator) section also lists the same four email addresses. At the bottom right are 'Voltar' (Back) and 'Salvar' (Save) buttons.

Figura 3.9: (Parte 1) Tela de criação da disciplina de Processamento da Informação pelo administrador.

parametrizados para correção automática no Moodle, utilizando o *plugin VPL*.

3.3.1 Criar turma com arquivo CSV

O professor é responsável por criar uma turma e manter os dados de seus estudantes atualizados, como discutido na Seção 2.2.4 – Turma. O primeiro passo é criar a turma e relacioná-la a uma disciplina, conforme ilustrado na Figura 2.17. Em seguida, o professor pode inserir os estudantes de três maneiras distintas:

1. Selecionando vários estudantes na lista com a tecla “Ctrl” pressionada, como demonstrado na Figura 2.18;
2. Incluindo um estudante de cada vez, conforme exemplificado na Figura 2.19 e/ou atualizando os dados de um estudante, como demonstrado na Figura 2.20;
3. Ou, de maneira mais eficiente, utilizando um arquivo CSV, vistos a seguir.

Para utilizar um arquivo CSV para preencher uma turma com estudantes, o professor deve primeiro criar a turma, como ilustrado na Figura 2.17, e em seguida fazer o *upload* do arquivo no

3.3. PROFESSOR

Incluir vários professores em um arquivo CSV

Nenhum arquivo escolhido

[Ver exemplos \(click com botão direito do mouse e salve o arquivo em seu computador, antes de fazer upload\)](#)

Sintaxe do arquivo CSV - Somente formato UTF-8 - Número máximo de caracteres para estudante é 45:

João, Silva, joao@ufabc.edu.br

Maria, Campos, maria@ufabc.edu.br

Se necessário, use este conversor de acentos: [conversor](#)

Após importar o arquivo CSV, verificar abaixo se os professores estão marcados em cinza

Criar várias turmas de um arquivo CSV

Nenhum arquivo escolhido

ATENÇÃO: este esquema irá remover todos os estudantes existentes em *TODAS AS TURMAS* desta disciplina

[Ver exemplos \(click com botão direito do mouse e salve o arquivo em seu computador, antes de fazer upload\)](#)

Sintaxe do arquivo CSV - Somente formato UTF-8 - Número máximo de caracteres para estudante é 45:

1234, João Silva, joao@ufabc.edu.br, 08h-SA-DA1, sala1, PClass, fzprof@ufabc.edu.br

4321, Maria Campos, maria@ufabc.edu.br, 10h-SB-DA2, sala2, TClass, fzprof@ufabc.edu.br

Se necessário, use este conversor de acentos: [conversor](#)

Somente admin pode criar/atualizar/apagar institutos/cursos/disciplinas

Somente coordenador pode atualizar disciplina

Figura 3.10: (Parte 2) Tela de criação da disciplina de Processamento da Informação pelo administrador. É possível incluir professores, turmas e estudantes pela importação de dois arquivos no formato CSV.

início da Figura 2.18, selecionando o arquivo em “Escolher arquivo” e, em seguida, clicando no botão “Importar-Estudantes”. O arquivo deve seguir o formato apresentado abaixo:

Arquivo CSV com dados completos:

123, João da Silva, joao@aluno.ufabc.edu.br

987, Maria Gonçalves, maria@aluno.ufabc.edu.br

Observe que a primeira linha do arquivo já representa o primeiro estudante, com sua identificação, nome completo e e-mail, separados por vírgula ou ponto e vírgula. É importante destacar que o e-mail é opcional, como no exemplo abaixo:

Arquivo CSV, sem e-mail:

123, João da Silva

987, Maria Gonçalves

3.3.2 Criar turma com arquivo CSV – restrições

O professor deve ter uma atenção especial aos acentos e símbolos especiais neste arquivo CSV, que deve seguir o formato UTF-8. Uma alternativa é converter esses símbolos para o formato L^AT_EX, utilizando, por exemplo, o recurso disponível na internet em w2.syrnex.com/jmr/latex-symbols-converter, conforme exemplo abaixo:

Arquivo CSV, com acentos no formato L^AT_EX:

```
123, João da Silva, joao@aluno.ufabc.edu.br
987, Maria Gonçalves, maria@aluno.ufabc.edu.br
```

Observações:

1. O nome do estudante no sistema tem um limite de 45 caracteres. Caso o arquivo CSV contenha um nome de estudante com mais de 45 caracteres, o sistema irá remover automaticamente o(s) sobrenome(s) do meio, de trás para frente, mantendo o último sobrenome, até atingir os 45 caracteres permitidos. Essa medida é tomada para garantir a integridade dos dados e evitar problemas de exceder o limite de caracteres ao incluir o nome do estudante em um exame. Um exemplo pode ser visto na Figura 3.11;
2. Caso o arquivo contenha um identificador já existente, o sistema não criará ou alterará o mesmo.

Mensagem

ESTUDANTES DA TURMA >> TURMA EXEMPLO

1; 123; João da Silva; joao@aluno.ufabc.edu.br
 ### BIG ###: Maria da Silva Andrade de Souza Santos do Campos Gonçalves ==> Maria da Silva Andrade de S Gonçalves
 2; 987; Maria da Silva Andrade de S Gonçalves; maria@aluno.ufabc.edu.br

[Voltar para detalhes da turma](#) [Link](#)

Figura 3.11: Tela mostrando o resultado do cadastro de estudantes utilizando a importação de um arquivo no formato CSV.

3.4 Considerações finais

Este capítulo expandiu as funcionalidades apresentadas no capítulo anterior, apresentando recursos avançados disponíveis no MCTest para usuários com diferentes papéis (administrador, coordenador e professor). Cada tipo de usuário tem acesso a um conjunto específico de ferramentas para criar e gerenciar as diversas entidades do sistema de avaliação, como institutos, cursos, disciplinas, tópicos, turmas, professores e estudantes.

As funcionalidades abordadas procuram facilitar o gerenciamento das entidades e a configuração do sistema conforme as necessidades de cada instituição. O administrador, em particular, tem permissões amplas para configurar o sistema e gerenciar o acesso dos usuários. Os coordenadores são responsáveis por gerenciar disciplinas, tópicos e turmas. Já os professores podem gerenciar questões, exames, turmas e estudantes sob sua responsabilidade.

Nos próximos capítulos da Parte II, serão abordados a criação e o gerenciamento de questões

3.4. CONSIDERAÇÕES FINAIS

no MCTest, com foco nas diversas formas de elaboração, revisão e aprovação das questões destinadas a serem aplicadas nos exames discutidos na Parte III.

Parte II

Questões no MCTest

Conteúdo

4 Questões estáticas	57
5 Questões paramétricas	71

Capítulo 4

Questões estáticas

Conteúdo

4.1	Tutoriais gerais sobre a navegação de questões	57
4.2	Questão de múltipla escolha (QM)	62
4.3	Questão dissertativa (QT)	66
4.4	Considerações finais	68

A criação de questões estáticas é fundamental para a avaliação educacional e existem diversas abordagens e formatos disponíveis. Neste capítulo, serão abordados os dois principais tipos de questões estáticas utilizados em avaliações educacionais: as questões de múltipla escolha (QMs) e as questões dissertativas ou de texto (QTs). O objetivo é apresentar as características e particularidades de cada tipo de questão, bem como as vantagens e desvantagens de sua utilização na avaliação da aprendizagem. É importante destacar que as questões apresentadas neste capítulo não são parametrizadas, sendo as únicas variações os sorteios das questões e alternativas, no caso das QMs. No próximo capítulo, serão abordadas as questões parametrizadas.

Neste capítulo, será discutida a visibilidade do professor no menu do MCTest, permitindo que ele faça a manutenção das questões, exames e turmas que criou. Além disso, o professor também pode visualizar e utilizar questões criadas por outros professores cadastrados nas mesmas disciplinas. É importante compreender como essa funcionalidade pode ser útil aos professores, proporcionando maior uniformidade e controle sobre as avaliações.

4.1 Tutoriais gerais sobre a navegação de questões

Na Figura 4.1, é apresentada a lista de questões que o usuário “fzprof” pode visualizar ao clicar no botão “Questões” à esquerda do menu superior. Esse menu exibe todas as questões criadas por todos os professores cadastrados nas mesmas disciplinas de “fzprof”.

A Figura 4.1 exibe a lista de questões, apresentando diversos atributos (colunas), como o “Tópico” da disciplina, o “Tipo” de questão (podendo ser QM ou QT), o número de alternativas (no caso do tipo QM, que neste exemplo é cinco), o “Grupo” da questão (para evitar que sejam sorteadas duas ou mais questões do mesmo grupo em um exame), a dificuldade variando de 1 a 5, se é paramétrica, a identificação da questão no banco de dados (“ID”), a “Descrição curta” (para facilitar o entendimento do que se trata a questão sem precisar abrir todo o enunciado) e, por fim, os botões de “Ações”, que permitem atualizar ou apagar a questão.

Tópico	Tipo	Grupo	Dif.	Par.	ID	Descrição curta	Ações
[DE]<Topico Exemplo>	QM 5		1	no	2443	QE1	<button>Atualizar</button> <button>Apagar</button>
[DE]<Topico Exemplo>	QM 5		1	no	2444	QE2	<button>Atualizar</button> <button>Apagar</button>
[DE]<Topico Exemplo>	QM 5		1	no	2445	QE3	<button>Atualizar</button> <button>Apagar</button>

Showing 1 to 3 of 3 entries

Questões das disciplinas em que participo

AGPLv3 Copyright © 2018-2023 por Francisco de Assis Zampirolli da UFABC e Colaboradores; Quem utiliza; Download; Manual; LEIA-ME.
Apoio: UFABC; #2009/14430-1 e #2018/23561-1 - FAPESP

Figura 4.1: Tela para o professor visualizar todas as questões de disciplinas que está cadastrado.

Caso o professor tente modificar uma questão que não é de sua autoria, será exibida a mensagem de erro apresentada na Figura 4.2.

4.1. TUTORIAIS GERAIS SOBRE A NAVEGAÇÃO DE QUESTÕES

Erro

- ERRO: Você não criou esta questão ou não é o coordenador do curso. Por favor, entre em contato com eles para alterar algo nesta questão.

Figura 4.2: Exibição de erro ao tentar modificar questão criada por outro autor.

Observações:

1. Embora a Figura 4.1 exiba os botões “Atualizar” e “Apagar”, é importante ressaltar que, caso o professor tente alterar ou apagar uma questão criada por outro professor, o sistema não permitirá a modificação ou exclusão da questão;
2. É interessante manter o botão “Atualizar” disponível, por permitir que qualquer professor da disciplina possa visualizar como a questão foi criada. Ao clicar nas colunas “Tipo” até “Descrição curta”, o professor pode conferir a questão em um modo diferente e simplificado, como mostrado na Figura 4.3. É importante ressaltar que essa não é a melhor forma de visualizar os detalhes da questão, mas o professor pode acessar o PDF completo de visualização da questão clicando no botão “Criar-PDF”.

Detalhar Questão

Criar-PDF

Voltar

Atualizar

Tópico: [DE]<Topico Exemplo>

Descrição curta: QE1

Grupo:

Descrição: Primeira Questao - Sempre a primeira alternativa criada eh a correta.

Tipo: QM

Quem criou: fzampirolli@ufabc.edu.br

Última atualização: 17 de Abril de 2023

Alternativa: 1a alternativa criada

Retorno:

Alternativa: 2a alternativa criada

Retorno:

Alternativa: 3a alternativa criada

Retorno:

Alternativa: 4a alternativa criada

Retorno:

Alternativa: 5a alternativa criada

Retorno:

Figura 4.3: Exibição da questão QE1 criada por outro autor.

Na Figura 4.4, é possível observar que o professor “fzprof” ainda não criou nenhuma questão. Para criar novas questões, o professor pode importar um arquivo no formato TXT, conforme exem-

plificado abaixo. Esse formato foi utilizado na versão 4 do MCTest, que executava no console de um computador e aceitava apenas três níveis de dificuldade: fácil (QE), médio (QM) e difícil (QH), representados em inglês por *easy*, *median* e *hard*, respectivamente. Já a versão web atual do MCTest aceita cinco níveis de dificuldade, de 1 a 5. Ao importar uma questão de um arquivo TXT na versão web, as questões com nível de dificuldade QE terão dificuldade 1, as QM terão dificuldade 3 e as QH terão dificuldade 5.

No exemplo a seguir, o tópico é “DE-matriz” e é necessário existir uma disciplina com esse tópico cadastrada no MCTest. Para evitar conflitos, é recomendável incluir algum código como prefixo do tópico, como o código da disciplina. Por exemplo, “DE-matriz” significa que o tópico matriz pertence à disciplina “Disciplina Exemplo” cadastrada no MCTest. Além disso, o campo “grupo” da questão é importante para evitar que duas questões do mesmo grupo sejam sorteadas em um mesmo exame. Isso será exemplificado em capítulos futuros ao explicar a elaboração de exames. Finalmente, as alternativas, se existirem, devem ser precedidas de “A.”, sendo a primeira alternativa sempre a correta. Ao visualizar uma questão, as alternativas serão sorteadas a cada vez que o PDF da questão for gerado. É possível criar várias QMs utilizando somente um arquivo TXT. Porém, se forem questões paramétricas, deve existir apenas uma questão por arquivo.

Arquivo CSV, com acentos no formato L^AT_EX:

```
QE::DE-matriz::grupo::  
Crie uma matriz $3 \times 5$ de inteiros, com elementos $(i, j) = i + j$,  
com índices começando em zero, imprima a soma dos elementos da matriz.  
A: 44 % alternativa correta - sempre a primeira  
A: 35  
A: 43  
A: 55  
A: 47
```

Na Seção 8.1 – Desenvolvendo QMs e uma QT para inclusão em um exame, serão apresentados mais exemplos de como criar questões utilizando um arquivo TXT, destinadas a serem utilizadas em um exame.

Melhorias:

Por enquanto, o botão “Upload-Questões-Json” apresentado na Figura 4.4 ainda não está funcionando. Essa funcionalidade será útil para um professor poder fazer *backup* de suas questões e recuperá-las no futuro, caso necessário.

Além de importar questões nos formatos TXT ou JSON, o MCTest oferece a opção de criar questões diretamente na plataforma. Para isso, o professor deve clicar no botão “Criar uma nova Questão”, conforme ilustrado na Figura 4.4. A ação abrirá a tela mostrada na Figura 4.5, na qual é possível criar uma nova questão. Cada questão deve estar associada a um único tópico, selecionado

4.1. TUTORIAIS GERAIS SOBRE A NAVEGAÇÃO DE QUESTÕES

Minha Lista de Questões

Não existe **Questão** registrada(o) ainda

[Criar uma nova Questão](#)

Importar questões de um arquivo (formato MCTest4) - Somente formato UTF-8

[Escolher arquivo](#) Nenhum arquivo escolhido [Upload-Questões](#) Escolher arquivo TXT para importar questões

Ver exemplos (click com botão direito do mouse e salve o arquivo em seu computador, antes de fazer upload)

Sintaxe do arquivo TXT - IMPORTANTE: CRIAR TÓPICOS NA DISCIPLINA EM QUE VOCÊ ESTÁ CADASTRADO ANTES DE FAZER UPLOAD, se não existir:

QE::matriz::grupo A:: % QE, QM or QH :: tópico :: grupo

Crie uma matriz 3 x 5 de inteiros, com elementos $(i, j) = i + j$, com índices começando em zero, imprima a soma dos elementos da matriz.

A: 44 % alternativa correta - sempre a primeira

A: 35

A: 43

A: 55

A: 47

Se necessário, use este conversor de acentos: [conversor](#)

Após importar o arquivo TXT, verificar se as novas questões foram criadas (questões com o mesmo enunciado não são inseridas no BD)

Importar questões de um arquivo (formato Json)

[Escolher arquivo](#) Nenhum arquivo escolhido [Upload-Questões-Json](#) Download uma questão antes para ver o formato Json

Questões das disciplinas em que participo

Entre em contato com o coordenador da disciplina

Figura 4.4: Tela para o professor visualizar suas questões e criar novas, incluindo importação de formato TXT.

na opção “Escolher Tópico”. Vale destacar que um mesmo tópico pode ser vinculado a várias disciplinas, permitindo o uso das questões em diferentes contextos. Em seguida, o professor deve preencher campos obrigatórios, como “Descrição curta”, “Grupo”, “Descrição”, e selecionar o “Tipo” da questão (QM ou QT), o nível de “Dificuldade” (de 1 para muito fácil a 5 para muito difícil) e a “Taxonomia de Bloom”. Por fim, é necessário indicar se a questão é paramétrica. A Figura 4.6 apresenta um exemplo de preenchimento dos campos para uma nova questão antes de clicar no botão “Salvar”.

Vale destacar que ainda serão necessários incluir novos experimentos para avaliar os atributos do formulário da questão. Por exemplo, é importante analisar as características de cada questão utilizando a Teoria de Resposta ao Item (TRI) (ZAMPIROLI, 2021; ZAMPIROLI, 2021). Adicionalmente, a inclusão da “Taxonomia de Bloom” foi motivada pelos artigos de referência de Calsavara, Serra, Zampirolli, Carvalho, Jonathan e Correia (2018) e Correia, Calsavara, Serra, Zampirolli e Jo-

nathan (2018), os quais podem ser consultados na Seção 8.6 – Exames adaptativos.

Criar uma nova Questão

Escolher Tópico	-----	▼
Descrição curta	Descrição curta	
Grupo	Somente uma questão por grupo será sorteada para cada exame (e para cada estudante)	
Aceita a descrição e parametrização do LaTeX usando a linguagem Python (consulte as publicações).		
Descrição		
Tipo Questões de Múltipla Escolha		
Dificuldade	Questão com nível muito fácil	
Taxonomia de Bloom	lembre-se: reconhecendo, lembrando	
Paramétrica	Não	
Voltar De Acordo Salvar		

Clique em Atualizar Questão em Minhas Questões para criar alternativas.

Questões das disciplinas em que participo

Entre em contato com o coordenador da disciplina

Figura 4.5: Tela para um professor criar uma nova questão.

4.2 Questão de múltipla escolha (QM)

Nesta seção, serão abordadas as QMs, após a revisão geral das questões apresentadas na seção anterior. A QM é um tipo de questão amplamente utilizado em avaliações educacionais. Ela consiste em apresentar ao estudante uma pergunta ou enunciado, seguido de diversas opções de resposta, variando geralmente de três a cinco alternativas, das quais apenas uma é a correta. O objetivo é verificar se o estudante possui o conhecimento necessário para identificar a resposta correta dentre as opções apresentadas. O MCTest também possibilita a existência de mais de uma alternativa correta, ou a atribuição de pesos diferentes em cada alternativa, mas esses tópicos serão abordados em capítulos futuros.

As QMs são frequentemente utilizadas em testes padronizados, exames de vestibular e concursos públicos, além de serem comuns em avaliações em sala de aula. Elas são consideradas uma forma eficiente e prática de avaliar o conhecimento dos estudantes, já que permitem avaliar inúmeras pessoas em pouco tempo e com baixo custo. No entanto, uma das desvantagens desse tipo de questão

4.2. QUESTÃO DE MÚLTIPLA ESCOLHA (QM)

Criar uma nova Questão

Escolher Tópico	[DE]<Topico Exemplo>	
Descrição curta	QE4	
Grupo	A	
Descrição	Quarta questão.	
Tipo		Questões de Múltipla Escolha
Dificuldade		Questão com nível muito fácil
Taxonomia de Bloom		lembre-se: reconhecendo, lembrando
Paramétrica		Não
Voltar De Acordo Salvar		

Clique em Atualizar Questão em Minhas Questões para criar alternativas.

Questões das disciplinas em que participo

Entre em contato com o coordenador da disciplina

Figura 4.6: Tela com valores preenchidos para criar uma nova questão, antes de clicar no botão “Salvar”.

é a possibilidade de plágio. O MCTest, porém, consegue minimizar esse problema com a utilização de sorteio das questões e alternativas, gerando exames individuais para cada estudante.

Após a criação da questão, realizada ao clicar no botão “Salvar” na Figura 4.6, pode ser necessário atualizá-la para adicionar alternativas, caso seja uma QM. Para isso, basta acessar a opção “Questões” à direita de “fzprof”, na Figura 4.1, e, em seguida, selecionar o botão “Atualizar” referente à questão desejada. Será apresentada uma nova tela, conforme ilustrado nas Figuras 4.8 e 4.9.

Na Figura 4.8, o professor pode visualizar a questão em formato PDF clicando no botão “Criar-PDF”, mostrado na Figura 4.10. Ao se tratar de uma questão paramétrica que inclui códigos em Python, o professor pode testar a questão separadamente no [Google Colab](#), clicando no botão “Compile-Colab”. O Colab é uma ferramenta muito útil durante a criação de questões paramétricas, já que o MCTest ainda não oferece um ambiente para compilar e depurar códigos. No entanto, esse tópico será abordado em capítulos futuros.

O botão “Salvar-Json” salva todas as questões criadas pelo usuário “fzprof” em um arquivo no formato JSON. A importação desse arquivo ainda não está disponível no MCTest, como observado

anteriormente.

Complementando a tela de criação de questões, apresentadas nas Figuras 4.5 e 4.6, a Figura 4.8 inclui os atributos “Quem criou” e a data da “Última atualização”. A seguir estão as políticas de edição e uso de questões no MCTest:

Observações:

1. Somente o criador da questão ou o coordenador da disciplina podem editar uma questão;
2. No entanto, todos os professores cadastrados na disciplina têm permissão para utilizar as questões em seus exames, se concordarem com os termos apresentados na Seção 1.3.2 - [Direitos autorais das questões disponíveis no BD](#), disponível na opção “De Acordo” da Figura 4.9. Essa é uma política adotada na UFABC, mas outras instituições podem adotar políticas diferentes.

Além disso, a Figura 4.8 apresenta campos para edição das alternativas da questão. Caso a questão seja dissertativa e possua conteúdo nesses campos, esses valores não serão utilizados ao criar exames.

É importante observar que a tela apresenta apenas uma alternativa, incluindo o “Texto da Resposta” e o “Retorno” desta alternativa” (este último é opcional). Para criar novas alternativas, é necessário clicar no botão “Salvar”. De maneira similar, para excluir uma ou várias alternativas já criadas, basta selecionar a caixa de seleção “Apagar” correspondente a cada alternativa que deseja excluir e, em seguida, clicar em “Salvar”.

Destaque:

As Figuras 4.8 e 4.9 foram atualizadas para permitir que o professor efetue o armazenamento no servidor do MCTest de imagens e as inclua em uma questão. O professor pode selecionar um arquivo no formato PNG para importação, sendo necessário tomar precauções quanto ao nome do arquivo, a fim de evitar a sobreposição com outro arquivo de mesmo nome. Uma sugestão consiste em acrescentar um prefixo ao nome do arquivo, tal como fz_PDI_image01.png. É importante observar que o nome do arquivo não deve conter caracteres especiais nem espaços em branco. Este arquivo permanecerá no servidor pelo período de 180 dias. Adicionalmente, é relevante ressaltar que a inclusão de imagens em um documento L^AT_EX pode resultar em considerável lentidão na geração do arquivo PDF.

4.2. QUESTÃO DE MÚLTIPLA ESCOLHA (QM)

Destaque:

A Figura 4.9 foi atualizada para incluir dois contadores: um para o número de correções realizadas para QM e outro para o número de questões que os estudantes acertaram. Com isso, é possível calcular a acurácia da questão (*corretas/correções*). Além disso, foram adicionados os parâmetros da Teoria de Resposta ao Item (TRI): Discriminação (a), Habilidade (b) e Chute (c). Ver Seção 8.6 – [Exames adaptativos](#) para mais detalhes do uso deste recurso.

O modelo TRI emprega uma função logística que pode ajustar até três parâmetros associados a um item (questão): a complexidade (ou habilidade), representada pelo modelo logístico de um parâmetro (b , ou 1PLM, conforme mostrado na Curva Característica do Item na Figura 4.7); essa característica é combinada com a capacidade discriminativa (modelo logístico de dois parâmetros a e b , ou 2PLM); e, por fim, essas características são integradas com a causalidade (acerto pelo chute) (modelo logístico de três parâmetros a , b e c , ou 3PLM). No estudo ([MIN; ARYADOUST, 2021](#)), os pesquisadores estimam um mínimo de 200, 500 e 1.000 participantes nos modelos 1PLM, 2PLM e 3PLM, respectivamente. A Figura 4.7 foi gerada através deste [Colab](#).

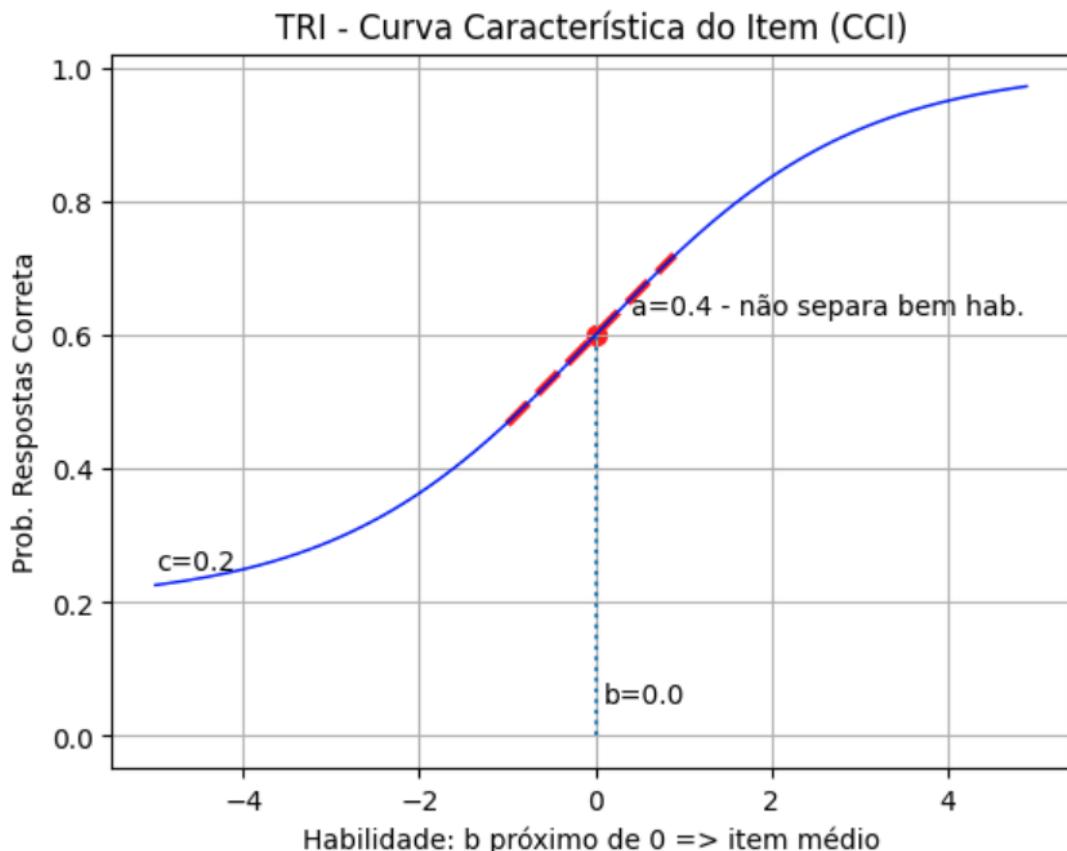


Figura 4.7: Curva Característica do Item (CCI) de 3-Modelo Logístico de Parâmetros, ou 3PLM. Adaptado de [Zampirolli, Batista, Josko, Steil e Trevisan \(2021\)](#).

Destaque:

Na Figura 4.9, há o botão “Duplicar esta Questão” para facilitar o processo de criar uma nova questão com base em uma já existente.

Atualizar Questão

Criar-PDF	Compile-Colab	Salva-Json
Ver esta questão no formato PDF	Copie-Cole a descrição da questão para testar no Colab do Google	Salvar questões de um arquivo (formato Json)
<input type="button" value="Escolher Tópico"/> [DE]<Topico Exemplo>		
<input type="button" value="Descrição curta"/> QE 4		
<input type="button" value="Grupo"/> Somente uma questão por grupo será sorteada para cada exame		
<input type="button" value="Descrição"/> Quarta questão.		
<input type="button" value="Tipo"/> Questões de Múltipla Escolha		
<input type="button" value="Dificuldade"/> Questão com nível muito fácil		
<input type="button" value="Taxonomia de Bloom"/> lembre-se: reconhecendo, lembrando		
<input type="button" value="Paramétrica"/> Não		
<input type="button" value="Quem criou"/> fzampirolli@ufabc.edu.br		
<input type="button" value="Última atualização"/> 07/12/2023		
Texto da Resposta: <input type="text"/>		
Retorno desta alternativa: <input type="text"/>		
Apagar: <input type="checkbox"/>		

Figura 4.8: (Parte 1) Tela de atualização de questão sem alternativas e *feedback*, previamente criada pelo professor.

A Figura 4.11 apresenta o PDF gerado de uma QM com cinco alternativas preenchidas, após o clique no botão “Criar-PDF”. O número em verde representa o ID da questão no banco de dados. Os valores em vermelho correspondem às alternativas incorretas, enquanto o valor em azul representa a alternativa correta, considerando a ordem de criação das alternativas definidas na Figura 4.8.

4.3 Questão dissertativa (QT)

Para criar uma questão dissertativa (QT – de texto), basta selecionar a opção “Questão Dissertativa” no campo Tipo, conforme apresentado na Figura 4.5. É importante observar que o processo de criação para QTs é similar ao processo para QMs, apresentado na seção anterior.

4.3. QUESTÃO DISSERTATIVA (QT)

Estatísticas de questões de múltipla escolha (QM): correções, precisão (corretas/correções) e TRI

Contador de Correções 0	Contador de Corretas 0	
0.00% (= 100 * 0 / 1)		
Discriminação (a) TRI: 0,0	Habilidade (b) 0,0	Chute (c) 0,0

[Voltar](#) [De Acordo](#) [Salvar](#)

[Duplicar esta Questão](#)

[Escolher arquivo](#) Nenhum arquivo escolhido [Upload-Image](#)

Escolha um arquivo PNG para importação. Tenha cuidado com o nome do arquivo para evitar sobre escrever outro arquivo com o mesmo nome. Uma sugestão é incluir um prefixo no nome do arquivo, como fz_PDI_image01.png. Não inclua caracteres especiais e espaços no nome do arquivo. Este arquivo será mantido por 180 dias no servidor. Lembre-se de que as imagens em LaTeX tornam a geração do PDF extremamente lenta! Veja abaixo como incluir esta imagem em uma questão.

```
\begin{figure}[h!]\centering
\includegraphics[scale=0.99]{fz_PDI_image01}
\caption{Example figure}
\end{figure}
```

[Apagar](#)

Questões das disciplinas em que participo
Entre em contato com o coordenador da disciplina

Figura 4.9: (Parte 2) Continuação da tela de atualização de questão, com estatísticas e ajuda para incluir figura.

MCTest

Topic: Topico Exemplo
Group: A
Short Description: QE4
Type: QM
Difficulty: 1
Bloom taxonomy: remember
Last update: 2023-06-04
Who created: fzprof@ufabc.edu.br
Parametric: NO

#2446 1. Quarta questão.

Figura 4.10: Recorte do PDF gerado após clicar no botão “Criar-PDF”, na Figura 4.8.

A Figura 4.12 apresenta um exemplo de PDF gerado para uma QT. Nesse tipo de questão, o estudante deve escrever uma resposta livremente, sem a necessidade de escolher entre alternativas predefinidas. As cinco linhas que aparecem nessa questão são impressas utilizando o comando pré-definido `\drawLines{5}`. É comum que as QTs sejam usadas para avaliar a compreensão do estudante sobre um determinado tema, suas habilidades de análise e argumentação, ou ainda para avaliar a

MCTest

Topic: Topico Exemplo
Group: A
Short Description: QE4
Type: QM
Difficulty: 1
Bloom taxonomy: remember
Last update: 2023-06-04
Who created: fzprof@ufabc.edu.br
Parametric: NO

#2446 1. Quarta questão.

A.^{*}1a alternativa criada B.^{*}3a alternativa criada C.^{*}01a alternativa criada D.^{*}3a alternativa criada E.^{*}5a alternativa criada

Figura 4.11: Recorte do PDF gerado após clicar no botão “Criar-PDF”, na Figura 4.8, para a questão atualizada com cinco alternativas.

sua capacidade de escrever de forma clara e organizada.

MCTest

Topic: Topico Exemplo
Group: B
Short Description: texto
Type: QT
Difficulty: 1
Bloom taxonomy: remember
Last update: 2023-06-29
Who created: fzampirolli@ufabc.edu.br
Parametric: NO

#2448 1. Descrição da questão dissertativa.

Figura 4.12: Recorte do PDF gerado após clicar no botão “Criar-PDF”, na Figura 4.8, para uma questão dissertativa (“Type: QT”).

Na Figura 4.13, apresenta-se a alteração do campo “Descrição” da Figura 4.8, utilizando a sintaxe do L^AT_EX, uma linguagem de marcação amplamente utilizada na produção de documentos acadêmicos. O PDF gerado desta questão é apresentado na Figura 4.14

A questão avaliará as competências e habilidades dos estudantes em relação aos conceitos de condicional e repetição na programação. Esses conceitos são fundamentais para o desenvolvimento de algoritmos e programas computacionais, e são frequentemente abordados em disciplinas introdutórias de programação.

4.4. CONSIDERAÇÕES FINAIS

Preencher a tabela com os valores corretos da variável soma e da condição correspondente em cada iteração do laço, para um valor de entrada N específico. Além disso, considerar os números das linhas do algoritmo para preencher a tabela para facilitar a referência.

```
\begin{verbatim}
1. Inicializar a variável N com 7
2. Inicializar a variável soma com 0
3. Inicializar a variável contador com 1
4. Enquanto contador <= N faça
5.   Se contador é par então
6.     soma = soma + contador
7.   Senão
8.     soma = soma - contador
9.   Fim se
10.  Incrementar contador em 1
11. Fim enquanto
12. Imprimir o valor da variável soma
\end{verbatim}
```

```
\begin{tabular}{|c|c|c|c|} \hline
\textbf{linha} & \textbf{contador} & \textbf{soma} \\ \hline
& & \\ \hline
\end{tabular}
```

Figura 4.13: Recorte da “Descrição” de uma QT de teste de mesa.

4.4 Considerações finais

O MCTest é uma plataforma completa que oferece diversas funcionalidades para a criação de exames. Desde QMs e QTs até questões paramétricas que envolvem códigos em Python. O MCTest oferece recursos para os professores poderem criar avaliações personalizadas para suas turmas. Além disso, a plataforma permite que as questões sejam organizadas em bancos de dados para facilitar a criação de exames.

No entanto, é importante destacar que o MCTest é apenas uma ferramenta auxiliar no processo de avaliação. A qualidade da avaliação depende não apenas das questões criadas, mas também da forma como são elaboradas e aplicadas. É fundamental que os professores sejam cuidadosos na escolha das questões e na definição do nível de dificuldade adequado para cada turma. A utilização do MCTest deve ser sempre acompanhada de uma análise crítica e cuidadosa sobre a adequação das questões e a efetividade do processo de avaliação.

MCTest

Topic: Topico Exemplo
Group: B
Short Description: texto
Type: QT
Difficulty: 1
Bloom taxonomy: remember
Last update: 2023-06-04
Who created: fzprof@ufabc.edu.br
Parametric: NO

#2447 1. Preencher a tabela com os valores corretos da variável soma e da condição correspondente em cada iteração do laço, para um valor de entrada N específico. Além disso, considerar os números das linhas do algoritmo para preencher a tabela para facilitar a referência.

1. Inicializar a variável N com 7
2. Inicializar a variável soma com 0
3. Inicializar a variável contador com 1
4. Enquanto contador <= N faça
5. Se contador é par então
6. soma = soma + contador
7. Senão
8. soma = soma - contador
9. Fim se
10. Incrementar contador em 1
11. Fim enquanto
12. Imprimir o valor da variável soma

linha	contador	soma

Figura 4.14: Recorte do PDF gerado da QT de teste de mesa.

Por fim, espera-se que este documento tenha sido útil para os professores interessados em utilizar o MCTest em suas disciplinas. No próximo capítulo, serão abordadas questões paramétricas que envolvem códigos em Python.

Capítulo 5

Questões paramétricas

Conteúdo

5.1	QM paramétrica	72
5.1.1	Um exemplo	73
5.1.2	QM paramétrica no MCTest	74
5.2	QT paramétrica	78
5.2.1	Exemplo 1 – Teste de mesa	78
5.2.2	Exemplo 2 – Teste de mesa e gabarito	79
5.2.3	Exemplo 3 – Teste de mesa e gabarito com <code>settrace</code>	80
5.3	QM paramétrica com múltiplas variações: um estudo de caso	85
5.3.1	Explicação do método <code>gerar_QM_itens</code>	86
5.3.2	Explicação do método <code>gerar_QM_itens</code> : Verifica parâmetros	87
5.3.3	Explicação do método <code>gerar_QM_itens</code> : Outros métodos	90
5.3.4	Explicação do método <code>gerar_QM_itens</code> : Laço Principal	93
5.3.5	Aplicação do método <code>gerar_QM_itens</code>	93
5.4	QT paramétrica, com código	98
5.4.1	Introdução ao Moodle e ao VPL	98
5.4.2	Questão com exercício de programação no Moodle+VPL	99
5.4.3	Questão com exercício de programação no MCTest+Moodle+VPL	101
5.5	Considerações finais	109

No capítulo anterior, foi apresentada a criação de questões estáticas de múltipla escolha (QMs) e dissertativas (QTs), incluindo uma questão típica utilizada em disciplinas de lógica de programação na UFABC, apresentada na Figura 4.14. É possível criar essas questões utilizando L^AT_EX, uma linguagem de preparação de documentos de alta qualidade e profissionalismo, com recursos avançados para controle de tipos de letra, espaçamento, numeração de seções, referências bibliográficas e equações

matemáticas, entre outros. L^AT_EX é gratuita e está disponível para vários sistemas operacionais.

A linguagem L^AT_EX foi criada no início da década de 1980 pelo matemático e cientista da computação Leslie Lamport ([LAMPORT, 1985](#)). Ela é baseada no T_EX, uma linguagem de marcação desenvolvida por Donald Knuth na década de 1970 para produzir documentos com alta qualidade tipográfica para sua série de livros “*The Art of Computer Programming*”. O T_EX se tornou popular para a produção de documentos científicos, técnicos e matemáticos e ainda é amplamente utilizado atualmente.

Para contornar a sintaxe complexa e de difícil programação do L^AT_EX, no MCTest foi introduzida a possibilidade de intercalar trechos em L^AT_EX com trechos de código em Python. Python é uma linguagem de programação de alto nível, interpretada e de propósito geral, criada no início da década de 1990 por Guido van Rossum, e atualmente é uma das linguagens de programação mais populares do mundo ¹. Com uma ampla variedade de aplicações em áreas como desenvolvimento web, ciência de dados, inteligência artificial e automação de processos, Python é conhecida por sua sintaxe clara e legível, facilitando a escrita e a leitura de código.

O MCTest proporciona aos usuários a capacidade de criar questões diversas e paramétricas, combinando o uso do formato L^AT_EX e da linguagem de programação Python. Essa funcionalidade será apresentada neste capítulo. Em capítulos futuros, na Parte - **IV - Experimentos**, serão exploradas com mais detalhes essa característica fundamental do MCTest de criar questões paramétricas. Essa abordagem proporciona maior flexibilidade na criação de questões personalizadas e permite uma fácil manipulação dos parâmetros definidos pelos professores. Dessa forma, a criação de questões torna-se mais eficiente e adaptável às necessidades específicas de cada disciplina.

Apesar dessa possibilidade, muitas vezes a criação de questões paramétricas ainda exige habilidades de programação por parte do professor. No entanto, quando há um grupo de professores interessados e uma grande demanda de avaliações para milhares de estudantes, o MCTest oferece a gestão de avaliações utilizando um banco de questões que podem ser facilmente reaproveitadas em avaliações futuras. Essa abordagem já foi avaliada em várias publicações científicas, demonstrando eficiência e eficácia em avaliações, como apresentado em [vision.ufabc.edu.br](#).

5.1 QM paramétrica

A criação de QMs é uma tarefa comum para professores de diversas áreas do conhecimento. Para tornar esse processo mais eficiente, é possível utilizar recursos de programação. Nesta seção, será abordada a QM paramétrica, que permite criar questões com variações de parâmetros. Isso possibilita a criação de inúmeras questões com facilidade e rapidez. Serão apresentados exemplos de como criar questões paramétricas em Python, utilizando o MCTest como plataforma. Serão abordadas diferentes técnicas de parametrização, permitindo ao professor escolher a que melhor se adapta à sua habilidade em programação.

¹Levantamento das linguagens de programação mais populares em 2023: [survey.stackoverflow.co/2023](#) e [pypl.github.io/PYPL.html](#).

5.1.1 Um exemplo

Suponha que você seja um professor de matemática e deseje criar várias QMs sobre a fórmula da área do círculo. Embora este exemplo seja bastante simples, é possível generalizar para questões mais complexas. Para alcançar esse objetivo, você pode utilizar o método em Python apresentado no Código 5.1.

Código: método `area_circulo(raio)`

```
1 import math
2
3 def area_circulo(raio):
4     return math.pi * raio ** 2
```

Código 5.1: Exemplo de método para calcular a área de um círculo.

Com base no método apresentado no Código 5.1, é possível gerar diversas QMs sobre a área do círculo, com as seguintes variações:

1. A questão deve ter 4 alternativas de resposta;
2. O raio do círculo deve variar aleatoriamente;
3. A alternativa correta deve ser calculada com base no valor do raio;
4. As demais alternativas erradas devem ter valores distintos.

A seguir, serão apresentados três exemplos de questões para calcular a área do círculo, com raios escolhidos aleatoriamente:

1) Qual é a área do círculo de raio 2?

- A. 3,14 B. 6,28 C. 12,56 D. 25,12

2) Qual é a área do círculo de raio 4?

- A. 12,56 B. 25,12 C. 37,68 D. 50,24

3) Qual é a área do círculo de raio 3?

- A. 9,42 B. 12,56 C. 15,70 D. 18,84

Nos exemplos ilustrativos apresentados, o método `area_circulo` pode ser utilizado para calcular a resposta correta para cada alternativa de resposta. A variação aleatória do raio pode ser realizada com a ajuda da biblioteca `random` do Python. Por fim, as alternativas erradas podem ser calculadas, também utilizando o método `area_circulo`, com base no valor do raio, garantindo a não repetição de alternativas. A seguir, será apresentado esse exemplo implementado no MCTest.

5.1.2 QM paramétrica no MCTest

É possível parametrizar QMs no MCTest de várias maneiras, dependendo das habilidades de programação em Python do professor. Algumas dessas maneiras são apresentadas nesta seção.

Observações:

1. A descrição da questão no MCTest (campo “Descrição” na Figura 4.8), que pode incluir textos em L^AT_EX integrados com códigos em Python, é a parte mais importante na criação de questões paramétricas. A partir deste capítulo, não será mais apresentada uma captura de tela da questão, como foram realizados nos dois capítulos anteriores. Para criar uma QM ou QT paramétrica basta alterar o parâmetro “Paramétrica” de “Não” para “Sim”, nas telas de criação ou alteração de questão;
2. A parte mais complicada na criação de questões paramétricas no MCTest é lidar com possíveis erros que podem ser retornados pelo MCTest ao tentar criar o PDF. Para tentar resolver esse problema, foi criado um *notebook* no Google Colab, acessado ao clicar no botão “Compile-Colab” nas telas de alteração de questão, como apresentado na Figura 4.8.

Exemplo 1

Veja uma descrição da questão paramétrica para calcular a área de um círculo, implementada no MCTest, no Código 5.2. Todo o texto que aparece até a linha 6, contendo `[[def:]`, fará parte do enunciado da questão e pode utilizar a sintaxe do L^AT_EX. As partes paramétricas deste enunciado ocorrem nas instruções em Python entre as marcações `[[code: e]]`. Geralmente, essas instruções são variáveis definidas em Python entre as marcações `[[def: e]]`, neste exemplo, entre as linhas 6 e 30.

Marcações principais definidas no MCTest:

1. Instruções que podem representar a parte paramétrica do enunciado da questão ou das alternativas são colocadas entre as marcações `[[code: e]]`, conforme exemplificado na linha 1 do Código 5.2;
2. Após esse enunciado, um único bloco de código em Python deve ser definido entre as marcações `[[def: e]]`, como ilustrado entre as linhas 6 e 30 do exemplo. Todo o texto após esse bloco é descartado do enunciado da questão.

Destaque:

Em alguns casos, ocorre um erro no retorno da expressão regular para capturar as variáveis ou instruções delimitadas por `[[code: e]]`. Nesses casos, é necessário adicionar um espaço antes de `[[code: e após]]`. O método `get_code` disponível no GitHub em [UtilsMCTest4.py](#) propõe uma sugestão de modificação na expressão regular. No entanto, essa sugestão apresenta uma falha quando ocorre uma sequência de `]]` dentro de `[[def: e]]`. Assim, para evitar problemas, use sempre espaços antes de `[[code: e após]]`.

Se as marcações de início e fim de bloco ou instruções aparecerem erroneamente, ou se houver algum erro no código Python, mensagens de erro do compilador Python poderão ser exibidas, mas sem fornecer muitos detalhes sobre o erro específico. Portanto, é fundamental testar o código Python fora do MCTest, por exemplo, utilizando o *notebook* do Google Colab. Para acessar o *notebook*, basta clicar no botão “Compile-Colab” nas telas de alteração de questão, como mostrado na Figura 4.8. Experimente adicionar espaços inesperados, como `[[code: raio1]]` ou `[[code:raio1]]`, salve e clique em “Cria-PDF”. Isso permitirá identificar erros e depurar a questão no MCTest, seguindo os seguintes exemplos:

Alguns exemplos de erros comuns que podem ocorrer:

Experimente incluir esses erros, salve e crie o PDF para identificar e corrigir possíveis problemas futuros:

1. `[[code: raio1]]` – espaço não esperado – retorna:
`ERROR in [[code: ...]]: unmatched ']' (<string>, line 1) raio1]]?`
2. `[[code:Raio1]]` – “R” maiúsculo – retorna:
`ERROR in [[code: ...]]: name 'Raio1' is not defined Raio1`
3. Bloco de código no método `area_circulo` sem *indentação* correta – retorna:
`ERROR in [[def: ...]]: unexpected indent (<string>, line 7)`
4. Sem a linha `import math` no método `area_circulo` – retorna:
`ERROR in [[def: ...]]: name 'math' is not defined`
5. Sem os `:` no final da linha do método `area_circulo` – retorna:
`ERROR in [[def: ...]]: invalid syntax (<string>, line 5)`
6. `[[def: ou [[def :` – espaço não esperado – retorna:
`ERROR in [[code: ...]]: name 'raio1' is not defined raio1`

Observe que, nesse sexto caso, o erro retornado é que a variável `raio1` não foi definida. Isso ocorreu porque o MCTest não encontrou o bloco de código Python entre as marcações `[[def: e]]`. Certifique-se de ter incluído corretamente o bloco de código necessário para definir a variável `raio1` e verifique se não há erros de digitação ou formatação nas marcações. Essa etapa é crucial para garantir o funcionamento correto da questão paramétrica no MCTest.

Questão:

```

1 Qual é a área do círculo de raio [[code:raio1]]?
2
3 % comentário em LaTeX. Observe que a variável raio1, após "code:"
4 % foi definida na linha 15 abaixo
5
6 [[def: # comentário em Python
7 # o bloco de código Python deve sempre iniciar com a linha anterior!
8 import random
9
10 def area_circulo(raio): # método para calcular a área do círculo
11     import math # necessário importar biblioteca(s) no método
12     return math.pi * raio ** 2
13
14 # lista de 4 raios aleatórios entre 20 e 50
15 raio1,raio2,raio3,raio4 = random.sample(range(20,51), 4)
16
17 # formata as áreas com 2 casas decimais
18 area1 = f"{area_circulo(raio1):.2f}"
19 area2 = f"{area_circulo(raio2):.2f}"
20 area3 = f"{area_circulo(raio3):.2f}"
21 area4 = f"{area_circulo(raio4):.2f}"
22
23 # incluir nas alternativas:
24 # [[code:area1]]
25 # [[code:area2]]
26 # [[code:area3]]
27 # [[code:area4]]
28
29 # o bloco de código Python deve sempre terminar com a próxima linha
30 ]]
31 Todo o texto que aparece após a linha anterior é ignorado.

```

Código 5.2: Exemplo de QM paramétrica para calcular a área do círculo.

Exemplo 2

Quando as alternativas de uma QM paramétrica assumem apenas valores inteiros, é possível utilizar o método `createWrongAnswers` desenvolvido e disponibilizado no MCTest. No entanto, algumas exigências são necessárias, como apresentado no Código 5.3.

Primeiramente, é necessário criar uma variável global chamada `correctAnswer`, conforme mostrado na linha 12 do código. Em seguida, atribui-se a resposta correta a essa variável, como ilustrado na linha 14.

No que diz respeito às alternativas da questão, é importante incluir na primeira alternativa o conteúdo da linha 17, sem o comentário. Na segunda alternativa, deve-se utilizar o conteúdo da linha 18, também sem o comentário. Na linha 18, o método irá gerar três valores aleatórios distintos entre `correctAnswer-10` e `correctAnswer+10`.

É importante observar que, nesse caso, as questões geradas não são consideradas “ótimas”,

5.1. QM PARAMÉTRICA

pois os estudantes podem deduzir que a resposta correta não está nos extremos dos valores fornecidos, conforme ilustrado na Figura 5.1. Em QMs, o texto em verde (#2450) representa a identificação da questão no banco de dados. Azul (#0) representa a primeira alternativa na tela de criação ou alteração da questão. As demais alternativas em vermelho (*1, *2 e *3) são as três alternativas erradas criadas pelo método `createWrongAnswers`.

Questão:

```
1 Qual é a área do círculo de raio [[code:raio1]]?
2
3 [[def:
4 import random
5
6 def area_circulo(raio):
7     import math # necessário importar biblioteca(s) no método
8     return math.pi * raio ** 2
9
10 raio1 = random.randint(10, 51) # sorteia valor entre 10 e 50
11
12 global correctAnswer # variável necessária para usar o método
13 # createWrongAnswers, a resposta deve ter valor inteiro
14 correctAnswer = int(area_circulo(raio1))
15
16 # incluir nas alternativas da questão:
17 # [[code:correctAnswer]] % primeira alternativa (sempre) correta
18 # [[code:createWrongAnswers([3,10])]] % cria 3 alter. erradas +/- 10
19 ]]
```

Código 5.3: Exemplo de QM paramétrica para calcular a área do círculo, utilizando o método `createWrongAnswers`.

MCTest

Topic: Topico Exemplo

Group: A

Short Description: área círculo - ex.2

Type: QM

Difficulty: 1

Bloom taxonomy: remember

Last update: 2023-06-30

Who created: fzampirolli@ufabc.edu.br

Parametric: YES

#2450 1. Qual é a área do círculo de raio 47?

A.*₂6932 B.*₁6930 C._{#0}6939 D.*₃6942

Figura 5.1: Recorte do PDF gerado para a questão definida no Código 5.3.

5.2 QT paramétrica

Os conceitos apresentados anteriormente para as QMs paramétricas são aplicáveis também às QTs paramétricas, com a distinção de que não há necessidade de considerar as alternativas da questão. Em vez disso, é possível definir uma única resposta correta, se necessário, exemplificado na Seção 8.1 – Desenvolvendo QMs e uma QT para inclusão em um exame. Nesta seção, serão fornecidos três exemplos adicionais da questão de teste de mesa introduzida no capítulo anterior (Seção 4.3 – Questão dissertativa (QT)), agora sob a perspectiva paramétrica.

No primeiro exemplo, foi criada uma questão com três parâmetros aleatórios. No segundo exemplo, foi adicionado um gabarito para esses parâmetros aleatórios, implementando a solução do pseudocódigo apresentado em Python e adicionando comandos `print` em locais estratégicos para imprimir a tabela com o gabarito. Por fim, no último exemplo, foi utilizado um recurso avançado de rastreamento de instruções, métodos e exceções. Esses exemplos mostram o poder da parametrização no MCTest para criar questões personalizadas e detalhadas.

5.2.1 Exemplo 1 – Teste de mesa

Nesta seção, será abordada a parametrização da questão de teste de mesa ilustrada na Figura 5.2 e apresentada no Código 5.4, para exemplificar o processo. As partes paramétricas estão localizadas nas linhas 4, 7, 8 e 9 na questão, com os parâmetros `N1`, `N2` e `N3`, cujos valores são definidos aleatoriamente no bloco de código Python nas linhas 38, 39 e 40, respectivamente. O PDF correspondente a essa questão é apresentado na Figura 5.2.

#2451 1. Preencha a tabela com os valores corretos das variáveis contador e soma em cada iteração do laço. Além disso, considere os números das linhas do código para preencher a tabela, a fim de facilitar a referência. Considere inicialmente `maximo=14`, `contador=2` e `soma=1`.

1. Inicializar a variável `maximo` com 14
2. Inicializar a variável `contador` com 2
3. Inicializar a variável `soma` com 1
4. Enquanto `contador <= maximo` faça
5. Se `contador` é par então
6. `soma = soma + contador`
7. Senão
8. `soma = soma - contador`
9. Fim se
10. Incrementar `contador` em 1
11. Fim enquanto
12. Imprimir o valor da variável `soma`

linha	maximo	contador	soma

Figura 5.2: Recorte do PDF gerado para a questão definida no Código 5.4.

Questão:

```
1 Preencha a tabela com os valores corretos das variáveis contador e soma
2 em cada iteração do laço. Além disso, considere os números das linhas do
3 código para preencher a tabela, a fim de facilitar a referência. Considere
4 inicialmente maximo=[[code:N1]], contador=[[code:N2]] e soma=[[code:N3]].
5
6 \begin{verbatim}
7 1. Inicializar a variável maximo com [[code:N1]]
8 2. Inicializar a variável contador com [[code:N2]]
9 3. Inicializar a variável soma com [[code:N3]]
10 4. Enquanto contador <= maximo faça
11   5.   Se contador é par então
12     6.     soma = soma + contador
13   7.   Senão
14     8.     soma = soma - contador
15   9.   Fim se
16 10.   Incrementar contador em 1
17 11. Fim enquanto
18 12. Imprimir o valor da variável soma
19 \end{verbatim}
20
21 \begin{tabular}{|c|c|c|c|} \hline
22 \textbf{linha} & \textbf{maximo} & \textbf{contador} & \textbf{soma} \\ \hline
23 & & & \\ \hline
24 & & & \\ \hline
25 & & & \\ \hline
26 & & & \\ \hline
27 & & & \\ \hline
28 & & & \\ \hline
29 & & & \\ \hline
30 & & & \\ \hline
31 & & & \\ \hline
32 & & & \\ \hline
33 & & & \\ \hline
34 \end{tabular}
35
36 [[def:
37 import random
38 N1 = random.randint(7,16)
39 N2 = random.randint(1,6)
40 N3 = random.randint(1,5)
41 ]]
```

Código 5.4: Exemplo simples de descrição de QT paramétrica para teste de mesa.

5.2.2 Exemplo 2 – Teste de mesa e gabarito

É possível aprimorar o exemplo anterior expandindo-o para gerar o gabarito correspondente a cada valor aleatório gerado para as variáveis N1, N2 e N3. Para essa finalidade, foi criado um método em Python que implementa o algoritmo da questão. Devido à quantidade de linhas na descrição dessa questão, os códigos correspondentes foram divididos e estão disponíveis nos fragmentos dos

Códigos 5.5 e 5.6.

Nesta nova versão, o gabarito foi adicionado ao PDF utilizando as linhas 37 a 41 do Código 5.5. Na linha 37, azul e o tamanho `small` foram definidos para a variável `tabela`, conforme mostrado na Figura 5.3. Caso o professor não deseje exibir esse gabarito na folha de atividade, basta utilizar a cor `white`. Na linha 39, a variável `tabela`, sendo uma *string*, é definida na segunda parte do código, no Código 5.6.

Nesta versão, foi criado o método `gerar_gabarito` para retornar a tabela atualizada para incluir as linhas específicas do algoritmo com os espaços correspondentes para preencher com os valores corretos das variáveis `maximo`, `contador` e `soma`.

O trecho entre as linhas 30 e 34 do Código 5.6 apresenta um exemplo de uso do código, no qual são gerados valores aleatórios para `N1`, `N2` e `N3` até que a tabela gerada tenha entre 10 e 15 linhas (exclusivamente). Esse trecho será analisado linha por linha:

`while True:` isso cria um *loop* infinito que será executado até que a condição na linha 33 seja satisfeita e o *loop* seja interrompido na linha 34;

`N1, N2, N3 = random.sample(range(4, 20), 3):` o método `sample` da biblioteca `random` é utilizado para gerar três valores aleatórios, sem repetição, no intervalo de 4 a 19. Esses valores são atribuídos às variáveis `N1`, `N2` e `N3`;

`tabela = gerar_gabarito(N1, N2, N3):` o método `gerar_gabarito` é chamado, recebendo como argumentos os valores aleatórios `N1`, `N2` e `N3`. Esse método retorna a tabela formatada com os valores corretos das variáveis `maximo`, `contador` e `soma`;

`if 10 < len(tabela.split('\n')) < 15: break:` o número de linhas na tabela gerada é verificado. Se o número de linhas estiver entre 10 e 15 (exclusivamente), a condição é satisfeita e o *loop* é interrompido com o comando `break`.

Dessa forma, esse trecho de código permite gerar valores aleatórios para as variáveis `N1`, `N2` e `N3` repetidamente até que a tabela gerada tenha o número de linhas desejado. Isso garante que todas as variações tenham dificuldades de resolução semelhantes, testando diferentes casos. Neste exemplo simples, é possível gerar $15 * 14 * 13$ variações (pois os valores sortados não se repetem usando o comando `random.sample(range(4, 20), 3)`), mas apenas aquelas que resultarem em tabelas com 11 a 14 linhas serão consideradas.

5.2.3 Exemplo 3 – Teste de mesa e gabarito com `settrace`

Um exemplo avançado de teste de mesa é o uso do método `settrace` em Python. `settrace` é um método da biblioteca `sys` que permite definir um rastreador de ações ocorridas no código. Ele é usado para registrar um rastreamento personalizado que será chamado sempre que ocorrer uma chamada de instrução, retorno de método ou exceção.

Ao chamar `sys.settrace`, é possível fornecer um rastreamento personalizado que será invocado automaticamente durante a execução do programa. Esse método de rastreamento recebe os argumentos:

Questão:

```
1 Preencha a tabela com os valores corretos das variáveis contador e soma
2 em cada iteração do laço. Além disso, considere os números das linhas do
3 código para preencher a tabela, a fim de facilitar a referência. Considere
4 inicialmente maximo=[[code:N1]], contador=[[code:N2]] e soma=[[code:N3]].
5
6 \begin{verbatim}
7 1. Inicializar a variável maximo com [[code:N1]]
8 2. Inicializar a variável contador com [[code:N2]]
9 3. Inicializar a variável soma com [[code:N3]]
10 4. Enquanto contador <= maximo faça
11   5.   Se contador é par então
12     6.     soma = soma + contador
13   7.   Senão
14     8.     soma = soma - contador
15   9.   Fim se
16 10.   Incrementar contador em 1
17 11. Fim enquanto
18 12. Imprimir o valor da variável soma
19 \end{verbatim}
20
21
22 \begin{tabular}{|c|c|c|c|} \hline
23 & \textbf{linha} & \textbf{maximo} & \textbf{contador} & \textbf{soma} \\ \hline
24 & & & & \\ \hline
25 & & & & \\ \hline
26 & & & & \\ \hline
27 & & & & \\ \hline
28 & & & & \\ \hline
29 & & & & \\ \hline
30 & & & & \\ \hline
31 & & & & \\ \hline
32 & & & & \\ \hline
33 & & & & \\ \hline
34 & & & & \\ \hline
35 \end{tabular}
36
37 {\color{blue} \small
38 \begin{verbatim}
39 [[code:tabela]]
40 \end{verbatim}
41 }}
```

Código 5.5: Exemplo prático de teste de mesa paramétrico mostrando o gabarito – Parte 1: Descrição de questão.

frame: é um objeto que representa o quadro de variáveis atual em uma pilha;

event: é uma *string* que indica o tipo de evento que ocorreu. Pode ser *call* (chamada de método), *return* (retorno de método) ou *exception* (exceção);

arg: é um argumento adicional que depende do tipo de evento. Para eventos de chamada e retorno,

Questão:

```

1  [[def:
2  import random
3
4  def gerar_gabarito(N1, N2, N3):
5      s = "linha maximo contador soma\n"
6      s += "-----\n"
7
8      maximo = N1
9      s += f" 1 {maximo:6d}\n"
10     contador = N2
11     s += f" 2 {maximo:6d} {contador:6d}\n"
12     soma = N3
13     s += f" 3 {maximo:6d} {contador:6d} {soma:7d}\n"
14
15
16     while contador <= maximo:
17         if contador % 2 == 0:
18             soma = soma + contador
19             s += f" 6 {maximo:6d} {contador:6d} {soma:7d}\n"
20         else:
21             soma = soma - contador
22             s += f" 8 {maximo:6d} {contador:6d} {soma:7d}\n"
23
24         contador = contador + 1
25         s += f" 10 {maximo:6d} {contador:6d} {soma:7d}\n"
26
27     return s
28
29 # Exemplo de uso:
30 while True:
31     N1, N2, N3 = random.sample(range(4, 20), 3)
32     tabela = gerar_gabarito(N1, N2, N3)
33     if 10 < len(tabela.split('\n')) < 15:
34         break
35 ]]

```

Código 5.6: Exemplo prático de teste de mesa paramétrico mostrando o gabarito – Parte 2: Bloco de código em Python.

é sempre *None*. Para eventos de exceção, é uma tupla contendo informações sobre a exceção disparada.

O método de rastreamento personalizado pode realizar várias ações, como exibir informações sobre as chamadas de métodos, coletar dados de execução, fazer análises dinâmicas, entre outros. É uma ferramenta poderosa para depurar e analisar o fluxo de execução de um programa Python.

É importante mencionar que o uso de `sys.settrace` pode ter um impacto significativo no desempenho do programa, uma vez que o método de rastreamento é chamado em cada evento de chamada, retorno ou exceção. Portanto, é recomendado usá-lo com cuidado e apenas quando necessário para fins de depuração ou análise.

Exemplo de uso do método settrace para rastreamento de código

O Código 5.7 define o método `trace` para ser chamada sempre que ocorrer um evento específico, como a execução de uma linha de código. Neste método, as informações relevantes são impressas no console, após a sua execução, como o evento, o número da linha e as variáveis locais. Em seguida, o método `gerar_gabarito` é definido para calcular a soma ou subtração com contador par e ímpar, respectivamente, em um determinado intervalo. O método é chamado com os argumentos 9, 7 e 6 para os parâmetros `maximo`, `contador` e `soma`, respectivamente. Por fim, o método `trace` é desativado na linha 16. Este código é um exemplo de como usar o recurso de rastreamento em Python para depurar o código e entender como as variáveis são alteradas durante a execução. O código rastreado não possui saída, mas o rastreamento pode ser modificado para imprimir informações adicionais ou para armazenar informações em um arquivo de *log*.

Ao salvar o Código 5.7 no arquivo `trace.py` e executá-lo no console do computador usando o comando `python trace.py`, a saída resultante será semelhante ao apresentado a seguir:

#2452 1. Preencha a tabela com os valores corretos das variáveis contador e soma em cada iteração do laço. Além disso, considere os números das linhas do código para preencher a tabela, a fim de facilitar a referência. Considere inicialmente `maximo=10`, `contador=8` e `soma=17`.

1. Inicializar a variável `maximo` com 10
2. Inicializar a variável `contador` com 8
3. Inicializar a variável `soma` com 17
4. Enquanto `contador <= maximo` faça
 5. Se `contador` é par então
 6. `soma = soma + contador`
 7. Senão
 8. `soma = soma - contador`
 9. Fim se
 10. Incrementar `contador` em 1
 11. Fim enquanto
 12. Imprimir o valor da variável `soma`

linha	maximo	contador	soma

```
linha maximo contador soma
-----
1 10
2 10     8
3 10     8     17
6 10     8     25
10 10    9     25
7 10     9     16
10 10    10    16
6 10    10     26
10 10   11     26
```

Figura 5.3: Recorte do PDF gerado para a questão definida nos Códigos 5.5 e 5.6.

Questão:

```

1  def trace(frame, event, arg_unused):
2      print(f"{event}\t{frame.f_lineno}\t{frame.f_locals}")
3      return trace
4
5  def gerar_gabarito(maximo, contador, soma):
6      while contador <= maximo:
7          if contador % 2 == 0:
8              soma = soma + contador
9          else:
10             soma = soma - contador
11             contador = contador + 1
12
13 import sys
14 sys.settrace(trace)
15 gerar_gabarito(9, 7, 6)
16 sys.settrace(None)

```

Código 5.7: Exemplo de uso de `sys.settrace`.**Saída do Código 5.7:**

```

1  call    5      {'maximo': 9, 'contador': 7, 'soma': 6}
2  line    6      {'maximo': 9, 'contador': 7, 'soma': 6}
3  line    7      {'maximo': 9, 'contador': 7, 'soma': 6}
4  line    10     {'maximo': 9, 'contador': 7, 'soma': 6}
5  line    11     {'maximo': 9, 'contador': 7, 'soma': -1}
6  line    6      {'maximo': 9, 'contador': 8, 'soma': -1}
7  line    7      {'maximo': 9, 'contador': 8, 'soma': -1}
8  line    8      {'maximo': 9, 'contador': 8, 'soma': -1}
9  line    11     {'maximo': 9, 'contador': 8, 'soma': 7}
10 line   6      {'maximo': 9, 'contador': 9, 'soma': 7}
11 line   7      {'maximo': 9, 'contador': 9, 'soma': 7}
12 line   10     {'maximo': 9, 'contador': 9, 'soma': 7}
13 line   11     {'maximo': 9, 'contador': 9, 'soma': -2}
14 line   6      {'maximo': 9, 'contador': 10, 'soma': -2}
15 return  6      {'maximo': 9, 'contador': 10, 'soma': -2}

```

É possível formatar essa saída e gerar uma questão completa no MCTest, conforme demonstrado a seguir.

Teste de mesa com rastreamento de código usando `settrace`

Neste terceiro exemplo de teste de mesa paramétrico, é utilizado o método `settrace`, da biblioteca `sys`, para gerar o gabarito. Este exemplo foi inspirado no artigo de [Teubl e Zampirolli \(2023\)](#). A vantagem dessa versão em relação à anterior é que pode-se criar o gabarito sem precisar incluir vários comandos `print` na implementação do pseudocódigo, tornando o processo mais genérico

para diferentes exemplos de teste de mesa. A seguir, serão adaptadas as versões anteriores, criando um código em Python que faz parte do enunciado da questão, apresentado no Código 5.8. No entanto, também é possível ter utilizado um pseudocódigo, conforme demonstrado por Teubl e Zampirolli (2023).

A parte com diferenças significativas em relação às anteriores é apresentada no Código 5.9. Nessa parte, o método `trace` é adaptado, conforme apresentado no Código 5.7, para formatar a tabela com o gabarito. Dentro desse método, é verificado se o evento atual é uma linha de código (`event == 'line'`). Em caso afirmativo, o método obtém o número da linha atual (`lineno`) e o dicionário de variáveis locais (`locals_dict`) do `frame` atual.

Em seguida, o método obtém os valores das variáveis `maximo`, `contador` e `soma` do dicionário de variáveis locais, usando o método `get`. Esses valores são usados para criar uma `string` formatada, armazenada em uma lista chamada `output`.

Por fim, o método retorna a si (`return trace`), o que permite que seja chamado novamente no próximo evento de linha. Esse processo se repete até que a execução do programa seja concluída ou até encontrar o comando `sys.settrace(None)`, que encerra o rastreamento.

Além disso, o código apresenta o método `gerar_gabarito`, semelhante ao apresentado no Código 5.7. Em seguida, o trecho de código inicia um loop infinito que realiza as seguintes ações: gera três números aleatórios distintos em um intervalo específico, inicializa uma lista vazia chamada `output` e adiciona cabeçalhos e uma linha de separação à lista. Em seguida, o rastreamento do código é iniciado com `sys.settrace(trace)`, o método `gerar_gabarito` é executado com os valores gerados e o rastreamento é finalizado com `sys.settrace(None)`. A lista `output` é convertida em uma única `string` chamada `tabela`. O código verifica se o número de linhas na tabela está entre 11 e 14. Se a condição for satisfeita, o loop é interrompido e o programa é finalizado. Essa estrutura de repetição permite gerar e obter uma tabela com um número específico de linhas no intervalo desejado.

5.3 QM paramétrica com múltiplas variações: um estudo de caso

Nesta seção, é apresentada uma questão paramétrica que oferece uma ampla variedade de configurações, nos Códigos 5.10, 5.11, 5.12, 5.13, 5.15 e 5.16. A descrição da questão é definida no Código 5.10, que inclui o parâmetro `itens_format`, contendo os itens (I, II, III, IV, ...) para que o estudante verifique qual(is) destes itens contém apenas afirmações verdadeiras, neste caso, elementos do conjunto dos operadores relacionais. Uma variação desta questão é apresentada na Figura 5.5.

Para implementar esta questão, foram definidos dois conjuntos: `setTrue`, que contém operadores relacionais, e `setFalse`, que reúne operadores não relacionais, como mostrado no Código 5.11. Essa abordagem pode ser generalizada para outros conjuntos, em que o primeiro agrupa afirmações verdadeiras e o segundo afirmações falsas. Os parâmetros da questão são configuráveis, possibilitando a criação de diversos cenários de avaliação. O número de itens da questão (`num_itens`) pode ser ajustado, respeitando o limite máximo de 10 itens. Além disso, é possível definir o número de alternativas exibidas (`num_alternativas`) e a quantidade de operadores presentes em cada item

Questão:

```

1 Preencha a tabela com os valores corretos das variáveis contador e soma
2 em cada iteração do laço, para valores de entrada específicos do método.
3 Além disso, considere os números das linhas do código para preencher a
4 tabela, a fim de facilitar a referência. Considere inicialmente
5 maximo=[[code:N1]], contador=[[code:N2]] e soma=[[code:N3]].
6
7 \begin{verbatim}
8 1. def gerar_gabarito(maximo, contador, soma):
9 2.     while contador <= maximo:
10 3.         if contador % 2 == 0:
11 4.             soma = soma + contador
12 5.         else:
13 6.             soma = soma - contador
14 7.         contador = contador + 1
15 \end{verbatim}
16
17 \begin{tabular}{|c|c|c|c|} \hline
18 \textbf{linha} & \textbf{maximo} & \textbf{contador} & \textbf{soma} \\ \hline
19 & & & \\ \hline
20 & & & \\ \hline
21 & & & \\ \hline
22 & & & \\ \hline
23 & & & \\ \hline
24 & & & \\ \hline
25 & & & \\ \hline
26 & & & \\ \hline
27 & & & \\ \hline
28 & & & \\ \hline
29 & & & \\ \hline
30 \end{tabular}
31
32 {\color{blue} {\small
33 \begin{verbatim}
34 [[code:tabela]]
35 \end{verbatim}
36 }}}
```

Código 5.8: Exemplo prático de teste de mesa paramétrico utilizando `sys.settrace` – Parte 1: Descrição de questão.

(`num_afirmacoes_por_item`). A questão também oferece flexibilidade para determinar quantos itens são classificados como corretos (`num_itens_corretos`), permitindo ajustar o grau de dificuldade da avaliação.

5.3.1 Explicação do método `gerar_QM_itens`

O Código 5.12 apresenta o início do método `gerar_QM_itens`, implementado no arquivo em [Utils.py](#), disponível no GitHub. Este método é responsável por gerar os itens da questão.

A partir da linha 14, o código é estruturado em vários métodos auxiliares que encapsulam as

Questão:

```

1  [[def:
2  import random, sys
3  global trace
4
5  def trace(frame, event, arg_unused):
6      global output
7      if event == 'line':
8          lineno = frame.f_lineno # pega a linha
9          locals_dict = frame.f_locals # pega todas variáveis locais
10         maximo = locals_dict.get('maximo') # pega a variável maximo
11         contador = locals_dict.get('contador') # pega a variável contador
12         soma = locals_dict.get('soma') # pega a variável soma
13         # armazena a string na lista
14         output.append(f'{lineno-16:3d} {maximo:7d} {contador:7d} {soma:7d}\n')
15     return trace
16
17 def gerar_gabarito(maximo, soma, contador):
18     while contador <= maximo:
19         if contador % 2 == 0:
20             soma = soma + contador
21         else:
22             soma = soma - contador
23         contador = contador + 1
24
25 # Exemplo de uso:
26 while True:
27     N1, N2, N3 = random.sample(range(4, 20), 3)
28     output = []
29     output.append("linha maximo contador soma\n")
30     output.append("-----\n")
31     sys.settrace(trace) # Inicializa o rastreamento
32     gerar_gabarito(N1, N2, N3)
33     sys.settrace(None) # Finaliza o rastreamento
34     tabela = ''.join(output) # converte a lista de strings em uma string
35     if 10 < len(tabela.split('\n')) < 15:
36         break
37     []]

```

Código 5.9: Exemplo prático de teste de mesa paramétrico utilizando `sys.settrace` – Parte 2: Bloco de código em Python.

diferentes etapas do processo de geração de itens. Esses métodos auxiliam na organização do código e facilitam a manutenção. Eles são apresentados nos Códigos 5.13, 5.15 e 5.16.

5.3.2 Explicação do método gerar_QM_itens: Verifica parâmetros

O método `verifica_parametros` valida os parâmetros para geração de variações de questões, focando em garantir a formação adequada dos itens com base em restrições matemáticas e combinatórias.

#2453 1. Preencha a tabela com os valores corretos das variáveis contador e soma em cada iteração do laço, para valores de entrada específicos do método. Além disso, considere os números das linhas do código para preencher a tabela, a fim de facilitar a referência. Considere inicialmente maximo=19, contador=4 e soma=18.

```
1. def gerar_gabarito(maximo, contador, soma):
2.     while contador <= maximo:
3.         if contador % 2 == 0:
4.             soma = soma + contador
5.         else:
6.             soma = soma - contador
7.         contador = contador + 1
```

linha	maximo	contador	soma
2	19	18	4
3	19	18	4
4	19	18	4
7	19	18	22
2	19	19	22
3	19	19	22
6	19	19	22
7	19	19	3
2	19	20	3

Figura 5.4: Recorte do PDF gerado para a questão definida nos Códigos 5.8 e 5.9.

Questão:

```
1 Qual(is) do(s) seguinte(s) item(ns) conta apenas com operadores relacionais?
2
3 \begin{enumerate}[label=(\Roman*)]\itemsep0pt\parskip0pt\parsep0pt
4 [[code:itens_format]]
5 \end{enumerate}
```

Código 5.10: Exemplo de QM paramétrica de operadores relacionais – Parte 1: Descrição de questão.

#2474 1. Qual(is) do(s) seguinte(s) item(ns) conta apenas com operadores relacionais?

- (I) \leq , $<$, \geq , $=$
 - (II) \leq , $\%$, and, $=$
 - (III) $=$, \geq , $<$, $>$
 - (IV) not, \leq , $>$, $\%$
 - (V) $>$, or, not, $=$

A.*₃I, II B.#₀I, III C.*₄II, V D.*₁IV, V E.*₂I, IV

Figura 5.5: Recorte do PDF gerado para a QM com operadores relacionais.

Sejam:

- T e F , os números de afirmações verdadeiras e falsas, respectivamente;

Questão:

```

1  [[
2  # Lista de afirmações verdadeiras: operadores relacionais
3  setTrue = ['\\>', '\\>=\\', '\\<=\\', '\\<\\', '\\!=\\', '\\==\\']
4
5  # Lista de afirmações falsas: operadores não relacionais
6  setFalse = ['\\text{and}', '\\text{not}', '\\text{or}',
7             '\\=(\\)', '\\+(\\)', '\\(%\\)']
8
9  # Definição dos parâmetros da questão
10 num_itens = 5           # N. itens na questão (máximo: 10)
11 num_alternativas = 5    # N. alternativas (máximo: 15)
12 num_itens_corretos = 2 # N. itens corretos na questão (máximo: num_itens)
13 num_afirmacoes_por_item = 4
14 # N. afirmacoes em cada item (menor que quant. de setTrue)
15
16 # chamada do método gerar_QM_itens
17 itens_format, descricoes = gerar_QM_itens(setTrue, setFalse, num_itens,
18                                             num_alternativas, num_itens_corretos, num_afirmacoes_por_item)
19
20     '''' Alternativas
21 [[code:descricoes[0]]]
22 [[code:descricoes[1]]]
23 ...
24 ...
25 ]]

```

Código 5.11: Exemplo de QM paramétrica de operadores relacionais – Parte 2: Bloco de código em Python com as declarações e a chamada do método `gerar_QM_itens`.

Questão:

```

1  # Parte 1: Bloco principal
2
3  def gerar_QM_itens(
4      setTrue=['1', '3', '5'],
5      setFalse=['2', '4'],
6      num_itens=5,           # N. itens na questão (máximo: 10)
7      num_alternativas=10,   # N. alternativas (máximo: 15)
8      num_itens_corretos=3,  # N. itens corretos na questão (máximo: num_itens)
9      num_afirmacoes_por_item=3):
10     # N. afirmacoes em cada item (menor que quant. de setTrue)
11
12     global itens_format, itens_str
13
14     itens_str = ['I', 'II', 'III', 'IV', 'V', 'VI', 'VII', 'VIII', 'IX', 'X']
15     # ...

```

Código 5.12: Método `gerar_QM_itens` – Parte 1: Bloco principal.

- N , o número total de itens;
- A , o número de alternativas;
- C , o número de itens corretos;
- P , o número de afirmações por item.

O método valida as seguintes condições:

1. Restrições de consistência:

$$C < N, \quad P \leq T \quad (5.1)$$

2. Combinações de itens corretos:

$$\binom{T}{P} \geq C \quad (5.2)$$

3. Formação de itens da questão:

$$\binom{T+F}{P} \geq N \quad (5.3)$$

4. Geração de alternativas (baseada no conjunto das partes dos itens):

$$2^N - 1 \geq A \quad (5.4)$$

O número de variações possíveis é calculado por:

$$\binom{\binom{T+F}{P}}{N} \quad (5.5)$$

O método retorna `pode_gerar` (variável lógica) e `itens_format` (mensagens de erro, se existirem).

Apenas como ilustração, ao modificar a chamada do método no Código 5.11, o Código 5.14 apresenta quatro cenários que podem invalidar a geração de questões, cada um representando uma restrição estrutural ou combinatória distinta.

A Figura 5.6 ilustra o PDF gerado pelo MCTest utilizando a primeira chamada do método `gerar_QM_itens`, com parâmetros inválidos.

5.3.3 Explicação do método `gerar_QM_itens`: Outros métodos

Esta seção apresenta métodos auxiliares do `gerar_QM_itens`, responsáveis por gerar e validar itens de questões de forma estruturada. Cada método desempenha uma função específica, desde a criação e embaralhamento dos itens até a verificação da conformidade com os critérios estabelecidos.

Os métodos a seguir estão apresentados no Código 5.15.

`gerar_itens()`: Este método utiliza a biblioteca `random` para embaralhar as listas de operadores `setTrue` e `setFalse`. Em seguida, gera uma lista de `num_itens` de itens aleatórios contendo `num_afirmacoes` de elementos, selecionados a partir da concatenação de `setTrue` e `setFalse`;

Questão:

```

1  # Verifica parâmetros para criar variações
2  # Parte 2 - Método verifica_parametros
3  def verifica_parametros(T, F, N, P, C, A):
4      from scipy.special import comb
5      output, pode_gerar = [], True
6      T, F = len(T), len(F)
7
8      # Valida consistência básica e combinatória de itens
9      if C >= N or P > T:
10         output.append(
11             f"ERRO: Não é possível gerar variações com {N} itens e {C} corretos")
12         pode_gerar = False
13     if comb(T, P) < C:
14         output.append(
15             f"ERRO: Não é possível gerar {C} itens distintos com {P} afirmações.")
16         pode_gerar = False
17     if comb(T + F, P) < N:
18         output.append(f"ERRO: Não é possível gerar {N} itens totais.")
19         pode_gerar = False
20
21     # Nova validação: o número de alt. deve ser menor que o Power Set dos itens
22     total_alt_posseiveis = (2**N) - 1
23     if total_alt_posseiveis < A:
24         output.append(
25             f"ERRO: {N} itens geram no máximo {total_alt_posseiveis} alt. únicas.")
26         pode_gerar = False
27
28     variacoes = (f'Número de variações possíveis: '
29                  f'{comb(comb(T + F, P):.0f},{N}) = {comb(comb(T + F, P), N):.0f}')
30     itens_format = ''
31     if not pode_gerar:
32         output.extend([
33             f'Items distintos com {P} afirmações verdadeiras: ',
34             f'{comb({T},{P}) = {comb(T, P):.0f},',
35             f'Items distintos com afirmações V+F: ',
36             f'{comb({T + F},{P}) = {comb(T + F, P):.0f}, variacoes}'])
37     itens_format = "\nitem " + "\nitem ".join(output)
38     itens_format += (f'\nitem T = {T}\nitem F = {F}\nitem N = {N}'
39                      f'\nitem A = {A}\nitem C = {C}\nitem P = {P}')
40     itens_format = itens_format.replace('_', '\_')
41     return pode_gerar, itens_format
42
43     pode_gerar, itens_format = verifica_parametros(
44         setTrue, setFalse, num_itens, num_afirmacoes_por_item,
45         num_itens_corretos, num_alternativas)
46     if not pode_gerar:
47         print(itens_format)
48         descricoes = ['' for _ in range(num_alternativas)]
49     # ...

```

Código 5.13: Método gerar_QM_itens – Parte 2: Método verifica_parametros().

Questão:

```

1  # 1. Restrição Inicial
2  # Exemplo: 2 itens corretos em 2 itens, tentando gerar 10 alternativas
3  gerar_QM_itens(setTrue=['1', '3', '5'], setFalse=['2', '4'],
4                  num_itens=2, num_alternativas=10,
5                  num_itens_corretos=2, num_afirmacoes_por_item=3)
6
7  # 2. Insuficiência de Combinações de Itens Corretos
8  # Exemplo: Tentar criar 4 itens corretos com apenas 2 afirmações verdadeiras
9  gerar_QM_itens(setTrue=['1', '3'], setFalse=['2', '4', '6'],
10                 num_itens=5, num_alternativas=10,
11                 num_itens_corretos=4, num_afirmacoes_por_item=2)
12
13 # 3. Impossibilidade de Gerar Itens Totais
14 # Exemplo: Tentar criar 6 itens com poucas afirmações disponíveis
15 gerar_QM_itens(setTrue=['1', '3'], setFalse=['2'],
16                 num_itens=6, num_alternativas=10,
17                 num_itens_corretos=3, num_afirmacoes_por_item=2)
18
19 # 4. Limitação de Alternativas
20 # Exemplo: Tentar gerar 10 alternativas com poucos itens
21 gerar_QM_itens(setTrue=['1', '3', '5'], setFalse=['2', '4'],
22                 num_itens=5, num_alternativas=10,
23                 num_itens_corretos=3, num_afirmacoes_por_item=1)

```

Código 5.14: Exemplo de chamadas de `gerar_QM_itens` com `verifica_parametros` retornando `pode_gerar = False`.

#2474 1. Qual(is) da(s) seguinte(s) alternativa(s) conta apenas com operadores relacionais?

- (I) ERRO: Não é possível gerar variações com 10 alternativas, 2 itens e 2 item(ns) correto(s)
- (II) ERRO: Não é possível gerar 2 itens distintos com 3 afirmações corretas por item.
- (III) ERRO: Não é possível gerar 10 alternativas
- (IV) Itens distintos com 3 afirmações verdadeiras: $\text{comb}(3,3) = 1$
- (V) Itens distintos com afirmações V+F: $\text{comb}(5,3) = 10$
- (VI) Número de variações possíveis: $\text{comb}(10,2) = 45$
- (VII) T = 3
- (VIII) F = 2
- (IX) N = 2
- (X) A = 10
- (XI) C = 2
- (XII) P = 3

A.[#0](#) B.[#0](#) C.[#0](#) D.[#0](#) E.[#0](#)

Figura 5.6: Recorte do PDF gerado para a QM com os parâmetros inválidos da primeira chamada do Código 5.14.

`gerar_itens_corretos()`: Este método verifica os itens gerados, contando o número de operadores relacionais presentes em cada item. Apenas os itens que possuem exatamente `num_afirmacoes` de operadores relacionais são considerados corretos e são armazenados em uma lista chamada `itens_corretos`;

`gerar_indices_itens_corretos()`: Este método tem como objetivo gerar uma lista de índices dos itens corretos, enquanto também cria um formato de exibição dos itens exibição na descrição

da questão. A variável global `itens_format` é utilizada para acumular uma representação formatada dos itens, que é construída no formato de lista (`\item`) para ser incorporada em um ambiente de lista em L^AT_EX. O código percorre os itens gerados, formata cada item como uma *string* e adiciona o índice correspondente à lista de itens corretos (`corretos`). Ao final, a método retorna os índices dos itens corretos, ordenados, para facilitar referências posteriores.

5.3.4 Explicação do método `gerar_QM_itens`: Laço Principal

O Código 5.16 encapsula o processo central para a geração automatizada de questões objetivas contendo itens aleatórios. As principais etapas são:

1. Geração dos itens através do método `gerar_itens()`, obtendo os operadores a partir de conjuntos pré-definidos;
2. Identificação dos itens corretos por meio do método `gerar_itens_corretos()`, que avalia a presença de operadores relacionais;
3. Verificação do número de itens corretos versus o parâmetro `num_itens_corretos`. Se diferente, gera nova questão;
4. Obtenção dos índices dos itens corretos com `gerar_indices_itens_corretos()` e formatação dos itens a serem incluídos na descrição da questão;
5. Construção da alternativa correta a partir dos itens corretos;
6. Geração de alternativas inválidas aleatoriamente distintas;
7. Nova verificação do número de alternativas. Se diferente, gera questão nova.

Dessa forma, é assegurada a obtenção de uma questão válida que atenda aos parâmetros estabelecidos de maneira aleatória. O processo se repete em laço até encontrar uma solução factível.

5.3.5 Aplicação do método `gerar_QM_itens`

Em 22 de fevereiro de 2025, o método `gerar_QM_itens` foi aplicado no processo seletivo da Especialização em Tecnologia de Sistemas da Informação (TSI) da UFABC, com a participação de 273 candidatos em diferentes polos do estado de São Paulo. As provas, geradas no MCTest, continham 30 questões parametrizadas sobre Fundamentos de Computação, sendo 5 questões sobre operações binárias e representação hexadecimal, 1 questão sobre endereçamento direto em memória RAM e 24 questões semelhantes à apresentada na Figura 5.5, compostas por conjuntos de afirmações categorizadas em `setTrue` e `setFalse`. O Código 5.17 exemplifica a configuração utilizada, com os parâmetros `num_itens=5`, `num_alternativas=10`, `num_itens_corretos=2` e `num_afirmacoes_por_item=2`.

Com base nas equações apresentadas na Seção 5.3.2, é possível gerar até 3003 variações de cada questão, sem considerar a ordem das 10 alternativas. Calcular as possíveis variações com os valores fornecidos:

Questão:

```

1  # Parte 3 - Outros Métodos
2
3  def gerar_itens(setTrue, setFalse, num_itens, num_afirmacoes):
4      todas_afirmacoes = setTrue + setFalse # Junta todas as afirmações
5      random.shuffle(todas_afirmacoes) # Embaralha a lista para aleatoriedade
6
7      # Gerar combinações únicas usando um conjunto para evitar duplicatas
8      itens = set()
9      while len(itens) < num_itens:
10         item = tuple(sorted(random.sample(todas_afirmacoes, num_afirmacoes)))
11         itens.add(item)
12
13     return list(itens) # Converte para lista antes de retornar
14
15 def gerar_itens_corretos(setTrue, setFalse, num_itens, num_afirmacoes, itens):
16     # Lista para armazenar os itens que atendem aos critérios
17     itens_corretos = []
18     for item in itens:
19         # Contar a quantidade de afirmações verdadeiras presentes no item
20         cont = 0
21         for op in item:
22             if op in setTrue:
23                 cont += 1
24         # Se o número de afirmações verdadeiras for igual ao número desejado
25         if cont == num_afirmacoes:
26             itens_corretos.append(item)
27     return itens_corretos
28
29 def gerar_indices_itens_corretos(itens, itens_corretos):
30     # Gerar índices dos itens corretos e criar itens_format
31
32     # formatar itens e declarar como variável global
33     global itens_format, itens_str
34     itens_format = ''
35
36     corretos = []
37     for i, item in enumerate(itens):
38         s = f'{i}; '.join(item)}
39         #itens_format += f"\item {itens_str[i]}. {s}\n"
40         itens_format += f"\item {s}\n"
41
42     # Adicionar o índice do item à lista de itens corretos
43     for c in itens_corretos:
44         if c == item:
45             corretos.append(i)
46
47     return sorted(corretos) # Ordenar os índices dos itens corretos
48     # ...

```

Código 5.15: Método gerar_QM_itens – Parte 3: Outros métodos.

Questão:

```

1  # Parte 4 - Laço principal
2
3  while pode_gerar:
4      itens = gerar_itens(setTrue, setFalse, num_itens,
5                          num_afirmacoes_por_item)
6
7      itens_corretos = gerar_itens_corretos(setTrue, setFalse, num_itens,
8                                              num_afirmacoes_por_item, itens)
9
10     # Se o n. de itens corretos não for igual ao desejado, gerar uma nova questão
11     if len(itens_corretos) != num_itens_corretos: continue
12
13     corretos = gerar_indices_itens_corretos(itens, itens_corretos)
14
15     # Alternativa correta (fixa com os índices verdadeiros)
16     aux_correto = sorted([itens_str[i] for i in corretos])
17
18     # 1. Gera o espaço amostral de todas as combinações possíveis de itens
19     todas_combinacoes = []
20     for r in range(1, num_itens + 1):
21         for combo in itertools.combinations(itens_str[:num_itens], r):
22             todas_combinacoes.append(list(combo))
23
24     # 2. Remove a correta para não sorteá-la como distrator
25     if aux_correto in todas_combinacoes:
26         todas_combinacoes.remove(aux_correto)
27
28     # 3. Sorteia as erradas necessárias (num_alternativas - 1)
29     erradas_sorteadas = random.sample(todas_combinacoes, num_alternativas - 1)
30
31     # 4. Une Correta + Erradas (a primeira posição é a correta para o MCTest)
32     itens_comb = [aux_correto] + [sorted(e) for e in erradas_sorteadas]
33
34     # Montar as descrições das alternativas
35     descricaoes = [f'{"; ".join(letra)}' for letra in itens_comb]
36
37     # Se o número de descrições não for igual ao número de alternativas,
38     # gerar uma nova questão
39     if len(descricaoes) != num_alternativas: continue
40
41     # Encerrar o loop, pois a questão foi gerada com sucesso
42     break
43
44     return itens_format, descricaoes

```

Código 5.16: Método gerar_QM_itens – Parte 4: Laço principal.

- $A = 10$ (número de alternativas);
- $N = 5$ (número total de itens);
- $F = 3, T = 3$ (número de afirmações falsas e verdadeiras);

Questão:

```

1  setTrue = [
2    "Operadores aritméticos (+, -, *, /) são usados para realizar cálculos \
3     matemáticos.",
4    "Operadores lógicos (e, ou, não) são usados para combinar ou negar condições.",
5    "Estruturas de controle (se, senão, enquanto, para) controlam o fluxo de \
6     execução do programa.",
7  ]
8  setFalse = [
9    "Operadores lógicos são usados principalmente para realizar cálculos matemáticos.",
10   "Estruturas de controle não são necessárias para escrever programas.",
11   "Um loop 'enquanto' repete um bloco de código um número específico de vezes.",
12 ]
13 num_itens=5           # N. itens na questão (máximo: 10)
14 num_alternativas=10    # N. alternativas (máximo: 15)
15 num_itens_corretos=2  # N. itens corretos na questão (máximo: num_itens)
16 num_afirmacoes_por_item=2 # N. afirmações em cada item (menor que quant. de setTrue)

```

Código 5.17: Trecho de QM paramétrica utilizada no processo seletivo do TSI.

- $P = 2$ (número de afirmações por item);
- $C = 2$ (número de itens corretos).

Agora, aplicando as fórmulas:

1. Combinações de itens corretos:

$$\binom{T}{P} = \binom{3}{2} = \frac{3!}{2!(3-2)!} = 3 \quad (5.6)$$

Como $3 \geq C = 2$, a restrição inicial apresentada na Seção 5.3.2 é satisfeita;

2. Formação de itens da questão:

$$\binom{T+F}{P} = \binom{3+3}{2} = \binom{6}{2} = \frac{6!}{2!(6-2)!} = 15 \quad (5.7)$$

Como $15 \geq N = 5$, a restrição inicial apresentada na Seção 5.3.2 é satisfeita;

item Geração de alternativas: Diferente do modelo de tamanho fixo, a nova implementação permite qualquer combinação de itens. Com $N = 5$:

$$2^N - 1 = 2^5 - 1 = 31 \quad (5.8)$$

Como $31 \geq A = 10$, a restrição é satisfeita com folga, aumentando a diversidade dos distratores;

3. Número total de variações possíveis: Considerando a composição dos itens:

$$\binom{(T+F)}{N} = \binom{15}{5} = 3003 \quad (5.9)$$

Ou seja, existem **3003 variações possíveis** para a geração das questões.

No MCTest foi criado um Exame com essas 30 questões parametrizadas e 10 variações. Foi escolhido salvar um arquivo no formato XML com essas 300 variações, por exemplo `test.xml`. Em seguida, todo texto no formato L^AT_EX deste arquivo foi convertido para o formato HTML para importação no Moodle, conforme Código 5.18 (para mais detalhes, ver Capítulo 9 – Variações de exames com QM+QT para o Moodle). Esse código necessita instalar com `pip` a biblioteca `latex2mathml`. O código lê um arquivo XML (`test.xml`) e processa os textos dentro das `tags <text>`, substituindo expressões L^AT_EX por equivalentes em HTML. Para isso, remove configurações desnecessárias, converte listas enumeradas L^AT_EX (`\begin{enumerate}` e `\item`) para `` e ``, transforma expressões matemáticas delimitadas por `\(...\)` em MathML usando `latex2mathml`, e substitui `\text{...}` por `...`. O resultado atualizado é salvo em um novo arquivo (`output.xml`).

Como demonstração deste processo seletivo no TSI, a Figura 5.7 apresenta a distribuição de frequência dos acertos nas 30 questões para cada um dos 273 candidatos, seguindo uma distribuição normal.

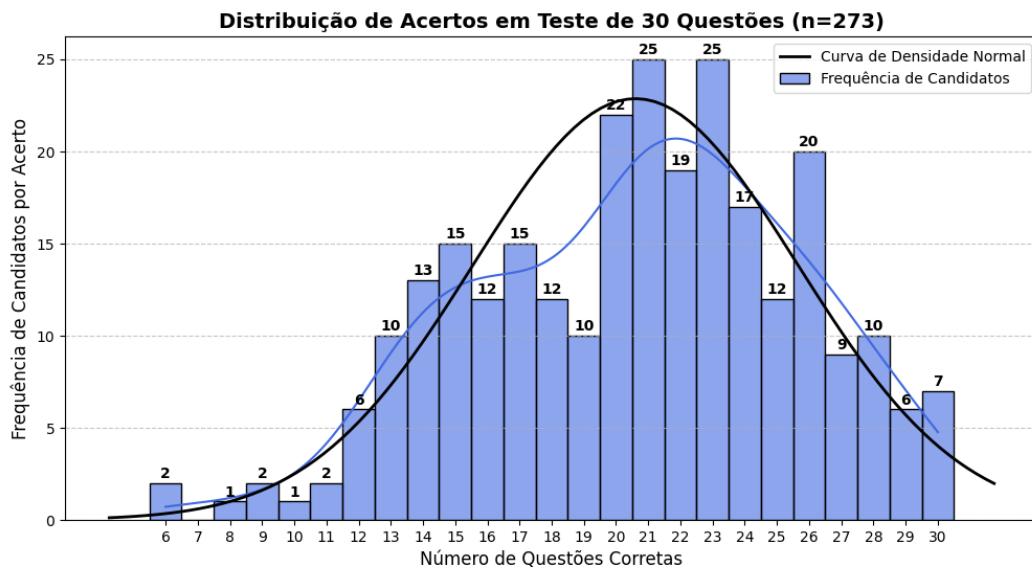


Figura 5.7: Frenquênciados acertos dos 273 candidatos no processo seletivo do TSI.

Assim, esse estilo de questão contendo itens foi facilmente adaptado para outros tipos de conjuntos `setTrue` e `setFalse`, não apenas os relacionais e não relacionais. Bastou que o primeiro contivesse elementos com premissas verdadeiras e o segundo, com premissas falsas. Por exemplo, o conjunto de elementos planares versus não planares em geometria; o conjunto dos animais vertebrados versus invertebrados em biologia; o conjunto dos números pares versus ímpares em matemática; o conjunto dos caracteres alfanuméricos permitidos em um nome de arquivo/variável versus caracteres proibidos em computação; entre outros.

Questão:

```

1  from bs4 import BeautifulSoup
2  import re
3  import latex2mathml.converter
4
5  def clean_latex(latex_text):
6      """Remove configurações LaTeX desnecessárias."""
7      return re.sub(r"\[label=\(\\\Roman\*\)\]\\\itemsepOpt\\parskipOpt\\parsepOpt",
8                  "", latex_text).strip()
9
10 def latex_to_html(latex_text):
11     """Converte expressões LaTeX para HTML e ajusta listas."""
12     text = clean_latex(latex_text)
13     text = text.replace(r"\begin{enumerate}", "<ol type='I'>") \
14         .replace(r"\end{enumerate}", "</ol>")
15     text = re.sub(r"\\\item\s*", "<li>", text) + "</li>"
16
17     # Converte expressões matemáticas LaTeX para MathML
18     for expr in re.findall(r"\\\\((.*?))\\\"", text):
19         text = text.replace(f"\\\\({expr})\\\"", \
20                             latex2mathml.converter.convert(expr))
21
22     # Converte \text{...} para <span>...</span>
23     return re.sub(r"\\text\{(.*)\}", r"<span>\1</span>", text).strip()
24
25 # Lê e processa o arquivo XML
26 with open("test.xml", "r", encoding="utf-8") as file:
27     soup = BeautifulSoup(file, "xml")
28
29 for tag in soup.find_all("text"):
30     if tag.string and tag.parent.name in ["questiontext", "answer"]:
31         tag.string = latex_to_html(tag.string)
32
33 # Salva o resultado
34 with open("output.xml", "w", encoding="utf-8") as file:
35     file.write(str(soup))

```

Código 5.18: Código para converter arquivo XML do formato L^AT_EX para HTML.

5.4 QT paramétrica, com código

Nesta seção, são abordadas QTs paramétricas que envolvem código, introduzindo o ambiente Moodle com o *plugin* VPL. Em seguida, é apresentado um exemplo de atividade que combina essas duas ferramentas. Posteriormente, é fornecido um exemplo de atividade que utiliza a combinação MCTest, Moodle e VPL.

5.4.1 Introdução ao Moodle e ao VPL

O Moodle (moodle.org) é uma plataforma de aprendizagem *online* amplamente utilizada em instituições de ensino em todo o mundo ([PRESEDO, 2015](#)). Ela oferece recursos abrangentes

5.4. QT PARAMÉTRICA, COM CÓDIGO

para apoiar o ensino e a aprendizagem, incluindo a capacidade de criar e disponibilizar atividades interativas para os estudantes. Um dos *plugins* populares do Moodle é o VPL (*Virtual Programming Lab* – vpl.dis.ulpgc.es) (PINO, 2012), desenvolvido especialmente para a correção automática de exercícios de programação (EPs).

O *plugin* VPL permite que os professores criem EPs e ofereçam aos estudantes um ambiente virtual para escrever e testar seu código. Essa ferramenta é particularmente útil em disciplinas de programação, nas quais os estudantes precisam praticar e aprimorar suas habilidades de codificação.

O VPL oferece recursos avançados de avaliação e correção automática. Os estudantes podem enviar seus códigos para o sistema, e o VPL executará testes automatizados para verificar se o código produz os resultados esperados. Os resultados são fornecidos instantaneamente, permitindo que os estudantes identifiquem os pontos em que cometem erros e os aspectos em que podem melhorar.

Além disso, o VPL permite que os professores configurem casos de teste personalizados para verificar a correção dos programas dos estudantes. Isso significa que os professores podem criar uma variedade de exercícios e testes para avaliar a compreensão e a habilidade dos estudantes em diferentes aspectos da programação.

O uso do VPL no Moodle traz diversos benefícios. Ele promove a prática ativa e produz *feedback* imediato aos estudantes, permitindo que eles experimentem e aprendam com seus erros. Além disso, o VPL agiliza o processo de correção, liberando mais tempo para que os professores se concentrem em fornecer orientação e apoio individualizado aos estudantes.

5.4.2 Questão com exercício de programação no Moodle+VPL

Nesta seção, será apresentada uma QT que requer a solução utilizando o *plugin* VPL do Moodle. Esse tipo de questão é definido neste livro como EP. O estudante é solicitado a inserir o código na atividade correspondente do Moodle, utilizando uma linguagem de programação definida pelo professor. É importante destacar que a questão apresentada a seguir não é paramétrica, embora possua diferentes casos de teste.

Um exemplo de atividade Moodle+VPL

Escreva um programa em Python para calcular o fatorial de um número inteiro fornecido pelo usuário.

Instruções

Segue um exemplo de questão de EP no Moodle que utiliza a linguagem Python para calcular o fatorial de um número:

1. Acesse o Moodle e vá para a disciplina em que deseja adicionar a questão;
2. Clique em “Ativar edição”, para conseguir incluir novas atividades;
3. Escolha um tópico do curso e clique em “Adicionar uma atividade ou recurso” e selecione “Laboratório Virtual de Programação” para criar uma nova atividade VPL;

4. Preencha os detalhes da atividade, como nome, descrição e pontuação. Finalize salvando e mostrando a atividade. Em seguida, clique no ícone de engrenagem para acessar as configurações avançadas da atividade e siga os próximos passos;
5. Na seção “Opções de execução”, selecione a linguagem “PYTHON-3: Using python3 with the first file” como “Script de execução”, deixando apenas “Avaliar” e “Atribuição automática de nota” como “Sim”, por exemplo. Se deixar em branco na linguagem, a atividade irá aceitar códigos em dezenas de linguagens suportadas.
6. Na seção “Casos de teste” (novamente no ícone de engrenagem), você pode definir os casos de teste da seguinte maneira:

Arquivo com os casos de teste:

```
case=caso1
input=5
output="O factorial de 5 é 120."
output="O factorial de 5 é 120"
case=caso2
input=10
output="O factorial de 10 é 3628800."
output="O factorial de 10 é 3628800"
```

7. Na seção “Arquivo de execução”, você pode deixar em branco, pois não é necessário alterações neste exemplo.

O Código 5.19 apresenta um método em Python para calcular o factorial de um número. O factorial de um número inteiro é o produto de todos os números inteiros positivos de 1 até esse número. O método `fatorial()` apresentado neste código utiliza um laço `for` para multiplicar todos os números de 1 até `n` e acumula o resultado na variável `r`. Finalmente, retorna o valor acumulado, que é o factorial de `n`.

Em seguida, o programa solicita ao usuário que digite um número inteiro. Esse número é armazenado na variável `numero`. O programa então chama o método `fatorial(numero)` para calcular o factorial desse número e armazena o resultado na variável `resultado`. Por fim, o programa exibe a frase “O factorial de” seguida do número digitado pelo usuário e do resultado calculado.

Avaliação

Para reproduzir a visão de estudante para a realização de testes, o professor pode clicar na seta para baixo ao lado do seu nome no canto superior direito e clicar em “Mudar papel para...”, escolhendo a opção “Estudante”.

Na atividade VPL, agora, com a visão de estudante, escolha “Editar” para selecionar um novo nome de arquivo, como por exemplo, `teste.py` (necessariamente com a extensão da linguagem escolhida). Para alterar esse nome, basta clicar duas vezes sobre ele.

Código: Exemplo de uso de atividade Moodle+VPL

```
1 def fatorial(n):
2     r = 1
3     for i in range(1, n + 1):
4         r = r * i
5     return r
6
7 numero = int(input())
8 resultado = fatorial(numero)
9 print(f"O fatorial de {numero} é {resultado}.")
```

Código 5.19: Programa em Python para cálculo do fatorial.

Inclua uma solução, conforme apresentado no Código 5.19, clique no ícone para salvar e solicite avaliação. A pontuação para esta questão será baseada no programa fornecido pelo estudante para ler as entradas, calcular o fatorial e gerar as saídas esperadas para os casos de teste fornecidos. Se o estudante fornecer apenas uma saída fixa, como `print("O fatorial de 5 é 120.")`, a questão receberá uma nota de 50% (considerando que foram criados dois casos de teste).

Portanto, é importante incluir uma variedade de casos de teste para avaliar corretamente o programa e identificar possíveis tentativas de trapaça. Quanto mais casos de teste forem utilizados, melhor será a avaliação, uma vez que aumenta a chance de identificar soluções corretas e detectar respostas fixas ou incorretas. Além disso, é possível adicionar arquivos específicos na atividade VPL para verificar se o estudante de fato elaborou o código; essa funcionalidade será introduzida na próxima seção.

A Figura 5.8 ilustra a situação após a solicitação de avaliação, na qual um espaço antes do ponto foi intencionalmente inserido no comando `print`. Esta interface do Moodle é da versão 4.1. Na parte direita da imagem, é possível observar diversas mensagens para cada caso de teste, sendo destacadas as seções `--- Input ---` para a entrada com o comando `input()`, `--- Program output ---` para a saída do programa, e `--- Expected output (exact text)---` para a saída esperada. Removendo esse espaço e solicitando uma nova avaliação, será atribuída `Nota proposta: 100 / 100`, observando a parte superior direito desta figura.

5.4.3 Questão com exercício de programação no MCTest+Moodle+VPL

Um problema na solução de EP utilizando apenas Moodle+VPL é não ser possível criar um conjunto específico de casos de teste para cada estudante. Além disso, não é possível criar uma questão distinta para cada estudante. Para resolver isso, será apresentado nesta seção um tipo de questão, utilizando EP no MCTest+Moodle+VPL, que apresenta dois desafios devido à falta de integração atual entre os bancos de dados do MCTest e do Moodle.

The screenshot shows a Moodle VPL (Virtual Programming Lab) interface. On the left, a code editor window titled "teste.py" contains Python code for calculating factorials. The code defines a function `fatorial` that takes an integer `n` and returns the factorial of `n` using a for loop. It also reads a number from input and prints the result. The code is as follows:

```

1 def fatorial(n):
2     result = 1
3     for i in range(1, n + 1):
4         result *= i
5     return result
6
7 numero = int(input())
8 resultado = fatorial(numero)
9 print(f'O factorial de {numero} é {resultado}.')
10

```

On the right, the student's profile "Francisco de Assis Zampirolli" is shown with a clock icon and a "Nota proposta: 0 / 100". Below it, the "Comentários" section is expanded, showing "Failed tests" for two cases: "caso1" and "caso2".

- Test 1: caso1**: Incorrect program output. Input: 5. Program output: "O factorial de 5 é 120 .". Expected output: "O factorial de 5 é 120.".
- Test 2: caso2**: Incorrect program output. Input: 10. Program output: "O factorial de 10 é 3628800 .". Expected output: "O factorial de 10 é 3628800.".

A summary at the bottom indicates 2 tests run, 0 passed.

Figura 5.8: Recorte da tela do Moodle de uma atividade VPL.

Primeiro desafio

O primeiro desafio consiste em definir a variação da atividade para cada estudante, uma vez que as atividades podem ser individuais. Para solucionar esse problema, o MCTest gera um arquivo no formato CSV chamado `students_variations.csv`, contendo o nome e o email do estudante, juntamente com a variação atribuída. Esse arquivo deve ser inserido no campo “Arquivo de execução” da atividade VPL no Moodle. Além disso, existem vários outros arquivos disponíveis no [mctest](#) do GitHub, detalhados em capítulos futuros deste livro, que também devem ser incluídos nesse campo. A versão utilizada neste segunda edição do livro está na pasta [V13-Moodle_v4.1](#) do GitHub.

Destaque:

É importante destacar que, a partir da versão 5.3 do MCTest, a busca pela variação do estudante deve ser realizada pelo e-mail do estudante no arquivo `students_variations.csv`. Anteriormente, essa busca era realizada pelo nome e sobrenome do estudante, o que apresentava diversos problemas de compatibilidade entre os nomes cadastrados no MCTest e no Moodle, além de questões relacionadas a estudantes com o mesmo nome e sobrenome.

Segundo desafio

O segundo desafio, ainda mais crítico, envolve a definição dos casos de teste para cada variação da questão. Esse desafio é enfrentado através da geração do arquivo `linker.json` pelo MCTest, no formato JSON. Esse arquivo contém as variações da atividade, incluindo as questões e seus respectivos casos de teste. Nesta seção, será abordado o processo de criação de uma QT paramétrica de EP no MCTest+Moodle+VPL. No entanto, é importante ressaltar que todos os detalhes desse processo serão abordados em capítulos futuros, uma vez que esses arquivos CSV e JSON são gerados ao criar um exame no MCTest, especificamente no Capítulo 10 – [Exames com MCTest+Moodle+VPL](#).

Um exemplo simples de cálculo de parcelas sem juros

O VPL considera cada linha como uma entrada de dados de um programa. No Python, essa entrada é capturada utilizando o comando `input()`, que recebe um texto. Por exemplo, o texto "5003.9\n10\n" contém duas entradas. Para uma questão de parcelas sem juros, é possível ter duas entradas e uma saída de dados no Código 5.20 em Python, que pode ser testado em uma atividade VPL no Moodle. Nesse exemplo, o programa captura o valor total e o número de parcelas informados

Código: Exemplo de uso de atividade Moodle+VPL – Parcelas

```
1 valor_total = float(input())
2 parcelas = int(input())
3
4 valor_parcela = valor_total / parcelas
5
6 print(f"{valor_parcela:.2f}")
```

Código 5.20: Programa em Python para cálculo de parcelas.

pelo usuário e calcula o valor de cada parcela. Em seguida, exibe o resultado utilizando o comando `print`, formatando com duas casas decimais.

O MCTest pode gerar automaticamente diversos casos de teste, seguindo uma estrutura específica de dicionário de dados. Por exemplo, a seguir estão apresentados dois casos de teste criados no MCTest para o exemplo de cálculo de parcelas sem juros:

Dicionário usado no MCTest contendo os casos de teste:

```
moodle_cases = {
    "input": ["5009.4\n10\n", "5003.9\n10\n"],
    "output": ["500.94", "500.39"]
}
```

Esse dicionário será formatado no arquivo `linker.json`, que será tratado por vários arquivos que deverão ser inseridos em “Arquivos de execução”, em uma atividade VPL no Moodle, que pode ser equivalente aos seguintes casos de teste.

Arquivo com os casos de teste:

```
case=caso1
input=5009.4
10
output=500.94
case=caso2
input=5003.9
10
output=500.39
```

Para realizar essa tarefa, na descrição da questão no MCTest, é necessário incluir um trecho de código utilizando o comando `comment` que contenha apenas o dicionário de casos de teste. Esse trecho indica ao gerador MCTest para incluir os casos de teste no arquivo `linker.json`, que será importado no Moodle.

Atenção:

O comando `comment` na descrição da questão é exclusivamente designado para a inclusão do dicionário de casos de teste no MCTest. Qualquer utilização além dessa finalidade pode comprometer a integridade do teste, resultando em possíveis complicações na execução da tarefa. A função deste comando é informar ao gerador MCTest sobre os casos de teste a serem integrados ao arquivo `linker.json` para subsequente importação no Moodle.

Criando uma QT paramétrica para integração MCTest+Moodle+VPL

A seguir, será apresentado um código completo referente à questão das parcelas sem juros, com as variáveis e instruções em Python destacadas entre os marcadores `[[code: e]]`. Este exemplo foi adaptado do trabalho de [Zampirolli, Sato, Savegnago, Batista e Kobayashi \(2021\)](#). As variáveis devem ser definidas no bloco de código Python localizado entre os marcadores `[[def: e]]`.

Para criar questões paramétricas, não apenas nos casos de teste, é necessário incluir variáveis aleatórias no enunciado da questão entre `[[code: e]]`. No exemplo das parcelas sem juros, foi

5.4. QT PARAMÉTRICA, COM CÓDIGO

incluído no enunciado o número de parcelas, sendo utilizado apenas o valor da compra como entrada para o caso de teste.

Atenção:

Para garantir que cada estudante receba uma questão diferente, é necessário incluir o maior número possível de variáveis aleatórias no enunciado. Por exemplo, ao lidar com um problema de parcelas sem juros, é possível adicionar variáveis aleatórias como o nome do cliente, o produto em questão, entre outros. Além disso, é importante que esses dados sejam impressos na saída esperada para os casos de teste. Caso contrário, um mesmo código pode funcionar para todas as variações de atividade, ignorando a necessidade de múltiplos casos de teste.

No Código 5.21, é detalhado o enunciado da questão, incorporando a parcela como um parâmetro. Além disso, o primeiro caso de teste é fornecido como exemplo no enunciado. Esta seção inicial é concluída com o bloco `comment`, essencial para que o MCTest comprehenda que se trata de uma questão a ser resolvida pelo estudante como uma atividade VPL no Moodle

Questão:

```
1 Escreva um programa que auxilia uma loja na hora de efetuar uma venda. Seu
2 programa deve perguntar o preço do produto e considerar que a loja parcela
3 este preço em somente em [[code:parcelas]] parcelas (sem juros). O seu
4 programa deve mostrar quanto será o valor de uma parcela (arredondando
5 para duas casas decimais apenas). Veja exemplo a seguir: \\
6
7 \noindent\textrm{Exemplo de Entrada:}
8 \begin{verbatim}
9 [[code:caso0_inp]]
10 \end{verbatim}
11
12 \noindent\textrm{Exemplo de Saída:}
13 \begin{verbatim}
14 [[code:caso0_out]]
15 \end{verbatim}
16
17 % necessário para gerar casos de testes no moodle
18 \begin{comment}
19 [[code:moodle_cases]]
20 \end{comment}
```

Código 5.21: Exemplo de QT paramétrica utilizando MCTest+Moodle+VPL – Parte 1: Descrição de questão.

A segunda parte da questão apresentada no Código 5.21 é fornecida no Código 5.22, que contém o bloco de código Python que define a parte paramétrica, incluindo o dicionário `moodle_cases` que contém os casos de teste. Este Código 5.22 consiste em quatro passos principais, destacados a seguir:

Passo 1: Um número inteiro aleatório entre 8 e 19 é gerado, representando o número de parcelas em uma QT paramétrica;

Passo 2: Os casos de teste são criados. Para cada caso de teste, um valor aleatório para o preço do produto é gerado, e as entradas e saídas correspondentes são formatadas;

Passo 3: Um dicionário contendo os casos de teste é criado e convertido para o formato JSON;

Passo 4: Por fim, um exemplo de caso de teste é mostrado no enunciado da questão. Esse código permite gerar casos de teste e parâmetros para questões paramétricas, proporcionando variação nos valores dos testes e facilitando a criação de questões personalizadas. É importante testar o código em uma IDE antes de utilizá-lo no MCTest.

Atenção:

Dentre os quatro passos apresentados, o professor precisa apenas modificar o Passo 1, no qual os parâmetros do enunciado da questão são definidos. Além disso, o trecho entre `#>>>` e `#<<<` deve ser alterado para criar diferentes casos de teste. Essas modificações permitem criar outras questões paramétricas, com variações nos parâmetros e nos casos de teste, tornando o processo de criação mais flexível e personalizado.

Veja o resultado desta questão criada no MCTest após clicar em “Criar-PDF” na Figura 5.9. Observa-se que, na descrição da questão, é exibido **Integration: Moodle+VPL**, indicando que o MCTest reconheceu essa questão como uma tarefa que o estudante deve realizar em uma atividade VPL no ambiente do Moodle, devido ao reconhecimento do comando `command` no enunciado da questão.

MCTest

Topic: Topico Exemplo
Group:
Short Description: questaoQT-EP-1
Type: QT
Difficulty: 1
Bloom taxonomy: remember
Last update: 2023-07-04
Who created: fzampirolli@ufabc.edu.br
Parametric: YES
Integration: Moodle+VPL

#2454 1. Escreva um programa que auxilia uma loja na hora de efetuar uma venda. Seu programa deve perguntar o preço do produto e considerar que a loja parcela este preço em somente em 8 parcelas (sem juros). O seu programa deve mostrar quanto será o valor de uma parcela (arredondando para duas casas decimais apenas). Veja exemplo a seguir:

Exemplo de Entrada:

5001.4

Exemplo de Saída:

625.17

Figura 5.9: Recorte do PDF gerado para a QT definida nos Códigos 5.21 e 5.22.

5.4. QT PARAMÉTRICA, COM CÓDIGO

Questão:

```
1  [[def:
2  # Antes de prosseguir, é recomendado testar o trecho de código
3  # Python a seguir em uma IDE para garantir seu funcionamento correto.
4  import json
5  import numpy as np # uso da biblioteca numpy para gerar número aleatório
6
7  # Passo 1: Criar os parâmetros do enunciado da questão
8  parcelas = np.random.randint(8,20)
9
10 # Passo 2: Criar os casos de teste
11 inp_list, out_list = [], [] # Listas vazias para armazenar os casos de teste
12 casos_teste = 2 # Número de casos de teste desejado
13 # Aumentar esse número após validar a questão também na atividade VPL do Moodle
14
15 # Para cada caso de teste:
16 for i in range(casos_teste):
17
18     #>>> begin - casos de teste
19
20     # Gerar um valor aleatório para o preço do produto
21     valor_total = np.random.randint(100)/10 + 5001
22
23     # Criar a entrada do caso de teste como uma string
24     inp = str(valor_total) + '\n'
25
26     # Calcular o valor da parcela arredondado para duas casas decimais
27     out = f'{(valor_total/parcelas):.2f}'
28
29     #<<< end - casos de teste
30
31     # Adicionar a entrada e saída do caso de teste às listas
32     inp_list.append(inp)
33     out_list.append(out)
34
35 # Passo 3: Criar o dicionário com os casos de teste
36 cases = {}
37 cases['input'] = np.array(inp_list).tolist()
38 cases['output'] = np.array(out_list).tolist()
39 moodle_cases = json.dumps(cases)
40
41 # Passo 4: Mostrar um exemplo no enunciado da questão
42 caso0_inp = cases['input'][0]
43 caso0_out = cases['output'][0]
44
45 #print(moodle_cases) # para testar em uma IDE
46 ]]
```

Código 5.22: Exemplo de QT paramétrica utilizando MCTest+Moodle+VPL – Parte 2: Bloco de código em Python.

As chaves do dicionário `linker.json`

O dicionário `linker.json` gerado pelo MCTest apresenta uma estrutura hierárquica organizada para representar dados relacionados à geração de casos de teste em um contexto educacional ou de avaliação. A primeira versão deste arquivo foi apresentada no trabalho de [Zampirolli, Sato, Savegnago, Batista e Kobayashi \(2021\)](#). A seguir, é apresentado o conteúdo deste arquivo gerado a partir de um exame contendo apenas a questão de parcelas sem juros, conforme discutido neste capítulo. Os detalhes sobre como gerar este exemplo são apresentados no Capítulo 10 – [Exames com MCTest+Moodle+VPL](#).

Arquivo com os casos de teste:

```
{
  "variations": [
    {
      "variant": "1",
      "questions": [
        {
          "key": "2454",
          "number": "1",
          "file": "Q1",
          "weight": "1",
          "language": ["all"],
          "skills": [],
          "answer": [],
          "description": [],
          "cases": [
            {
              "case": "test_1",
              "input": ["5007.5\n"],
              "output": ["294.56"]},
            {
              "case": "test_2",
              "input": ["5005.8\n"],
              "output": ["294.46"]}]}]}
```

O dicionário `linker.json` é composto por uma série de elementos-chave organizados de maneira hierárquica para representar um conjunto de variações contendo questões parametrizadas. A estrutura geral é dividida em três níveis principais: `"variations"`, `"questions"`, e `"cases"`, detalhados a seguir:

variations: Este nível contém informações sobre as diferentes variantes (ou modelos) do conjunto de questões. Cada variante é identificada por um número único, como exemplificado pela entrada `"variant": "1"`. Dentro de cada variante, encontramos informações específicas sobre

5.5. CONSIDERAÇÕES FINAIS

as questões associadas;

questions: Este nível representa as questões individuais dentro de cada variante. Cada questão é identificada por uma chave única ("key"), e são fornecidas informações adicionais, como o número da questão ("number"), o nome do arquivo associado ("file") – que deve ser o mesmo nome no arquivo incluído na atividade VPL –, o peso da questão ("weight"), a linguagem de programação ("language"), habilidades requeridas ("skills"), solução da questão ("answer"), e uma descrição textual da questão ("description"). Esta descrição pode ser apresentada na interface VPL do Moodle;

cases: Este é o nível mais detalhado, representando os casos de teste específicos para cada questão. Cada caso de teste é identificado por um nome único ("case") e inclui informações sobre as entradas ("input") e as saídas esperadas ("output"). No exemplo fornecido, são apresentados dois casos de teste (`test_1` e `test_2`) para a questão identificada pela chave "2454".

A utilização desse formato de dicionário permite a descrição estruturada de questões de programação parametrizadas, facilitando a integração entre o MCTest e o *plugin* VPL do Moodle para a correção automática de questões individualizadas para cada estudante.

5.5 Considerações finais

Neste capítulo, foram exploradas algumas possibilidades de criação de questões paramétricas no MCTest, combinando recursos de L^AT_EX e Python. Foram apresentados exemplos simples na parte inicial do capítulo, mas é importante ressaltar que é possível criar questões paramétricas muito mais complexas, como a apresentada na Seção 5.3 – [QM paramétrica com múltiplas variações: um estudo de caso](#) (outros exemplos complexos serão apresentados em capítulos futuros). Essa capacidade de criação personalizada é uma das partes mais importantes do MCTest, ao permitir a geração e compartilhamento de questões para os mais diferentes fins.

Embora a criação de questões paramétricas possa exigir habilidades de programação por parte dos professores, o MCTest oferece a vantagem de gerenciar avaliações com um banco de questões reutilizáveis. Isso é especialmente útil quando há um grupo de professores interessados e uma grande demanda de avaliações para milhares de estudantes.

No caso das QMs paramétricas, foram apresentados exemplos de como criar questões com variações de parâmetros, permitindo a geração fácil e rápida de inúmeras questões. Também foram discutidas diferentes técnicas de parametrização, oferecendo ao professor a liberdade de escolher a que melhor se adapta às suas habilidades de programação.

No contexto das QTs paramétricas, foi ressaltado que os conceitos e técnicas utilizados para as QMs também são aplicáveis a esse formato de questão. Entretanto, nas QTs, não há necessidade de se preocupar com as alternativas. Exemplos de questões paramétricas foram apresentados, demonstrando a variação dos parâmetros e proporcionando uma ampla gama de variações para os estudantes.

Por fim, foram abordadas QTs paramétricas que envolvem código e introduzido o ambiente Moodle e o *plugin* VPL. Mostrou-se como criar questões de EPs utilizando o Moodle+VPL e discutiram-se os benefícios dessa abordagem, incluindo a prática ativa, o *feedback* imediato e a agilidade na correção. Também foram mencionados os desafios de integrar o MCTest, Moodle e VPL para questões paramétricas de EP. A combinação dessas ferramentas oferece aos professores uma poderosa maneira de criar e avaliar questões paramétricas personalizadas, enriquecendo a experiência de aprendizagem dos estudantes.

Parte III

Exames no MCTest

Conteúdo

6 Visão geral dos exames	113
7 Exames com Quadro de Respostas (QR)	127
8 Exames com QR+QM e/ou QT	141
9 Variações de exames com QM+QT para o Moodle	163
10 Exames com MCTest+Moodle+VPL	183

Capítulo 6

Visão geral dos exames

Conteúdo

6.1	Acesso à tela de exames	114
6.2	Recorte I da tela de exame	116
6.2.1	Área à esquerda	116
6.2.2	Área central	119
6.2.3	Área à direita	120
6.3	Recorte II da tela de exame – turmas	121
6.4	Recorte III da tela de exame – tópicos	121
6.5	Recorte IV da tela de exame – questões	122
6.6	Recorte V da tela de exame – configurando os detalhes	122
6.7	Considerações finais	125

A realização dos exames desempenha um papel fundamental no MCTest, sendo o elemento central do sistema. Nessa etapa, é possível ter a flexibilidade de incluir uma ou várias turmas de estudantes em um exame, selecionar as questões que farão parte dele, decidir sobre o formato de impressão (como uma questão por folha ou de maneira ecologicamente sustentável, com várias questões por folha), determinar a quantidade de variações a serem criadas e escolher se o exame será impresso ou enviado por e-mail aos estudantes, entre outros aspectos relevantes.

Neste capítulo, será fornecida uma visão geral da tela de exames utilizada no MCTest. Como essa tela é extensa, ela será explicada em partes.

Adicionalmente, nos capítulos subsequentes, serão abordadas as diferentes modalidades de exames já utilizadas, contendo: (1) Questões com apenas o quadro de respostas (QR); (2) Questões com QR+QT, incluindo os enunciados; (3) QTs exclusivamente, que requerem correção manual sendo enviadas aos estudantes por e-mail; (4) QTs exclusivamente, nas quais os estudantes devem fornecer soluções, como exercícios de programação (EPs) em atividades VPL do Moodle. Além disso, será

tratada a combinação desses estilos distintos em um único exame.

6.1 Acesso à tela de exames

Após efetuar o *login*, o professor cadastrado em uma disciplina terá acesso à tela de exames no menu superior direito. Ao acessar essa tela, ilustrada na Figura 6.1, o professor poderá observar que ainda não foram criados exames. Para iniciar o processo de criação de um novo exame, basta clicar no botão “Criar um novo Exame” disponível nessa mesma tela.

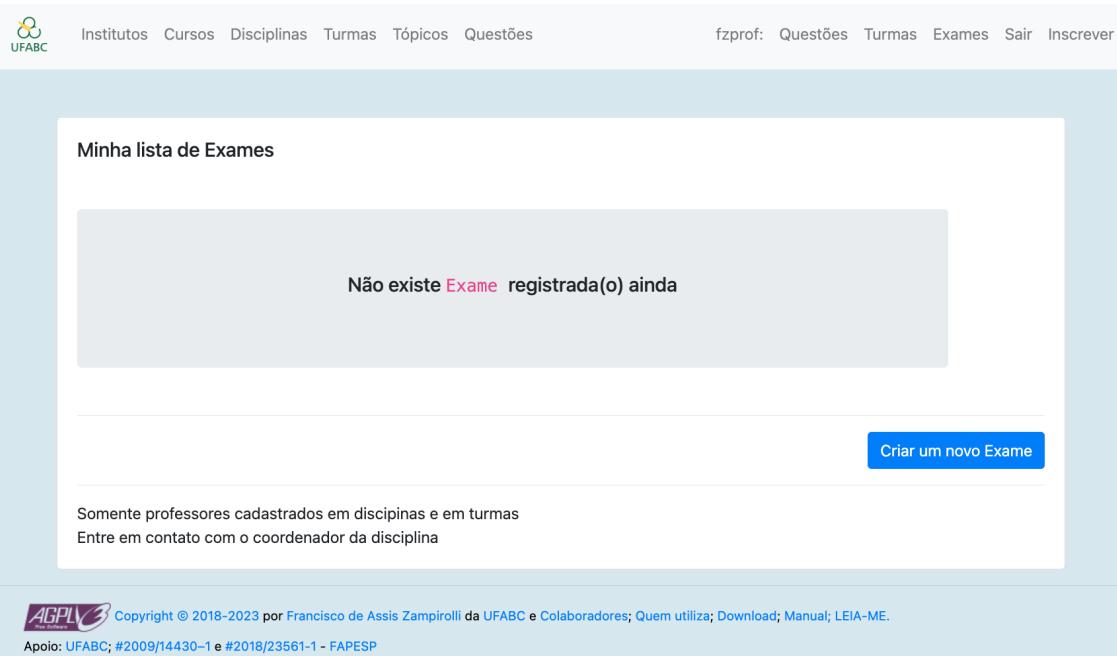


Figura 6.1: Tela de exames do professor.

Na tela de criação de um novo exame (consulte a Figura 6.2), é necessário selecionar uma turma com pelo menos um estudante e atribuir um nome ao exame. Em seguida, clique no botão “Salvar”. A criação de turma por um professor foi apresentada nas Seções 2.2.4 – **Turma** e 3.3.1 – **Criar turma com arquivo CSV**.

Atenção:

1. É essencial ressaltar que um exame só pode ser criado se estiver associado a uma turma criada pelo professor. Portanto, antes de excluir uma turma, certifique-se de apagar todos os exames associados a ela;
2. Ao acessar a tela de atualização do exame, é extremamente importante preencher os campos com muita atenção. Essa parte do sistema é especialmente sensível e requer a criação de várias mensagens de erro para lidar com o preenchimento incorreto dos campos.

6.1. ACESSO À TELA DE EXAMES

Criar um novo Exame

Nome fz-exameTesteQR

Listar Turmas

copiar csv excel pdf imprimir Search:

Disciplina	Código	Tipo	Sala	Período	N. Estudantes	Profs	Ver
<input checked="" type="checkbox"/> Disciplina Exemplo	fz-turmaTeste	PClass	virtual	2023.2	1	fzprof	670

Showing 1 to 1 of 1 entries

As ordenações de colunas ainda não estão funcionando corretamente, use o recurso de Busca

Voltar Salvar

Se você não é um coordenador, crie um exame SOMENTE para sua classe
Entre em contato com o coordenador da disciplina

Figura 6.2: Tela para criar um novo exame.

Após a criação do exame, é possível realizar atualizações e configurar todos os atributos necessários para sua conclusão. Devido à extensão dessa parte na versão atual do MCTest, a tela de atualização de exame foi dividida em várias seções, que serão apresentadas a seguir. A tela de exame pode ser acessada por meio do botão “Atualizar” exibido na Figura 6.1 correspondente ao exame em questão recém criado.

Além disso, antes de prosseguir com a explicação detalhada das seções, é possível gerar um PDF do exame com as configurações padrão ao clicar no botão “Cria-PDF”, conforme detalhado na próxima seção. Esse PDF é ilustrado na Figura 6.3.

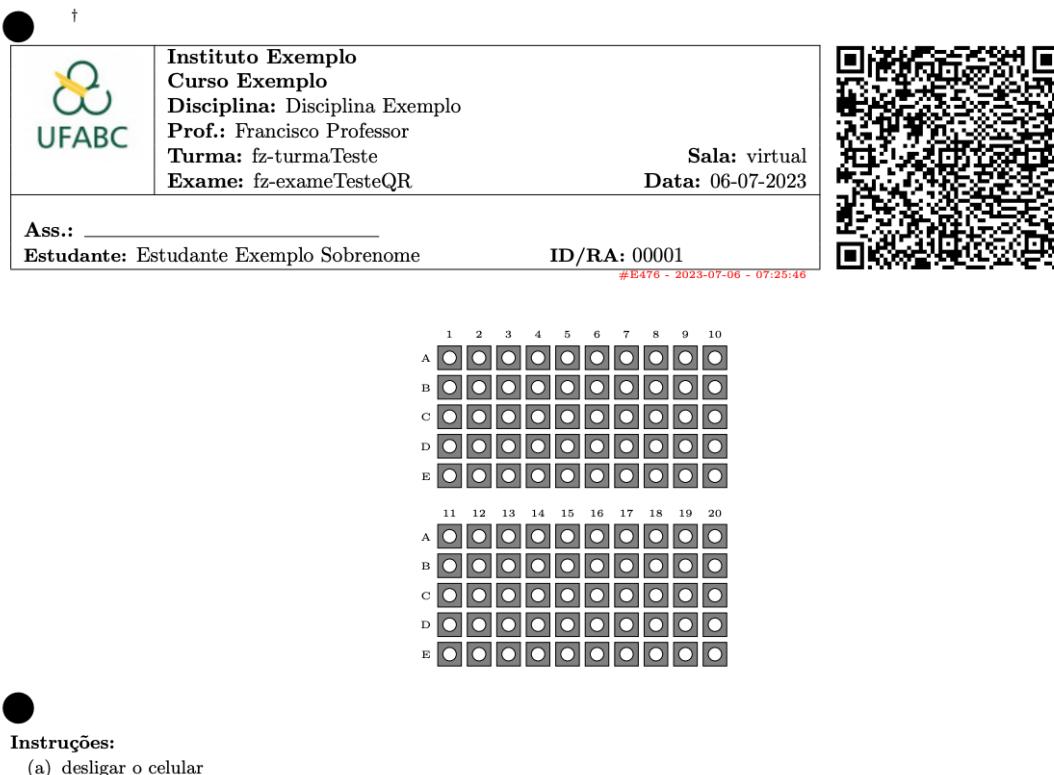


Figura 6.3: Recorte do PDF gerado com as configurações padrão, contendo apenas o cabeçalho e o QR.

6.2 Recorte I da tela de exame

Na Figura 6.4, é apresentado o Recorte I da tela de atualização de exame, que contém os principais botões para a criação e correção de exames. Cada um desses botões ou combinações foi submetido a experimentos e será detalhado em capítulos futuros deste livro. Esses capítulos fornecerão informações mais específicas sobre a utilização e os resultados dessas funcionalidades.

6.2.1 Área à esquerda

É importante ressaltar que o botão “Criar-PDF” deve ser pressionado somente no final do processo de atualização do exame, apesar de estar localizado no início da tela. Essa posição inicial é justificada pelo fato de que, ao compartilhar um exame entre várias turmas, basta alterar a(s) turma(s), clicar em “Salvar” no final da tela de exame e, em seguida, clicar nesse botão “Criar-PDF”. Dessa forma, todas as alterações feitas no exame serão registradas e o PDF final poderá ser gerado com base nas turmas atualizadas.

Um exemplo prático desse recurso é aplicar o mesmo exame em várias ofertas da disciplina, embora essa não seja considerada uma boa prática pedagógica. No entanto, caso o exame seja utilizado como uma atividade formativa e não avaliativa, essa abordagem não representaria um problema significativo. É importante destacar que a utilização de exames idênticos em diferentes ofertas da disciplina pode comprometer a equidade e a validade da avaliação, pois os estudantes de diferentes turmas poderiam compartilhar informações sobre o exame. Recomenda-se a elaboração de

6.2. RECORTE I DA TELA DE EXAME

Atualizar Exame

ATENÇÃO: Cada vez que você clicar em Criar–Variações, um exame diferente será gerado! Depois de imprimir o PDF com o exame, SALVAR NO COMPUTADOR e NÃO MUDE os atributos desta página; Caso contrário, a correção automática não será possível nos exames digitalizados em Upload–PDF.

The screenshot shows the Moodle Exam creation interface. On the left, there's a sidebar with 'Criar-PDF' and a video link. Below it are settings for font size (11pt), exam type (Adaptive), and statistical methods (WPC, MLE, SAT). A note about discoloring is present. In the center, there are three main sections: 'Upload-PDF' (with a note about not changing file attributes), 'Enviar-Retornos-Estudantes Questões dissertativas' (with detailed instructions about generating zip files for each student), and 'Enviar-Retornos-Estudantes' (for sending individual feedback). At the bottom, there are several output format options: 'Criar-Variações' (highlighted in red), 'Json', 'Template', 'Aiken', 'XML', and 'LaTeX+PDF'. The URL 'fz-exameTesteQR' is visible at the bottom of the browser window.

Figura 6.4: Recorte I da tela de exame – ver explicações no texto sobre os botões.

diferentes versões ou variações do exame para cada oferta da disciplina, a fim de garantir a diversidade e a individualidade das avaliações. Para alcançar esse objetivo, é possível criar novas variações do exame.

Nas situações em que se deseja criar exclusivamente exames com QRs, como ilustrado na Figura 6.3, é possível clicar diretamente no botão “Criar-PDF” sem acionar o botão vermelho “Criar-Variações”. A etapa de criação de variações do exame, que envolve a seleção de questões e outros atributos, não é necessária quando o objetivo é gerar apenas exames com QRs.

O botão de vídeo ao lado do botão “Criar-PDF” foi criado com o intuito de fornecer uma explicação detalhada do processo de criação de exames com exercícios de programação e correção automática na atividade VPL do Moodle. Esse tópico será abordado em um capítulo futuro, com mais detalhes e instruções específicas.

Destaque:

A primeira barra de rolagem abaixo do botão “Criar-PDF” na Figura 6.4 define o tamanho da fonte que irá aparecer no PDF. O valor padrão é 11pt, o mínimo é 10pt e o valor máximo é 12pt. É muito importante observar que o QRCode que irá aparecer no PDF deve ficar no interior do retângulo imaginário definido pelos quatro discos pretos, conforme exemplificado na Figura 6.3. Esse QRCode vai deslocando para a direita, conforme o tamanho do nome do estudante. Assim, recomenda-se fortemente verificar se esse QRCode está posicionado corretamente em todos os exames contidos no PDF gerado.

Destaque:

A segunda barra de rolagem, localizada abaixo do botão “Criar-PDF” na Figura 6.4, tem a função de configurar a geração de exames adaptativos, levando em consideração exames anteriores. Até o momento, esse recurso se aplica exclusivamente a exames compostos apenas por questões de múltipla escolha disponíveis no banco de dados. Essa barra de rolagem permite selecionar a quantidade de exames anteriores a serem considerados, variando de 1 a infinitos exames (se o valor for 10, assumimos todos os exames anteriores), ordenados pela data especificada na tela do exame. A configuração padrão é não utilizar esses exames adaptativos, pois as opções dos tipos de estatísticas não estão escolhidas. Mais detalhes sobre esses exames adaptativos estão descritos na Seção 8.6 – Exames adaptativos.

Ao acionar o botão “Criar-PDF” na Figura 6.4, será gerado um arquivo PDF para cada turma. Se houver mais de uma turma vinculada ao exame, o navegador irá retornar um arquivo ZIP contendo todos os arquivos PDF. Esse arquivo em formato PDF, ou o arquivo ZIP, também serão enviados para o e-mail do professor. Para imprimir o PDF, siga as seguintes orientações:

Área à esquerda da Figura 6.4 – botão “Criar-PDF”

1. Escolha uma folha A4 de boa qualidade de impressão;
2. No caso de exames com QM, é importante garantir que os círculos estejam preenchidos corretamente, pois falhas na marcação podem interferir no funcionamento do leitor óptico. Se houver falhas, recomenda-se trocar de impressora e imprimir novamente;
3. Antes de aplicar o exame, é altamente recomendável imprimir uma folha de teste, preenchê-la, digitalizá-la e seguir as instruções indicadas no centro da página para o envio do arquivo digitalizado pelo botão “Upload-PDF”.

Atenção:

Em exames contendo questões do BD, cada vez que o botão “Criar-Variações” for acionado, diferentes exames serão gerados e armazenados no BD, com questões e respostas sorteadas. Após imprimir o PDF do exame, é importante salvar o arquivo no computador com segurança e NÃO ALTERAR mais os atributos da página de exame, exceto ao modificar as turmas e gerar novos PDFs usando o botão “Criar-PDF”. Caso contrário, a correção automática não será possível nos exames digitalizados através do botão “Upload-PDF”, detalhado a seguir, e nem nas correções automáticas em atividades VPL do Moodle. Além disso, ao clicar em “Criar-PDF”, será sorteada uma variação diferente para cada estudante.

6.2.2 Área central

Após a aplicação das atividades com QMs, é possível realizar a correção de forma automática, digitalizando apenas a primeira página de cada estudante e agrupando em um único PDF todos os estudantes da turma. No entanto, antes de prosseguir com esse processo, é necessário seguir as seguintes recomendações:

Área central da Figura 6.4 – botão “Upload-PDF”

1. Antes de digitalizar os exames realizados pelos estudantes, verifique se todos os círculos foram preenchidos corretamente. Se houver falhas ou borrões no contorno dos círculos, o corretor pode não funcionar adequadamente;
2. Digitalize a página de frente (para exames com QMs) em níveis de cinza, com uma resolução de 150 dpi (se a decodificação do código QR falhar, use 200 dpi) e salve como um arquivo PDF por turma;
3. Verifique se os 4 discos pretos estão completamente preenchidos e sem falhas;
4. Se a opção “Respostas” foi selecionada no campo “Questões/Respostas/Ambos” na tela de Exame, no Recorte V, a primeira página do PDF deve conter o gabarito, e todas as questões marcadas no Recorte IV serão desconsideradas;
5. Se a opção “Retorno: SIM” foi escolhida na tela de Exame, ao cadastrar os estudantes da turma, será necessário incluir também o e-mail do estudante. Nesse caso, siga as instruções no lado direito da página “Enviar Retornos aos Estudantes” para cada estudante receber a correção do seu exame por e-mail.

Destaque:

O conteúdo da Figura 6.4 foi modificado para viabilizar a opção do professor fornecer *feedback* aos estudantes, possibilitando o envio tanto das questões quanto dos gabaritos. Essa operação é realizada marcando a opção “Retorno=SIM”, salvando o formulário, selecionando a caixa de verificação abaixo da opção “Upload-PDF” e, por fim, submetendo o arquivo PDF digitalizado ao clicar no botão “Upload-PDF”. Se essa caixa de verificação não for selecionada, apenas a imagem e informações do estudante, juntamente com os pontos obtidos, serão anexados ao PDF.

6.2.3 Área à direita

Esta área é destinada aos exames com QTs e correção manual realizadas nas folhas impressas do exame. No artigo de Zampirolli, Goya, Pimentel e Kobayashi (2018) foi realizado um experimento utilizando esse processo avaliativo e será detalhado no Capítulo 12 – MCTest versões 4, 4.G e 5. No botão “Enviar Retornos aos Estudantes Questões Dissertativas”, na página do Exame, na Figura 6.4, se foram utilizados exames com QTs no formato NÃO ecológico, então foi gerada uma questão por folha, correto? Se desejar enviar as correções feitas manualmente nos PDFs digitalizados para cada estudante, será necessário seguir alguns passos:

Área à direita da Figura 6.4 – botão “Enviar Retornos aos Estudantes Questões Dissertativas”

1. No passo anterior (botão “Upload-PDF”), ao fazer o *upload* de todos os exames digitalizados, foi gerado um arquivo ZIP com uma pasta para cada questão (por exemplo, `Download.zip`). Dentro dessa pasta, foi gerado um arquivo PDF para cada estudante no formato `_e1_c2_q3_p001_5.pdf`, em que 1 é o ID do exame, 2 é o ID da turma, 3 é o ID da questão, 001 é o número da página e 5 é o ID do estudante. O professor pode fazer correções adicionando anotações diretamente no PDF OU pode digitalizar os exames com anotações realizadas à caneta;
2. Altere manualmente os nomes dos arquivos de cada questão da seguinte forma: `_A;e1,e2,...;e1_c2_q3_p001_5.pdf`, com A sendo um conceito ou nota atribuída e ei são números de códigos de erros (opcional). Na pasta de cada questão, pode ser incluído um arquivo `_msg.txt` (um para cada questão) contendo uma mensagem a ser enviada para cada estudante;
3. Compacte a pasta de cada questão (por exemplo, `_e1_q3.zip`) e faça o *upload* utilizando o botão “Enviar Retornos aos Estudantes”. Será enviado para cada estudante o PDF com as correções, e também será gerado um arquivo CSV contendo os conceitos de cada estudante.

6.3 Recorte II da tela de exame – turmas

A Figura 6.5 apresenta a seção para selecionar as turmas associadas ao exame. Nessa seção, o professor pode escolher as turmas nas quais deseja aplicar a avaliação, marcando a caixa de seleção na primeira coluna. Após realizar as seleções, as turmas escolhidas serão sempre exibidas no topo da lista de turmas ao atualizar a página clicando no botão “Salvar”.

Na Figura 6.5, também são apresentados os seguintes detalhes sobre cada turma: disciplina, código, tipo (prática ou teórica), localização da sala, período ou semestre letivo, número de estudantes e professor(es) responsável(is) pela turma. A última coluna, intitulada “Ver”, contém um botão com o ID da turma no banco de dados. Ao clicar nesse botão, será exibida uma lista com todos os estudantes da respectiva turma.

Listar Turmas										
Listar Turmas										
Novo										
Excluir										
Disciplina	Código	Tipo	Sala	Período	N. Estudantes	Profs	Ver			
<input checked="" type="checkbox"/> Disciplina Exemplo	fzTurmaTeste	PClass	virtual	2023.3	0	fzampirolli	670			

Showing 1 to 1 of 1 entries

Figura 6.5: Recorte II da tela de exame, com as turmas do professor.

6.4 Recorte III da tela de exame – tópicos

No MCTest 5.3, foi adicionada uma nova área à tela do exame, que permite a seleção dos tópicos da disciplina, ver Figura 6.6. Esse recurso se mostrou necessário, pois a tela anterior estava demorando muito para carregar todas as questões cadastradas nos tópicos da disciplina. Agora, o professor pode primeiro escolher o(s) tópico(s), clicar no botão “Salvar” e, no próximo Recorte IV da tela, serão apresentadas apenas as questões correspondentes ao(s) tópico(s) selecionado(s).

Listar Tópicos			
		Search:	
> □	↑\n Tópico	↓\n	Ver ↑\n
<input checked="" type="checkbox"/>	Topico Exemplo		238
<input type="checkbox"/>	DE-matriz		243
<input type="checkbox"/>	DE-TIC		242

Showing 1 to 3 of 3 entries

Figura 6.6: Recorte III da tela de exame, com os tópicos da disciplina.

6.5 Recorte IV da tela de exame – questões

O professor tem a opção de selecionar as questões desejadas para aplicar na avaliação, conforme ilustrado na Figura 6.7. Essa seleção é feita marcando a caixa de seleção na primeira coluna correspondente a cada questão desejada. As questões marcadas serão sempre exibidas no topo da lista de questões, após clicar no botão “Salvar”. Essa lista contém todas as questões da disciplina, dos tópicos selecionados anteriormente, à qual o exame está vinculado.

Na Figura 6.7, também são exibidos os seguintes detalhes sobre cada questão: tópico, descrição curta, tipo QM ou QT, seguido pelo número de alternativas (é importante ressaltar que todas as questões devem ter o mesmo número de alternativas em cada exame), dificuldade (as questões de dificuldade 1 são apresentadas primeiro, seguidas pelas questões de dificuldade 2, e assim por diante; em seguida, são apresentadas as QTs, também nessa ordem), grupo (apenas uma questão por grupo é sorteada em cada exame) e, por fim, se a questão é paramétrica ou não.

A última coluna, intitulada “Ver”, exibe o ID da questão. Ao clicar nesse botão, o professor poderá visualizar a questão em detalhes.

6.6 Recorte V da tela de exame – configurando os detalhes

Na Figura 6.8, são apresentados os campos de configuração de um exame. Cada questão possui uma dificuldade variando de 1 a 5. O professor deve selecionar a quantidade de QMs (na Figura 6.7) que deseja incluir em cada exame. As questões serão exibidas em ordem crescente de dificuldade, começando pela dificuldade 1 e assim por diante. É importante observar que todas as

6.6. RECORTE V DA TELA DE EXAME – CONFIGURANDO OS DETALHES

Listar Questões

	Tópico	Descrição	Tipo	Dif.	Grupo	Par.	Ver
<input type="checkbox"/>	Topico Exemplo	DE-mru	QM 4	1		yes	2467
<input type="checkbox"/>	Topico Exemplo	equação da reta	QM 4	1		yes	2468
<input type="checkbox"/>	Topico Exemplo	equação e figura	QM 4	1		yes	2469
<input type="checkbox"/>	Topico Exemplo	grafo	QM 4	1		yes	2471
<input type="checkbox"/>	Topico Exemplo	idades irmãos	QM 4	1		yes	2465
<input type="checkbox"/>	Topico Exemplo	QE1	QM 5	1		no	2443
<input type="checkbox"/>	Topico Exemplo	QE2	QM 5	1		no	2444
<input type="checkbox"/>	Topico Exemplo	QE3	QM 5	1		no	2445

Showing 1 to 19 of 19 entries

Figura 6.7: Recorte IV da tela de exame, com as questões da disciplina, do(s) tópico(s) selecionado(s).

QMs de um exame devem ter o mesmo número de alternativas. Além disso, é possível criar exames que contenham QTs, seguindo a ordem de dificuldade de cada questão.

Alguns exemplos de erros comuns que podem ocorrer:

Na Figura 6.8, o número de questões por dificuldade de 1 a 5 é aplicável apenas a QM. O número de QTs não é contabilizado por dificuldade. Quanto ao número de questões por dificuldade:

1. É comum ocorrer um erro ao definir uma quantidade inferior de QMs em relação à quantidade selecionada anteriormente (Figura 6.5) para cada dificuldade, especialmente quando as questões possuem grupos. Lembre-se de que em cada exame é sorteada apenas uma questão por grupo. Portanto, se você selecionar 10 questões do grupo X, deve considerar apenas uma questão para contabilizar a quantidade correta;
2. Também é comum ocorrer o erro de selecionar QMs com um número diferente de alternativas;
3. Finalmente, ocorre erro ao contabilizar questões com diferentes dificuldades.

Na Figura 6.8, à direita, é possível visualizar o estilo do(s) bloco(s) de respostas ou QRs, nos quais os estudantes devem marcar as alternativas. Esses QRs aparecem na primeira folha da avaliação. Diversos estilos estão disponíveis em vision.ufabc.edu.br/MCTest/MCTest5-Experiments. Se a opção “Ecológico” for definida como “NÃO”, cada QT será desenhada em uma única folha, frente e verso.

Número de questões por Exame		Estilo do Exame	
Dificuldade 1	20	Questões por bloco	10
Dificuldade 2	0	Max Blocos Horiz.	1
Dificuldade 3	0	Quadro de respostas	Horizontal
Dificuldade 4	0	Respostas/Questões/Ambos	Respostas
Dificuldade 5	0	Ecológico	Sim
Alternativas por questão	5	Retorno	Não
Dissertativa	0	Data	07/12/2023
Variações	2	Período	Primeiro quadrimestre
<p>Quem criou: fzampirolli@ufabc.edu.br</p> <p>Instruções: \item desligar o celular</p>			
<p>Voltar Salvar</p> <p>Apagar</p>			
<p>Exame das disciplinas que eu participo Entre em contato com o coordenador da disciplina</p>			

Figura 6.8: Recorte V da tela de exame, configurando os detalhes do exame.

Atenção:

- Se a opção “Retorno” for selecionada como “SIM” na tela de Exame, ao pressionar o botão “Criar-PDF”, será enviado um e-mail para cada estudante contendo o exame em um arquivo PDF anexado;
- Um exame só existe se estiver associado a uma turma. Portanto:
 - Antes de excluir as turmas, certifique-se de que todos os exames estejam associados a uma única turma. Somente após isso, você poderá excluir as outras turmas;
 - Se um exame for deixado sem uma turma selecionada na Figura 6.5 e você clicar em “Salvar”, ocorrerá um erro. Nesse caso, será possível corrigir apenas se o administrador fizer alterações diretamente no BD, o que não é recomendado;
 - Foram adicionadas várias mensagens de erro para tentar lidar com esses problemas.

Alguns exemplos de erros comuns que podem ocorrer:

Um erro frequente ocorre quando o campo “Respostas/Questões/Ambos” é incompatível com os demais atributos do exame. Certifique-se de que as opções selecionadas nesse campo sejam consistentes com as demais configurações do exame.

1. A opção “Respostas” exibe apenas o QRs, sem as questões. Essa opção é útil quando o professor deseja aplicar apenas QMs, mas fornece os enunciados das questões em outro documento, não gerado pelo MCTest. Nesse caso, não é possível ter variações de exame;
2. A opção “Questões” é geralmente usada para avaliações com QTs;
3. A opção “Ambos” permite incluir tanto o QR quanto as QMs, podendo também incluir QTs.

6.7 Considerações finais

A realização dos exames desempenha um papel fundamental no MCTest, sendo o elemento central do sistema. Neste capítulo, foi fornecida uma visão geral da tela de exames utilizada no MCTest, abordando os principais aspectos relacionados à criação e configuração dos exames.

Ao acessar a tela de exames, o professor tem a flexibilidade de incluir uma ou várias turmas de estudantes em um exame, selecionar as questões que farão parte dele e definir diversos aspectos relevantes, como o formato de impressão, a quantidade de variações a serem criadas e a entrega do exame aos estudantes.

Foram exploradas as diversas seções presentes na tela de exames, dando ênfase ao processo de seleção das turmas associadas ao exame, à escolha dos tópicos e das questões relacionadas a serem incluídas, bem como à configuração detalhada do exame, incluindo a definição da quantidade de QMs e QTs.

Ao longo dos capítulos seguintes, serão aprofundadas as modalidades de exames disponíveis no MCTest, abordando questões com apenas o QR, questões com o enunciado completo, QTs que requerem correção manual e exames compostos por diferentes estilos de questões.

É importante estar ciente de possíveis erros comuns ao criar e configurar os exames, como a definição incorreta do número de questões por dificuldade, especialmente quando as questões possuem grupos, e a seleção de QMs com um número diferente de alternativas. O sistema conta com mensagens de erro para auxiliar na identificação e correção desses problemas.

Por fim, foi destacada a importância de associar corretamente os exames a uma turma e de verificar as configurações antes de realizar exclusões, garantindo a integridade dos dados e o correto funcionamento do sistema.

No próximo capítulo, será abordada a modalidade de exames com apenas QR, explorando suas características e funcionalidades.

Capítulo 7

Exames com Quadro de Respostas (QR)

Conteúdo

7.1 Utilizando exames exclusivamente com QR	128
7.1.1 Estilo vertical	128
7.1.2 Estilo horizontal	130
7.2 Restrições nos QRs	130
7.3 Corrigindo exames com QR	132
7.3.1 Arquivo CSV com as correções	135
7.3.2 Teoria de Resposta ao Item	136
7.3.3 Feedback ao estudante	138
7.4 Considerações finais	138

Este capítulo se concentra na criação de exames que exclusivamente apresentam o quadro de resposta (QR). Essa abordagem é particularmente útil quando os enunciados das questões são fornecidos separadamente. É uma prática amplamente adotada na Escola Preparatória da UFABC (EPUFABC) para processos seletivos e simulados que envolvem inúmeros estudantes anualmente.

Será demonstrado como acomodar centenas de questões em uma única página em formato PDF, que pode ser impressa e distribuída aos estudantes para preencherem manualmente as respostas. Posteriormente, os QRs são digitalizados e enviados ao MCTest para correção. O sistema encaminha ao professor um e-mail contendo arquivos CSV com as marcações de cada estudante e as respostas corretas. Além disso, em poucos minutos, o MCTest fornece uma síntese da Teoria de Respostas ao Item (TRI) com base nas correções realizadas, a menos que ocorra algum erro no processo de correção, como a não decodificação do QRCode.

O gabarito, ou seja, as respostas corretas, é disponibilizado na primeira página do PDF digitalizado, que pode conter as respostas de todos os estudantes de uma turma em sequência.

7.1 Utilizando exames exclusivamente com QR

Conforme visto no capítulo anterior, na Seção 6.1 – [Acesso à tela de exames](#), ao criar um exame, a configuração padrão é para um exame com exclusivamente o QR, como mostrado na Figura 6.3. Para esse estilo de exame, o professor só precisa selecionar a(s) turma(s) e seguir as instruções descritas na Seção 6.6 – [Recorte V da tela de exame – configurando os detalhes](#) para configurar os detalhes do exame. Nesse caso, é necessário definir o número de questões em uma escala de dificuldade de 1 a 5 e o número de alternativas de cada questão na parte esquerda da Figura 6.8. Na parte direita, você deve informar a quantidade de questões por bloco/quadro e escolher a quantidade de blocos por linha, além de definir se o estilo do exame será horizontal ou vertical. Certifique-se de marcar as opções “Questões/Respostas/Ambos=Respostas”, “Ecológico=Sim” e “Retorno=Não”. As instruções do exame podem ser definidas no campo “Instruções” e é possível utilizar a formatação de itens com o comando `\item`.

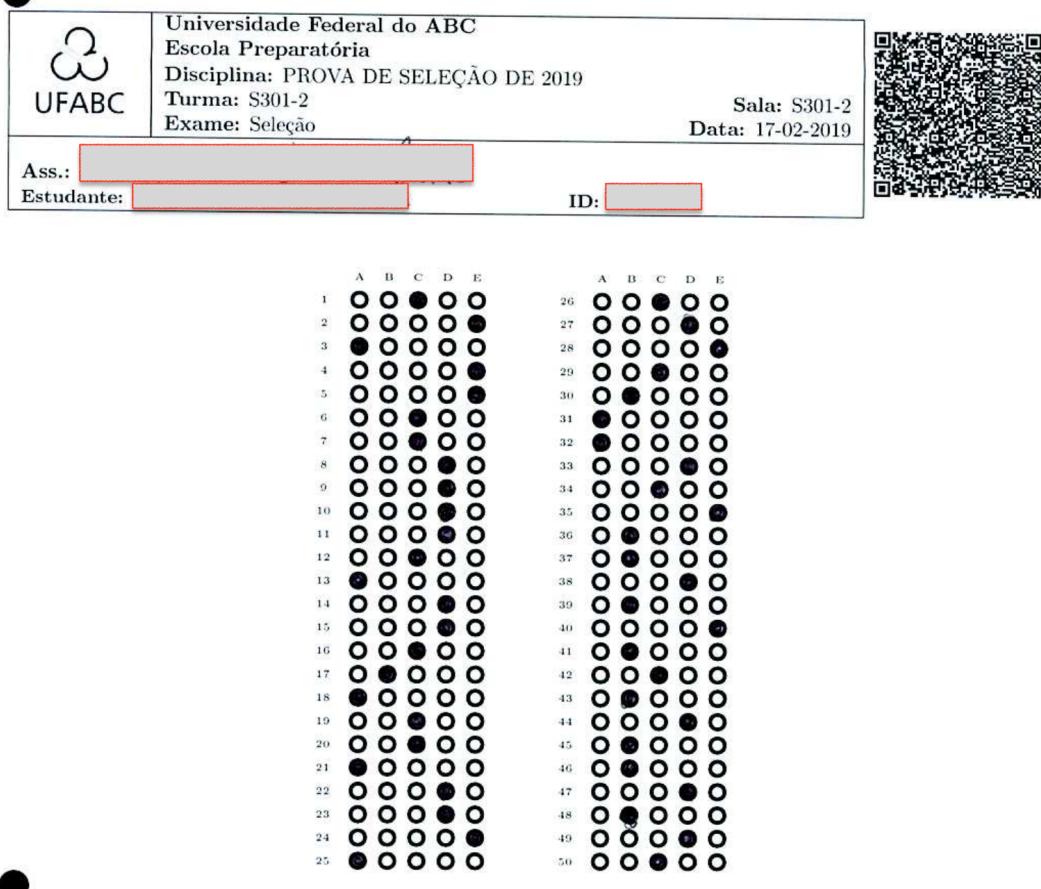
7.1.1 Estilo vertical

Na Figura 7.1, é apresentado um recorte do cabeçalho do exame, que inclui os dados do candidato e o QR. No processo seletivo da EPUFABC, realizado em 2019, houve a digitalização de 2553 QRs. Nesse caso específico, foi adotado o estilo vertical, com 25 questões por bloco e dois blocos por linha. Observa-se no cabeçalho a presença de um QRcode (*Quick Response Code*) criptografado e compactado, contendo informações do exame e do candidato. Apenas 13 exames tiveram o código não reconhecido (normalmente, basta digitalizá-lo novamente com uma resolução melhor para funcionar). Além disso, somente 5 candidatos preencheram o QR incorretamente. Por exemplo, na Figura 7.1, é possível observar que o candidato ultrapassou o local de marcação na questão 48. Quando isso acontece, é possível corrigir a marcação errada utilizando um corretivo ou um editor de PDF que permite a edição de imagens no próprio documento, como o site [ilovepdf.com](#). No entanto, é importante ter cuidado para não remover parte dos círculos ao realizar a correção.

Um exemplo de instruções, que contou com a colaboração do Prof. Dr. Leonardo Steil, então Pró-Reitor de Extensão, para redigir o texto, é mostrado na Figura 7.2, utilizado no processo seletivo de 2019 da EPUFABC.

Com esse exemplo realizado na EPUFABC, foi destacada a utilidade do MCTest na realização de avaliações para inúmeros estudantes. No caso em questão, foi criado um único arquivo CSV para preencher as 29 turmas, uma em cada sala de aula, conforme descrito na Seção 3.2.2 - [Criar várias turmas com arquivo CSV](#). Em seguida, foi configurado um único exame selecionando todas as turmas e gerados os PDFs, um por turma, conforme detalhado na Seção 6.2 - [Recorte I da tela de exame](#). Os exames foram impressos e aplicados aos estudantes. Posteriormente, foi realizada a digitalização dos QRs, conforme explicado na Seção 6.2. O MCTest então enviou as correções dos exames. Em casos em que ocorreu algum erro durante o processo de correção do QR, uma imagem da questão e das alternativas foi salva e incluída em um arquivo ZIP que também foi enviado ao professor por e-mail.

7.1. UTILIZANDO EXAMES EXCLUSIVAMENTE COM QR



Instruções:

Figura 7.1: Recorte de um exame digitalizado em formato PDF no estilo vertical.

Instruções:

- (a) Confira os números de seu RG ou CPF no local apropriado;
- (b) Marque as respostas neste CARTÃO-RESPOSTA, no campo correspondente a cada questão;
- (c) O CARTÃO-RESPOSTA é o único documento que será utilizado para a correção de sua prova. Não amasse, não dobre nem rasure este CARTÃO-RESPOSTA. As marcações no CARTÃO-RESPOSTA só podem ser feitas com caneta esferográfica de tinta preta. Não será permitido o uso de lápis e/ou lapiseira (grafite);
- (d) Em nenhuma hipótese haverá substituição deste CARTÃO-RESPOSTA por erro de preenchimento do candidato;
- (e) Não é permitida marca identificadora nesta parte do CARTÃO-RESPOSTA;
- (f) A marcação dos círculos correspondentes às respostas deve ser feita PREENCHENDO TODO O ESPAÇO COMPREENDIDO PELO CÍRCULO E NÃO PODE PINTAR FORA DO CÍRCULO, PINTAR DE FORMA CONTÍNUA E DENSA. A LEITORA ÓTICA é sensível a marcas escuras; portanto, preencha os campos de marcação completamente, sem deixar claros;
- (g) o que NÂO pode fazer: passar branquinho - ex.1 - ex.2 ; desenhar fora do círculo - ex.3 ; usar a parte do gabarito acima como rascunho.

Figura 7.2: Recorte de um exame em formato PDF no estilo vertical, com as instruções.

7.1.2 Estilo horizontal

Para minimizar os erros nas marcações dos círculos, foi desenvolvido o estilo horizontal, como ilustrado na Figura 7.3. É importante observar que, na questão 15 desta figura, o candidato utilizou corretivo, o que impossibilitou o algoritmo de visão computacional de corrigir automaticamente esse exame. Esse estilo foi aplicado no processo seletivo da EPUFABC em 2020, com 2053 exames digitalizados, 3 QRCodes não identificados e apenas uma falha nas correções das respostas, conforme mostrado na Figura 7.3. Além disso, o estilo horizontal ocupa menos espaço na folha de exame, o que é especialmente útil quando se deseja incluir tanto o QR quanto as questões em uma única folha, como utilizado no processo seletivo da Especialização em Tecnologias e Sistemas da Informação (TSI), que será apresentado no próximo capítulo.

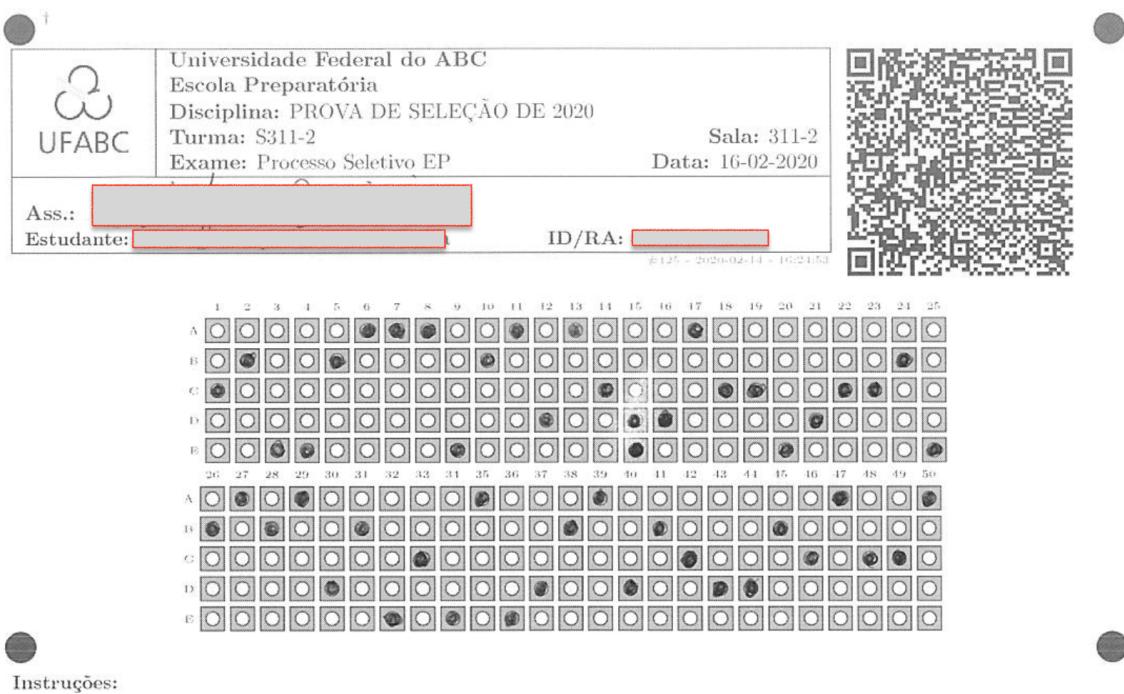


Figura 7.3: Recorte de um exame digitalizado em formato PDF no estilo horizontal.

O restante deste capítulo fornecerá mais detalhes sobre esse estilo de exame que inclui exclusivamente o QR.

7.2 Restrições nos QRs

Na Figura 7.4, são apresentados exemplos de exames com a quantidade máxima de questões nos estilos horizontal (245 questões) e vertical (200 questões). É importante ressaltar que, se o número de alternativas for menor do que cinco, esses valores máximos serão ainda maiores. **O número mínimo de alternativas permitidas por questão é três.** Além disso, é relevante destacar que apenas a página da frente pode ser utilizada para a construção dos QRs.

A seguir, serão apresentadas mais restrições relacionadas à organização dos QRs. **O número mínimo de questões por bloco é três.** Portanto, se for criado um exame com 17 questões, cinco

7.2. RESTRIÇÕES NOS QRS

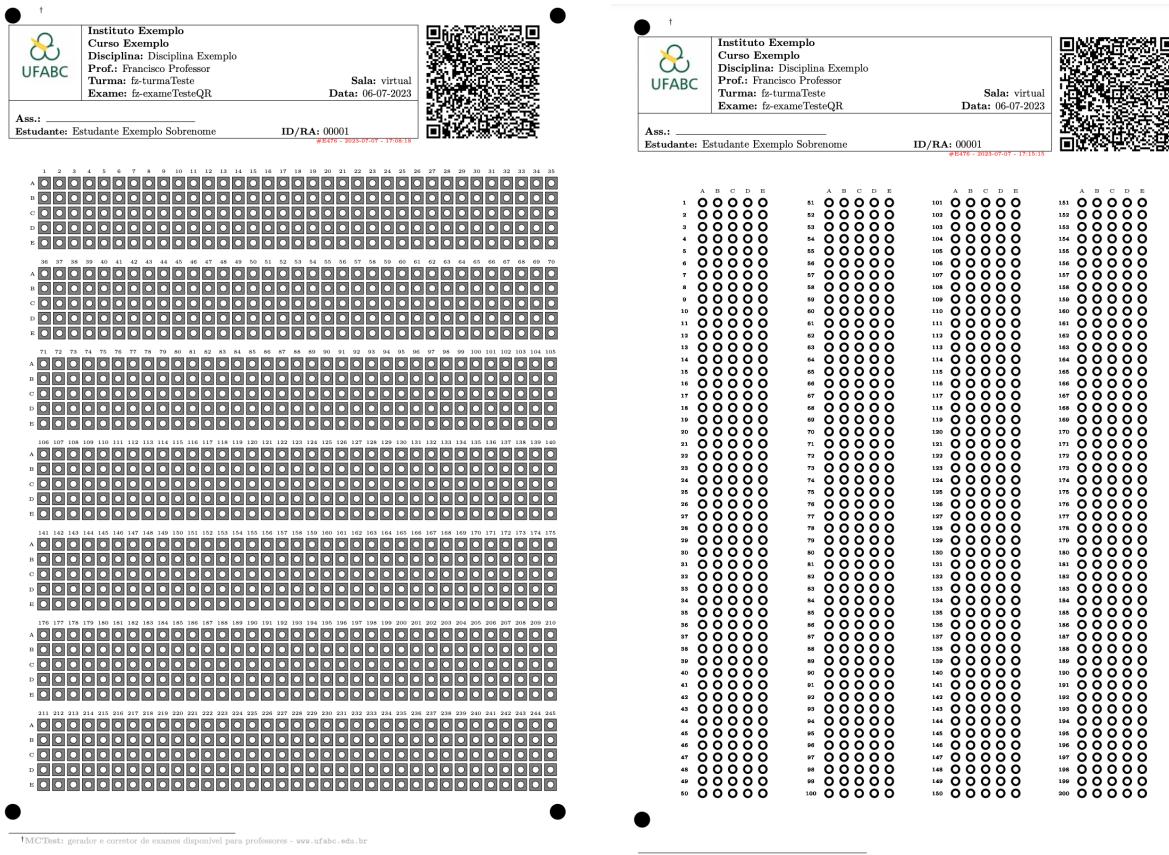


Figura 7.4: Exemplos de PDFs de exames com 245 questões no estilo horizontal e 200 questões no estilo vertical.

questões por bloco e dois blocos por linha, ocorrerá um erro, pois não é possível ter um bloco com apenas duas questões. Uma alternativa nesse caso seria criar blocos com seis questões cada um, por exemplo. Na Figura 7.5, é mostrado um exemplo contendo 18 questões, cinco questões por bloco e dois blocos por linha.

 Instituto Exemplo Curso Exemplo Disciplina: Disciplina Exemplo Prof.: Francisco Zampirolli Turma: Turma Exemplo Exame: fz-exameTesteQR	Sala: virtual Data: 04-01-2024																																																																																																																																										
Ass.: _____ Estudante: Gabarito Primeira Página ID/RA: 101 <small>#E488 - 2024-01-04 - 08:40:40</small>																																																																																																																																											
																																																																																																																																											
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 5%;">1</td> <td style="width: 5%;">2</td> <td style="width: 5%;">3</td> <td style="width: 5%;">4</td> <td style="width: 5%;">5</td> <td style="width: 5%;">6</td> <td style="width: 5%;">7</td> <td style="width: 5%;">8</td> <td style="width: 5%;">9</td> <td style="width: 5%;">10</td> </tr> <tr> <td>A</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td>A</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> <tr> <td>B</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td>B</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> <tr> <td>C</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td>C</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> <tr> <td>D</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td>D</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> <tr> <td>E</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td>E</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> <tr> <td colspan="5"></td> <td>11</td> <td>12</td> <td>13</td> <td>14</td> <td>15</td> <td>16</td> <td>17</td> <td>18</td> </tr> <tr> <td colspan="5"></td> <td>A</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td></td> <td></td> <td>A</td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> <tr> <td colspan="5"></td> <td>B</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td>B</td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> <tr> <td colspan="5"></td> <td>C</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td>C</td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> <tr> <td colspan="5"></td> <td>D</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td>D</td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> <tr> <td colspan="5"></td> <td>E</td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td><input type="radio"/></td> <td>E</td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> </table>		1	2	3	4	5	6	7	8	9	10	A	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	C	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	C	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	D	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	D	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	E	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	E	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>						11	12	13	14	15	16	17	18						A	<input type="radio"/>	<input type="radio"/>			A	<input type="radio"/>	<input type="radio"/>						B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	B	<input type="radio"/>	<input type="radio"/>						C	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	C	<input type="radio"/>	<input type="radio"/>						D	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	D	<input type="radio"/>	<input type="radio"/>						E	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	E	<input type="radio"/>	<input type="radio"/>
1	2	3	4	5	6	7	8	9	10																																																																																																																																		
A	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	A	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>																																																																																																																																		
B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>																																																																																																																																		
C	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	C	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>																																																																																																																																		
D	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	D	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>																																																																																																																																		
E	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	E	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>																																																																																																																																		
					11	12	13	14	15	16	17	18																																																																																																																															
					A	<input type="radio"/>	<input type="radio"/>			A	<input type="radio"/>	<input type="radio"/>																																																																																																																															
					B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	B	<input type="radio"/>	<input type="radio"/>																																																																																																																															
					C	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	C	<input type="radio"/>	<input type="radio"/>																																																																																																																															
					D	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	D	<input type="radio"/>	<input type="radio"/>																																																																																																																															
					E	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	E	<input type="radio"/>	<input type="radio"/>																																																																																																																															

Instruções:
(a) desligar o celular

Figura 7.5: Exemplo de PDF de um exame com 18 questões, cinco questões por bloco e dois blocos por linha.

7.3 Corrigindo exames com QR

Foi criado um PDF destinado a uma turma com três estudantes, sendo o primeiro exclusivamente para armazenar o gabarito, conforme ilustrado na Figura 7.5. Após a impressão desse PDF, o gabarito foi preenchido e digitalizado, como mostrado na Figura 7.6-(a). Em seguida, mais dois QRs foram preenchidos e digitalizados, conforme apresentado nas Figuras 7.6-(b) e (c). Todas as três digitalizações foram armazenadas em um único PDF, sendo a primeira página estritamente destinada ao gabarito.

No segundo QR, Figura 7.6-(b), optou-se por introduzir vários erros no preenchimento das alternativas nos terceiro e quarto blocos; os detalhes estão disponíveis na Figura 7.7. Nesse caso específico, os erros foram gerados intencionalmente ao marcar mais de uma alternativa em algumas questões, para avaliar a capacidade do método de visão computacional em detectar essas marcações incorretas.

7.3. CORRIGINDO EXAMES COM QR

Na terceira página do PDF, Figura 7.6-(c), decidiu-se repetir o gabarito para simular um estudante que obteve a nota máxima nesta avaliação. No entanto, a questão 9 não foi considerada devido à marcação insuficiente.

O algoritmo implementado para considerar uma marcação correta exige que mais de 50% da área esteja marcada dentro do círculo. Se houver mais de uma marcação, em alguns casos, é considerada a marcação com maior área. Essa lógica pode ser observada no quarto bloco, tanto na Figura 7.6-(b) quanto na Figura 7.7.

É importante mencionar que a Figura 7.6-(c) exibe a digitalização da página rotacionada de propósito. Isso pode ser feito sem problemas com rotações inferiores a $+/- 45$ graus, desde que os quatro discos pretos externos estejam intactos na digitalização.

Destaque:

Se algum dos quatro discos não aparecer durante a digitalização, uma sugestão seria pintar o que estiver faltando com o mesmo diâmetro e realizar uma nova digitalização. Outra opção é realizar a pintura diretamente no arquivo digitalizado, utilizando, por exemplo, a ferramenta disponível em ilovepdf.com. Lembrando que os quatro discos devem formar um retângulo imaginário com as marcações e o QRcode internos.

Após fazer o *upload* deste PDF digitalizado, seguindo as instruções descritas na Seção 6.2 - Recorte I da tela de exame, serão gerados vários arquivos que serão compactados em um arquivo ZIP, conforme apresentado a seguir:

Arquivos gerados ao solicitar a correção do exame no botão “Upload-PDF”

```
_e488_fzampirolli@ufabc.edu.br_e488_varID_0_class_672_scan_RETURN__.csv  
_e488_fzampirolli@ufabc.edu.br_e488_varID_0_class_672_scan_RETURN_p002_s3_q002.png  
_e488_fzampirolli@ufabc.edu.br_e488_varID_0_class_672_scan_RETURN_p002_s3_q003.png  
_e488_fzampirolli@ufabc.edu.br_e488_varID_0_class_672_scan_RETURN_p002_s3_q004.png  
_e488_fzampirolli@ufabc.edu.br_e488_varID_0_class_672_scan_RETURN_p002_s3_q005.png  
_e488_fzampirolli@ufabc.edu.br_e488_varID_0_class_672_scan_RETURN_p002_s4_q001_D_OK.png  
_e488_fzampirolli@ufabc.edu.br_e488_varID_0_class_672_scan_RETURN_irt.csv  
_e488_fzampirolli@ufabc.edu.br_e488_varID_0_class_672_scan_RETURN_statistics.csv  
studentEmail_e488.zip
```

Nesses arquivos, o prefixo `_e488_` representa o ID do exame no banco de dados, seguido pelo e-mail do professor logado no MCTest e pelo nome do arquivo digitalizado e enviado para correção. O sufixo `_RETURN_` indica que esses arquivos serão compactados e enviados ao e-mail do professor vinculado à conta logada no MCTest.

CAPÍTULO 7. EXAMES COM QUADRO DE RESPOSTAS (QR)

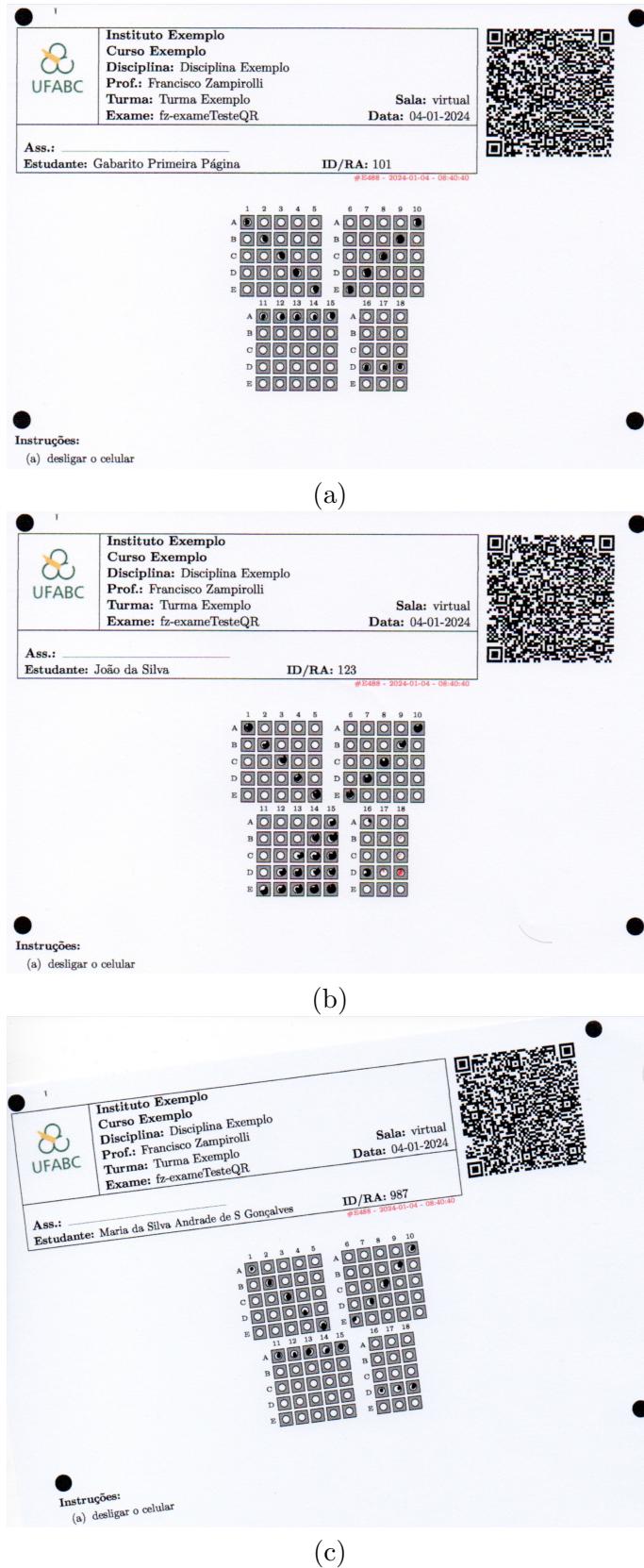


Figura 7.6: Exemplos de exames digitalizados incluem: (a) o gabarito obtido a partir do PDF apresentado na Figura 7.5; (b) e (c) representam dois exemplos adicionais de respostas de estudantes.

7.3.1 Arquivo CSV com as correções

O conteúdo do primeiro arquivo CSV na lista anterior, `*_.csv`, apresenta a seguinte estrutura: a primeira linha contém o cabeçalho, no qual `Pag` representa a página do PDF, `ID` é a identificação do estudante obtida por meio do QR code (com este `ID` obtém-se o nome `Student` e o email `Email` do estudante), `Resp` indica a quantidade de alternativas, `Quest` representa a quantidade de questões, `Inv` indica as questões inválidas (sem marcação ou com mais de uma marcação), e `Grade` mostra a quantidade de questões marcadas corretamente, conforme o gabarito presente na segunda linha. A segunda linha contém a compilação do gabarito (compilação da primeira página do PDF), enquanto a linha seguinte representa o exame com erros no preenchimento (compilação da segunda página do PDF). É possível observar que existem cinco questões inválidas identificadas pela coluna `Inv`, correspondentes às questões de 12 a 15 do terceiro bloco (identificado como `s3` na lista acima), ver Figura 7.6–(b), além da questão 17 no quarto bloco.

Essas questões foram salvas como arquivos PNG e estão exibidas na Figura 7.7. Essas imagens são úteis, pois o professor pode analisá-las para verificar possíveis alterações na quantidade de respostas corretas do estudante. A quarta linha deste arquivo CSV corresponde à repetição do gabarito (compilação da terceira página do PDF), com exceção da marcação inválida na questão 9.

Optou-se por não enviar os recortes das questões sem marcação ou com baixa marcação (0/*, como mostram as questões 17 da segunda página e 9 da terceira página), pois é comum o estudante deixar questões em branco, o que geraria muitas imagens no arquivo ZIP.

Conteúdo do arquivo `*_.csv` gerado (as colunas são separadas por vírgula)

Pag	ID	Student	Email	Resp	Quest	Inv	Grade
1	101	Gabarito Primeira Página	fzampirolli@gmail.com	5	18	0	0
2	123	João da Silva	fzampirolli@gmail.com	5	18	5	12
3	987	Maria da Silva Andrade de S Gonçalves	fzampirolli@gmail.com	5	18	1	17

Continuação do arquivo `*_.csv`

Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18
A	B	C	D	E	E	D	C	B	A	A	A	A	A	A	D	D	D
A	B	C	D	E	E	D	C	B	A	E/A	2/A	3/A	4/A	5/A	D	0/D	D
A	B	C	D	E	E	D	C	0/B	A	A	A	A	A	D	D	D	

Atenção:

1. Quando ocorre um erro na decodificação do QRcode, é apresentado **ERROR** na coluna **Pag**;
2. Quando ocorre um erro na compilação dos QRs, como apresentados nas Figuras 7.1 e 7.3, as colunas **Resp** e **Quest** apresentam números diferentes dos definidos no formulário do exame.

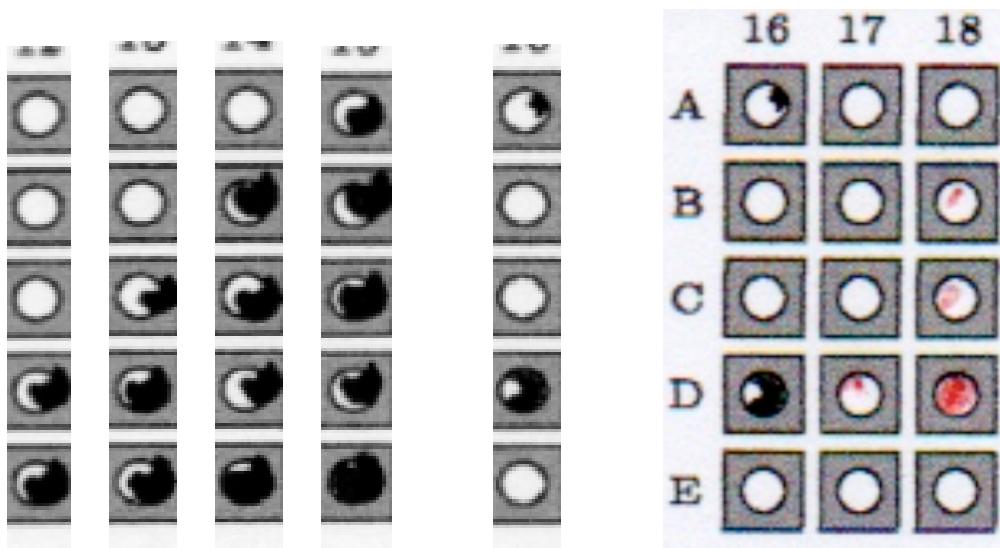


Figura 7.7: À esquerda, são apresentados quatro recortes automáticos das questões erroneamente marcadas no terceiro bloco, na Figura 7.6-(b). No centro, encontra-se outro recorte automático referente ao quarto bloco, em que a alternativa D foi considerada correta. À direita, também no quarto bloco, a imagem é apresentada de forma ampliada para destacar múltiplas marcações em vermelho.

7.3.2 Teoria de Resposta ao Item

O segundo arquivo CSV, com sufixo `_irt.csv`, contém as respostas corretas (representadas pelo valor 1) e incorretas (representadas pelo valor 0), a partir da segunda linha (o gabarito da primeira linha é desconsiderado), conforme exemplificado a seguir. Esse arquivo será utilizado para realizar os cálculos da Teoria de Resposta ao Item (TRI) ([BIRNBAUM, 1968](#)).

Conteúdo do arquivo `*.irt.csv` gerado (as colunas são separadas por vírgula)

```
1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 1 0 1
1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1
```

Os resultados da TRI foram adaptados do site criado por [Yasuhiro Okamoto](#), utilizando a biblioteca PyMC3 do Python. A TRI é amplamente utilizada na medição de habilidades, sendo o

7.3. CORRIGINDO EXAMES COM QR

Exame Nacional do Ensino Médio (ENEM) um exemplo conhecido dessa aplicação. No entanto, em psicologia, é comum adotar a abordagem de estágios para representar o desenvolvimento dessas habilidades. Esse modelo permite analisar dados de habilidades representadas por estágios, nos quais a probabilidade de uma resposta correta aumenta à medida que o estágio avança. No caso do MCTest, utilizou-se um número de estágios $K=5$, representando os cinco conceitos que o estudante pode alcançar (A, B, C, D e F) na graduação da UFABC. O *script* adaptado calcula as probabilidades de respostas corretas entre esses estágios.

Ao detalhar um pouco mais esse processo, é importante mencionar que o MCTest calcula automaticamente a TRI de cada PDF digitalizado submetido para correção. No entanto, para calcular a TRI dos 2553 candidatos que participaram do processo seletivo da EPUFABC em 2019, foi necessário reunir todos esses arquivos `_irt.csv` obtidos dos PDFs submetidos em um único arquivo. Para otimizar o tempo de processamento, realizou-se a ordenação e criou-se um novo CSV contendo os dados dos 1000 candidatos com melhor classificação. Em seguida, utilizou-se o *script* disponibilizado no GitHub, no arquivo github.com/fzampirolli/mctest/blob/master/_irt_pymc3_shell.py. Especificamente para o trabalho de [Zampirolli, Batista, Josko, Steil e Trevisan \(2021\)](#), foi utilizada também a linguagem de programação R para processar todos os candidatos.

Melhorias:

Para trabalhos futuros, é necessário adaptar esse processo de cálculo da TRI para lidar de forma mais eficiente com um número maior de candidatos, visto que a abordagem atual se limita aos 1000 mais bem classificados.

Os resultados obtidos por meio da TRI serão apresentados no terceiro arquivo CSV, com sufixo `_statistics.csv`, conforme demonstrado a seguir. Nesse arquivo de estatísticas, a primeira coluna, `id`, representa o número da questão. A coluna `corr` indica o número de respostas corretas, enquanto a coluna `fail` indica o número de respostas marcadas incorretamente. A coluna `aver` representa a média de acertos, e a coluna `str` indica o desvio padrão. Em seguida, têm-se as colunas `%corr` e `%fail`, que representam a porcentagem de respostas corretas e incorretas, respectivamente.

No entanto, é importante ressaltar que um experimento completo realizado na EPUFABC, que resultou no trabalho de [Zampirolli, Batista, Josko, Steil e Trevisan \(2021\)](#), será apresentado na parte de experimentos deste livro.

Conteúdo do arquivo *_statistics.csv gerado (as colunas são separadas por vírgula)

id	corr	fail	aver	std	%corr	%fail
1	2	0	1.0	0.0	1.0	0.0
2	2	0	1.0	0.0	1.0	0.0
3	2	0	1.0	0.0	1.0	0.0
4	2	0	1.0	0.0	1.0	0.0
5	2	0	1.0	0.0	1.0	0.0
6	2	0	1.0	0.0	1.0	0.0
7	2	0	1.0	0.0	1.0	0.0
8	2	0	1.0	0.0	1.0	0.0
9	1	1	0.5	0.5	0.5	0.5
10	2	0	1.0	0.0	1.0	0.0
11	1	1	0.5	0.5	0.5	0.5
12	1	1	0.5	0.5	0.5	0.5
13	1	1	0.5	0.5	0.5	0.5
14	1	1	0.5	0.5	0.5	0.5
15	1	1	0.5	0.5	0.5	0.5
16	2	0	1.0	0.0	1.0	0.0
17	1	1	0.5	0.5	0.5	0.5
18	2	0	1.0	0.0	1.0	0.0

7.3.3 *Feedback* ao estudante

Se a opção “Retorno=Sim” na Seção 6.6 – Recorte V da tela de exame – configurando os detalhes for selecionada, ao clicar em “Upload–PDF”, as correções do exame de cada estudante serão enviadas por e-mail em formato PDF, conforme apresentado na Figura 7.8. É possível observar que as marcações incorretas são indicadas por um círculo na alternativa correta. No final do PDF, é fornecido um resumo contendo os dados do estudante e a quantidade de acertos. No próximo capítulo, serão apresentados mais exemplos de QR que incluem as QMs.

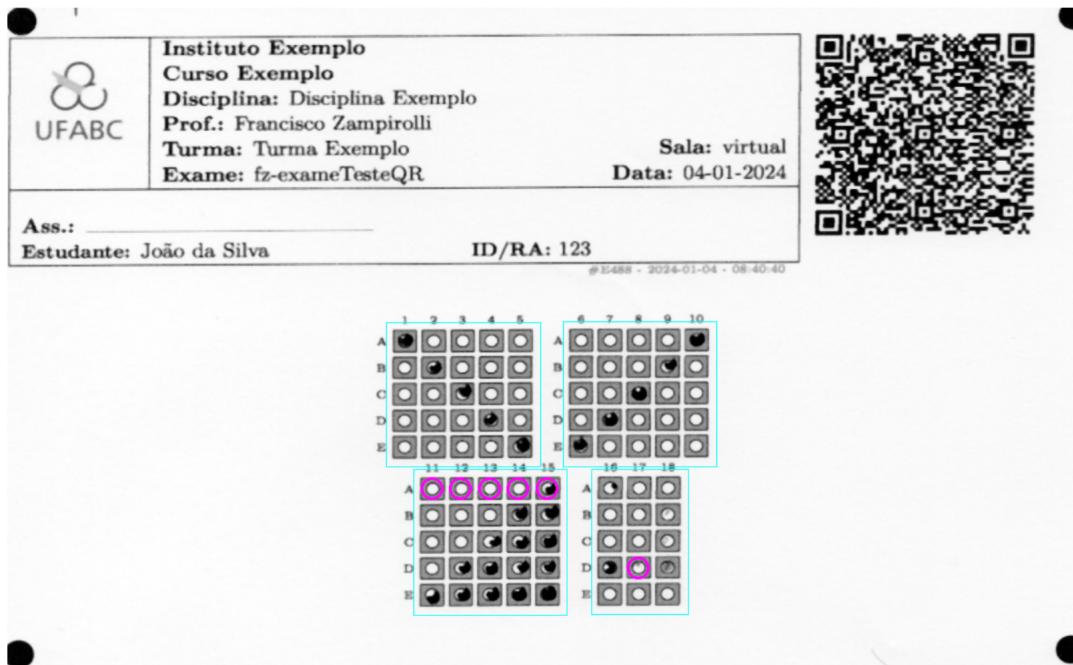
7.4 Considerações finais

Este capítulo abordou a criação de exames exclusivamente com o QR, uma prática comumente adotada na Escola Preparatória da UFABC (EPUFABC) para processos seletivos e simulados que envolvem inúmeros estudantes anualmente. Foi demonstrado como acomodar centenas de questões em uma única página no formato PDF, que pode ser impressa e distribuída aos estudantes para preencherem manualmente as respostas.

Em seguida, os QRs são digitalizados e enviados ao MCTest para correção. O sistema envia ao professor um e-mail contendo arquivos CSV com as marcações de cada estudante e as respostas corretas. Além disso, o MCTest fornece uma síntese da Teoria de Respostas ao Item (TRI) com base nas correções realizadas. O gabarito, ou seja, as respostas corretas, é fornecido na primeira página do PDF digitalizado, que pode conter as respostas de todos os estudantes de uma turma em sequência.

Ao utilizar exames exclusivamente com QR, é importante configurar corretamente os detalhes

7.4. CONSIDERAÇÕES FINAIS



Estudante: João da Silva

ID: 123

Email: fzampirolli@gmail.com

Nota: 12/18 (66.667%)

Figura 7.8: PDF com o *feedback* das correções enviado para o e-mail do estudante.

do exame, como o número de questões, o número de alternativas por questão e o estilo do exame (horizontal ou vertical). Foi destacado que é possível corrigir eventuais erros de marcação utilizando ferramentas de edição de PDF, mas é necessário ter cuidado para preservar as marcações corretas.

Por fim, foi apresentada a possibilidade de fornecer *feedback* aos estudantes, através da opção “Retorno=Sim” no MCTest. Nesse caso, o sistema envia por e-mail um PDF com as correções do exame para cada estudante, contendo marcações incorretas indicadas por círculos na alternativa correta, bem como um resumo dos dados do estudante e a quantidade de acertos.

No próximo capítulo, serão apresentados mais exemplos de QR que incluem as questões, ampliando as possibilidades de utilização dessa abordagem no contexto educacional.

Capítulo 8

Exames com QR+QM e/ou QT

Conteúdo

8.1	Desenvolvendo QMs e uma QT para inclusão em um exame	142
8.1.1	QT com valor exato	144
8.1.2	Elaborar exame com questões formuladas	144
8.2	Criando as variações	144
8.3	Detalhando os gabaritos no arquivo CSV	146
8.4	Criando o PDF do exame com QR+QM+QT	148
8.5	Corrigindo exames com QR+QM	151
8.6	Exames adaptativos	154
8.6.1	Exemplificando o uso do método SAT	156
8.6.2	Estimativa dos parâmetros do modelo TRI	159
8.7	Recomendações para realização tranquila de exames presenciais	160
8.8	Considerações finais	162

Este capítulo complementa o anterior, fornecendo detalhes sobre o processo de criação e correção de exames que envolvem tanto o Quadro de Respostas (QR) quanto os enunciados das questões de múltipla escolha (QMs), como exemplificado nos Capítulos 4 – Questões estáticas e 5 – Questões paramétricas para questões com código em Python.

Esse estilo de exame com QR+QM tem sido amplamente utilizado na Especialização em Tecnologia e Sistemas de Informação (TSI) da UFABC, como introduzido no Capítulo 1, nos processos seletivos desde 2017. Anteriormente, desde 2012, já se utilizava o MCTest em versões anteriores, com QR+QM, porém em um formato de exame que empregava editores de texto como o Word ou o BOffice, com uma única folha frente e verso, para evitar a necessidade de grampear as folhas de milhares de candidatos nos processos seletivos. O avanço significativo no MCTest ocorreu na versão 4, quando passou a ser possível criar os exames utilizando o L^AT_EX, o que permitiu também a inclusão

de blocos de código Python para a elaboração de questões paramétricas, conforme apresentado no Capítulo 5 – [Questões paramétricas](#). A versão mais recente do MCTest, disponível na web, possibilitou o armazenamento das questões em um banco de dados, juntamente com toda uma estrutura para gerenciar um sistema acadêmico voltado exclusivamente para avaliações. Esses avanços têm sido objeto de diversas publicações e também motivaram a escrita deste livro, com o intuito de registrar as principais características de tudo o que foi desenvolvido e validado em avaliações dos mais diversos estilos.

Além das QMs, também é possível incluir questões dissertativas (QTs) nos exames, após a apresentação das QMs. Essa abordagem permite uma maior diversidade nos tipos de questões e uma avaliação mais abrangente das habilidades dos estudantes.

No Capítulo 6 – [Visão geral dos exames](#), foi introduzido como criar exames contendo QR+QM, abordando detalhes sobre a tela do exame. Agora, será colocada em prática a criação de exemplos de exames contendo QR+QM+QT, além de detalhar o processo de correção automática.

8.1 Desenvolvendo QMs e uma QT para inclusão em um exame

Uma forma prática de criar questões não paramétricas no MCTest é utilizando um arquivo TXT com todas as questões, conforme introduzido na Seção 4.1 – [Tutorialis gerais sobre a navegação de questões](#). Antes de prosseguir com a criação das questões, é necessário criar o tópico, por exemplo, DE-TIC da disciplina, seguindo as instruções apresentadas na Seção 2.2.5 – [Tópico](#).

A seguir, serão apresentadas sete questões que serão incluídas no MCTest. Serão duas questões para o nível de dificuldade fácil (QE), ou dificuldade 1, duas questões para o nível de dificuldade médio (QM), ou dificuldade 3, e três questões para o nível de dificuldade difícil (QH), ou dificuldade 5. Para otimizar o processo de *upload*, sugere-se agrupá-las em um único arquivo TXT, como disponibilizado no GitHub em [cap08_exemplosQM.txt](#).

Arquivo CSV, questões fáceis

QE::DE-TIC::

Qual é o componente principal de um computador que armazena dados e programas?

A: Memória RAM % alternativa correta - sempre a primeira

A: Placa-mãe

A: Processador

A: Teclado

A: Placa de vídeo

QE::DE-TIC::

Qual dos seguintes dispositivos é usado exclusivamente para entrada de dados em um computador?

A: Teclado

A: Impressora

A: Caixa de som

A: Monitor

A: Placa de vídeo

Arquivo CSV, questões de nível médio

QM::DE-TIC::

O que significa a sigla HTML?

A: HyperText Markup Language

A: High Technical Machine Language

A: Home Tool Management Language

A: Hyper Transfer Machine Learning

A: Hardware Technical Maintenance Language

QM::DE-TIC::

Qual das seguintes linguagens de programação é amplamente utilizada para o desenvolvimento de sites dinâmicos?

A: JavaScript

A: HTML

A: CSS

A: Python

A: C++

Arquivo CSV, questões difíceis

QH::DE-TIC::

Qual das seguintes opções representa corretamente um endereço IP válido?

A: 127.0.0.1

A: 192.168.0.300

A: 10.0.0.256

A: 172.16.0.0

A: 256.256.256.256

QH::DE-TIC::

Qual dos seguintes protocolos de internet é utilizado para enviar e-mails?

A: SMTP

A: FTP

A: HTTP

A: DNS

A: SSH

QH::DE-TIC::

Qual é o resultado da soma dos números hexadecimais 7C e 54?

A: D0

A: F0

A: D4

A: D1

A: E0

8.1.1 QT com valor exato

A questão apresentada no Código 8.1 é um exemplo de QT que contém um valor exato como resposta correta, delimitado pelos marcadores `%%{` e `}%%`, conforme indicado na linha 5 do código. Neste caso, é utilizada a palavra “exata” como exemplo para ilustrar o funcionamento dessa abordagem. Na linha 7 do código, são desenhadas 10 linhas para que o estudante possa responder à questão. Na próxima seção, será explorada essa funcionalidade com mais detalhes.

Questão:

```
1 Descreva brevemente a importância da tecnologia da informação (TI) na
2 sociedade moderna e mencione pelo menos duas áreas onde a TI desempenha
3 um papel fundamental.
4
5 %%{exata}%%
6
7 \drawLines{10}
```

Código 8.1: Exemplo de QT com resposta exata.

8.1.2 Elaborar exame com questões formuladas

Após importar esse arquivo TXT com as novas questões no tópico “DE-TIC” da “Disciplina Exemplo”, criar um novo exame. Antes disso, crie uma turma de teste para essa disciplina. Ao abrir a tela para atualizar esse exame, escolha o tópico “DE-TIC” e, em seguida, clique no botão “Salvar”. Dessa forma, todas as questões recentemente criadas desse tópico irão aparecer na tela do exame. Selecione essas questões, conforme Figura 8.1. Em seguida, defina duas questões em cada nível de dificuldade, conforme ilustrado na Figura 8.2. Além disso, inclua uma questão dissertativa (QT) como exemplo. Certifique-se de marcar a opção “Respostas/Questões/Ambos=Ambos” para incluir tanto o QR quanto as questões (QM e QT) no exame. Mais informações sobre essa configuração podem ser encontradas na Seção 6.6 – Recorte V da tela de exame – configurando os detalhes.

8.2 Criando as variações

Antes de criar o PDF do exame, é necessário criar as variações clicando no botão “Criar Variações” (observe que ao lado deste botão há um ID inicialmente definido como 0, que será alterado quando o botão for pressionado). É importante detalhar mais essa opção, conforme ilustrado na Figura 8.3.

Ao marcar a opção “Json”, se a questão for de integração entre MCTest, Moodle e VPL, será enviado ao e-mail do professor um arquivo `linker.json` contendo os casos de teste a serem incluídos na atividade VPL do Moodle. Se a opção “Template” for selecionada, será criado um arquivo CSV contendo o gabarito das QMs, bem como QTs com respostas curtas, para ser possível comparar as respostas exatas, caractere por caractere.

8.2. CRIANDO AS VARIAÇÕES

Listar Questões

	Tópico	Descrição	Tipo	Dif.	Grupo	Par.	Ver
<input checked="" type="checkbox"/>	DE-TIC	DE-TIC000	QM 5	1		no	2455
<input checked="" type="checkbox"/>	DE-TIC	DE-TIC001	QM 5	1		no	2456
<input checked="" type="checkbox"/>	DE-TIC	DE-TIC002	QM 5	3		no	2457
<input checked="" type="checkbox"/>	DE-TIC	DE-TIC003	QM 5	3		no	2458
<input checked="" type="checkbox"/>	DE-TIC	DE-TIC004	QM 5	5		no	2459
<input checked="" type="checkbox"/>	DE-TIC	DE-TIC005	QM 5	5		no	2460
<input checked="" type="checkbox"/>	DE-TIC	DE-TIC006	QM 5	5		no	2461
<input checked="" type="checkbox"/>	DE-TIC	texto sobre TI	QT	1		no	2462

Showing 1 to 8 of 8 entries

Figura 8.1: Recorte da tela com as questões escolhidas do exame.

Número de questões por Exame		Estilo do Exame	
Dificuldade 1	2	Questões por bloco	10
Dificuldade 2	0	Max Blocos Horiz.	1
Dificuldade 3	2	Quadro de respostas	Horizontal
Dificuldade 4	0	Respostas/Questões/Ambos	Ambos
Dificuldade 5	2	Ecológico	Sim
Alternativas por questão	5	Retorno	Não
Dissertativa	1	Data	09/07/2023
Variações	2	Período	Primeiro quadrimestre
Quem criou fzampirolli@ufabc.edu.br			
Instruções	\item desligar o celular		

Figura 8.2: Recorte da tela de configuração do exame.

As opções “Aiken” e “XML” são utilizadas para gerar arquivos que podem ser importados para criar um banco de questões no Moodle. Por fim, ao selecionar a opção “LaTeX+PDF”, o

professor receberá um PDF contendo todas as variações do exame. No exemplo ilustrado na Figura 8.2, foram criadas apenas duas variações. Vale ressaltar que existem artigos validando cada método utilizado nesses botões, os quais serão detalhados na parte de experimentos deste livro.

Antes de criar os exames no botão acima, primeiro crie as variações. É necessário criar novas variações do exame cada vez que você muda as questões e o número de variações nos atributos abaixo. As opções marcadas abaixo serão enviadas para o seu e-mail.



Figura 8.3: Recorte da tela de configuração do exame, detalhando o botão “Criar-Variações”.

Assim, antes de criar o PDF do exame, marque a opção “Template” na Figura 8.3 para receber o gabarito e clique em “Criar-Variações”. Nesse exemplo, uma nova aba será aberta no navegador exibindo a mensagem apresentada na Figura 8.4.



Figura 8.4: Mensagem após clicar no botão “Criar-Variações”.

8.3 Detalhando os gabaritos no arquivo CSV

Após clicar em “Criar-Variações”, selecionando antes a opção “Template”, o professor receberá um e-mail contendo o seguinte arquivo com os gabaritos, conforme ilustrado na Figura 8.5.

Arquivo CSV, com os gabaritos (as colunas são separadas por vírgula)

variation	Q1	Q2	Q3	Q4	Q5	Q6	Q7	K1	K2	K3	K4	K5	K6
0	A	D	D	E	B	A	exata	245604231	245521304	245843102	245723410	246130421	246001432
1	D	D	E	C	B	E	exata	245521403	245632104	245814230	245721034	246010243	246113240

No arquivo em questão, a primeira coluna `variation` apresenta a variação do exame. No exemplo mostrado na Figura 8.2, foram escolhidas apenas duas variações, identificadas como variação 0 e variação 1. Em seguida, são apresentadas as respostas corretas para cada uma das sete questões do exame, identificadas como `Q1` até `Q7`.

A questão `Q7` é uma QT e foi definida em seu enunciado a sequência `%%{exata}%%`, indicando para o MCTest que essa questão tem uma resposta exata contendo o texto “exata”. É importante

8.3. DETALHANDO OS GABARITOS NO ARQUIVO CSV

 webmctest@ufabc.edu.br
Caixa d...ufabc.edu.br 10:35
MCTest: files created automatically; Exam: exameTesteQR-QM; 2023-12-09 - 10:35:02
Para: Francisco Zampirolli <fzampirolli@ufabc.edu.br>

Prezada(o),

Esta mensagem contém arquivos criados automaticamente, com todas as variações deste exame.

CSV: Gabaritos das questões.

Ver detalhes:

A. <https://doi.org/10.5753/cbie.sbie.2020.51>. Este arquivo CSV pode ser usado para correção automática utilizando Google Forms + Sheets.

B. "Automated assessment with multiple-choice questions using weighted answers" (CSEDU21)



report_Exam_48
5_templates.csv

Figura 8.5: E-mail enviado ao professor após clicar no botão “Criar-Variações”.

ressaltar que é possível incluir questões paramétricas, nas quais cada variação pode apresentar uma resposta correta diferente. Essa flexibilidade é discutida em detalhes no trabalho de [Zampirolli, Batista, Iriarte e Junior \(2020\)](#), que descreve a aplicação dessas técnicas na disciplina de Cálculo 1. Na parte de experimentos deste livro, essas abordagens serão apresentadas de forma mais detalhada, explorando suas aplicações e resultados.

Após as respostas corretas, são criadas novas colunas de K1 até K6 para detalhar o sorteio realizado em cada variação do exame para as QMs. Por exemplo, na variação 1 (segunda variação), a questão K1 tem o valor 245521403. Como são QMs com 5 alternativas, o valor 21403 indica a ordem das alternativas da questão 2455 (ID da questão no banco de dados). Sabendo que a alternativa correta sempre é a primeira (índice 0), pode-se concluir que a alternativa correta para essa questão é a letra D. Portanto, a alternativa A corresponde à terceira alternativa no banco de dados (índice 2), a alternativa B corresponde à segunda (índice 1), a alternativa C corresponde à quinta alternativa (índice 4), a alternativa D é a correta (índice 0) e a alternativa E corresponde à quarta (índice 3), resultando na sequência 21403.

Ao analisar o sorteio das questões em ambas as variações, foram selecionadas sete QMs. No entanto, para cada exame, apenas seis questões serão sorteadas, conforme a configuração previamente definida. No arquivo CSV, é possível observar que apenas as questões Q3 e Q4 não apresentaram variações diferentes (consulte as colunas K3 e K4, com questões de ID 2458 e 2457, respectivamente). No entanto, em todas as questões houve embaralhamento das alternativas.

Uma estrutura de atribuição de pesos diferentes para cada alternativa marcada é detalhada no trabalho de [Zampirolli, Batista, Iriarte e Junior \(2020\)](#), no qual são apresentadas abordagens para avaliar o desempenho dos estudantes considerando a relevância das respostas em QMs.

Atenção:

É extremamente importante verificar se o gabarito gerado corresponde ao PDF que será distribuído aos estudantes.

1. O ID da variação será alterado toda vez que o botão “Criar-Variações” for acionado;
2. Esse número de ID também será impresso no PDF gerado, abaixo do cabeçalho do exame, em vermelho. Por exemplo, #E485#V61031 -- 2023-12-09 -- 10:59:04. No exemplo mencionado, o exame possui o ID 485 e a variação é 61031;
3. No momento em que o botão “Criar-Variações” é acionado, uma “fotografia” das questões selecionadas no exame é armazenada no banco de dados para cada variação. Essa “fotografia” representa o estado atual das questões e é independente de eventuais alterações realizadas posteriormente pelo coordenador da disciplina ou professor que criou alguma destas questões. Esse recurso de segurança garante que as variações do exame permaneçam consistentes, mesmo que haja modificações nas questões originais.

8.4 Criando o PDF do exame com QR+QM+QT

Após a criação das variações, a próxima etapa do processo de criação do exame consiste em gerar o PDF contendo os QRs, as QMs e QTs, caso estejam presentes no exame. Essa etapa é realizada ao clicar no botão “Criar-PDF” no painel de configuração do exame.

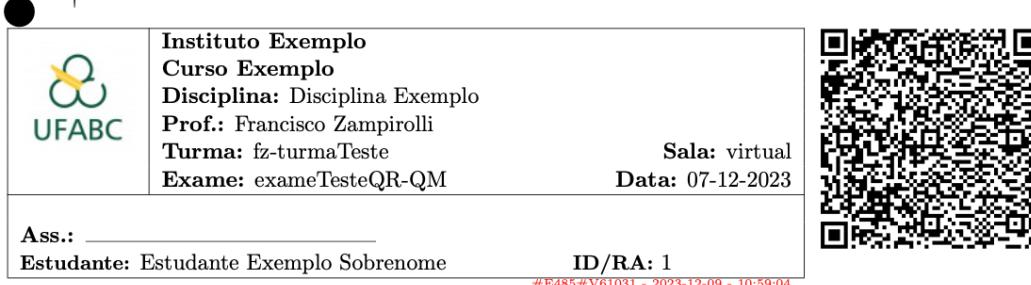
Ao gerar o PDF, todas as variações do exame serão incluídas, garantindo a diversificação das questões e respostas. Os QRs serão apresentados para os estudantes poderem preencher suas respostas, e as QMs e QTs serão apresentadas conforme a configuração do exame. A ordem das questões impressas no PDF gerado segue a ordem de dificuldade de 1 a 5 para as QMs. Em seguida, as QTs serão apresentadas na mesma ordem.

É de extrema importância revisar o PDF gerado para garantir a correta formatação das questões, respostas e demais elementos antes de disponibilizá-lo aos estudantes. Além disso, é válido ressaltar que será criado um PDF para cada turma selecionada no exame, conforme detalhado na Seção 6.3 – Recorte II da tela de exame – turmas. No contexto do exame criado nas seções anteriores, foi gerado o PDF ilustrado na Figura 8.6.

Após a geração do PDF do exame, será enviado por e-mail ao professor o arquivo L^AT_EX correspondente a cada turma selecionada. Esse arquivo pode ser útil caso o professor deseje realizar alterações manuais no exame, embora seja importante ressaltar que o QRcode é específico de cada estudante, para garantir a correção automática. No exemplo fornecido, o arquivo gerado é o _e485_class_671_varID_61031.tex.

Além do arquivo L^AT_EX, o e-mail também conterá um arquivo CSV anexado, que apresenta as variações sorteadas para cada estudante. No exemplo mencionado, o arquivo é denominado report_Exam_485_varID_61031_students_variations.csv. Caso haja mais turmas selecionadas

8.4. CRIANDO O PDF DO EXAME COM QR+QM+QT



Instruções:

- desligar o celular

Questões de Múltipla Escolha:

- Qual é o componente principal de um computador que armazena dados e programas?
A. Memória RAM B. Placa de vídeo C. Processador D. Teclado E. Placa-mãe
- Qual dos seguintes dispositivos é usado exclusivamente para entrada de dados em um computador?
A. Caixa de som B. Impressora C. Monitor D. Teclado E. Placa de vídeo
- Qual das seguintes linguagens de programação é amplamente utilizada para o desenvolvimento de sites dinâmicos?
A. C++ B. Python C. HTML D. JavaScript E. CSS
- O que significa a sigla HTML?
A. Home Tool Management Language B. Hyper Transfer Machine Learning C. Hardware Technical Maintenance Language D. High Technical Machine Language E. HyperText Markup Language
- Qual é o resultado da soma dos números hexadecimais 7C e 54?
A. D1 B. D0 C. E0 D. D4 E. F0
- Qual dos seguintes protocolos de internet é utilizado para enviar e-mails?
A. SMTP B. FTP C. SSH D. DNS E. HTTP

Questões Dissertativas

- Descreva brevemente a importância da tecnologia da informação (TI) na sociedade moderna e mencione pelo menos duas áreas onde a TI desempenha um papel fundamental.

Figura 8.6: Recorte do PDF do exame gerado após a ação do botão “Criar-PDF”.

para o exame, todos os estudantes de todas as turmas serão listados em um único arquivo CSV, que corresponde ao exame com o ID 485.

Ao observar a Figura 8.7, é possível verificar um trecho do e-mail encaminhado ao professor após acionar o botão “Criar-PDF” contendo os três arquivos a seguir:

Arquivos anexados ao e-mail enviado ao professor

_e485_varID_61031_class_671.pdf
_e485_varID_61031_class_671.tex
_e485_varID_61031_class_671_students_variations.csv
_e485_varID_61031_class_671_variations.csv

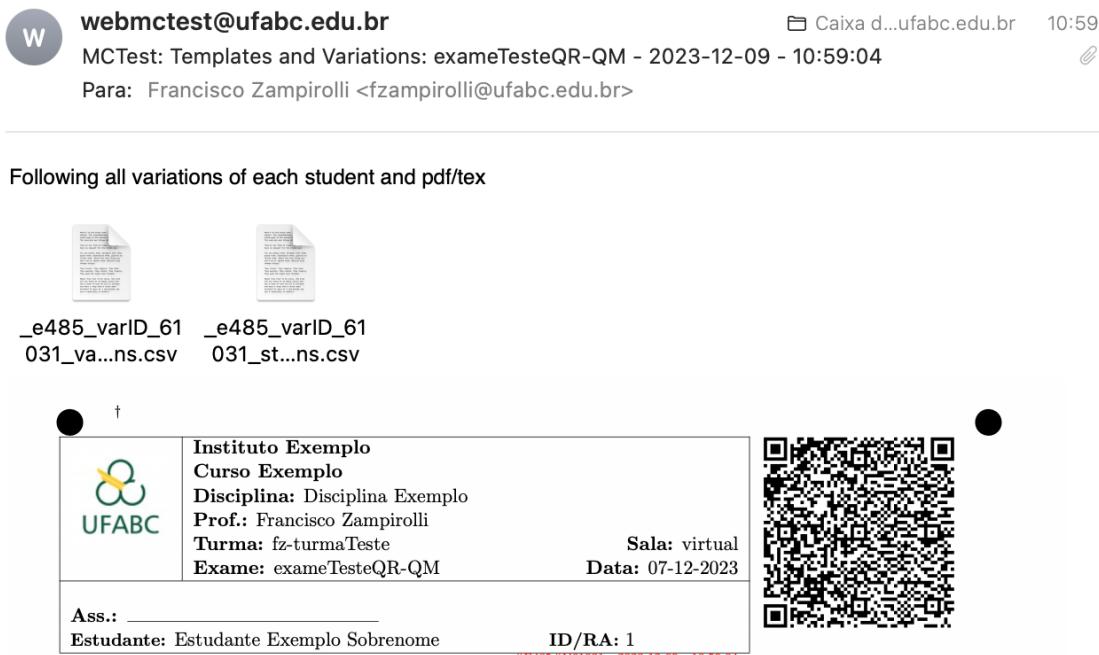


Figura 8.7: E-mail enviado ao professor após clicar no botão “Criar-PDF”.

Arquivo _e485_varID_61031_students_variations.csv, com as variações de cada estudante

Name	Email	Variation
Estudante Exemplo Sobrenome	fzampirolli@ufabc.edu.br	0

Arquivo _e485_varID_61031_variations.csv, com as variações de cada estudante, com turma e média das notas de exames anteriores

Room	ID	Name	Email	Variation	Mean	PreviousAbilities
fz-turmaTeste	1	Estudante Exemplo Sobrenome	fzampirolli@ufabc.edu.br	0		0

Atenção:

Foi adotado o seguinte critério de sorteio de variações para os estudantes:

1. Se o número de variações definido for menor que o número de estudantes em cada turma, será sorteada uma variação para os estudantes baseado no *hash* gerado a partir do nome e sobrenome do estudante. Dessa forma, ao clicar em “Criar-PDF”, será atribuída a mesma variação para cada estudante;
2. Caso contrário, ou seja, se o número de variações definido no exame for maior ou igual que o número de estudantes da turma, será sorteada uma variação diferente para cada estudante a cada vez que o botão “Criar-PDF” for acionado.

8.5 Corrigindo exames com QR+QM

Ao corrigir exames que contêm QR e QMs utilizando o MCTest, o processo é mais simples em comparação com a correção de exames apenas com QR, conforme detalhado na Seção 7.3 – Corrigindo exames com QR.

O gabarito de cada exame não é mais armazenado no servidor do MCTest em formato TXT. Agora, ele é identificado por meio da variação contida no QRcode. As informações contidas no QRcode, conforme ilustrado na Figura 8.6, são criptografadas e compactadas.

É possível verificar as respostas correspondentes ao gabarito apresentado no CSV na Seção 8.3 – Detalhando os gabaritos no arquivo CSV, considerando a variação 0, conforme apresentado no CSV da seção anterior. Portanto, as respostas para as questões Q1 até Q7 são, respectivamente: A, D, D, E, B, A, exata.

Imprimindo o PDF apresentado na Figura 8.6, selecionando a opção “Retorno=Sim” e salvando o exame, preenchendo as questões, digitalizando e enviando para o botão “Upload-PDF”, o estudante receberá o *feedback* apresentado na Figura 8.8 por e-mail. Certifique-se de marcar a caixa abaixo desse botão para também enviar os enunciados das questões, as alternativas corretas e as inválidas.

Além disso, o professor receberá as correções realizadas em um arquivo ZIP, conforme detalhado na Seção 7.3 – Corrigindo exames com QR, que conterá os seguintes arquivos:

Arquivos gerados ao solicitar a correção do exame no botão “Upload-PDF”

```
studentEmail_e485.zip  
_e485_fzampirolli@ufabc.edu.br_ScanFileQR_QM_RETURN__.csv  
_e485_fzampirolli@ufabc.edu.br_ScanFileQR_QM_RETURN_irt.csv  
_e485_fzampirolli@ufabc.edu.br_ScanFileQR_QM_RETURN_p001_s1_q006.png  
_e485_fzampirolli@ufabc.edu.br_ScanFileQR_QM_RETURN_statistics.csv
```

O arquivo `studentEmail_e485.zip` contém todos os arquivos PDF enviados aos estudantes,

W webmctest@ufabc.edu.br Caixa... - fzampirolli 11:37
 Mensagem automática enviada pelo MCTest 
 Para: Francisco Zampirolli <fzampirolli@gmail.com>

Prezado(a) Estudante Exemplo Sobrenome

Segue em anexo a sua atividade: Lista de Exercícios ou Exame.

Se receber em anexo um arquivo '.bin', renomear para '.pdf'.

Não retornar este email para webmctest@ufabc.edu.br, pois não será monitorado.

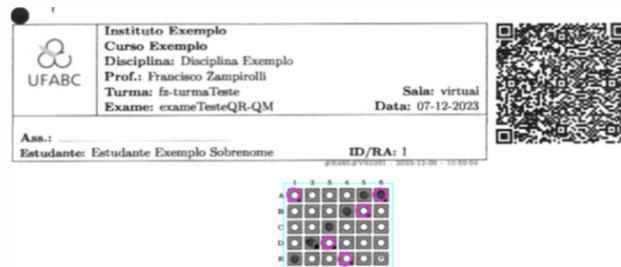
Se tiver alguma dúvida, entre em contato com o seu professor.

====

O MCTest é um software de código aberto (disponível no GitHub) para gerar e corrigir questões (em especial as parametrizadas - com variações) de forma automática e está sendo desenvolvido com apoio das instituições públicas:

- * Universidade Federal do ABC (UFABC)
- * FAPESP

====



Estudante: Estudante Exemplo Sobrenome

ID: 1

Email: fzampirolli@gmail.com

Nota: 1/6 (16.667%)

Questões de Múltipla Escolha:

1. Qual é o componente principal de um computador que armazena dados e programas?
Alternativa correta: (A) Memória RAM
Sua alternativa: (E) Placa-mãe
2. Qual dos seguintes dispositivos é usado exclusivamente para entrada de dados em um computador?
Alternativa correta: (D) Teclado
3. Qual das seguintes linguagens de programação é amplamente utilizada para o desenvolvimento de sites dinâmicos?
Alternativa correta: (D) JavaScript
Sua alternativa: (C) HTML
4. O que significa a sigla HTML?
Alternativa correta: (E) HyperText Markup Language
Sua alternativa: (B) Hyper Transfer Machine Learning
5. Qual é o resultado da soma dos números hexadecimais 7C e 54?
Alternativa correta: (B) D0
Sua alternativa: (A) D1
6. Qual dos seguintes protocolos da internet é utilizado para enviar e-mails?
Alternativa correta: (A) SMTP

Figura 8.8: Recorte do e-mail com parte do PDF do exame enviado como *feedback* ao estudante, após a ação do botão “Upload-PDF”. Neste PDF, o enunciado da QT também é apresentado após as QMs.

caso tenham escolhido “Retorno=SIM” na tela do exame. Um exemplo é apresentado na Figura 8.8,

8.5. CORRIGINDO EXAMES COM QR+QM

exibindo também as questões, as respostas do estudante e a alternativa correta. Esses enunciados são incluídos no PDF quando a caixa abaixo do botão “Upload-PDF” é marcada; caso contrário, apenas a imagem, as informações do estudante e sua nota são enviadas.

Destaque:

Um destaque incluído na versão mais recente do MCTest é apresentar as questões e as alternativas armazenadas nas variações salvas no banco de dados. Ou seja, quando há questões paramétricas, é exibida a “fotografia” da questão criada no momento em que o botão “Criar Variações” foi pressionado. O *feedback* é apresentado obtendo o atributo também da “fotografia” gerada, que agora pode ser parametrizado, sendo incluído no PDF a ser enviado ao estudante. Para isso, foi necessário incluir no QRCode informações da chave da variação do exame sorteada para cada estudante.

Conforme explicado na Seção [7.3.1 – Arquivo CSV com as correções](#), o primeiro arquivo CSV, `*_.csv`, apresenta o seguinte conteúdo:

Conteúdo do arquivo `*_.csv` gerado (as colunas são separadas por vírgula)

Pag	ID	Student	Email	Resp	Quest	Inv	Grad
1	1	Estudante Exemplo Sobrenome	fzampirolli@gmail.com	5	6	1	1

Continuação do arquivo `*_.csv`

Q1	Q2	Q3	Q4	Q5	Q6	K1	K2	K3	K4	K5	K6
E/A	D	C/D	B/E	A/B	2/A	245604231	245521304	245843102	245723410	246130421	246001432

No arquivo em questão, é possível observar nas colunas Q1 até Q6 a notação apresentada na Seção [7.3.1 – Arquivo CSV com as correções](#), agora com a adição de **resposta marcada / resposta correta**. Por exemplo, na questão Q1 tem a notação E/A, indicando que a alternativa marcada pelo estudante foi a E, enquanto a resposta correta é a A. Também é possível notar a presença de uma questão inválida na coluna Inv, referente à questão Q6, que apresenta o conteúdo 2/E. Isso indica que duas alternativas foram marcadas, sendo que na Figura [8.8](#) é possível observar que a alternativa A foi marcada, mas também foi realizada uma pequena marcação na alternativa E, invalidando a questão. Neste exemplo, a única marcação correta é na questão Q2. Observa-se que ainda não é possível corrigir a QT com resposta exata, e a questão 7 não está incluída no arquivo CSV anterior.

A Figura [8.8](#) apresenta um relatório com os dados do estudante, a nota obtida e os enunciados das questões, incluindo as marcações realizadas e a alternativa correta de cada questão. As colunas K1 até K6 foram previamente explicadas na Seção [8.3 – Detalhando os gabaritos no arquivo CSV](#).

8.6 Exames adaptativos

Abaixo do botão “Criar-PDF” na tela de exame, conforme ilustrado na Figura 6.4, encontram-se duas barras de rolagem. A primeira permite definir o tamanho da fonte do arquivo PDF gerado com o exame. A segunda barra de rolagem possibilita definir o número de exames anteriores realizados por cada estudante a ser considerado. Por exemplo, se o valor selecionado for 1, será considerada apenas o último exame para definir qual variação deste será destinada aos estudantes.

Foram estabelecidos três distintos tipos de estatísticas aplicadas a esses exames anteriores, conforme evidenciado na Figura 8.9. O número mínimo de exames anteriores na barra de rolagem é 1 e estende-se até 10, sendo 10 o valor padrão. Ao alcançar o valor 10, todas as avaliações anteriores serão consideradas, caracterizando um conjunto infinito.

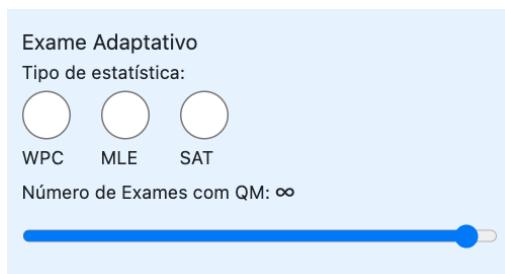


Figura 8.9: Tipos de exames adaptativos.

O processo de cálculo desses tipos de exames adaptativos considera que, para cada exame anterior e para cada questão respondida pelo estudante:

1. Recupere os dados da questão;
2. Calcule o parâmetro b (dificuldade da questão) com base no tipo de teste adaptativo escolhido;
3. Armazene o parâmetro b na lista `b_vector`;
4. Armazene se o estudante acertou (valor 1 acertou e 0 caso contrário) a questão na lista `u_vector`.

No item 2 anterior, para calcular o parâmetro b , são apresentados 3 tipos de teste/exame adaptativo:

WPC: Se o teste for do tipo *Weighted Probability of Correctness* (WPC) ou Porcentagem Ponderada de Acertos, o parâmetro b é calculado como a diferença entre 1 e a fração ponderada de respostas corretas em relação à quantidade total de questões corrigidas. O valor de b foi normalizado entre -5 e 5. No WPC, o método aprimora a avaliação atribuindo pesos às respostas corretas, levando em consideração a dificuldade individual de cada questão.

MLE: Se o teste for do tipo *Maximum Likelihood Estimation* (MLE) ou Estimativa de Máxima Verossimilhança, o parâmetro b é a habilidade associada à questão. Neste caso, esse valor de b foi normalizado entre -5 e 5. O MLE é um método usado para determinar os valores dos

8.6. EXAMES ADAPTATIVOS

parâmetros de um modelo estatístico que maximizam a probabilidade de observar os dados. Esse método ajusta dinamicamente a dificuldade das questões com base nas suas respostas anteriores, seguindo a TRI.

SAT: Se o teste for do tipo *Semi-Adaptive Testing* (SAT) ou Avaliação Semi-Adaptativa, o parâmetro b é determinado pelo índice da taxonomia de Bloom da questão, variando entre 1 e 6. Neste caso, esse valor de b foi normalizado entre -2 e 3. O SAT é um método de avaliação adaptativa em que o professor define manualmente o nível da Taxonomia de Bloom para cada questão. Este método combina adaptabilidade com avaliação personalizada, permitindo que o sistema ajuste dinamicamente a dificuldade das questões com base nas respostas dos estudantes em exames anteriores, ao mesmo tempo em que considera os níveis cognitivos predefinidos pelo professor.

Nos tipos WPC e SAT, calcula-se a habilidade do estudante multiplicando elemento a elemento das listas `b_vector` e `u_vector`. Ao realizar essa operação e calcular a média dessas multiplicações, obtém-se um valor que representa a habilidade do estudante com base em suas respostas e nas dificuldades das questões.

Para uma boa calibração das questões, os tipos WPC e MLE exigem que cada questão tenha sido respondida por um número mínimo de participantes anteriormente. Por exemplo, o MLE, que se baseia na Teoria de Resposta ao Item (TRI), requer um mínimo de 1.000 participantes no modelo 3PLM ([MIN; ARYADOUST, 2021](#)). Se a questão ainda não tiver sido respondida, a Taxonomia de Bloom é utilizada para classificá-la.

Atenção:

Para que o método adaptativo funcione corretamente:

1. Atribuir a Taxonomia de Bloom a cada questão. Essa atribuição é considerada uma condição inicial para o cálculo dos níveis de dificuldade das questões, tanto no método WPC quanto no MLE, antes de corrigir qualquer questão;
2. Certificar-se de que todos os exames anteriores tenham sido corrigidos e que as estatísticas de desempenho, tanto para o método WPC quanto para o MLE, tenham sido atualizadas para cada questão. Isso pode ser feito por meio da digitalização dos exames anteriores e do uso do botão “Upload-PDF” para realizar as correções;
3. Antes de escolher “Criar PDF” para distribuir os exames adaptativos, utilizando um dos três métodos apresentados na Figura 8.9, verifique se todas as turmas dos exames anteriores estão selecionadas nas respectivas telas de configuração de exames. Adicionalmente, as datas dos exames nos formulários devem estar corretas, pois apenas exames anteriores serão considerados nos cálculos das estatísticas.

8.6.1 Exemplificando o uso do método SAT

Para que este método de exame adaptativo funcione adequadamente, é necessário definir a taxonomia de Bloom para cada questão, nos 6 possíveis níveis: lembrar, compreender, aplicar, analisar, avaliar e criar. Cada um desses níveis recebe um valor de 1 a 6, respectivamente. Quando, para cada nível de dificuldade, são escolhidas várias questões com taxonomias distintas, cada variação do exame poderá apresentar pesos diferentes considerando essas taxonomias. Por exemplo, se um exame definir 3 questões com dificuldade 1, mas houver 6 questões com diferentes taxonomias marcadas para esse mesmo nível 1, e, além disso, foi decidida a criação de 20 variações de exames, teremos então variações com médias com diferentes pesos dos níveis dessa taxonomia.

Para ilustrar esse processo, foi criado um exame com 3 QM e 1 QT, todas com dificuldade 1. Além disso, foram selecionadas 6 QM para serem sorteadas, conforme Figura 8.10, com apenas 3 delas sendo escolhidas com base no desempenho do estudante em exames recentes.

Listar Questões

Listar Questões									
copiar csv excel pdf imprimir colunas visíveis ▾									
Search: <input type="text"/>									
> □	↑↓	Tópico	↑↓	Descrição	↑↓	Tipo	↑↓	Dif.	↑↓
Grp.	↑↓	Par.	↑↓	Ver	↑↓				
<input checked="" type="checkbox"/>		Topico Exemplo		QE1		QM 5		1	
									2443
<input checked="" type="checkbox"/>		Topico Exemplo		QE2		QM 5		1	
									2444
<input checked="" type="checkbox"/>		Topico Exemplo		QE3		QM 5		1	
									2445
<input checked="" type="checkbox"/>		DE-TIC		DE-TIC000		QM 5		1	
									2455
<input checked="" type="checkbox"/>		DE-TIC		DE-TIC001		QM 5		1	
									2456
<input checked="" type="checkbox"/>		DE-TIC		texto sobre TI		QT		1	
									2462
<input checked="" type="checkbox"/>		DE-matriz		DE-criar matriz		QM 2		1	
									2468

Figura 8.10: Seleção das questões na tela do exame para sorteio.

Na barra de rolagem da Figura 8.9, foi escolhido o número 5 para o exame adaptativo, ou seja, será calculada a média das 5 últimas habilidades dos exames realizados pelo estudante. Foram solicitadas a criação de 20 variações de exames antes de clicar no botão “Criar-Variações”. Também foram selecionadas duas turmas. Após clicar no botão “Criar-PDF”, os arquivos foram enviados para o email do professor, além de um *download* de um arquivo compactado contendo os seguintes arquivos (esse *download* ocorre apenas quando tem mais de uma turma selecionada).

Arquivos gerados ao solicitar criar PDF com o botão “Criar-PDF”

```
_e486_varID_61093_adaptive_test_variations.csv  
_e486_varID_61093_adaptive_test.csv  
_e486_varID_61093_class_671.pdf  
_e486_varID_61093_class_671.tex  
_e486_varID_61093_class_672.pdf  
_e486_varID_61093_class_672.tex  
_e486_varID_61093_students_variations.csv  
_e486_varID_61093_variations.csv
```

O arquivo `*_adaptive_test_variations.csv` contém informações sobre as variações do exame, ordenadas pela coluna `MeanAbilities`. O segundo arquivo `_adaptive_test.csv` contém informações dos exames anteriores realizados por todos os estudantes, ordenados pela última coluna `MeanPreviousAbilities`. Em seguida, temos o PDF gerado a partir do `TEX` de cada turma. O penúltimo arquivo, `*_students_variations.csv`, contém um resumo do arquivo seguinte com a variação sorteada de cada estudante, em ordem alfabética, por turma. Finalmente, o arquivo `*_variations.csv` contém a variação de cada estudante e a média das habilidades, ordenadas por essa média. Assim, as informações dos exames anteriores realizados por cada estudante da disciplina são detalhadas no arquivo `*_adaptive_test.csv` e a partir dele são resumidas em outros dois arquivos: `*_students_variations.csv` e `*_variations.csv`. Todos esses arquivos procuram tornar mais didáticos os cálculos estatísticos realizados para a escolha dos exames adaptativos para cada estudante. Os principais arquivos serão detalhados a seguir.

Arquivo `*_adaptive_test_variations`

O primeiro arquivo `*_adaptive_test_variations.csv` possui, na primeira coluna, as 20 variações do exame, na segunda coluna a chave das variações no banco de dados e na terceira coluna a média dos níveis da taxonomia de Bloom para as questões sorteadas para cada variação, em ordem crescente. Observam-se as questões com variações 4, 9, 12, 17 e 19 com o mesmo nível de dificuldade média de -0.333. Além disso, as variações 2, 5, 14 e 18 têm dificuldade média 2. A coluna seguinte armazena o desvio padrão dessas médias. As colunas seguintes são para calibrar os parâmetros a_i e b_i da TRI de cada questão $i \in \{1, \dots, 5\}$. Finalmente, as últimas colunas k_i armazenam as chaves das questões QM no banco de dados; além disso, mostram em seguida a(s) chave(s) das QT, se existir(em).

Arquivo *_adaptive_test_variations

Variation	VariationID	MeanAbilities	STD	a1	a2	a3	b1	b2	b3	k1	k2	k3	k4
4	61096	-0.333	1.247	0	0	0	0	1	-2	2443	2444	2455	2462
9	61101	-0.333	1.247	0	0	0	-2	0	1	2455	2443	2444	2462
12	61104	-0.333	1.700	0	0	0	-2	-1	2	2455	2456	2445	2462
17	61109	-0.333	1.247	0	0	0	0	1	-2	2443	2444	2455	2462
19	61111	-0.333	1.700	0	0	0	2	-2	-1	2445	2455	2456	2462
6	61098	0.000	2.160	0	0	0	3	-1	-2	2468	2456	2455	2462
15	61107	0.000	2.160	0	0	0	-2	-1	3	2455	2456	2468	2462
1	61093	0.333	1.247	0	0	0	2	-1	0	2445	2456	2443	2462
3	61095	0.667	1.700	0	0	0	-1	0	3	2456	2443	2468	2462
11	61103	0.667	2.055	0	0	0	-2	1	3	2455	2444	2468	2462
13	61105	0.667	2.055	0	0	0	1	-2	3	2444	2455	2468	2462
16	61108	0.667	1.700	0	0	0	3	0	-1	2468	2443	2456	2462
7	61099	1.000	1.633	0	0	0	1	-1	3	2444	2456	2468	2462
10	61102	1.000	1.633	0	0	0	-1	1	3	2456	2444	2468	2462
8	61100	1.333	1.247	0	0	0	0	1	3	2443	2444	2468	2462
20	61112	1.333	1.700	0	0	0	2	-1	3	2445	2456	2468	2462
2	61094	2.000	0.816	0	0	0	2	3	1	2445	2468	2444	2462
5	61097	2.000	0.816	0	0	0	3	2	1	2468	2445	2444	2462
14	61106	2.000	0.816	0	0	0	2	3	1	2445	2468	2444	2462
18	61110	2.000	0.816	0	0	0	3	2	1	2468	2445	2444	2462

Arquivo *_adaptive_test.csv

O arquivo `*_adaptive_test.csv` contém informações sobre turmas, códigos das turmas, nomes de estudantes e seus e-mails. Há duas turmas identificadas, 671 e 672. A primeira possui um estudante, enquanto a segunda tem dois estudantes associados.

Arquivo *_adaptive_test.csv

RoomID	RoomCode	NameStudent	EmailStudent
672	Turma Exemplo	João da Silva	fzampirolli@gmail.com
672	Turma Exemplo	Maria da Silva Andrade de S Gonçalves	fzampirolli@gmail.com
671	fz-turmaTeste	Estudante Exemplo Sobrenome	fzampirolli@gmail.com

Considere clicar em “Criar PDF” escolhendo SAT, ver Figura 8.9, em um novo exame adaptativo de nome `exameTesteAdaptQR-QM`, com data de 19 de abril de 2024. As demais colunas desse arquivo `*_adaptive_test.csv` a seguir representam informações específicas de cada exame realizado pelos estudantes em datas anteriores. As colunas incluem a data de realização do exame, o ID do exame e o nome do exame. O conteúdo de cada coluna contém as habilidades obtidas em cada exame. Observa-se que o primeiro exame com ID 485 ocorreu em 7 de dezembro de 2023, e a média de acertos, considerando a taxonomia de Bloom, foi de -0.333. Os outros dois estudantes não tiveram

8.6. EXAMES ADAPTATIVOS

notas registradas nessa avaliação, e foi atribuído o valor mínimo de habilidade igual a -5.0. A coluna `MeanPreviousAbilities` indica a média das habilidades nas avaliações dos estudantes.

Continuação do arquivo *_adaptive_test.csv

2023-12-07:485:exameTesteQR-QM	MeanPreviousAbilities
-5.0	-5.0
-5.0	-5.0
-0.333	-0.333

Arquivo *_variations.csv

O arquivo `*_variations.csv` a seguir resume esses dados, com a variação sorteada para cada estudante. Observa-se que as variações 12 e 19, com a menor média pela taxonomia de Bloom, foram sorteadas para os estudantes da turma `Turma Exemplo`. Como apresentado anteriormente, essas variações são 4, 9, 12, 17 e 19. Por outro lado, a variação 2 foi sorteada entre 2, 5, 14 e 18, que apresentam a maior dificuldade.

Arquivo *_variations.csv

Room	ID	Name	Variation	MeanPreviousAbilities
Turma Exemplo	123	João da Silva	12	-5.0
Turma Exemplo	987	Maria da Silva Andrade de S Gonçalves	19	-5.0
fz-turmaTeste	1	Estudante Exemplo Sobrenome	2	-0.333

Uma parte crucial desse processo adaptativo reside na seleção apropriada das questões, classificadas com base na taxonomia de Bloom. É essencial assegurar que as questões sejam precisamente definidas e estejam alinhadas aos objetivos de aprendizagem. Ademais, o método adaptativo em questão deve estimular os estudantes a manterem-se engajados na disciplina, evitando a imposição de exames excessivamente difíceis como forma de penalização ou excessivamente fáceis, o que resultaria em uma atribuição injusta de conceitos. Uma estratégia promissora talvez seja a atribuição de notas de participação, independentemente do desempenho obtido nos exames adaptativos. Dessa forma, essas atividades são consideradas formativas e não avaliativas.

8.6.2 Estimativa dos parâmetros do modelo TRI

Ao corrigir um exame com QMs, diversas estatísticas são calculadas, incluindo os parâmetros a , b e c da Teoria da Resposta ao Item/Questão (TRI). Estes parâmetros representam, respectivamente, a discriminação, a habilidade e o chute do item, conforme detalhado na Seção 4.2. A Figura 4.7 ilustra a Curva Característica do Item (CCI) com base nesses três parâmetros.

O código desenvolvido no MCTest v.5.3 tem como objetivo estimar os parâmetros a , b e c da TRI para cada QM de um exame, utilizando o método da máxima verossimilhança.

Inicialmente, uma função gera um dicionário contendo as respostas de cada estudante para cada questão no formato de 1 para acerto e 0 para erro. A chave do dicionário é o ID da questão no banco de dados.

Em seguida, para cada questão é feita a estimativa dos parâmetros utilizando a função de *likelihood* negativa. Os parâmetros iniciais são definidos com base nos valores já armazenados no banco de dados ou valores fixos. A minimização é realizada utilizando o algoritmo L-BFGS-B, com restrições nos intervalos de cada parâmetro, com a seguinte importação `from scipy.optimize import minimize`.

Após a convergência do algoritmo de otimização, os valores estimados dos parâmetros a , b e c são atualizados no modelo da questão no banco de dados. Desta forma, os parâmetros TRI são recalculados a cada aplicação e correção do exame, permitindo acompanhar possíveis mudanças na dificuldade ou discriminação das questões ao longo do tempo.

Estes cálculos estimam os parâmetros do modelo tripáramétrico (3PLM) para cada QM de um exame utilizando máxima verossimilhança. A função de verossimilhança negativa $L(\theta, a, b, c|\mathbf{x})$ é definida como:

$$L(\theta, a, b, c|\mathbf{x}) = - \sum_{i=1}^N x_i \log P(X_i = 1|\theta_i, a, b, c) + (1 - x_i) \log[1 - P(X_i = 1|\theta_i, a, b, c)],$$

em que $P(X_i = 1|\theta_i, a, b, c) = c + (1 - c)F(a(\theta_i - b))$ e F é a função logística cumulativa.

A estimativa é realizada minimizando $L(\theta, a, b, c|\mathbf{x})$ usando o algoritmo L-BFGS-B, com restrições $\theta \in [-5, 5]$, $a \in (-\infty, +\infty)$, $b \in [-5, 5]$ e $c \in [0, 1]$. Valores iniciais são obtidos dos parâmetros armazenados ou valores fixos. Após a convergência, os parâmetros estimados a , b e c são atualizados no banco de dados para cada questão. Dessa forma, o código implementa a estimativa dos parâmetros do modelo TRI para cada questão por máxima verossimilhança. Esses cálculos são realizados através do método `estimateIRT_parameters(exam)`, disponível no arquivo `CVMCTest.py`¹, chamado após as correções dos exames digitalizados, ao pressionar o botão “Upload-PDF” na tela do exame.

8.7 Recomendações para realização tranquila de exames presenciais

Com base na experiência de inúmeros exames aplicados e corrigidos, é possível destacar algumas recomendações essenciais para garantir a realização tranquila do processo e permitir o melhor desempenho de todos os envolvidos:

1. Revisar cuidadosamente todas as questões, a fim de evitar erros ou ambiguidades que comprometam a aplicação ou correção. Questões mal elaboradas podem prejudicar a validade de todo o exame;
2. Após selecionar as questões e configurar o estilo, crie as variações de exame marcando a caixa “Template” ao lado do botão “Criar-Variações”. Isso assegura que o sistema gere os gabaritos

¹ (<https://github.com/fzampirolli/mctest/blob/master/exam/CVMCTest.py>)

8.7. RECOMENDAÇÕES PARA REALIZAÇÃO TRANQUILA DE EXAMES PRESENCIAIS

corretos e possibilite a correção automática por meio do QRCode (se houver algum erro na leitura do QRCode durante a correção automática, o gabarito enviado por e-mail deverá ser usado – certifique-se de guardar o arquivo CSV com cuidado);

3. Cada vez que você clicar em “Criar-Variações”, o ID será alterado, gerando novas variações com gabaritos diferentes no banco de dados. Para aplicar o exame, verifique se o ID ao lado deste botão corresponde ao ID em vermelho no PDF do exame (não se esqueça de atualizar a página para atualizar o ID). Isso é fundamental para garantir a correção automática, pois o ID gerado deve sempre coincidir com o do exame aplicado;
4. É viável empregar as variações existentes no banco de dados em múltiplas turmas, com a simples modificação da turma no exame, seguida dos botões “Salvar” e “Criar-PDF”;
5. Deixar “Retorno” como “Não” ao criar os exames, caso contrário elas serão enviadas aos estudantes por e-mail antes da aplicação;
6. Imprima um único exemplar do exame antes de produzir toda a tiragem, a fim de validar. Preencha o exame, digitalize e solicite a correção automática enviando o PDF e selecionando o botão “Upload-PDF”. Isso permite verificar se o sistema gerou o exame corretamente e realizou a correção automática adequadamente. Além disso, verifique se todos os exames ocupam apenas uma página frente e verso, evitando a necessidade de grampear folhas. Se todos os PDFs gerados estiverem corretos, solicite a impressão no modo frente e verso, utilizando uma impressora com bom toner. Por fim, verifique se a impressão foi realizada com sucesso, garantindo que os quatro discos pretos que delimitam o QR estão intactos e sem cortes na folha;
7. Organizar os exames em ordem alfabética sobre as filas de carteiras da sala, de modo que os estudantes não tenham dúvidas sobre qual é a deles. Isso garante que todos tenham acesso aos seus exames de forma rápida e organizada. Esse processo geralmente leva cerca de 10 minutos para 100 exames;
8. Instrua os estudantes a não utilizar fluidos de correção (conhecidos como “branquinho”) ou fazer rascunhos nos espaços designados pelos quatro discos pretos, a fim de evitar a interferência na correção automática dos gabaritos;
9. Digitalizar somente a frente dos exames em um único arquivo PDF por turma para permitir a correção em larga escala de maneira ágil e livre de erros. Evite o uso de impressoras que possam “engolir” as folhas;
10. Se ocorrer erro na decodificação do QRCode, é possível digitalizar apenas os exames com erro, utilizando diferentes configurações de digitalização da impressora. Outra alternativa é a correção manual, usando o gabarito enviado por e-mail. Certifique-se de verificar qual é a variação do exame do estudante recebido por e-mail ao clicar em “Criar-PDF” e o gabarito da variação recebida também por e-mail ao clicar em “Criar Variação”;

11. Envie os resultados aos estudantes somente após validar as correções para assegurar que tudo ocorreu conforme o esperado e não há erros. Se desejar enviar um gabarito aos estudantes apenas com QR, sem as questões, marque “Retorno” como “SIM” e não selecione a caixa abaixo do botão “Upload-PDF”. Em seguida, escolha o arquivo e clique no botão para realizar a correção automática e enviar os *feedbacks* aos estudantes por e-mail.

Seguir essas recomendações é crucial para a realização de exames presenciais tranquilamente. É essencial revisar as questões, criar variações e verificar os IDs tanto na página do exame quanto no PDF. Garantir a impressão correta do PDF, sem cortes no QR, é importante. Além disso, realizar a validação das correções antes de enviar os resultados assegura a qualidade e precisão do processo. Ao seguir essas recomendações, é possível promover a eficiência da correção automatizada.

8.8 Considerações finais

Este capítulo abordou a criação e correção de exames que envolvem tanto o QR quanto as QMs, além das QTs. Essa combinação de questões permite uma avaliação abrangente das habilidades dos estudantes e tem sido amplamente utilizada na Especialização em Tecnologia e Sistemas de Informação (TSI) da UFABC, como mencionado no Capítulo 1.

No processo de criação dos exames, é necessário criar as QMs e QTs, seguindo as instruções fornecidas neste capítulo, bem como nos Capítulos 4 – Questões estáticas e 5 – Questões paramétricas. A utilização de arquivos TXT para criar as questões não paramétricas é uma abordagem prática e eficiente para inseri-las no MCTest. Além disso, é fundamental criar as variações do exame por meio do botão “Criar-Variações”. Essa etapa garante a diversificação das questões e respostas disponibilizadas aos estudantes.

Após a criação das variações, é possível gerar o PDF do exame, incluindo os QR, QM e QT, através do botão “Criar-PDF”. É essencial revisar o PDF gerado para garantir a formatação correta das questões, respostas e demais elementos antes de disponibilizá-lo aos estudantes.

Na etapa de correção dos exames, o processo é simplificado quando se utiliza o MCTest, pois o gabarito é armazenado no servidor. O gabarito é identificado através da decodificação do QRcode e pode ser verificado a partir do CSV dos gabaritos correspondentes, caso a opção “Template” tenha sido selecionada antes de clicar em “Criar-Variações”. As respostas corretas para cada questão podem ser obtidas dessa forma, como exemplificado no capítulo.

Ao enviar o PDF preenchido pelos estudantes através do botão “Upload-PDF”, e selecionar a opção “Retorno=Sim”, o estudante receberá um *feedback* detalhado por e-mail, contendo as correções realizadas. O professor também receberá um arquivo ZIP com as correções realizadas em formato CSV, além de vários outros arquivos já citados anteriormente, facilitando a análise e o registro das correções.

No próximo capítulo, serão explorados os formatos de arquivos Aiken e XML ao criar variações de um exame. Esses recursos são altamente úteis para a criação de bancos de questões no Moodle.

Capítulo 9

Variações de exames com QM+QT para o Moodle

Conteúdo

9.1	Criando exames e exportando arquivos Aiken e XML	164
9.1.1	QM – equação de primeiro grau – idade	164
9.1.2	QM – equação da reta	165
9.1.3	QT com resposta exata – ordena parte do vetor	166
9.1.4	QT para a integração MCTest+Moodle+CodeRunner	169
9.1.5	Criando exame e exportando as variações	170
9.2	Importando arquivo XML no Moodle	179
9.3	Considerações finais	180

Neste capítulo, será apresentado um recurso interessante do MCTest que permite exportar as variações de um exame para arquivos nos formatos Aiken e XML, que podem ser utilizados pelo Moodle para criar um banco de questões. Será utilizado um exemplo de questão da área do círculo apresentada no Capítulo 8 – Exames com QR+QM e/ou QT, bem como a criação de novas questões para exemplificar melhor o processo. O artigo de Zampirolli, Batista, Pimentel e Braga (2021) detalha este recurso, e neste capítulo é resumido todo o processo.

Embora o Moodle suporte a criação de questões parametrizadas, que utilizam valores cullinga, essas questões têm limitações em relação às funções disponibilizadas pela linguagem PHP. Utilizar o MCTest para criar o banco de questões é vantajoso porque suporta muito mais variações de questões. É possível, por exemplo, criar um exame contendo QMs e QTs, com respostas exatas. Em seguida, é configurado o exame para gerar variações, salvando com um destes formatos Aiken ou XML. É possível então importar esses arquivos para o banco de questões no Moodle e criar uma atividade, incluindo as questões recém-importadas. Dessa forma, é possível criar atividades muito

mais parametrizadas do que as suportadas pelo Moodle utilizando os curingas.

Assim, apesar dos recursos disponíveis no Moodle, ele ainda apresenta limitações na criação de questões. Neste capítulo, será abordado como tentar contornar essas restrições para criar avaliações com diversas variações.

9.1 Criando exames e exportando arquivos Aiken e XML

Será criado um exame com três QMs paramétricas, cada uma contendo quatro alternativas, e uma QT com resposta exata. A primeira questão abordará a área do círculo, conforme apresentado no segundo exemplo da Seção 5.1.2 – QM paramétrica no MCTest. As outras três questões serão definidas nas seções subsequentes.

9.1.1 QM – equação de primeiro grau – idade

O Código 9.1 apresenta um algoritmo que soluciona o problema de encontrar as idades de duas pessoas, dada a soma e a diferença entre elas. O método `calcular_idades` recebe como parâmetros esses dois valores e utiliza equações matemáticas para determinar as idades da pessoa e do irmão. As idades são retornadas como uma *string* no formato “x e y”, em que x e y são as idades da pessoa e do irmão, respectivamente. Caso as idades não sejam números inteiros positivos, o método retorna uma *string* vazia.

O programa utiliza o método `random.sample` para gerar aleatoriamente pares de valores de soma e diferença no intervalo de 2 a 29. Para cada par de valores gerado, o programa chama o método `calcular_idades` e armazena as respostas válidas em uma lista. A alternativa correta da questão, apresentada na linha 36, considera os dados do enunciado e utiliza o método `calcular_idades` para determinar as idades dos irmãos apresentados na Figura 9.1. Esta resposta, `respostas[3]`, deve ser incluída na primeira alternativa (correta) do formulário da questão. As demais, encontradas nos elementos de índice 0, 1 e 2 da lista `respostas`, devem ser inseridas nas demais alternativas (erradas). Se desejar gerar uma questão com um maior número de alternativas, por exemplo, 10, basta modificar a linha com o comando `while` para ser menor que 10, e a alternativa correta estará no elemento 9 da lista `respostas`.

MCTest

Topic: Topico Exemplo
Group:
Short Description: idades irmãos
Type: QM
Difficulty: 1
Bloom taxonomy: remember
Last update: 2023-07-13
Who created: fzampirolli@ufabc.edu.br
Parametric: YES

#2465 1. Uma pessoa tem x anos e seu irmão tem y anos. Se a diferença entre suas idades é de 17 anos, e a soma das idades é 29, determine as idades da pessoa e do irmão, respectivamente.

A.*₃16 e 13 B.*₂18 e 5 C.*₁ 9 e 3 D.*₀23 e 6

Figura 9.1: Recorte do PDF gerado para a questão das idades dos irmãos, referente ao Código 9.1.

Questão:

```
1 Uma pessoa tem x anos e seu irmão tem y anos. Se a diferença entre suas idades é
2 de [[code:diferenca]] anos, e a soma das idades é [[code:soma]], determine as
3 idades da pessoa e do irmão, respectivamente.
4
5 [[def:
6     import random
7
8     def calcular_idades(soma, diferenca):
9         """Calcula idades, dados soma e diferença.
10        x + y = soma e x - y = diferenca ==> 2x = (soma + diferenca)"""
11
12        # Calcular a idade da pessoa
13        x = (soma + diferenca) / 2
14
15        # Calcular a idade do irmão
16        y = soma - x
17
18        if not x%1 and x>0 and y>0: # somente inteiros positivos
19            return f"{x:2.0f} e {y:2.0f}"
20        else:
21            return ""
22
23    # Lista de respostas
24    respostas = []
25
26    # Gerar 4 equações de reta aleatórias
27    while len(respostas) < 4:
28        # Gerar 2 valores aleatórios
29        soma, diferenca = random.sample(range(2, 30), 2)
30        r = calcular_idades(soma, diferenca)
31        if not r: # somente inteiros positivos
32            continue
33        if r not in respostas: # respostas distintas
34            respostas.append(r)
35
36    # [[code:respostas[3]]] # última será a correta
37 ]]
```

Código 9.1: Exemplo de QM paramétrica para calcular as idades dos irmãos.

9.1.2 QM – equação da reta

O Código 9.2 implementa um algoritmo para gerar quatro equações de reta aleatórias a partir de pontos com coordenadas inteiras entre 1 e 9. O método `equacao_reta` é utilizado para calcular a equação da reta que passa pelos dois pontos fornecidos. Para evitar duplicatas, o código verifica se a equação já está na lista de respostas antes de adicioná-la. Caso uma reta vertical seja gerada, ela é ignorada. A última alternativa da questão na linha 26 considera os dois pontos fornecidos no enunciado, apresentado na Figura 9.2.

Questão:

```

1 Qual é a equação da reta que passa pelos pontos ([[code:x1]], [[code:y1]]) e
2 ([[code:x2]], [[code:y2]])?  
3
4 [[def:
5 import random
6
7 def equacao_reta(x1, y1, x2, y2):
8     """Calcula a equação da reta que passa pelos pontos (x1, y1) e (x2, y2)"""
9     m = (y2 - y1) / (x2 - x1)
10    b = y1 - m * x1
11    return f"y = {m:.2f}x + {b:.2f}"
12
13 # Lista de respostas
14 respostas = []
15
16 # Gerar 4 equações de reta aleatórias e distintas
17 while len(respostas) < 4:
18     x1,y1,x2,y2 = random.sample(range(1,10), 4) # Gerar 2 pontos aleatórios
19     if x1 == x2: # A reta é vertical: não é possível calcular a inclinação m
20         continue
21
22     equacao = equacao_reta(x1, y1, x2, y2)
23     if equacao not in respostas:
24         respostas.append(equacao)
25
26 # [[code:respostas[3]]] # última será a correta
27 ]]

```

Código 9.2: Exemplo de QM paramétrica para calcular a equação da reta.

MCTest

Topic: Topico Exemplo
Group:
Short Description: equação da reta
Type: QM
Difficulty: 1
Bloom taxonomy: remember
Last update: 2023-07-13
Who created: fzampirolli@ufabc.edu.br
Parametric: YES

#2464 1. Qual é a equação da reta que passa pelos pontos (2, 9) e (4, 7)?

A. $y = 2.00x + 1.00$ B. $y = -1.00x + 11.00$ C. $y = 0.43x + 2.57$ D. $y = 2.67x + -7.00$

Figura 9.2: Recorte do PDF gerado para a questão da equação da reta definida no Código 9.2.

9.1.3 QT com resposta exata – ordena parte do vetor

Na Seção 8.1 – Desenvolvendo QMs e uma QT para inclusão em um exame, foi apresentado o Código 8.1, que descreve como criar uma QT com resposta exata. Nesta seção, será apresentado um

exemplo prático deste recurso, por meio da adaptação dos Códigos 5.21 e 5.21 de questão paramétrica integrada ao VPL.

Os Códigos 9.3 e 9.4 implementam uma questão paramétrica exata, utilizando o método `ordena_parte_vetor`. Esse método recebe como entrada um vetor de números inteiros, um índice de início e um índice de fim, e ordena apenas uma parte do vetor, delimitada pelos índices de início e fim.

O método utiliza um algoritmo de ordenação por seleção simples, que percorre o trecho do vetor a ser ordenado e, em cada iteração, seleciona o menor elemento da parte não ordenada e o coloca em sua posição correta na parte ordenada. Esse algoritmo é implementado por meio de dois laços aninhados: o primeiro percorre os elementos da parte do vetor a ser ordenada, enquanto o segundo percorre os elementos ainda não ordenados. A cada iteração do segundo laço, é verificado se o elemento atual é menor do que o elemento selecionado pelo primeiro laço. Se for o caso, os elementos são trocados de posição. O método retorna o vetor ordenado, e a ordenação é realizada *in-place*, ou seja, o vetor original é modificado diretamente pelo método. A Figura 9.3 apresenta um exemplo para esta questão.

A questão apresentada nos Códigos 9.3 e 9.4 já está preparada para ser aplicada em atividades VPL, seguindo os Códigos 5.21 e 5.21. No entanto, a parte crítica é a formatação da questão, limitada pelo Moodle, como serão apresentadas nas próximas seções.

Esta questão de ordenar vetor é um exemplo típico que não pode ser implementado no Moodle utilizando os valores curinga. Isso ilustra que para questões mais complexas, o MCTest é uma ferramenta adequada para criar um banco de questões e exportá-lo para o Moodle.

Questão:

```
1 Dado um vetor de inteiros de tamanho \(\ n = [[code:n]]\), ordenar o trecho do
2 vetor que começa no índice \(\ inicio = [[code:inicio]] \) e termina no índice
3 \(\ fim = [[code:fim]]\). Onde, \(\ 0 < inicio < fim < n \), ou seja, o trecho a
4 ser ordenado está entre início (inclusive) e fim (inclusive). Vale ressaltar
5 que os índices do vetor começam em 0 e terminam em \(\ n-1\).
6
7 Considere esta entrada:}
8 [[code:caso0_inp]]
9
10 %{{[[code:caso0_out]]}}%
```

Código 9.3: Exemplo de questão paramétrica exata – Parte 1: Descrição de questão.

Questão:

```

1  [[def:
2   import json, numpy as np
3
4   # Passo 1: Criar os parâmetros do enunciado da questão
5   n = np.random.randint(20,40)
6   inicio = np.random.randint(2,n//2-2)
7   fim = np.random.randint(n//2+2,n-2)
8
9   # Passo 2: Criar os casos de teste
10  inp_list, out_list = [], [] # Listas vazias para armazenar os casos de teste
11  casos_teste = 1 # Número de casos de teste desejado
12
13 def ordena_parte_vetor(vetor,inicio,fim):
14     """ Ordenação da parte do vetor entre inicio e fim"""
15     for i in range(inicio, fim+1):
16         for j in range(i+1, fim+1):
17             if vetor[i] > vetor[j]:
18                 vetor[i], vetor[j] = vetor[j], vetor[i]
19     # Retorno do vetor ordenado
20     return vetor
21
22 # Para cada caso de teste:
23 for i in range(casos_teste):
24
25     #>>> begin - casos de teste
26     # Gerar valores aleatórios entre 0 e 9 para o vetor
27     v = np.random.randint(10, size=n)
28
29     # Criar a entrada do caso de teste como uma string
30     inp = ' '.join(str(i) for i in v) + '\n'
31
32     # Calcular o vetor de saída
33     out = ' '.join(str(i) for i in ordena_parte_vetor(v,inicio,fim)) # + '\n'
34     #<<< end - casos de teste
35
36     # Adicionar a entrada e saída do caso de teste às listas
37     inp_list.append(inp), out_list.append(out)
38
39 # Passo 3: Criar o dicionário com os casos de teste
40 cases = {}
41 cases['input'] = np.array(inp_list).tolist()
42 cases['output'] = np.array(out_list).tolist()
43
44 # Passo 4: Mostrar um exemplo no enunciado da questão
45 caso0_inp, caso0_out = cases['input'][0], cases['output'][0]
46 ]]

```

Código 9.4: Exemplo de questão paramétrica exata – Parte 2: Bloco de código.

MCTest

Topic: Topico Exemplo
Group:
Short Description: resposta exata - vetor
Type: QT
Difficulty: 1
Bloom taxonomy: remember
Last update: 2023-07-13
Who created: fzampirolli@ufabc.edu.br
Parametric: YES

#2466 1. Dado um vetor de inteiros de tamanho $n = 28$, ordenar o trecho do vetor que começa no índice $início = 8$ e termina no índice $fim = 20$. Onde, $0 < início < fim < n$, ou seja, o trecho a ser ordenado está entre início (inclusive) e fim (inclusive). Vale ressaltar que os índices do vetor começam em 0 e terminam em $n - 1$.

Considere esta entrada: 6 3 2 7 8 5 3 2 9 0 9 6 9 8 8 8 0 0 7 9 1 2 0 5 9 9 0 6

Figura 9.3: Recorte do PDF gerado para a questão com resposta exata definida pelos Códigos 9.3 e 9.4.

Destaque:

1. No final da linha 33 do Código 9.4, foi necessário adicionar um comentário para assegurar uma formatação adequada da questão no Moodle. Essa informação está presente no enunciado da questão, na linha 10 do Código 9.3, com o intuito de armazenar a resposta correta da questão.
2. As mudanças de linha são obrigatórias a cada entrada/saída nas questões com integração VPL, conforme apresentado no Capítulo 5 – Questões paramétricas e também serão destacadas no próximo capítulo.

9.1.4 QT para a integração MCTest+Moodle+CodeRunner

A nova versão do MCTest 5.3 permite gerar XML para questões QT, compatível com o plugin CodeRunner. Para utilizar essa funcionalidade, configure a questão como nos exemplos dos Códigos 9.3 e 9.4.

Após a linha 42 do Código 9.4, adicione a seguinte linha para inserir a solução da questão em código:

```
cases['answer'] = ['''sua solução''']
```

O método `createFileDB_xml()` contido no arquivo `UtilsLatex` na pasta de exames no github.com/fzampirolli/mctest pode ser consultado para obter mais detalhes sobre a geração do XML. Vale notar que o texto da questão no formato LATEX é convertido de forma automática para o formato HTML utilizando o pacote `pandoc`; veja o método `latex_to_html()` neste mesmo arquivo `UtilsLatex`.

9.1.5 Criando exame e exportando as variações

Com as questões definidas anteriormente, é possível agora criar um exame seguindo as instruções apresentadas no Capítulo 6 – Visão geral dos exames, e exportar as variações necessárias.

Após a criação do exame, é possível selecionar as quatro questões apresentadas na Figura 9.4. Os detalhes do exame são apresentados na Figura 9.5. É importante notar que foi decidido criar somente cinco variações.

Listar Questões								
	Tópico	Descrição	Tipo	Dif.	Grupo	Par.	Ver	
<input checked="" type="checkbox"/>	Topico Exemplo	equação da reta	QM 4	1		yes	2464	
<input checked="" type="checkbox"/>	Topico Exemplo	idades irmãos	QM 4	1		yes	2465	
<input checked="" type="checkbox"/>	Topico Exemplo	área círculo - ex.1	QM 4	1	A	yes	2449	
<input checked="" type="checkbox"/>	Topico Exemplo	resposta exata	QT	1		yes	2466	

Figura 9.4: Recorte da tela do exame com as quatro questões marcadas.

Número de questões por Exame		Estilo do Exame	
Dificuldade 1	3	Questões por bloco	10
Dificuldade 2	0	Max Blocos Horiz.	1
Dificuldade 3	0	Quadro de respostas	Horizontal
Dificuldade 4	0	Respostas/Questões/Ambos	Ambos
Dificuldade 5	0	Ecológico	Sim
Alternativas por questão	4	Retorno	Não
Dissertativa	1	Data	13/07/2023
Variações	5	Período	Primeiro quadrimestre

Figura 9.5: Recorte da tela do exame com os detalhes da configuração.

Para prosseguir com a criação das variações, é necessário selecionar as opções “Aiken” e “XML”, conforme ilustrado na Figura 9.6, e em seguida clicar em “Criar-Variações”. O professor receberá um e-mail contendo dois anexos: um no formato Aiken (TXT) e outro no formato XML, como mostrado na Figura 9.7. No caso deste exame com ID 480 recém-criado, os arquivos recebidos serão:

Arquivos em formatos Aiken (TXT) e XML

```
report_Exam_480_variations_DB_aiken.txt
report_Exam_480_variations_DB.xml
```

Antes de criar os exames no botão acima, primeiro crie as variações. É necessário criar novas variações do exame cada vez que você muda as questões e o número de variações nos atributos abaixo. As opções marcadas abaixo serão enviadas para o seu e-mail.



Figura 9.6: Recorte da tela do exame com os detalhes da configuração.

 webmctest@ufabc.edu.br Caixa d...fabc.edu.br 18:38
MCTest: files created automatically; Exam: criarVariações; 2023-07-13 - 18:38:17
Para: Francisco Zampirolli < fzampirolli@ufabc.edu.br >

Prezada(o),

Esta mensagem contém arquivos criados automaticamente, com todas as variações deste exame.

AIKEN: Banco de questões para o Moodle.

Ver detalhes: "Facilitating the Generation of Parametric Questions and their export to Moodle" (FIE21)

XML: Banco de questões para o Moodle.

Ver detalhes: "Facilitating the Generation of Parametric Questions and their export to Moodle" (FIE21)



report_Exam_480_variations_DB_aiken.txt
report_Exam_480_variations_DB.xml

Figura 9.7: Recorte do e-mail recebido após clicar em “Criar-Variações”.

Detalhando o arquivo no formato Aiken

A seguir, são apresentadas as três QMs da primeira variação. Vale ressaltar que, na Figura 9.5, foram solicitadas cinco variações no formato Aiken, geradas pelo MCTest. No entanto, antes de importá-las no Moodle, é necessário remover os comentários adicionados pelo MCTest apenas para controle. Em versões de produção do MCTest, esses comentários podem ser facilmente removidos. Além disso, um aspecto desagradável deste formato é que não é possível importar questões com apenas resposta exata.

Conteúdo do arquivo *_aiken.txt, variação 0 e questão 1.

```
##### variation #####
#c:1 #id:2465 #topic:Topico Exemplo #type:QM #diff:1
```

Uma pessoa tem x anos e seu irmão tem y anos. Se a diferença entre suas idades é de 2 anos, e a soma das idades é 28, determine as idades da pessoa e do irmão, respectivamente.

- A) 15 e 13
- B) 20 e 1
- C) 11 e 2
- D) 18 e 1

ANSWER: A

...

Conteúdo do arquivo *_aiken.txt, variação 0 e questão 2.

...

```
#c:2 #id:2449 #topic:Topico Exemplo #type:QM #diff:1
```

Qual é a área do círculo de raio \((25 \))?

% comentário em LaTeX. Observe que a variável `raio1`, após "code:"% foi definida na linha 18 abaixo

- A) 2827.43
- B) 3019.07
- C) 1809.56
- D) 1963.50

ANSWER: D

```
#c:3 #id:2464 #topic:Topico Exemplo #type:QM #diff:1
```

Qual é a equação da reta que passa pelos pontos (2, 7) e (8, 6)?

- A) $y = 1.00x + -2.00$
- B) $y = -0.17x + 7.33$
- C) $y = 2.00x + 1.00$
- D) $y = 1.00x + 4.00$

ANSWER: B

Detalhando o arquivo no formato XML para QM

A seguir é detalhado o arquivo no formato XML que contém a primeira QM da primeira variação. Diferentemente do formato Aiken, os comentários iniciados com o símbolo “#” neste formato XML não precisam ser removidos. No entanto, antes de importá-las no Moodle, é necessário remover os comentários em L^AT_EX dentro de cada questão e gerar novamente o arquivo. Realizar esse

processo diretamente no arquivo XML pode ser mais complexo e propenso a erros, especialmente se for necessário gerar muitas variações. Por isso, é recomendado gerar uma única variação e verificar como ela é formatada no Moodle antes de prosseguir com a geração de outras variações.

Cabe salientar que a sintaxe no formato XML é complexa e não é destinada ao consumo humano. Por isso, foi apresentada somente a primeira QM gerada a seguir.

Foi criada, no código XML a seguir, uma hierarquia de categorias para as questões com base nas suas características, tais como tipo, tópico e nível de dificuldade. Essa organização visa tornar o banco de questões mais estruturado no Moodle. Além disso, foi criada uma categoria para cada questão do MCTest, facilitando a localização das questões no Moodle. No exemplo a seguir, é apresentada a hierarquia de categorias `multichoice/Topico_Exemplo/diff1/área círculo - ex.1`.

Conteúdo do arquivo *.xml, variação 0 e questão 1 – categoria.

```
<?xml version="1.0" encoding="UTF-8"?>
<quiz>
    <!-- category: #id:2449 #type:multichoice #topic:Topico_Exemplo #diff:1 #descr:área círculo - ex.1 -->

    <question type="category">

        <category>
            <text>
                $course$/top/multichoice/Topico_Exemplo/diff1/área círculo - ex.1
            </text>
        </category>

        <info format="moodle_auto_format">
            <text></text>
        </info>

        <idnumber></idnumber>
    </question>
    ...

```

Conteúdo do arquivo *.xml, variação 0 e questão 1.

```
...
<!-- question:#c:2#id:2449#type:multichoice#topic:Topico Exemplo#diff:1#descr:área círculo - ex.1#var:1-->

<question type="multichoice">

    <name>
        <text>Topico: Topico Exemplo Dificuldade: 1 </text>
    </name>

    <questiontext format="moodle_auto_format">

        <text><![CDATA[<p> Qual é a área do círculo de raio \(( 25 \)?<br></p>]]>
        % comentário em LaTex. Observe que a variável raio1, após "code:"
        % foi definida na linha 18 abaixo

        <br></p>]]>
        </text>

    </questiontext>
...

```

Conteúdo do arquivo *.xml, variação 0 e questão 1.

```
...
<generalfeedback format="moodle_auto_format">
  <text></text>
</generalfeedback>

<defaultgrade>1.000000</defaultgrade>

<penalty>0.3333333</penalty>

<hidden>0</hidden>

<idnumber></idnumber>

<single>true</single>

<shuffleanswers>true</shuffleanswers>

<answernumbering>abc</answernumbering>

<correctfeedback format="moodle_auto_format">
  <text>Sua resposta está correta.</text>
</correctfeedback>

<partiallycorrectfeedback format="moodle_auto_format">
  <text>Sua resposta está parcialmente correta.</text>
</partiallycorrectfeedback>

<incorrectfeedback format="moodle_auto_format">
  <text>Sua resposta está incorreta.</text>
</incorrectfeedback>

<shownumcorrect/>
...
...
```

Conteúdo do arquivo *.xml, variação 0 e questão 1 – alternativas.

```
...
<answer fraction="0" format="moodle_auto_format">

  <text><! [CDATA[<p> 2827.43
<br></p>]]></text>

  <feedback format="moodle_auto_format">
    <text></text>
  </feedback>

</answer>
...
...
```

Conteúdo do arquivo *.xml, variação 0 e questão 1 – alternativas.

```
...
<answer fraction="0" format="moodle_auto_format">

    <text><! [CDATA[<p> 3019.07
<br></p>]]></text>

    <feedback format="moodle_auto_format">
        <text></text>
    </feedback>

</answer>
...
```

Conteúdo do arquivo *.xml, variação 0 e questão 1 – alternativas.

```
...
<answer fraction="0" format="moodle_auto_format">

    <text><! [CDATA[<p> 1809.56
<br></p>]]></text>

    <feedback format="moodle_auto_format">
        <text></text>
    </feedback>

</answer>
...
```

Conteúdo do arquivo *.xml, variação 0 e questão 1 – alternativas.

```
...
<answer fraction="100" format="moodle_auto_format">

    <text><! [CDATA[<p> 1963.50
<br></p>]]></text>

    <feedback format="moodle_auto_format">
        <text></text>
    </feedback>

</answer>

</question>
...
```

Detalhando o arquivo no formato XML – questão com resposta exata

A seguir, serão apresentados os detalhes do arquivo XML referente à questão com resposta exata. A hierarquia desta questão foi `top/shortanswer/Topico Exemplo/diff1/resposta exata`. É relevante ressaltar que essa questão foi agrupada na categoria `shortanswer`, diferentemente das

9.1. CRIANDO EXAMES E EXPORTANDO ARQUIVOS AIKEN E XML

QMs, que utilizam a categoria `multichoice`. É importante lembrar que é necessário criar as questões no formato adequado no MCTest para poderem ser importadas e exibidas corretamente no Moodle, que possui sua própria formatação.

Conteúdo do arquivo *.xml, variação 0 e questão com resposta exata – categoria.

```
...
<!-- category:#id:2466#type:shortanswer#topic:Topico Exemplo#diff:1#descr:resposta exata - vetor-->
<question type="category">
    <category>
        <text>
            $course$/top/shortanswer/Topico Exemplo/diff1/resposta exata - vetor
        </text>
    </category>

    <info format="moodle_auto_format">
        <text></text>
    </info>

    <idnumber></idnumber>
</question>
...
...
```

Conteúdo do arquivo *.xml, variação 0 e questão com resposta exata – questão.

```
...
<!--question:#c:4#id:2466#type:shortanswer#topic:Topico Exemplo#diff:1#descr:resposta exata - vetor#var:1-->

<question type="shortanswer">

    <name>
        <text>Topico: Topico Exemplo Dificuldade: 1 </text>
    </name>

    <questiontext format="moodle_auto_format">

        <text>

            <![CDATA[<p> Dado um vetor de inteiros de tamanho \(( n = 31 \), ordenar o trecho do
vetor que começa no índice \(( inicio = 10 \) e termina no índice
\(( fim = 24 \). Onde, \(( 0 < inicio < fim < n \), ou seja, o trecho
a ser ordenado está entre início (inclusive) e fim (inclusive). Vale
ressaltar que os índices do vetor começam em 0 e terminam em \(( n-1 \).
</p>]]>
            Considere esta entrada:
            8 1 4 1 2 0 1 6 1 9 6 3 7 2 7 8 9 3 7 2 0 7 6 4 3 0 1 1 9 4 1
            <br></p>]]>

        </text>
    </questiontext>
...
...
```

Conteúdo do arquivo *.xml, variação 0 e questão com resposta exata – resposta.

...

```
<generalfeedback format="moodle_auto_format">
    <text></text>
</generalfeedback>

<defaultgrade>1.000000</defaultgrade>

<penalty>0.3333333</penalty>

<hidden>0</hidden>

<idnumber></idnumber>

<single>true</single>

<shuffleanswers>true</shuffleanswers>

<answernumbering>abc</answernumbering>

<correctfeedback format="moodle_auto_format">
    <text>Sua resposta está correta.</text>
</correctfeedback>

<partiallycorrectfeedback format="moodle_auto_format">
    <text>Sua resposta está parcialmente correta.</text>
</partiallycorrectfeedback>

<incorrectfeedback format="moodle_auto_format">
    <text>Sua resposta está incorreta.</text>
</incorrectfeedback>

<shownumcorrect/>

<answer fraction="100" format="moodle_auto_format">
    <text>
        8 1 4 1 2 0 1 6 1 9 0 2 2 3 3 3 4 6 6 7 7 7 7 8 9 0 1 1 9 4 1
    </text>
</answer>

</question>

...
```

9.2 Importando arquivo XML no Moodle

Como o formato Aiken apresenta algumas limitações, como a impossibilidade de criar questões com resposta exata, será apresentado a seguir um tutorial para importar e criar questionários no Moodle usando o formato XML.

Considerando a versão 4.1 do Moodle, é necessário importar o arquivo XML gerado pelo MCTest. Para fazer isso, acesse a opção “Banco de questões”, disponível no ícone de engrenagem no canto superior direito ou no menu “Administração” à esquerda. Em seguida, basta escolher a opção “Importação”, selecionar o formato XML e mover o arquivo correspondente para a área especificada. Ao clicar em “Importar”, as 20 questões serão carregadas e poderão ser visualizadas, como ilustrado na Figura 9.8 para uma QM e na Figura 9.9 para uma questão com resposta exata. Na Figura 9.5 foram 3 QMs, 1 QT e 5 variações $((3 + 1) * 5 = 20)$.

Vale ressaltar que o uso de formatação em L^AT_EX no Moodle apresenta muitas limitações em comparação com o PDF gerado pelo MCTest, como ilustrado na Figura 9.9.

The screenshot shows a Moodle questionnaire interface. At the top, it says "Questão 1" and "Resposta salva Vale 1,00 ponto(s)". Below that is a question: "Qual é a equação da reta que passa pelos pontos (7, 6) e (8, 2)?". Underneath the question are four options:

- a. $y = 4.00x + -20.00$
- b. $y = 5.00x + -14.00$
- c. $y = -4.00x + 34.00$
- d. $y = 0.17x + 0.50$

Below the options is a red link "Limpar minha escolha". At the bottom are five buttons: "Começar de novo", "Salvar", "Preencher com respostas corretas", "Enviar e finalizar" (highlighted in green), and "Fechar pré-visualização".

Figura 9.8: Exemplo de QM no Moodle.

Em termos de comparação, observe na Figura 9.10 como seria um exame em PDF gerado pelo MCTest com essas variações das 3 QMs, seguido por uma QT com resposta exata. O professor também pode corrigir esse estilo de exame digitalizando-o para corrigir as QMs. Quanto à QT, o professor pode corrigi-la manualmente, comparando a resposta exata do estudante com o gabarito gerado pelo MCTest ao escolher a opção “Template” antes de clicar em “Criar-Variações”. O artigo de Zampirolli, Batista, Iriarte e Junior (2020) detalha esse processo de correção automática utilizando a opção “Template”, também resumido na Seção 12.2.2 – Artigo descrevendo o MCTest em conjunto

Questão 1 Resposta salva Vale 1,00 ponto(s).

Dado um vetor de inteiros de tamanho $n = 26$, ordenar o trecho do vetor que começa no índice $inicio = 10$ e termina no índice $fim = 22$. Onde, $0 < inicio < fim < n$, ou seja, o trecho a ser ordenado está entre início (inclusive) e fim (inclusive). Vale ressaltar que os índices do vetor começam em 0 e terminam em $n - 1$. Considere esta entrada: 7 1 2 0 6 2 0 9 7 3 2 8 3 4 1 9 9 7 5 9 0 7 1 2 9 0

Resposta:

[Começar de novo](#) [Salvar](#) [Preencher com respostas corretas](#) [Enviar e finalizar](#) [Fechar pré-visualização](#)

Figura 9.9: Exemplo da questão exata do Moodle.

com *Google Forms* e *Google Sheets*.

Instituto Exemplo
Curso Exemplo
Disciplina: Disciplina Exemplo
Prof.: Francisco Zampirolli, Francisco Professor
Turma: fz-turmaTeste
Exame: criarVariações

Sala: virtual
Data: 13-07-2023

Ass.: _____
Estudante: Estudante Exemplo Sobrenome ID/RA: 00001

#E480#V60994 - 2023-07-14 - 08:51:22



A	1	2	3
B	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
C	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
D	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Instruções:

- (a) desligar o celular

Questões de Múltipla Escolha:

1. Qual é a equação da reta que passa pelos pontos (9, 2) e (1, 7)?
A. $y = -1.33x + 10.67$ B. $y = -0.62x + 7.62$ C. $y = -7.00x + 58.00$ D. $y = 0.80x + 2.40$
2. Qual é a área do círculo de raio 27?
A. 6082.12 B. 5026.55 C. 7542.96 D. 2290.22
3. Uma pessoa tem x anos e seu irmão tem y anos. Se a diferença entre suas idades é de 11 anos, e a soma das idades é 17, determine as idades da pessoa e do irmão, respectivamente.
A. 12 e 4 B. 14 e 3 C. 23 e 4 D. 10 e 3

Questões Dissertativas

4. Dado um vetor de inteiros de tamanho $n = 21$, ordenar o trecho do vetor que começa no índice $inicio = 7$ e termina no índice $fim = 13$. Onde, $0 < inicio < fim < n$, ou seja, o trecho a ser ordenado está nos limites do vetor. Vale ressaltar que os índices do vetor começam em 0 e terminam em $n - 1$. Considere esta entrada: 4 6 4 9 6 1 8 9 6 6 9 6 3 8 6 8 4 1 3 6 4

Figura 9.10: Exemplo exame gerado pelo MCTest.

9.3 Considerações finais

Neste capítulo apresentou-se a utilização do recurso do MCTest que permite exportar variações de um exame nos formatos Aiken e XML para o Moodle, a fim de criar um banco de questões.

9.3. CONSIDERAÇÕES FINAIS

Utilizou-se uma questão apresentada no Capítulo 8 - [Exames com QR+QM e/ou QT](#), além de criar novas questões para exemplificar o processo.

Verificou-se que o uso do MCTest para criar o banco de questões é vantajoso em comparação à criação de questões parametrizadas diretamente no Moodle, uma vez que o MCTest suporta uma variedade maior de parametrizações de questões. Após configurar o exame para gerar variações, é possível exportá-las nos formatos Aiken ou XML e importá-las para o banco de questões do Moodle, criando uma atividade com as questões importadas. Dessa forma, é possível criar atividades muito mais parametrizadas do que aquelas suportadas apenas pelos valores curinga no Moodle.

Na seção de criação do exame e exportação das variações, foram demonstrados exemplos de como criar um exame com três QMs paramétricas e uma QT com resposta exata, explicando como o MCTest pode ser utilizado para gerar variações.

Foi detalhado o conteúdo dos arquivos nos formatos Aiken e XML para as QMs e QTs com respostas exatas, destacando a necessidade de remover os comentários adicionados pelo MCTest nos arquivos Aiken antes de importá-los para o Moodle. Quanto aos arquivos XML, ressaltou-se a complexidade da sintaxe e a importância de remover os comentários em L^AT_EX dentro de cada questão no MCTest. Também foi necessário prestar atenção especial à formatação utilizada, como a matemática, já que pode ser inválida no Moodle.

Por fim, apresentou-se um tutorial para importar e criar questionários no Moodle utilizando o formato XML, mostrando como acessar a opção de importação, escolher o formato XML e carregar o arquivo correspondente. Demonstrou-se que as questões são carregadas no Moodle, mesmo que haja algumas limitações na formatação em L^AT_EX.

Concluiu-se que o uso do MCTest em conjunto com o Moodle é uma abordagem eficaz para criar exames com uma variedade maior de questões parametrizadas, superando as limitações do Moodle na criação dessas questões. A exportação das variações nos formatos Aiken e XML permite uma integração mais fluida e flexível entre as duas plataformas, possibilitando a criação de atividades mais personalizadas e de qualidade no Moodle.

No próximo capítulo, serão abordados a criação de exames contendo questões com correção automática em atividades VPL do Moodle, apresentando exemplos e detalhando todo o processo de correção automática desses exames.

Capítulo 10

Exames com MCTest+Moodle+VPL

Conteúdo

10.1 Criando um exame com QT integrado ao VPL	184
10.1.1 Criando uma nova questão com matriz integrada ao VPL	184
10.1.2 Criando as variações e imprimindo o PDF do exame	188
10.2 Criando e configurando a atividade VPL para o exame	189
10.3 Soluções da atividade VPL para o exame	191
10.4 Detalhando o arquivo <code>linker.json</code>	193
10.5 Considerações finais	196

Neste capítulo, serão descritos exames que contêm questões dissertativas (QTs) paramétricas, nas quais os estudantes devem submeter exercícios de programação (EPs) em atividades VPL do Moodle. Esses tipos de questões foram detalhados na Seção 5.4 – QT paramétrica, com código. A questão de parcelas sem juros criada nesta seção será utilizada em um novo exame, no qual será detalhado todo o processo neste capítulo.

Esse tipo de exame tem sido amplamente utilizado na UFABC em disciplinas que envolvem programação, como Bases Computacionais da Ciência (equivalente à CS0 – *Computer Science 0*), para estudantes ingressantes. Nessa disciplina, a linguagem de programação Python é empregada juntamente com a biblioteca `pandas`, permitindo que os estudantes pratiquem estatísticas em arquivos no formato CSV.

Outra disciplina que faz uso frequente dessa integração com o *plugin* VPL é a de Processamento da Informação (PI), equivalente à CS1 (*Computer Science 1*). CS1 representa a primeira disciplina introdutória em Ciência da Computação, geralmente oferecida no primeiro ano de graduação em cursos dessa área. A definição CS1 foi estabelecida pela ACM (*Association for Computing Machinery*), uma das principais associações profissionais de Ciência da Computação do mundo (HOGAN, 2023).

Ao longo dos anos, diversos professores da UFABC têm contribuído para o desenvolvimento de questões, resultando em um total de 2.516 questões cadastradas até janeiro de 2024. Desses questões, 691 são do tipo paramétricas, como indicado em mctest.ufabc.edu.br.

É importante destacar que, embora essa metodologia de avaliação seja adotada em diversas turmas, muitos professores ainda não utilizam essa abordagem ou outras similares. Isso pode indicar uma falta de padronização dos conteúdos apresentados aos estudantes, o que pode prejudicar o processo de ensino-aprendizagem, com detalhado por [Zampirolli, Goya, Pimentel e Kobayashi \(2018\)](#).

Todo o processo de criação de exames descrito nos capítulos anteriores pode ser aplicado a exames que utilizam a integração do VPL do Moodle, com algumas adaptações. Este capítulo apresentará especificamente essa integração, que contou com a ajuda do Prof. Dr. Paulo Henrique Pisani e de seu orientando Heitor Rodrigues Savegnago na adaptação dos arquivos de configuração do VPL. Essa colaboração resultou em diversas publicações a partir de 2019, resumidas em vision.ufabc.edu.br. Neste capítulo, não serão detalhadas as modificações realizadas nesses arquivos de configuração, mas apenas será destacado como incluir os arquivos gerados pelo MCTest, necessários para a correção automática no *plugin* VPL.

10.1 Criando um exame com QT integrado ao VPL

Como já foi introduzido na Seção [5.4 – QT paramétrica, com código](#), o VPL é um *plugin* do Moodle para correção de atividades em que os estudantes devem submeter seus EPs para correção automática. Basicamente, são dois arquivos criados pelo MCTest que devem ser inseridos na atividade VPL do Moodle. O primeiro é o arquivo `linker.json`, que contém todas as variações do exame, juntamente com os detalhes de cada questão, incluindo os casos de teste. Esse arquivo é enviado para o e-mail do professor ao clicar no botão “Criar-Variações” na tela de exame. Assim, toda vez que o professor clicar nesse botão, será gerado um novo conjunto de variações que deverá ser atualizado na atividade VPL. O segundo arquivo é o `students_variations.csv`, que contém a variação do exame sorteada para cada estudante. Esse arquivo é enviado para o e-mail do professor ao clicar no botão “Criar-PDF”, também na tela do exame.

Nesta seção, um exame será elaborado utilizando a questão sobre parcelas sem juros, conforme demonstrado nos Códigos [5.21](#) e [5.22](#). Uma versão alterada dessa questão também foi ilustrada na Figura [5.9](#). Posteriormente, uma questão simples será acrescentada, com integração ao VPL, para ser incluída neste novo exame.

10.1.1 Criando uma nova questão com matriz integrada ao VPL

Considere a seguinte questão sobre matrizes para a disciplina CS1, apresentada nos Códigos [10.1](#) e [10.2](#). Nesta questão, o usuário deve ler os elementos de uma matriz de dimensões paramétricas, calcular a soma dos elementos e exibir o resultado, como apresentado na Figura [10.1](#).

O Código [10.2](#) utiliza as bibliotecas `json` e `numpy` para criar valores paramétricos e casos de teste de uma questão envolvendo matrizes. No Passo 1, são definidos os parâmetros Linhas e Colunas, que representam o tamanho da matriz e serão incluídos no enunciado da questão. No Passo

Questão:

```
1 Considere uma matriz de inteiros de dimensões \(([code:Linhas]] \times
2 [[code:Colunas]])\). Escreva um programa capaz de calcular a soma de todos
3 os elementos dessa matriz. O programa deve solicitar ao usuário os números
4 inteiros da matriz, linha por linha. Ao final, o programa deve exibir a
5 soma de todos os elementos da matriz. Veja exemplo a seguir: \\
6
7 \noindent\textbf{Exemplo de Entrada:}
8 \begin{verbatim}
9 [[code:caso0_inp]]
10 \end{verbatim}
11
12 \noindent\textbf{Exemplo de Saída:}
13 \begin{verbatim}
14 [[code:caso0_out]]
15 \end{verbatim}
16
17 % necessário para gerar casos de testes no moodle
18 \begin{comment}
19 [[code:moodle_cases]]
20 \end{comment}
```

Código 10.1: Exemplo de questão com matriz utilizando MCTest+Moodle+VPL – Parte 1: Descrição de questão.

2, são criados casos de teste com valores aleatórios na matriz e, em seguida, é gerada a entrada e saída para cada caso. O método `matriz2texto` converte a matriz em formato de texto. Os casos de teste são armazenados em listas de entrada (`inp_list`) e saída (`out_list`). No Passo 3, é criado um dicionário com os casos de teste e, em seguida, é convertido em formato JSON. Finalmente, no Passo 4, é mostrado um exemplo no enunciado da questão, exibindo a entrada e saída do primeiro caso de teste. O código pode ser utilizado para gerar casos de teste para uma questão relacionada a matrizes e cálculos.

Um método mais genérico para `matriz2texto` é apresentado no Código 10.3 para formatar uma matriz com elementos de vários dígitos. O método recebe uma matriz `f` representada como uma matriz `numpy ndarray`. O método tem o propósito de converter essa matriz em uma representação de texto adequada para impressão. Primeiro, o método obtém o número de linhas (`l`) e colunas (`c`) da matriz `f` usando `f.shape`. Em seguida, é feita uma verificação para determinar o formato adequado para a representação dos elementos da matriz na `string` de texto. Se o valor mínimo da matriz for menor que zero, a variável `num_digito` é criada usando a formatação `%Xd`, em que `X` é a quantidade de dígitos necessários para representar o valor máximo da matriz em texto. Caso contrário, a variável `num_digito` é criada usando `%Yd`, em que `Y` é a quantidade de dígitos necessários para representar o valor máximo da matriz em texto com valores negativos. Posteriormente, o método itera sobre cada elemento da matriz e formata cada valor usando a variável `num_digito`, concatenando-os na `string` de texto. A cada linha, é adicionado um caractere de quebra de linha '`\n`' para iniciar uma nova linha na representação final. O método retorna a `string` de texto contendo a representação da

Questão:

```

1  [[def:
2   import json, numpy as np
3
4   # Passo 1: Criar os parâmetros do enunciado da questão
5   Linhas, Colunas = np.random.randint(4, 8, size=2)
6
7   # Passo 2: Criar os casos de teste
8   inp_list, out_list = [], [] # Listas vazias para armazenar os casos de teste
9   casos_teste = 2 # Número de casos de teste desejado
10  # Aumentar esse número após validar a questão também na atividade VPL do Moodle
11
12 def matriz2texto(M):
13     """Converte uma matriz em formato de texto."""
14     texto = ''
15     for linha in M:
16         texto += ' '.join(f"elemento:{id}" for elemento in linha) + '\n'
17     return texto
18
19 # Para cada caso de teste:
20 for i in range(casos_teste):
21
22     #>>> begin - casos de teste
23     # Gerar valores aleatórios entre 0 e 9 para a matriz
24     M = np.random.randint(10, size=(Linhas, Colunas))
25
26     # Criar a entrada do caso de teste como uma string
27     inp = matriz2texto(M) + '\n'
28
29     # Calcular o valor da parcela arredondado para duas casas decimais
30     out = f'soma = {np.sum(M)}'
31     #<<< end - casos de teste
32
33     # Adicionar a entrada e saída do caso de teste às listas
34     inp_list.append(inp), out_list.append(out)
35
36 # Passo 3: Criar o dicionário com os casos de teste
37 cases = {}
38 cases['input'] = np.array(inp_list).tolist()
39 cases['output'] = np.array(out_list).tolist()
40 moodle_cases = json.dumps(cases)
41
42 # Passo 4: Mostrar um exemplo no enunciado da questão
43 caso0_inp, caso0_out = cases['input'][0], cases['output'][0]
44 #print(moodle_cases) # para testar em uma IDE
45 ]]

```

Código 10.2: Exemplo de questão com matriz utilizando MCTest+Moodle+VPL – Parte 2: Bloco de código em Python.

matriz. Esse método é útil para visualizar o conteúdo da matriz `f` de forma legível em uma saída de texto, permitindo uma inspeção mais clara dos valores contidos na matriz.

Questão:

```
1 def matriz2texto2(f):
2     l, c = f.shape
3     if np.min(f) < 0:
4         num_digitos = '%' + str(1 + len(str(np.max(f)))) + 'd'
5     else:
6         num_digitos = '%' + str(len(str(np.max(f)))) + 'd'
7     #print("'" + num_digitos + "'")
8     texto = ''
9     for i in range(l):
10        for j in range(c):
11            texto += num_digitos % f[i][j]
12        texto += '\n'
13    return texto
```

Código 10.3: Método mais genérico que `matriz2texto` para formatar uma matriz.

Essa questão apresentada nos Códigos 10.1 e 10.2 é parametrizada apenas nas dimensões da matriz, que variam entre 4 e 7, conforme indicado na linha 5 do Código 10.2. Para torná-la ainda mais parametrizável, seria possível variar o enunciado para calcular a soma, máximo, mínimo ou média dos elementos, considerando os índices ou valores pares, ou ímpares. Com essas variações, seria possível criar um total de 16 variações, sem considerar as variações nas dimensões da matriz, que poderiam ser aumentadas para criar ainda mais variações.

Dessa forma, seria possível criar uma ampla variedade de questões utilizando a mesma estrutura básica de código, tornando o processo de criação e correção de questões mais eficiente e flexível. É importante lembrar que, ao criar questões parametrizadas, é fundamental garantir que todas as variações possíveis sejam coerentes e bem formuladas, além de estarem no mesmo nível de dificuldade na resolução, a fim de garantir a qualidade do exame.

#2463 1. Considere uma matriz de inteiros de dimensões 7×6 . Escreva um programa capaz de calcular a soma de todos os elementos dessa matriz. O programa deve solicitar ao usuário os números inteiros da matriz, linha por linha. Ao final, o programa deve exibir a soma de todos os elementos da matriz. Veja exemplo a seguir:

Exemplo de Entrada:

```
4 0 4 8 5 9
2 1 8 5 8 6
7 2 2 7 6 6
5 0 0 3 7 5
1 3 9 2 0 0
7 9 0 8 0 3
3 8 4 8 3 1
```

Exemplo de Saída:

```
soma = 179
```

Figura 10.1: Recorte do PDF gerado para a questão de matriz definida nos Códigos 10.1 e 10.2.

10.1.2 Criando as variações e imprimindo o PDF do exame

Para criar um novo exame, siga os passos descritos no Capítulo 6 – Visão geral dos exames. Após definir o nome do exame, escolha a(s) turma(s), tópico(s) e as questões, preencha o restante da tela conforme apresentado na Figura 10.2, realizando as devidas alterações nos campos “Dificuldade 1”, “Dissertativa” e “Respostas/Questões/Ambos”. Finalize clicando no botão “Salvar”.

Figura 10.2: Recorte da tela de exame, configurando os detalhes do exame com integração VPL.

Agora, para criar as variações, é importante selecionar a opção “Json” antes de clicar no botão “Criar Variações”, conforme apresentado na Figura 10.3.

Antes de criar os exames no botão acima, primeiro crie as variações. É necessário criar novas variações do exame cada vez que você muda as questões e o número de variações nos atributos abaixo. As opções marcadas abaixo serão enviadas para o seu e-mail.



Figura 10.3: Recorte da tela do exame para criar as variações e enviar o arquivo no formato JSON.

Ao fazer isso, um arquivo no formato JSON com o nome *_linker.json será gerado e enviado para o e-mail do professor. Esse arquivo deve ser renomeado para linker.json. Veja na Figura 10.4 o e-mail recebido, com remetente webmctest@ufabc.edu.br. O arquivo JSON contém informações detalhadas sobre as variações do exame, incluindo as questões, as entradas utilizadas e as respostas esperadas.

Após criar as variações, é possível gerar um PDF por turma contendo os exames dos estudantes. No entanto, antes de fazer isso, uma boa estratégia seria validar a atividade VPL no Moodle. Para isso, crie uma turma de teste incluindo somente o professor como estudante. Es-

 webmctest@ufabc.edu.br Caixa d...fabc.edu.br 06:48
MCTest: files created automatically; Exam: exameTesteVPL; 2023-07-11 - 06:48:01
Para: Francisco Zampirolli < fzampirolli@ufabc.edu.br>

Prezada(o),

Esta mensagem contém arquivos criados automaticamente, com todas as variações deste exame.

JSON: Casos de teste a serem inseridos no Moodle para correção automática dos códigos enviados pelos alunos, seguindo os seguintes passos:

1. Use o PDF com os resultados gerados com esta data e hora (EXATAMENTE): "2023-07-11 - 06:48:01"
 2. Salve e renomeie os arquivos "linker.json" e "students_variations.csv"
 3. Após criar uma atividade VPL no Moodle fazer upload em "arquivos de execução", de "linker.json" e "students_variations.csv"
 4. Adicionar também os arquivos disponíveis em github.com/fzampirolli/mctest/VPL_modification
 5. Também habilite esses arquivos em "Arquivos a serem mantidos durante a execução"
- Observação: "students_variations.csv" será enviado por e-mail ao criar PDFs dos exames dos alunos.
Ver detalhes:
A. <https://doi.org/10.5753/cbie.sbie.2020.1573> - v.01
B. "Automated assessment of parametric programming in a large-scale course" (LACLO21) - v.10
C. <https://sol.sbc.org.br/index.php/sbie/article/view/18135/17969>

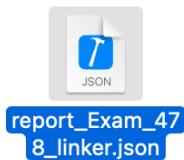


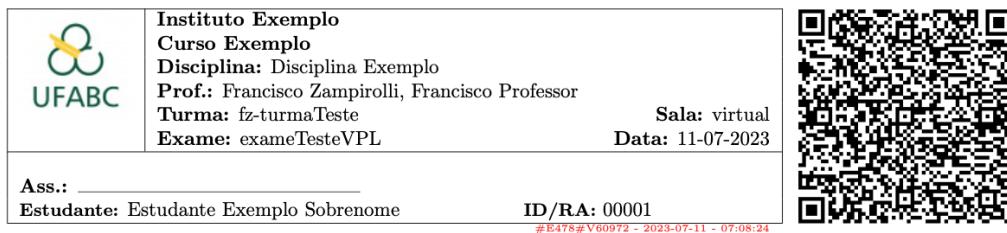
Figura 10.4: Recorte do e-mail recebido após clicar em “Criar-Variações”.

colha essa turma no exame e, em seguida, clique no botão “Criar-PDF”. Um PDF com o exame será enviado ao professor, conforme mostrado na Figura 10.5. No mesmo e-mail, será anexado o arquivo *_students_variations.csv, que deve ser renomeado para students_variations.csv. Este arquivo contém as variações de cada estudante.

10.2 Criando e configurando a atividade VPL para o exame

Adaptando as instruções definidas na Seção 5.4 – QT paramétrica, com código para criar uma atividade VPL no Moodle:

1. Acesse o Moodle e vá para a disciplina em que deseja adicionar um exame;
2. Clique em “Adicionar uma atividade ou recurso” e selecione “Laboratório Virtual de Programação” para criar uma nova atividade VPL;
3. Preencha os detalhes da atividade, como nome, descrição e pontuação. Em “Número máximo de arquivos”, defina 2, pois o exame criado neste capítulo possui duas questões. Finalize salvando e mostrando a atividade. Em seguida, clique no ícone de engrenagem para acessar as configurações avançadas da atividade e siga os próximos passos;
4. Na seção “Opções de execução”, selecione a linguagem “Python” como “Script de execução”, deixando apenas “Avaliar” e “Atribuição automática de nota” como “Sim”, por exemplo;

**Instruções:**

(a) desligar o celular

Questões Dissertativas

1. Considere uma matriz de inteiros de dimensões 7×6 . Escreva um programa capaz de calcular a soma de todos os elementos dessa matriz. O programa deve solicitar ao usuário os números inteiros da matriz, linha por linha. Ao final, o programa deve exibir a soma de todos os elementos da matriz. Veja exemplo a seguir:

Exemplo de Entrada:

```
8 2 8 8 7 3
0 4 9 3 9 8
5 1 5 1 6 2
2 1 0 7 7 1
5 8 6 6 3 3
4 7 9 5 9 9
3 3 7 2 1 4
```

Exemplo de Saída:

```
soma = 201
```

2. Escreva um programa que auxilia uma loja na hora de efetuar uma venda. Seu programa deve perguntar o preço do produto e considerar que a loja parcela este preço em somente em 12 parcelas (sem juros). O seu programa deve mostrar quanto será o valor de uma parcela (arredondando para duas casas decimais apenas). Veja exemplo a seguir:

Exemplo de Entrada:

```
5009.8
```

Exemplo de Saída:

```
417.48
```

Figura 10.5: Recorte da tela do PDF do exame com integração VPL.

10.3. SOLUÇÕES DA ATIVIDADE VPL PARA O EXAME

5. Na seção “Arquivo de execução”, inclua os arquivos disponíveis na última versão disponível em github.com/fzampirolli/mctest/tree/master/VPL_modification/V13-Moodle_v4.1. Inclua também os novos arquivos `linker.json` e `students_variations.csv`, que foram gerados recentemente pelos botões “Criar-Variações” e “Criar-PDF”, respectivamente;
6. Para que o Moodle selecione a variação correta do exame, é crucial que o endereço de e-mail seja idêntico tanto no Moodle quanto no arquivo `students_variations.csv`;
7. Em “Arquivos a serem mantidos durante a execução”, marque todas as opções e clique em Salvar;
8. Para realizar a atividade do exame, clique na opção “Atividade de teste” e em seguida em “Editar”. Será aberta uma janela solicitando o nome do arquivo, que deve ser exatamente `Q1.py`. Caso utilize o nome `q1.py`, a atividade não funcionará corretamente. Caso deseje criar um segundo arquivo, clique no ícone “+” e o ícone para criar um novo arquivo, em seguida utilize exatamente o nome do arquivo `Q2.py`.

Destaque:

1. No item 6, na versão 5.3 do MCTest, o email foi incluído no arquivo `students_variations.csv` como parte das informações do estudante, juntamente com o nome completo e a variação sorteada;
2. Nas versões anteriores, a alteração do nome do estudante de **Estudante Exemplo Sobrenome** para **Francisco Exemplo Zampirolli**, conforme mostrado na Figura 10.5, era necessária porque a chave de busca era o Nome e Sobrenome do estudante;
3. Agora, com a versão 5.3, a chave de busca é o email do estudante.

10.3 Soluções da atividade VPL para o exame

Após criar os arquivos `Q1.py` e `Q2.py`, inclua soluções para resolver essas questões. Por exemplo, confira no Código 10.4 uma solução para a primeira questão de matriz. A parte crítica desta questão é o estudante ter que ler 7 linhas, contendo 6 elementos em cada linha, conforme ilustrado na Figura 10.5.

Uma solução mais concisa e genérica para a questão Q1 é apresentada no Código 10.5. É importante ressaltar que essa solução é viável porque todos os casos de teste gerarão entradas no mesmo formato, como exemplificado na Figura 10.5.

Essa solução consegue ler matrizes de quaisquer dimensões, caso possuam os mesmos tipos de elementos, que neste exemplo são inteiros, separados por espaços, e cada linha da matriz seja lida com apenas um comando `input()`. Essa abordagem mais compacta permite uma maior legibilidade do código e pode ser útil em casos em que a entrada segue um formato padrão e bem definido.

Questão:

```

1 import numpy as np
2
3 # Definição das variáveis que indicam o número de linhas e colunas da matriz
4 linhas, colunas = 7, 6
5
6 # Inicialização da matriz M como uma lista vazia
7 M = []
8
9 # Loop para preenchimento da matriz M
10 for l in range(linhas):
11     # Inicialização da linha l da matriz M como uma lista vazia
12     linha = []
13
14     # Leitura da linha l da matriz como uma string separada por espaços em branco
15     linhaChar = input().split(' ')
16
17     # Loop para preencher a linha l da matriz M com os valores lidos pelo usuário
18     for c in linhaChar:
19         # Verificação se o caractere não é vazio
20         if c:
21             # Conversão e inserção do valor na linha l da matriz M
22             valor = int(c)
23             linha.append(valor)
24
25     # Adição da linha l à matriz M
26     M.append(linha)
27
28 # Cálculo da soma de todos os elementos da matriz M utilizando o método sum
29 soma = np.sum(M)
30
31 # Impressão do resultado da soma
32 print(f'soma = {soma}')

```

Código 10.4: Exemplo de solução para a questão de matriz do exame no VPL.

No entanto, é importante ressaltar que essa solução pode não ser adequada para cenários em que a entrada varie em formato ou tamanho. Nesses casos, é mais indicado utilizar uma abordagem mais específica para a leitura da entrada. Além disso, é fundamental que o enunciado da questão seja claro e objetivo, permitindo que o estudante compreenda as restrições e especificações do problema. Dessa forma, o estudante poderá desenvolver uma solução adequada e eficiente, considerando todas as competências a serem avaliadas.

Por fim, é importante mencionar que a escolha da linguagem de programação pode impactar o desenvolvimento da solução, e que, portanto, é recomendável que o enunciado seja elaborado para permitir soluções em diferentes linguagens. Assim, os estudantes terão a oportunidade de aplicar seus conhecimentos e habilidades em diferentes contextos.

A segunda questão sobre parcelas sem juros, a ser inserida no arquivo de código `Q2.py` da atividade VPL, pode ser resolvida com o seguinte comando: `print(f"float(input())/12:.2f")`.

Questão:

```
1 import numpy as np
2
3 def lerMatriz():
4     """ler matriz de inteiros"""
5     M, ler_linha = [], input()
6     while ler_linha:
7         M.append([int(i) for i in ler_linha.split(' ') if i])
8         ler_linha = input()
9     return M
10
11 print(f'soma = {np.sum(lerMatriz())}')
```

Código 10.5: Exemplo de solução compacta e genérica para a questão de matriz do exame no VPL.

Este comando lê a entrada fornecida, converte para o formato `float`, divide por 12 parcelas (como definido no enunciado da segunda questão na Figura 10.5) e formata o resultado com duas casas decimais.

10.4 Detalhando o arquivo linker.json

Esta seção detalha o arquivo `linker.json`, sendo amplamente utilizado para trocar dados entre sistemas e fornece uma maneira fácil de armazenar e acessar informações estruturadas. Com o arquivo `linker.json`, o professor pode facilmente importar as variações do exame em um sistema de gerenciamento de exames, como o Moodle com VPL, para criar e administrar o exame.

A versão mais recente, a 13, disponível no [GitHub](#), considera o arquivo `linker.json` criado pela nova versão 5.3 do MCTest. Isso possibilita aproveitar todos os novos recursos ao criar uma questão no MCTest com integração VPL. Destaca-se a adição de uma nova chave no dicionário, chamada `skills`. Essa chave contém um conjunto de habilidades que o estudante deve desenvolver para resolver a questão. Por exemplo, se a questão avalia a competência do estudante em criar laços de repetição utilizando o comando `while`, as adaptações nos arquivos do VPL que devem ser incluídas verificarão se o estudante realmente utilizou essa habilidade. É importante ressaltar que para cada disciplina devem existir conjuntos específicos de habilidades a serem avaliadas e implementadas nestes arquivos VPL, e esses conjuntos devem crescer progressivamente. Essas implementações no VPL estão em processo de construção; no entanto, a estrutura necessária no arquivo JSON já está definida.

Dessa forma, o arquivo `linker.json` se torna uma ferramenta valiosa para avaliar as competências dos estudantes de forma mais precisa e objetiva. Com as adaptações nos arquivos do VPL, é possível verificar se as habilidades necessárias para resolver a questão foram efetivamente utilizadas, garantindo uma avaliação mais coerente para aprendizagem da disciplina.

Na Seção 5.4.3 – Questão com exercício de programação no MCTest+Moodle+VPL, é apresentado o dicionário `moodle_cases` de uma QT com integração VPL para popular o arquivo

`linker.json`. O trecho de código a seguir exemplifica como o arquivo `linker.json` é estruturado.

O arquivo JSON apresentado a seguir consiste em uma variação de questões do exame, identificada pela chave `variant`, e uma lista de questões. Cada questão é identificada pela chave `key`, sendo um valor único do banco de dados associado a cada questão. Além disso, a questão possui um número de identificação, indicado pela chave `number`, que representa a ordem da questão na variante do exame, um arquivo de código relacionado, indicado pela chave `file`, que deve ser utilizado com o mesmo nome e extensão em alguma linguagem, como `Q1.py` para Python, e um peso, indicado pela chave `weight`. Este peso é a dificuldade da questão, definida na Seção 4.1 – [Tutorialis gerais sobre a navegação de questões](#) (ver Figura 4.5). Por exemplo, se um exame tem duas questões VPL, com dificuldades 1 e 2, respectivamente, a primeira questão terá uma nota fracionada de 1/3, enquanto a segunda questão valerá 2/3, sendo que a segunda questão tem o dobro da dificuldade da primeira.

Conteúdo do arquivo `linker.json` até a primeira questão:

```
{  
    "variations": [  
        {  
            "variant": "1",  
            "questions": [  
                {  
                    "key": "2463",  
                    "number": "1",  
                    "file": "Q1",  
                    "weight": "1",  
                    "language": [  
                        "all"  
                    ],  
                    "skills": [],  
                    "answer": [],  
                    "description": [],  
                    ...  
                }  
            ]  
        }  
    ]  
}
```

A questão também pode ter uma lista de linguagens de programação possíveis para a sua resolução, indicada pela chave `language`, uma lista de habilidades que o estudante deve desenvolver para resolvê-la, indicada pela chave `skills`, e uma descrição, indicada pela chave `description`, que pode ser o enunciado da questão a ser apresentado na descrição da questão no Moodle em formato TXT. Por fim, a questão contém uma lista de casos de teste, cada um contendo uma entrada e uma saída esperada, indicados pelas chaves `input` e `output`, respectivamente.

Vale relembrar que a definição do arquivo JSON contou com a valorosa colaboração do Prof. Dr. Paulo Henrique Pisani e seu orientando Heitor Rodrigues Savegnago para a adaptação dos arquivos de configuração do VPL.

Essa estrutura permite uma administração mais eficiente e precisa do exame, possibilitando a criação de diferentes variações de questões com pesos e habilidades diferentes.

Continuação do conteúdo do arquivo linker.json até a primeira questão:

```
{  
...  
    "cases": [  
        {  
            "case": "test_1",  
            "input": [  
                "8 2 8 8 7 3\n0 4 9 3 9 8\n5 1 5 1 6 2\n2 1 0 7 7 1\n5 8 6 6 3 3\n4 7 9 5 9 9\n3 3 7 2 1 4\n"],  
            "output": [  
                "soma = 201"  
            ]  
        },  
        {  
            "case": "test_2",  
            "input": [  
                "7 6 1 8 5 6\n1 6 2 4 2 3\n9 1 1 4 2 5\n1 9 5 8 1 3\n7 1 5 0 1 8\n2 8 2 0 0 2\n6 8 2 4 0 6\n"],  
            "output": [  
                "soma = 162"  
            ]  
        }  
    ],  
    ...
```

A seguir, é apresentada a segunda questão da primeira variação do exame, seguindo a mesma ordem da questão anterior. Como mencionado anteriormente, essa estrutura permite que o arquivo `linker.json` seja facilmente gerenciado e adaptado para diferentes necessidades, possibilitando a criação e administração de exames de forma mais eficiente e precisa. Dessa forma, é possível criar diferentes variações do exame, com questões e pesos diferentes, para avaliar as competências dos estudantes de forma mais abrangente.

Conteúdo do arquivo linker.json da segunda questão:

```
...  
{  
    "key": "2454",  
    "number": "2",  
    "file": "Q2",  
    "weight": "1",  
    "language": [  
        "all"  
    ],  
    "skills": [],  
    "answer": [],  
    "description": [],  
    ...
```

Continuação do conteúdo do arquivo `linker.json` até a segunda questão:

```
{
...
    "cases": [
        {
            "case": "test_1",
            "input": [
                "5009.8\n"
            ],
            "output": [
                "417.48"
            ]
        },
        {
            "case": "test_2",
            "input": [
                "5009.7\n"
            ],
            "output": [
                "417.47"
            ]
        }
    ]
},
...
}
```

10.5 Considerações finais

Neste capítulo, foi apresentado como criar exames integrados ao *plugin* VPL do Moodle, contendo QTs paramétricas nas quais os estudantes devem submeter EP. Esse tipo de avaliação tem sido amplamente utilizado em disciplinas de cursos de graduação e pós-graduação da UFABC, principalmente nas disciplinas introdutórias envolvendo programação.

A integração do VPL com o Moodle permite a correção automática das atividades de programação submetidas pelos estudantes. Para isso, são necessários dois arquivos gerados pelo MCTest: o `linker.json`, contendo as variações do exame e os detalhes de cada questão, principalmente os casos de teste; e o `students_variations.csv`, contendo a variação atribuída a cada estudante. Esses arquivos devem ser inseridos na configuração da atividade VPL para a correção automática ocorrer. Foram detalhados todos os passos necessários para criar uma atividade VPL para um exame, incluindo a configuração dos arquivos citados.

O arquivo `linker.json` é uma ferramenta essencial para transferir dados entre sistemas, contendo de maneira estruturada todas as informações de um exame, como variações de questões, descrições, pesos, habilidades, casos de teste, entre outros. A versão mais atual desse arquivo está sendo validada e em breve será disponibilizada, contendo novas chaves como `skills`, que especifica as habilidades que o estudante deve desenvolver para resolver cada questão. Isso permite uma avaliação

10.5. CONSIDERAÇÕES FINAIS

mais precisa das competências dos estudantes.

Apesar dos benefícios da avaliação por meio de atividades práticas na aprendizagem ativa, muitos docentes ainda não utilizam essa metodologia. A adoção de exames padronizados, como os apresentados neste trabalho, pode auxiliar na uniformização dos conteúdos e habilidades desenvolvidas pelos estudantes, fortalecendo o processo de ensino-aprendizagem.

Conclui-se esta parte sobre a criação e correção de exames de diferentes estilos, incluindo quadros de respostas (QRs), além de QMs e QTs. Estas últimas podem conter EP que são passíveis de correção automática utilizando o *plugin* VPL do Moodle. Espera-se que, até este ponto do livro, o potencial do MCTest para criar e corrigir exames tenha sido evidenciado ao professor. Para comprovar seus benefícios, nos próximos capítulos serão apresentados resumos dos principais resultados publicados até o momento.

Parte IV

Experimentos

Conteúdo

11 MCTest versões 1, 2 e 3 - Quadro de Respostas (QR)	201
12 MCTest versões 4, 4.G e 5	209
13 MCTest+Moodle+VPL	237
14 Conclusões	251
A Acompanhamento de acesso de estudantes no Moodle	255
B Exames Adaptativos na disciplina CS1	275
C Sobre o SEB	301
D Sobre validação de código	305
Bibliografia	319

Capítulo 11

MCTest versões 1, 2 e 3 - Quadro de Respostas (QR)

Conteúdo

11.1 MCTest-1: versão no Matlab	202
11.2 MCTest-2: versão no Android	204
11.3 Considerações finais	206

O objetivo desta parte do livro é relatar os experimentos realizados no MCTest desde a sua primeira versão (MCTest-1), os quais foram publicados como artigos científicos. Como cada nova versão do MCTest incorpora as funcionalidades das versões anteriores, em geral, é possível reproduzir os experimentos realizados nos processos de geração e correção de exames nesta versão web mais recente do MCTest (MCTest-5). Nos próximos capítulos, serão introduzidos alguns dos experimentos apresentados nos artigos, caso ainda não tenham sido abordados anteriormente. Esses exemplos serão salvos no banco de dados SQL relacionado, que estará disponível em github.com/fzampirolli/mctest, arquivo `book/1ed-br/mctestLivro.sql`, juntamente com o conteúdo deste livro, com o intuito de enriquecer o texto e torná-lo mais interessante.

Antes de iniciar este relato de experiências, é importante mencionar que todas as publicações realizadas não teriam sido possíveis sem a participação fundamental dos colaboradores. Além disso, gostaria de destacar o apoio e contribuição dos professores da Universidade Federal do ABC (UFABC): Guiou Kobayashi, José Artur Quilici-Gonzalez e Rogério Perino de Oliveira Neves. Eles participaram do processo seletivo da Especialização em Tecnologias e Sistemas da Informação (TSI) da UFABC e colaboraram nas primeiras versões do MCTest desde 2012. Agradeço sua valiosa participação e contribuição no desenvolvimento deste projeto.

O artigo de [Kobayashi, Zampirolli e Quilici-Gonzalez \(2016\)](#) detalha as três primeiras edições do TSI na modalidade de Ensino a Distância (EaD), oferecido pela UFABC, as quais foram lançadas

nos anos de 2010, 2012 e 2014. O objetivo do curso é suprir a necessidade por profissionais qualificados em Tecnologia da Informação no estado de São Paulo, abrangendo localidades distantes até 200 km. O curso foi ministrado por meio de uma plataforma web. Os experimentos realizados focaram em superar desafios relacionados ao EaD, como a escalabilidade de professores, o isolamento e a autenticação dos estudantes. O projeto foi financiado pelo programa UAB (Universidade Aberta do Brasil), vinculado a CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) do governo brasileiro, permitindo que o curso fosse oferecido gratuitamente. A avaliação positiva dos cursos demonstra que uma universidade pública pode complementar a formação profissional de centenas de estudantes por meio de cursos de EaD de alta qualidade. Os resultados dos experimentos forneceram referências importantes que podem ser utilizadas em projetos futuros de EaD.

Neste primeiro capítulo de experimentos, será apresentado um resumo dos artigos que descrevem as experiências de criação de exames com QM estáticas, utilizando editores de texto como o Word ou BOffice.

11.1 MCTest-1: versão no Matlab

A primeira versão do MCTest (MCTest-1) foi utilizada no processo seletivo do TSI entre os anos de 2012 e 2017. Durante esse período, o exame era realizado no Word ou no BOffice, e a automatização era aplicada apenas na correção do exame. Para esta versão, foi publicado o seguinte artigo.

Artigo descrevendo a parte de visão computacional do MCTest-1

O artigo de [Zampirolli, Quilici-Gonzalez e Neves \(2013\)](#) descreve o processo de transformação de imagens capturadas dos QRs dos exames, como ilustrado na Figura 11.1, em matrizes binárias reduzidas contendo as respostas às questões, juntamente com alguns elementos de controle na última coluna para as variações do exame e na última linha para atribuir pesos diferentes às questões.

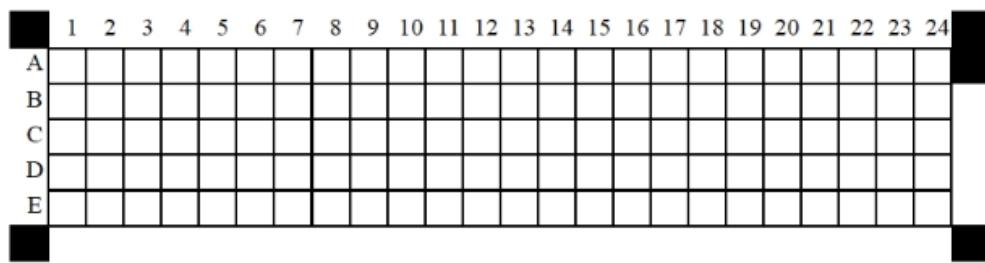


Figura 11.1: Recorte do PDF com o QR utilizado no MCTest-1.

Método

Como o artigo foi apresentado no Workshop de Visão Computacional, o método destaca o Processamento Digital de Imagens para converter uma imagem capturada pela câmera do com-

putador em uma matriz reduzida contendo as respostas marcadas e alguns elementos de controle adicionais.

O método foi aplicado ao problema real da correção automática de exames com QMs. Inicialmente, o usuário posiciona o QR em frente à câmera do computador para salvar a imagem no disco. Em seguida, a imagem é analisada para extrair o gabarito das variações do exame. Posteriormente, os QRs dos estudantes podem ser lidos usando o mesmo processo, e a pontuação do exame é exibida na tela e/ou salva em um arquivo.

Experimentos

Neste artigo, foi utilizada uma base de dados composta por 674 exames preenchidos por candidatos ao TSI, na oferta de 2012. Cada exame foi impresso em uma folha de papel A4 e continha 24 QMs, com cinco alternativas cada, sendo apenas uma a resposta correta. Foram preparados oito conjuntos diferentes de exames, identificados pela última coluna da Figura 11.1, cada um com variações nas questões e respostas, como detalhado na Figura 11.3.

O processamento dos exames foi conduzido em um MacBook Pro equipado com um processador Intel Core 2 Duo de 2,26 GHz, 2 GB de RAM e uma placa de vídeo Nvidia GeForce 9400M, que dispõe de 256 MB de memória compartilhada. Essa aquisição foi realizada por meio do projeto FAPESP [2009/14430-1](#). As capturas dos QRs preenchidos e das variações foram realizadas utilizando a câmera embutida de 2 megapixels. O processamento das imagens foi realizado por meio do MATLAB versão 2011. O processamento das imagens dos exames seguiu os seguintes passos:

1. Leitura das variações para cada conjunto de exames;
2. Salvamento de uma imagem no formato TIFF contendo apenas a área do QR da variação, conforme ilustrado na Figura 11.1;
3. Salvamento de uma matriz bidimensional contendo as respostas corretas para cada tipo de exame;
4. Leitura dos QRs dos exames dos estudantes;
5. Salvamento da imagem de cada exame em um arquivo TIFF separado;
6. Adição dos resultados processados dos exames a um arquivo CSV.

Os resultados dos experimentos demonstraram que o método proposto consegue corrigir automaticamente exames com QMs com alta precisão. A média de erros foi de apenas 0,2%, representando uma precisão significativamente maior em comparação aos métodos tradicionais de correção, como a correção manual e a correção por meio de softwares comerciais testados na época.

Apesar das instruções do exame, que deixavam claro que os quadrados deveriam ser preenchidos completamente com tinta para garantir uma identificação correta, foram identificadas respostas inválidas em 6,7% dos 674 exames (42 exames). Essas respostas foram prontamente identificadas visualmente pelo operador e a pontuação foi ajustada conforme necessário.

11.2 MCTest-2: versão no Android

Artigo descrevendo o MCTest-2 para dispositivos móveis

O MCTest, Versão 2 (MCTest-2), é um aplicativo desenvolvido para dispositivos móveis, especificamente para Android, que realiza a correção automática de exames com QMs. Essa versão foi detalhado do artigo de [China, Zampirolli, Neves e Quilici-Gonzalez \(2016\)](#) e é uma adaptação da primeira versão desenvolvida para computador pessoal (PC) utilizando o MATLAB. Nesta versão MCTest-2, as técnicas de processamento digital de imagens foram otimizadas para uso em dispositivos móveis, e a biblioteca [OpenCV](#) foi empregada em substituição à “*Toolbox de Processamento de Imagem*” do MATLAB.

O estudante de Iniciação Científica Rodrigo China foi responsável pelo desenvolvimento do MCTest-2, e os detalhes sobre a parte de Processamento Digital de Imagens foram publicados no Workshop de Visão Computacional ([ZAMPIROLI, 2015](#)). Além disso, o trabalho também foi apresentado como pôster em um evento de Iniciação Científica ([CHINA; ZAMPIROLI, 2014](#)).

Método

O artigo de [China, Zampirolli, Neves e Quilici-Gonzalez \(2016\)](#) descreve o MCTest-2 desenvolvido em Java e pode ser utilizado em dispositivos com câmera embutida e sistema operacional Android, Versão 4.0 ou superior. No entanto, a resolução da câmera pode limitar o número máximo de questões e respostas no papel do exame.

O processo de utilização do aplicativo consiste em três etapas. Na primeira etapa, é possível criar um novo projeto ou carregar um projeto previamente salvo. O projeto é armazenado em uma subpasta com o nome correspondente. Na segunda etapa, os gabaritos com as respostas corretas são salvos. Esses gabaritos podem ser criados utilizando qualquer software de edição de texto. Por fim, na terceira etapa, é feita a comparação entre os exames dos estudantes e os gabaritos correspondentes. A Figura 11.2 demonstra o aplicativo utilizado em um gabarito impresso.

A Figura 11.3 exibe o QR. Esse QR representa um exame com QMs com 10 questões e 5 possíveis respostas para cada questão. Uma linha e uma coluna adicionais são incluídas para armazenar informações sobre a variação de exame e a avaliação ponderada. Para identificar a variação do exame, é utilizada a coluna da direita, com os bits 0-3 indicando o número da variação em formato binário. Isso permite a criação de gabaritos de exame diferentes, com questões e respostas embaralhadas manualmente para evitar trapaças. O número de variações de exames possíveis é determinado pelo número de alternativas de resposta, e a avaliação ponderada é permitida na última linha da tabela.

O QR processado e apresentado no formato CSV a seguir, gerado pelo aplicativo, contém informações sobre as respostas de dois exames. A primeira linha mostra o número de questões, respostas e quantos QRs foram processados. Em seguida, os cabeçalhos das colunas, que incluem número, ID, tipo/variação, quantidade de respostas inválidas, quantidade de respostas erradas, quantidade de respostas corretas e pontuação final. A partir da segunda linha, cada linha corresponde a

11.2. MCTEST-2: VERSÃO NO ANDROID

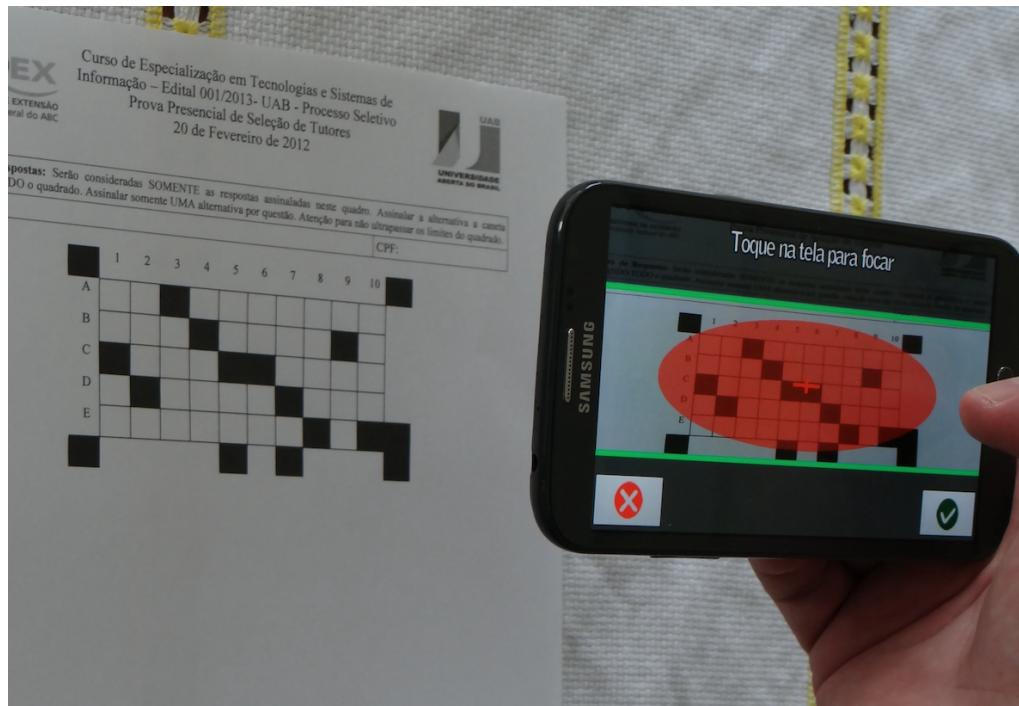


Figura 11.2: Demonstração da captura a partir de um gabarito impresso.

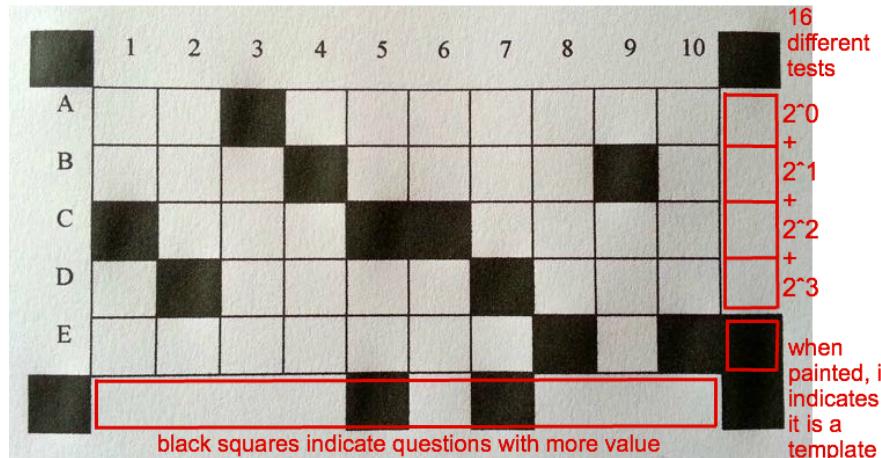


Figura 11.3: As questões marcadas na parte inferior indicam um valor ponderado atribuído a questão, as marcas à direita representam diferentes gabaritos e a identificação do gabarito é a marca à direita e abaixo. Fonte: [China, Zampirolli, Neves e Quilici-Gonzalez \(2016\)](#).

um participante do exame. As colunas seguintes representam as respostas para cada uma das dez questões do exame.

Além disso, o aplicativo também oferece a opção de visualizar estatísticas por turma, conforme ilustrado na Figura 11.4. Esse aplicativo possui várias telas apresentadas e detalhadas no trabalho de [China, Zampirolli, Neves e Quilici-Gonzalez \(2016\)](#).

Conteúdo do arquivo CSV com estatísticas e as marcações realizadas.

10	5	2	1																
Numero	ID	Tipo	Invalidas	Erradas	Corretas	Final	Ques:	1	2	3	4	5	6	7	8	9	10		
1		0	0	1	2	8	Ans: C	D	A	B	C	C	D	E	O	C			
2		0	0	0	2	8	Ans: C	D	A	B	C	C	D	E	A	C			

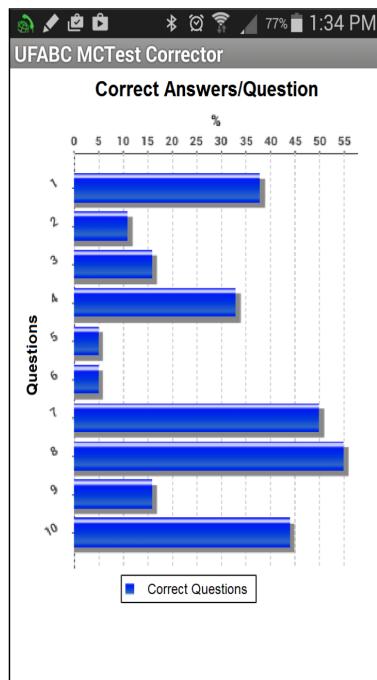


Figura 11.4: Estatísticas sobre o número de respostas corretas em uma turma. Fonte: [China, Zampirolli, Neves e Quilici-Gonzalez \(2016\)](#).

Experimentos

O aplicativo foi testado em dois cenários reais, envolvendo 41 e 64 estudantes, respectivamente. Em ambos os casos, o MCTest apresentou uma precisão de 100% quando os estudantes preencheram corretamente os QRs. O aplicativo considera os blocos preenchidos se eles contiverem pelo menos 50% da área pintada. É importante incluir instruções claras no exame, explicando como preencher completamente o QR usando caneta preta ou lápis, a fim de evitar pontuações mais baixas.

O tempo de processamento para cada exame varia conforme o dispositivo utilizado. Contudo, mesmo em dispositivos mais acessíveis, o tempo de processamento é mais rápido do que o observado na versão anterior no computador pessoal (PC), no caso do MCTest-1.

11.3 Considerações finais

Nas diferentes versões do MCTest, existem duas abordagens para realizar os experimentos descritos. A primeira abordagem envolve fornecer as questões separadamente, como demonstrado

11.3. CONSIDERAÇÕES FINAIS

no Capítulo 7 - [Exames com Quadro de Respostas \(QR\)](#), como utilizado na Escola Preparatória da UFABC. A segunda abordagem mais recente é incluir as questões no banco de questões do MCTest, conforme descrito no Capítulo 8 - [Exames com QR+QM e/ou QT](#). Além disso, nas versões mais recentes, não é mais necessário posicionar o QR diante da câmera do computador ou celular, como nas versões 1 e 2. Agora, é possível digitalizar todos os exames em um arquivo PDF, tornando o processo menos repetitivo e permitindo a correção de exames com centenas de estudantes.

A Versão 3 do MCTest (MCTest-3) foi desenvolvida em Python, utilizando o mesmo QR das versões anteriores. Nessa versão, as questões eram fornecidas separadamente e a biblioteca OpenCV era utilizada para o processamento digital de imagens. Os exames eram digitalizados em um arquivo PDF para a correção automática.

Por fim, todas as versões apresentadas neste capítulo foram registradas no INPI, conforme mencionado na Seção 1.2.2 - [Versões do MCTest](#).

A Versão 4 do MCTest também foi desenvolvida em Python, mas agora utiliza QRs criados no L^AT_EX. Além disso, as questões são geradas utilizando uma versão diferente, a 4.G (de *generate*), também com o L^AT_EX. Os experimentos realizados com essas versões serão apresentados no próximo capítulo.

Capítulo 12

MCTest versões 4, 4.G e 5

Conteúdo

12.1 MCTest-4: gerando e corrigindo questões criadas no L ^A T _E X	210
12.1.1 Artigo descrevendo as versões do MCTest 4 e 4.G	210
12.1.2 Artigo comparando as modalidades semipresencial e presencial CS1 com base na utilidade do MCTest-4	212
12.1.3 Artigo definindo a parametrização do MCTest após a versão 4	214
12.2 MCTest-5: a versão web	218
12.2.1 Artigo apresentando a primeira versão web do MCTest	218
12.2.2 Artigo descrevendo o MCTest em conjunto com <i>Google Forms</i> e <i>Google Sheets</i>	227
12.2.3 Artigo descrevendo a adoção de QMs com pesos diferentes nas alternativas	231
12.3 Considerações finais	234

Neste capítulo, são apresentadas as versões adicionais do MCTest, nomeadamente o MCTest-4, MCTest-4.G e MCTest-5.

O MCTest-4 é um software desenvolvido para a **correção** de exames com QMs. Ele lê um arquivo PDF contendo os QRs dos estudantes, identifica automaticamente cada estudante utilizando código de barras e corrige exames individuais com questões e alternativas aleatórias. O programa realiza a correção dos exames comparando as respostas marcadas com o gabarito e gera um arquivo CSV contendo a correção de cada questão, bem como o número total de questões corretas e inválidas.

O MCTest-4.G (de *generate*) é um software desenvolvido para a **geração** de exames com questões. Ele lê uma estrutura de pastas e arquivos contendo dados de estudantes e matrícula, além de um banco de QMs ou QTs. O MCTest-4.G utiliza esses dados para gerar exames distintos e aleatórios para cada estudante. Nas folhas de exame, além do nome e matrícula do estudante, há

um código de barras referente à sua matrícula.

Além disso, neste capítulo, são apresentados os experimentos conduzidos na versão mais recente, o MCTest-5, que agora é baseado na web e usa o banco de dados MySQL. No entanto, a integração com as atividades VPL do Moodle será tratada em um capítulo distinto, devido à sua complexidade e à extensa literatura existente sobre o assunto.

12.1 MCTest-4: gerando e corrigindo questões criadas no L^AT_EX

Tanto o gerador de exames (MCTest-4.G) quanto o corretor (MCTest-4) operam de maneira independente. No entanto, para assegurar a precisão na correção dos exames, é importante utilizar os exames gerados pelo MCTest-4.G. Em caso contrário, a única alternativa viável será considerar uma única variação do exame para todos os estudantes, e a identificação individual de cada estudante se baseará na sequência em que eles aparecem no documento PDF. Dessa forma, a correção terá de ser realizada manualmente, como exemplo, por meio do uso de uma planilha eletrônica, em que as respostas dos estudantes no CSV gerado serão comparadas com um gabarito, assemelhando-se ao procedimento nas versões anteriores do MCTest.

A partir desta seção, será considerada apenas a versão MCTest-4, incorporando as funcionalidades das duas versões.

12.1.1 Artigo descrevendo as versões do MCTest 4 e 4.G

A versão 4 do MCTest (MCTest-4) representou um avanço significativo em relação às versões anteriores, introduzindo melhorias na correção dos exames. No artigo de [Zampirolli, Batista e Quilici-Gonzalez \(2016\)](#), é apresentada uma solução inovadora para simplificar a geração e correção de QMs, destacando a importância da confiabilidade dos resultados.

Método

No MCTest-4, o programa gera um arquivo L^AT_EX e seu respectivo arquivo PDF compilado, contendo o exame individual de cada estudante. O exame consiste em uma página inicial seguida por uma lista de QMs e, opcionalmente, QTs. A página inicial é projetada para a marcação das respostas das QMs e inclui um cabeçalho. É possível gerar exames individuais com respostas únicas para cada estudante. Todas essas opções são configuráveis em um arquivo de configuração separado.

A pasta `MCTest4-master`, mostrada na Figura 12.1, contém cinco subpastas, incluindo a pasta `courses`, destinada a armazenar diferentes turmas de uma mesma disciplina ou de disciplinas diferentes. Por exemplo, se você estiver lecionando Programação Orientada a Objetos (POO) e Engenharia de Software (ES) em um semestre de 2016, poderá criar as pastas `course1` e `course2` dentro de `courses` para as turmas de POO e ES, respectivamente. Cada pasta conterá listas de estudantes e seus respectivos IDs em arquivos CSV, como `16_POO_class1.csv` e `16_ES_class2.csv`, seguindo a formatação definida na Seção 3.3.1 – Criar turma com arquivo CSV. As questões são armazenadas na pasta `questions` em arquivos TXT, seguindo a mesma estrutura apresentada na Seção 4.1 – Tutoriais gerais sobre a navegação de questões. A geração de um exame para uma turma

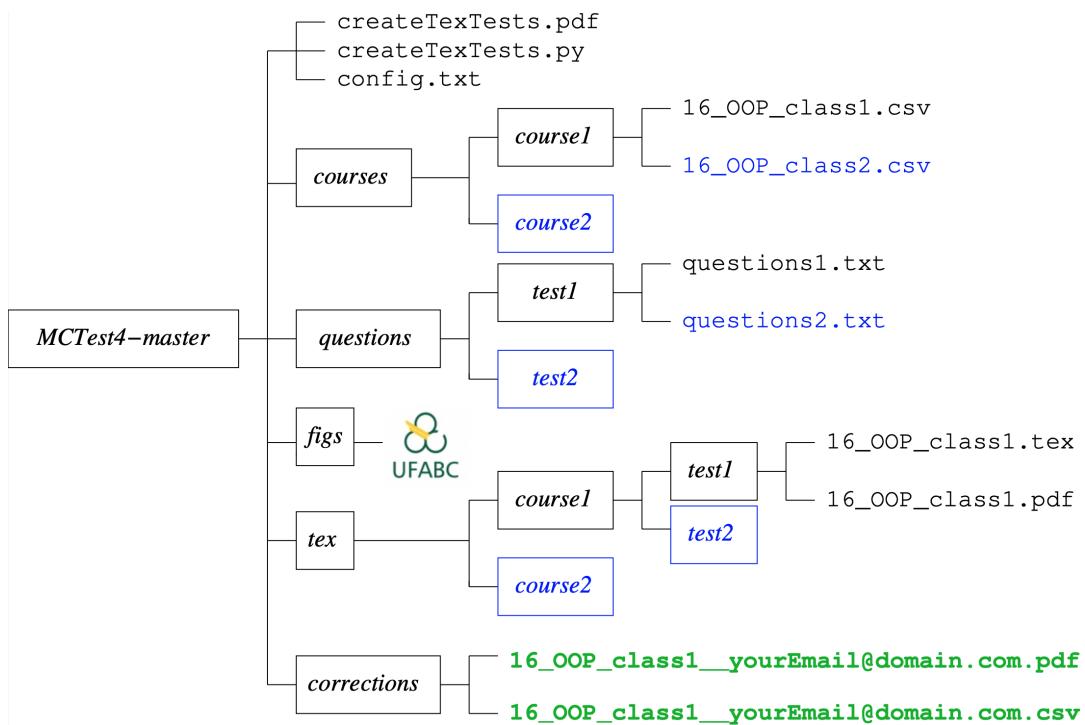


Figura 12.1: Estrutura de pastas do MCTest-4. Fonte: [Zampirolli, Batista e Quilici-Gonzalez \(2016\)](#).

específica requer a configuração no arquivo `config.txt`. Para mais detalhes sobre essa estrutura de pastas e arquivos, consulte o artigo de [Zampirolli, Batista e Quilici-Gonzalez \(2016\)](#).

Os estudantes preenchem os QRs na página inicial do exame, que já contém seus nomes e números de identificação em um código de barras, no caso de exames individuais. Após a conclusão do exame, todas as páginas com os QRs podem ser digitalizadas em um único arquivo PDF, que o software utiliza para realizar a correção automática. As notas finais são armazenadas em um arquivo CSV que contém o ID de cada estudante, seguindo o mesmo estilo apresentado na Seção 7.3.1 - Arquivo CSV com as correções.

O programa `createTexTests.py`, apresentado na Figura 12.1, é responsável por gerar um arquivo GAB contendo os gabaritos das respostas individuais de cada estudante, caso sejam utilizados exames aleatórios. O corretor, por sua vez, lê um arquivo PDF contendo as páginas digitalizadas dos exames e utiliza o arquivo GAB para calcular as notas. As respostas dos estudantes são identificadas através da análise das marcações nas questões. Caso ocorram erros de leitura do código de barras ou das marcações, o corretor sinaliza esses problemas nas notas. O arquivo de notas é gerado em formato CSV.

O MCTest-4 foi elaborado utilizando a linguagem de programação Python e se encontra disponível gratuitamente em github.com/fzampirolli/MCTest4. Professores podem fazer o *download* do código em Python, acompanhado da estrutura de pastas e exemplares de arquivos, para gerar os exames.

Para utilizar o programa, é necessário que o professor instale o Python e as bibliotecas necessárias, como a OpenCV. Uma alternativa é enviar os arquivos GAB e PDF digitalizados para o

servidor do MCTest-4 por meio do FTP. Neste servidor, há um *script* que detecta automaticamente esses arquivos e os processa. O arquivo de notas resultante pode ser enviado por e-mail ou baixado do servidor. Todo esse processo é detalhado no artigo de [Zampirolli, Batista e Quilici-Gonzalez \(2016\)](#).

Experimentos

Segundo os autores do artigo, são descritos três experimentos realizados com o MCTest-4. O primeiro experimento foi realizado com uma turma de 66 estudantes da disciplina de Programação Orientada a Objetos. Os exames foram gerados com 10 QMs e uma QT. Os resultados foram satisfatórios, com apenas um erro de leitura. O segundo experimento foi realizado com uma turma de 130 estudantes da disciplina de Processamento de Informação. Os exames foram gerados com 12 QMs e uma QT. Os resultados também foram satisfatórios, sem nenhum erro de leitura. O terceiro experimento foi realizado com 6772 estudantes de escolas públicas do Brasil. Os exames foram gerados com 11 QMs para os estudantes do ensino médio e 13 questões para os estudantes do ensino fundamental. Os resultados foram satisfatórios, com uma taxa de erros de leitura de 1,88%.

No terceiro experimento, foi utilizado um servidor adquirido com recursos da FAPESP, processo [2009/14430-1](#), com um microprocessador, Intel (R) Core (TM) i7, CPU X980@3.33GHz com 12GB de RAM, sistema operacional Linux Ubuntu 64 bits versão 15.04 e executáveis gerados por Python 2.7.9. Um *script* do MCTest-4 é executado a cada minuto e procura por arquivos com extensão PDF (sem arquivos GAB), para os quais os arquivos CSV correspondentes ainda não foram criados. O corretor automático levou 102 minutos para processar os 6772 exames, ou seja, 0,9 segundos por exame.

Os resultados dos experimentos indicam que o MCTest-4 é uma ferramenta eficaz para a avaliação de estudantes. O programa consegue gerar exames com questões de qualidade e de corrigilos de forma rápida e precisa. Além disso, o programa é fácil de usar e pode ser personalizado para atender às necessidades específicas de cada professor.

12.1.2 Artigo comparando as modalidades semipresencial e presencial CS1 com base na utilidade do MCTest-4

O artigo de [Zampirolli, Goya, Pimentel e Kobayashi \(2018\)](#), publicado na revista *Computer Applications in Engineering Education*, descreve um processo de avaliação desenvolvido para a disciplina de Processamento da Informação (ou Introdução à Programação, IP, ou também CS1) oferecido pela Universidade Federal do ABC. O estudo compara a modalidade presencial (IP-FF – *face-to-face*) e a modalidade semipresencial *blended learning* (IP-BL), na qual apenas as avaliações são realizadas presencialmente.

Método

O processo de avaliação foi desenvolvido a partir da identificação de problemas no processo de avaliação de IP-FF. Os problemas identificados foram: Avaliação de critérios diferentes entre

professores; falta de uniformidade na formatação dos exames; dificuldade de calcular a nota final do estudante.

Para resolver esses problemas, o processo de avaliação proposto utiliza: um banco de questões com diferentes níveis de dificuldade; um sistema para gerar exames únicos para cada estudante; um sistema para corrigir os exames de forma semi-automática.

Experimentos

Os experimentos foram conduzidos em 10 turmas da disciplina IP-BL e demonstraram que o processo de avaliação conseguiu padronizar os critérios de avaliação, aprimorar a formatação dos exames e simplificar o cálculo da nota final. Os exames foram elaborados utilizando adaptações no software MCTest-4.G, contendo três QTs em cada exame. Para aumentar a diversidade, as questões foram replicadas manualmente com modificações em alguns parâmetros.

Foram criados *scripts* em Python para auxiliar nas avaliações da disciplina. Essas ferramentas automatizadas incluem a geração automática de exames individuais, a organização dos exames digitalizados por questão e estudante, a compilação do código submetido pelos estudantes e o envio automático de e-mails com as notas e os erros apontados pelos professores. Todos esses *scripts* foram adaptados e incorporados na versão web mais recente, MCTest-5.

A primeira avaliação da disciplina foi realizada em formato impresso, na qual os estudantes tinham que resolver três EPs. Uma das questões envolvia o uso de teste de mesa, conforme ilustrado na Figura 4.14. Um exemplo de exame utilizado para essa primeira avaliação, em que cada questão ocupava uma folha frente e verso, é mostrado na Figura 12.2. As resoluções dos estudantes eram digitalizadas e enviadas aos professores, sendo que cada professor corrigia uma única questão de todos os estudantes da disciplina na modalidade IP-BL.

Na segunda avaliação, os estudantes também tinham que resolver três EPs, mas desta vez elas foram enviadas por meio da plataforma AVA (Ambiente Virtual de Aprendizagem) tidia4.ufabc.edu.br. Todos os códigos submetidos pelos estudantes foram salvos em um computador e executados automaticamente para verificar possíveis erros. Os professores deveriam atribuir um conceito a cada arquivo de código.

Essas correções unificadas ocorreram nos anos de 2016.3, 2017.1 e 2017.2, com um total de 235, 185 e 161 estudantes, respectivamente. A UFABC utiliza três períodos letivos por ano.

Esses arquivos de códigos dos estudantes foram também utilizados como *dataset* para o artigo de [Souza, Zampirolli e Kobayashi \(2019\)](#), que apresenta um método para validar as notas atribuídas por professores aos EPs de estudantes em IP-BL. Foi utilizado um modelo de rede neural convolucional treinado com códigos fonte de estudantes avaliados por diferentes professores. Os códigos foram pré-processados e convertidos em representações *embedding*, utilizadas como entrada para a rede neural. O modelo obteve uma precisão média de 74,9% na classificação das notas dos exercícios. Essa abordagem pode ajudar a detectar possíveis erros no processo de avaliação realizado pelos professores.

Os resultados mostraram a redução da dispersão nas notas após a implementação do processo de avaliação unificado. No entanto, ainda é preciso enfrentar desafios como a resistência dos

	Universidade Federal do ABC
Disciplina: Processamento da Informação - BC0505 - Primeira Prova	
Professor(es): _____	
Quadrimestre: 2/2018	
Aluno: _____	Modalidade: Semipresencial
	Data: 21/03/2018
RA: _____	Turma: SA-S204-0
Ass: _____ Instruções: <ul style="list-style-type: none"> • Não é permitido o uso de qualquer dispositivo eletrônico, incluindo celular (DESLIGUE-O) e nem fones de ouvido. • Nas questões 2 e 3 você poderá escolher entre Portugol Studio ou JAVA para apresentar a solução. • Nas questões 2 e 3, por clareza, use LETRAS DE FORMA para escrever o algoritmo. • Nas questões 2 e 3, caso o código, de alguma maneira, esteja de difícil leitura, o mesmo não será corrigido. Por exemplo: lápis muito claro, rasuras em cima do código correto, letra de difícil entendimento, etc. Sugere-se fazer rascunho à lápis na página de enunciado de exercício, e escrever o código claro e completo na página em branco do verso da questão, à CANETA. • A organização do código (indentação-recuo) será parte integrante da avaliação, portanto deixe o seu código indentado e mais claro possível - recomenda-se uso de linhas de comentários. 	
	

Questão 1: (PESO 2.5) Simule a execução do **ALGORITMO** abaixo realizando um **TESTE DE MESA** e faça o que se pede nas alternativas (a) e (b) a seguir:

(a) Considere como entrada de dados **a=16, b=21 e x=2**. Anote, na tabela **TESTE DE MESA**, todas as linhas que modifiquem **um dos valores contidos nas variáveis a, b, c, x e d** até que o algoritmo seja encerrado. Simultaneamente, anote na tabela **SAÍDA DE DADOS** todas as saídas (comando escreva) do programa e as linhas que fazem a saída.

ALGORITMO		TESTE DE MESA				
LINHA		a	b	c	x	d
1	programa { a = 16 b = 21 c = 2 x = 2 d = 0 while (a > 0) { d = d + 1 a = a - 1 if (a == 0) { b = b - 1 if (b == 0) { c = c + 1 if (c == 0) { x = x + 1 if (x == 0) { d = d + 1 break } } } } }					

Figura 12.2: Recorte de um exame gerado com MCTest-4.G.

professores em adotar critérios unificados e a incorporação de ferramentas de correção automática no AVA. O artigo ressalta a importância de atividades de motivação e acompanhamento dos estudantes para melhorar os índices de aprovação na disciplina.

12.1.3 Artigo definindo a parametrização do MCTest após a versão 4

Todo o conteúdo apresentado no Capítulo 5 – Questões paramétricas, exceto a seção referente ao VPL, foi originalmente publicado no artigo de Zampirolli, Teubl e Batista (2019a). Em resumo, para criar questões paramétricas no MCTest-4, são necessários dois delimitadores especiais: `[[code:...]]` e `[[def:...]]`. O primeiro deve ser colocado no enunciado da questão ou nas alternativas e define instruções em Python ou variáveis a serem usadas na geração do exame de cada estudante. Já o segundo deve ser definido no final do enunciado e deve conter os possíveis valores para as variáveis aplicadas anteriormente.

Esse artigo apresenta duas questões paramétricas utilizando arquivos TXT, conforme apresentado na Seção 4.1 - Tutoriais gerais sobre a navegação de questões, reproduzidas a seguir. Lembrando que a contribuição desse artigo criou questões paramétricas e na época da escrita do artigo, a versão web ainda não estava disponível. A seguir, são apresentadas essas questões, porém, na versão web do MCTest.

Movimento Retilíneo Uniforme com a biblioteca symbols

O Código 12.1 implementa um algoritmo para calcular a variável tempo (`t`) em um problema de Movimento Retilíneo Uniforme (MRU), dado a posição inicial (`s0`), a velocidade (`v`) e a posição final (`s`) do corpo. O método `algorithm(a)` recebe como entrada uma lista contendo três valores inteiros: a posição inicial (`s0`), a velocidade (`v`) e a posição final (`s`), nessa ordem. Na primeira linha

do método, são importados os módulos `symbols` e `solve` da biblioteca SymPy (docs.sympy.org), que permite criar símbolos matemáticos para serem usados em cálculos simbólicos e também resolver uma equação, respectivamente. São definidos os símbolos `s` e `t` utilizando o método `symbols`, e a posição `s` é calculada como a soma da posição inicial `s0` com a velocidade `v` multiplicada pelo tempo `t`. Por fim, o método `solve` é utilizado para resolver a equação `s-a2=0` em relação à variável `t`. O resultado é armazenado na variável `r` como um número decimal (do tipo `float`) e retornado pelo método `algorithm(a)` como o tempo necessário para o objeto percorrer a distância entre as posições `s0` e `s` com velocidade `v` em MRU. Um exemplo de saída desta questão pode ser visualizado na Figura 12.3.

Questão:

```
1 Um automóvel percorre uma estrada com função horária \(\ s=[[code:a0]] +  
2 [[code:a1]]t \), onde \(\(s\)\) é dado em quilômetros e \(\(t\)\) em horas.  
3 O automóvel passa pelo km [[code:a2]] após:  
4  
5 [[def:  
6 def generate_numbers():  
7     """ Gera três números aleatórios e retorna-os como uma lista. """  
8     import random  
9     global a0, a1, a2 # necessário declarar como global para utilizar no enunciado  
10    a0 = random.randrange(-6, 12, 1) # s0 - posição inicial  
11    a1 = random.randrange(3, 8, 1)    # v - velocidade  
12    a2 = random.randrange(3, 8, 1)    # s - posição final  
13    return [a0, a1, a2]  
14  
15 def algorithm(a):  
16     """ Executa um algoritmo matemático utilizando os valores passados como  
17     parâmetro. Retorna o resultado do algoritmo. """  
18     from sympy import symbols, solve  
19     a0, a1, a2 = a # s0, v, s  
20     s, t = symbols('s,t')  
21     s = a0 + a1 * t  
22  
23     r = float(solve(s - a2, t)[0])  
24     return r  
25  
26 # Gera os números aleatórios  
27 numbers = generate_numbers()  
28  
29 # Executa o algoritmo utilizando os números gerados  
30 correctAnswer = algorithm(numbers)  
31 # [[code:f"{correctAnswer:.2f}"]]  
32  
33 # Opções de respostas erradas com base no resultado do algoritmo  
34 # [[code:f"{correctAnswer+1.2:.2f}"]]  
35 # [[code:f"{correctAnswer+4.8:.2f}"]]  
36 # [[code:f"{correctAnswer-0.9:.2f}"]]  
37 ]]
```

Código 12.1: Exemplo de QM paramétrica para calcular MRU.

MCTest

Topic: Topico Exemplo
Group:
Short Description: mru
Type: QM
Difficulty: 1
Bloom taxonomy: remember
Last update: 2023-07-18
Who created: fzampirolli@ufabc.edu.br
Parametric: YES

#2467 1. Um automóvel percorre uma estrada com função horária $s = -1 + 5t$, onde s é dado em quilômetros e t em horas. O automóvel passa pelo km 3 após:

- A. 5.60 B. 0.80 C. 2.00 D. -0.10

Figura 12.3: Recorte do PDF gerado para a questão de MRU do Código 12.1.

Atenção:

1. Toda variável declarada em um método e utilizada no enunciado da questão deve ser definida como global, como pode ser visto na linha 9 do Código 12.1;
2. As importações de bibliotecas também devem ser realizadas nos métodos, como podem ser vistos nas linhas 8 e 18.

Criação de Matriz

A questão a seguir é uma adaptação daquela apresentada na Seção 4.1 – Tutoriais gerais sobre a navegação de questões, porém, agora é parametrizada com as dimensões da matriz e os elementos atribuídos. O Código 12.2 é responsável por criar uma matriz M com dimensões [[code:Linhas]] \times [[code:Colunas]]. Cada elemento da matriz é calculado utilizando a fórmula $(i, j) = (((i + 1) \times [[code:a2]]) + ((j + 1) \times [[code:a3]])) \bmod 100$, em que i representa as linhas e j representa as colunas da matriz. Na prática, o código gera aleatoriamente o número de linhas e colunas da matriz, num intervalo específico, assim como os valores de $a2$ e $a3$, escolhidos aleatoriamente entre as opções [7, 13, 19] e [11, 17, 23], respectivamente. A matriz é inicializada com zeros e, em seguida, preenchida utilizando dois laços aninhados. Cada elemento $M[i, j]$ recebe o resultado do cálculo mencionado anteriormente. Por fim, a variável global `correctAnswer` é definida como a soma de todos os elementos da matriz M . Um exemplo de saída desta questão pode ser visualizado na Figura 12.4.

Experimentos

O método proposto para gerar questões paramétricas foi aplicado na disciplina de Processamento da Informação (ou Introdução à Programação, IP, ou também CS1) no segundo trimestre de 2018, com um total de 167 estudantes matriculados. A disciplina foi oferecida em um formato semi-presencial, com aulas presenciais ocorrendo em quatro momentos ao longo do semestre: na abertura, no primeiro exame, no projeto e no segundo exame.

Questão:

```

1 Criar uma matriz \([[[code:Linhas]] \times [[code:Colunas]]]\) com elementos
2 \((i,j) = (((i+1) * [[code:a2]]) + ((j+1) * [[code:a3]])) \% 100\).
3 Calcular a soma dos elementos desta matriz. Os índices \((i)\) são as linhas
4 e \((j)\) as colunas iniciadas com o valor \((0)\).
5
6 [[def:
7 import numpy as np, random
8
9 # Gerar número aleatório de linhas e colunas
10 Linhas = np.random.randint(4, 8)
11 Colunas = np.random.randint(4, 8)
12
13 # Escolher valores aleatórios para a2 e a3
14 a2 = random.choice([7, 13, 19])
15 a3 = random.choice([11, 17, 23])
16
17 # Inicializar matriz com zeros
18 M = np.zeros((Linhas, Colunas), dtype=int)
19
20 # Preencher a matriz com os cálculos
21 for i in range(Linhas):
22     for j in range(Colunas):
23         M[i, j] = (((i+1) * a2) + ((j+1) * a3)) % 100
24
25 # Calcular a soma dos elementos da matriz
26 correctAnswer = np.sum(M)
27
28 # Incluir nas alternativas da questão:
29 # [[code:correctAnswer]] % primeira alternativa (sempre) correta
30 # [[code:createWrongAnswers([3,10])]] % cria 3 alter. erradas +/- 10
31 ]]

```

Código 12.2: Exemplo de QM paramétrica para criar uma matriz.

MCTest

Topic: DE-matriz
Group:
Short Description: DE-criar matriz
Type: QM
Difficulty: 1
Bloom taxonomy: remember
Last update: 2023-07-18
Who created: fzampirolli@ufabc.edu.br
Parametric: YES

- #2468 1. Criar uma matriz 4×4 com elementos $(i, j) = (((i+1) * 19) + ((j+1) * 11)) \bmod 100$. Calcular a soma dos elementos desta matriz. Os índices i são as linhas e j as colunas iniciadas com o valor 0.

A.*904 B.*897 C.*900 D.*892

Figura 12.4: Recorte do PDF gerado para a questão para criar uma matriz do Código 12.2.

A disciplina CS1 faz parte de um programa de bacharelado interdisciplinar oferecido pela universidade, e os estudantes que a cursam têm origens diversas. O método se baseia no software

MCTest-4, que permite gerar várias versões diferentes da mesma questão sem repetições. Para cada estudante, foram geradas três questões de dificuldades diferentes (fácil, média e difícil), porém com a preocupação de não deixar uma variação de exame mais difícil que a outra.

Os resultados dos experimentos mostraram que o método proposto foi eficaz na geração de questões paramétricas para exames impressos. Além disso, o uso do MCTest-4 facilitou a correção automática dos exames com QMs.

12.2 MCTest-5: a versão web

A partir deste ponto, todas as publicações foram conduzidas utilizando a versão web MCTest-5, que será mencionada simplesmente como MCTest. Esta versão possibilitou que os docentes empregassem o MCTest sem a necessidade de instalações locais em seus dispositivos, resultando em notável simplificação tanto na criação como na avaliação de exames, além da viabilização do compartilhamento de questões. A seguir, serão apresentados os artigos diretamente vinculados a esta edição *online*, corroborando sua eficácia e usabilidade, porém, ainda sem a integração do VPL no Moodle.

12.2.1 Artigo apresentando a primeira versão web do MCTest

O artigo de [Zampirolli, Teubl e Batista \(2019b\)](#) apresentou a primeira versão web do MCTest para geração e correção de questões parametrizadas, aplicável a diversas disciplinas e instituições educacionais. O método desenvolvido permite criar exames com variações nos textos, mantendo o mesmo conteúdo e nível de dificuldade. Para facilitar o processo, foram desenvolvidas interfaces gráficas que permitem coletar informações sobre instituições, cursos, disciplinas, tópicos, turmas, exames e questões.

O artigo apresenta as primeiras versões dessas interfaces gráficas, com ênfase na versão mais recente discutida nos Capítulos [2 - Visão geral do MCTest](#), [4 - Questões estáticas](#) e [5 - Questões paramétricas](#). Os resultados detalhados da implementação desse método são apresentados neste Capítulo [5](#).

O processo de geração de questões parametrizadas utiliza principalmente a linguagem de programação Python, com alguns elementos de L^AT_EX presentes no enunciado das questões. Os resultados obtidos demonstram a eficiência do sistema, mesmo em sua primeira versão, que foi implementada em páginas web escritas em Django. O sistema é gratuito e oferece aos professores a capacidade de gerar e corrigir exames automaticamente para centenas ou milhares de estudantes.

A seguir, serão apresentados exemplos das questões paramétricas discutidas neste artigo.

Equações paramétricas com equação e figura utilizando a biblioteca sympy

A seguir é apresentado um exemplo de questão paramétrica que inclui uma equação matemática e uma figura. Neste caso, a equação envolve funções seno, cosseno e integração.

O Código [12.3](#) primeiro importa as bibliotecas `sympy` e `random`. A biblioteca SymPy, disponível em docs.sympy.org, fornece funções para trabalhar com expressões simbólicas. O código

12.2. MCTest-5: A VERSÃO WEB

então gera três números aleatórios, `a1`, `a2` e `a3`, escolhidos entre 3 e 6, 2 e 4, e 1 e 3, respectivamente. Esses números serão usados para definir os parâmetros da função. O código então define a variável simbólica `x` e a função `f`. A função `f` é uma combinação de funções seno e cosseno com os parâmetros `a1`, `a2` e `a3`. O código então desenha o gráfico da função `f` no intervalo de -5 a 5. A imagem do gráfico é salva no arquivo `./tmp/imgs180days/fz_figIntegral_01_{a1}_{a2}_{a3}_h.png`. O código então calcula a integral da função `f`. A integral é calculada usando a função `integrate` da biblioteca SymPy. O código então converte a integral para notação L^AT_EX. Por fim, o código imprime as seguintes expressões em notação L^AT_EX: a integral da função `f`, a resposta correta à pergunta e três respostas incorretas.

Atenção:

1. A inclusão de imagens em arquivo L^AT_EX pode tornar a compilação para gerar o PDF muito lenta;
2. Uma sugestão é salvar as imagens no servidor do MCTest, disponível no seguinte *link*: [\(http://mctest.ufabc.edu.br:8000/tmp/imgs180days/\)](http://mctest.ufabc.edu.br:8000/tmp/imgs180days/). Esse servidor antes do `/tmp/` deve ser alterado a cada implantação do MCTest. O professor pode incluir um *link* para essa imagem no enunciado da questão, permitindo que os estudantes accessem durante a realização do exame.
3. Todos os arquivos temporários são eliminados automaticamente do servidor após 180 dias; contudo, essa configuração pode ser ajustada;
4. É fundamental incluir os parâmetros no nome do arquivo da imagem caso a questão seja paramétrica e a imagem varie conforme esses parâmetros. Isso pode ser feito como demonstrado nas linhas 22 a 29 do Código 12.3, em que uma *string* contendo os valores dos parâmetros é usada para gerar um objeto *hash*, convertido em uma *string* legível e adicionado ao nome do arquivo de imagem. Dessa forma, cada imagem gerada terá um nome único, garantindo que não haja conflitos entre elas.

Merece uma explicação especial o nome do arquivo em que será salva a imagem gerada. Esse arquivo deve ser único, minimizando a chance de criar um arquivo com o mesmo nome de outro que já existe. Para garantir isso, é utilizada a biblioteca `hashlib` na linha 23 do Código 12.3. Em seguida, o código cria uma *string* `s` que contém os valores dos parâmetros `a1`, `a2` e `a3`. Essa *string* é usada para criar um objeto *hash*, que será usado para gerar um nome único para o arquivo de imagem. Depois, o objeto *hash* é convertido em uma *string* legível e adicionado ao nome do arquivo de imagem. Esse nome é composto pelo prefixo `./tmp/imgs180days/fz_figIntegral_01_`, seguido pelos valores dos parâmetros `a1`, `a2` e `a3` e pelo *hash* gerado anteriormente. Por fim, o sufixo `.png` indica que a imagem será salva em formato PNG. Dessa forma, é possível gerar imagens únicas para diferentes valores dos parâmetros.

Questão:

```

1 A integral da função \([[code:a0]]) tem seu gráfico representado abaixo.
2 Calcule o valor da integral e marque a alternativa correta.
3
4 \begin{figure}[ht!]
5 \centering \includegraphics[scale=0.7]{[[code:figura]]}
6 \centerline{Figura da integral}
7 \end{figure}
8
9 [[def:
10 from sympy import symbols, sin, cos, plot, latex, Integral, integrate
11 import random
12
13 # Geração dos parâmetros aleatórios
14 a1 = random.randrange(3, 6, 1)
15 a2 = random.randrange(2, 4, 1)
16 a3 = random.randrange(1, 3, 1)
17
18 # Definição da variável simbólica e da função
19 x = symbols('x')
20 f = a1*sin(a2*x) - a3*cos(x)
21
22 # Plotagem da função
23 import hashlib
24 s = str([a1, a2, a3])
25 h = hashlib.md5(s.encode()) # create hash - arquivo único
26 h = str(h.hexdigest())
27 p = plot(f, (x, -5, 5), show=False)
28 figura = f'./tmp/imgs180days/fz_figIntegral_01_{a1}_{a2}_{a3}_{h}.png'
29 p.save(figura)
30
31 # Cálculo da integral e conversão para notação LaTeX
32 a0 = latex(Integral(f, x))
33 correctAnswer = latex(integrate(f, x))
34 error1 = latex(integrate(f + x, x))
35 error2 = latex(integrate(f - x, x))
36 error3 = latex(integrate(x**5 + x + 1, x))
37 # nas alternativas: \([[code:correctAnswer]]\); \([[code:error1]]\); ...
38 ]]

```

Código 12.3: Questão paramétrica com equação e figura.

Atenção:

A principal dificuldade ao lidar com questões paramétricas usando a biblioteca SymPy é definir corretamente o intervalo das variáveis aleatórias e também as respostas incorretas das alternativas, sem gerar soluções inexistentes.

MCTest

Topic: Topico Exemplo

Group:

Short Description: equação e figura

Type: QM

Difficulty: 1

Bloom taxonomy: remember

Last update: 2023-07-18

Who created: fzampirolli@ufabc.edu.br

Parametric: YES

#2469 1. A integral da função $\int (5 \sin(2x) - \cos(x)) dx$ tem seu gráfico representado abaixo. Calcule o valor da integral e marque a alternativa correta.

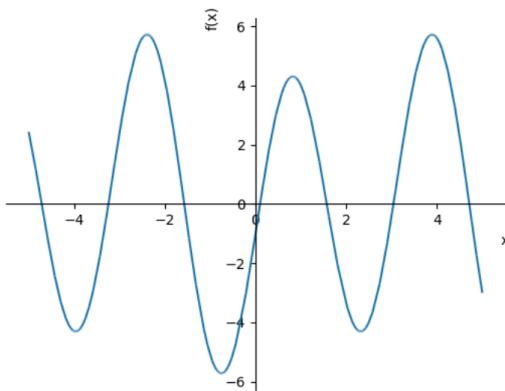


Figura da integral

A. $\frac{x^6}{6} + \frac{x^2}{2} + x$ B. $-\frac{x^2}{2} - \sin(x) - \frac{5 \cos(2x)}{2}$ C. $-\sin(x) - \frac{5 \cos(2x)}{2}$ D. $\frac{x^2}{2} - \sin(x) - \frac{5 \cos(2x)}{2}$

Figura 12.5: Recorte do PDF gerado para a questão paramétrica com equação e figura referente ao Código 12.3.

Questão paramétrica com grafo e figura

A questão paramétrica apresentada no Código 12.4 cria o grafo $G = (V, E)$, em que $V = n_1, n_2, \dots, n_p$ é um conjunto de nós (ou vértices) e E é um conjunto de arestas (ou arcos), sendo que cada $e \in E$ é dado por $e = (n_i, n_j)$ para $n_i, n_j \in V$. As arestas podem ter peso e o exemplo resolve o problema de encontrar o caminho de peso mínimo. Um caminho é uma sequência de vértices adjacentes, que determina uma sequência de arestas, e o peso do caminho é calculado somando os pesos de suas arestas.

Para gerar a imagem apresentada na Figura 12.6, será utilizado o método `draw` da biblioteca NetworkX (networkx.org) em conjunto com a biblioteca Matplotlib (matplotlib.org). No entanto, é importante notar que o código precisa ser modificado para tratar o caso em que existem dois ou mais caminhos mínimos com o mesmo peso. Uma possível solução seria modificar o código para descartar os pesos gerados nas linhas 16 e 17 quando existirem dois caminhos mínimos com o mesmo peso. Essa modificação é análoga à condição de aceitar apenas caminhos com pelo menos 5 vértices, verificada nas linhas 21 e 22. Dessa forma, o código garantiria que apenas um caminho mínimo seria considerado na geração da imagem, o que é necessário para garantir que a QM com alternativas

definidas nas linhas 30 e 31 tenha apenas uma correta.

Questão:

```

1 Qual é o caminho de Dijkstra neste grafo, entre os nós \(\text{a}\) e \(\text{e}\), com os
2 seguintes pesos \\ \verb|[ [code:grafo]]|?
3
4 \begin{figure}[!h]
5 \centering\includegraphics[scale=0.7]{ [ [code:pathGrafico]]}
6 \end{figure}
7
8 [[def:
9 import matplotlib.pyplot as plt, networkx as nx
10 from itertools import permutations
11
12 while True:
13     plt.clf()
14     G = nx.Graph() # Definição do grafo
15     w1,w2,w3,w4,w5,w6,w7 = random.sample(range(1,10), 7)
16     edges=[('a','b',w1),('b','c',w2),('a','c',w3),('c','d',w4),
17            ('b','d',w5),('d','e',w6),('e','c',w7)]
18     G.add_weighted_edges_from(edges)
19     grafo = str(edges)
20
21     v = nx.dijkstra_path(G, 'a', 'e') # Cálculo do caminho mínimo
22     if len(v)<5: continue # aceita pelo menos 5 nós
23
24     perm_list = [] # criar permutações
25     for perm in permutations(v):
26         if perm[0] == v[0] and perm[-1] == v[-1]:
27             perm_list.append(perm)
28     break
29
30     correctAnswer = str(perm_list[0])
31     error1,error2,error3 = str(perm_list[1]), str(perm_list[2]), str(perm_list[3])
32     # nas alternativas: \verb|[ [code:correctAnswer]]|\\ ; \verb|[ [code:error1]]|\\ ...
33
34     pos = nx.spring_layout(G) # Plotagem do grafo
35     nx.draw(G, pos=pos)
36     nx.draw_networkx_labels(G, pos=pos)
37     nx.draw_networkx_edge_labels(G, pos=pos)
38
39     import hashlib # Salvando a figura
40     s =str(edges)
41     h = hashlib.md5(s.encode()) # create hash - arquivo único
42     pathGrafico = 'figGrafo(' + str(h.hexdigest()) + ')'
43     plt.savefig(pathGrafico)
44 ]]
```

Código 12.4: QM paramétrica com grafo e figura.

MCTest

Topic: Topico Exemplo

Group:

Short Description: grafo

Type: QM

Difficulty: 1

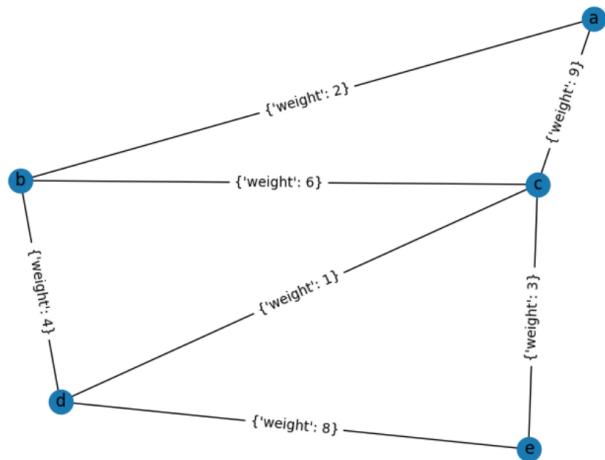
Bloom taxonomy: remember

Last update: 2023-07-18

Who created: fzampirolli@ufabc.edu.br

Parametric: YES

- #2471 1. Qual é o caminho de Dijkstra neste grafo, entre os nós a e e , com os seguintes pesos
[('a', 'b', 2), ('b', 'c', 6), ('a', 'c', 9), ('c', 'd', 1), ('b', 'd', 4), ('d', 'e', 8), ('e', 'c', 3)]?



- A.^{•3}('a', 'd', 'c', 'b', 'e')
B.^{•2}('a', 'd', 'b', 'c', 'e')
C.^{•0}('a', 'b', 'd', 'c', 'e')
D.^{•1}('a', 'b', 'c', 'd', 'e')

Figura 12.6: Recorte do PDF gerado para a questão paramétrica com grafo e figura referente ao Código 12.4.

Questão paramétrica com matriz e figura

No arquivo `static/Utils.py`, localizado no GitHub github.com/fzampirolli/mctest, há vários métodos úteis para a criação de questões paramétricas. Um desses métodos é o `drawMatrix()`, que permite desenhar uma matriz, conforme serão demonstrados nos códigos desta seção.

A questão apresentada nos Códigos 12.6 e 12.7 propõe a criação de um programa que manipula matrizes para identificar e ativar elementos com determinada condição em relação aos seus vizinhos. O programa recebe uma matriz de entrada E e cria uma matriz de saída F , inicializada com o valor “-1”. Um elemento (i, j) da matriz E é considerado “maior” ou “menor” em relação aos seus vizinhos nas direções especificadas. As direções são codificadas numericamente conforme a tabela fornecida na questão (ver Figura 12.7). O programa deve percorrer a matriz de entrada E e, para cada elemento que atenda à condição de ser “maior” ou “menor” em relação aos seus vizinhos, o valor correspondente na matriz de saída F deve ser atualizado. Após a ativação de todos os elementos que atendem à condição, o programa deve imprimir a matriz F . Além disso, são fornecidos exemplos de uma matriz de entrada e a saída correspondente, exibidos na forma de imagens. O programa deve

Método: desenha matriz

```

1  def drawMatrix(A, myfile):
2      """
3          Desenha uma representação visual de uma matriz e salva a imagem em um arquivo.
4          Exemplo de uso:
5          No enunciado da questão, desenhe a seguinte figura
6          (copie o código abaixo para uma nova questão):
7          \begin{figure}[h!]
8              \centering
9              \includegraphics[scale=0.55]{[[code:f"./tmp/imgs180days/{pathGraficoA}"]]}
10             \end{figure}
11
12 [[def:
13     import numpy as np
14     Lin, Col = 4, 5
15     A = np.random.randint(10, size=(Lin, Col))
16     # Plotagem da função
17     import hashlib
18     s = str([Lin, Col])
19     h = hashlib.md5(s.encode()) # create hash - arquivo único
20     h = str(h.hexdigest())
21     fig0 = f'fz_figExample01{Lin}_{Col}_{h}.png'
22     drawMatrix(A, fig0)
23 ]] """
24 import matplotlib.pyplot as plt
25
26 fig, ax = plt.subplots()
27 mat = ax.imshow(A, cmap='Pastel1', interpolation='nearest')
28
29 for x in range(A.shape[0]):
30     for y in range(A.shape[1]):
31         ax.annotate(str(A[x, y])[0], xy=(y, x), horizontalalignment='center',
32                     verticalalignment='center')
33
34 plt.show()
35 fig.savefig(f'./tmp/imgs180days/{myfile}.png', dpi=300)
36 plt.close()
```

Código 12.5: Método para desenhar matriz.

conseguir lidar com matrizes de qualquer dimensão e garantir que os elementos de borda da matriz F sejam mantidos como “-1”. O exemplo fornecido mostra uma matriz de entrada e a matriz de saída resultante após a ativação dos elementos que atendem à condição especificada. As imagens mostram a representação gráfica das matrizes. Para simplificar a resolução desta questão, é crucial destacar que $a_0 = (i, j)$ não se encontra em um pixel de borda. Em outras palavras, na matriz F , todos os elementos correspondentes às bordas devem ser mantidos com o valor “-1”. Um exemplo da saída dessa questão pode ser observado na Figura 12.7.

Questão:

```

1 Um elemento  $\textbf{(a}_0=(i,j)}$  é de uma matriz dito ser \textbf{[code:direcao]}
2 \textbf{[[code:maiormenor]]} se seu valor é \textbf{[[code:maiormenor]]} do que o valor de
3 seus vizinhos nas direções \textbf{[code:direcao_viz]}, considerando a seguinte
4 codificação para as possíveis direções:
5 \
6 \begin{table}[h!]
7 \centering\vspace{-2mm}
8 \begin{tabular}{lllll}
9 \cline{1-3}
10 \multicolumn{1}{|l|}{\textbf{(a}_8=\textbf{)}} Noroeste} & \multicolumn{1}{|l|}{\textbf{(a}_1=\textbf{)}} Norte} &
11 \multicolumn{1}{|l|}{\textbf{(a}_2=\textbf{)}} Nordeste} & & \\\ \cline{1-3}
12 \multicolumn{1}{|l|}{\textbf{(a}_7=\textbf{)}} Oeste} & & \multicolumn{1}{|l|}{\textbf{(a}_0=(i,j)\textbf{)}} &
13 \multicolumn{1}{|l|}{\textbf{(a}_3=\textbf{)}} Leste} & & & \\\ \cline{1-3}
14 \multicolumn{1}{|l|}{\textbf{(a}_6=\textbf{)}} Sudoeste} & \multicolumn{1}{|l|}{\textbf{(a}_5=\textbf{)}} Sul} &
15 \multicolumn{1}{|l|}{\textbf{(a}_4=\textbf{)}} Sudeste} & & \\\ \cline{1-3}
16 \end{tabular}
17 \end{table}
18 \vspace{-2mm}
19 \noindent Faça um programa para criar e ler uma matriz de entrada  $\textbf{(E)}$  de inteiros.
20 Seu programa deve criar também uma matriz de saída  $\textbf{(F)}$  e inicializa-la com o valor
21  $\textbf{-1}$ . Diz-se que um elemento  $\textbf{(i,j)}$  de  $\textbf{(F)}$  é \textbf{ativado} quando é atribuída a
22  $\textbf{(F[i,j])}$  o valor de  $\textbf{(E[i,j])}$ , isto é, quando  $\textbf{(F[i,j] \leftarrow E[i,j])}$ . O
23 seu programa deve também imprimir a matriz  $\textbf{(F)}$  depois de ativar todos os seus
24 elementos que são \textbf{\textbf{[code:direcao]} [[code:maiormenor]]}. Veja a seguir
25 um exemplo  $\textbf{([code:Linhas] \times [code:Colunas])}$  com a entrada e a saída
26 correspondente. \\
27
28 \noindent {\bf Obs.:} considere  $\textbf{(a}_0=(i,j)}$  não pertencer a um pixel de borda, isto
29 é, na matriz  $\textbf{(F)}$  todos os elementos de borda ficam com valor  $\textbf{-1}$ . O seu programa
30 deve funcionar para matrizes de quaisquer dimensões.
31
32 \begin{table}[h!] \vspace{-4mm}
33 \begin{tabular}{cc}
34 \includegraphics[scale=0.56]{pathGraficoA} &
35 \includegraphics[scale=0.56]{pathGraficoB}
36 \end{tabular}
37 \end{table}
38 \end{table}
39
40 \end{table}

```

Código 12.6: Questão paramétrica com matriz – Parte 1: Descrição de questão.

Resultados

O MCTest foi aplicado na disciplina de Processamento da Informação. Os resultados indicam que o método é eficaz na geração de exames com variações nos textos, mantendo o mesmo conteúdo e nível de dificuldade. Além disso, os estudantes consideraram as questões claras e relevantes para o conteúdo da disciplina.

Foram apresentados três exemplos de questões paramétricas, mas a quantidade de tipos e variações está diretamente relacionada à criatividade do professor que as prepara.

Questão:

```

1  [[def:
2    import json, random, numpy as np
3
4    def direcoes(offset, dir):
5      """ Método que retorna as direções adjacentes à direção dada. """
6      dir1, dir2 = (dir - 1) % 8 + 1, (dir + 1) % 8 + 1
7      return (dir1, dir, dir2)
8
9    # Parâmetros utilizados no enunciado da questão
10   #>>> BEGIN
11   dir = random.choice([1,2,3,4,5,6,7,8])
12   direcaoAll = ["Norte", "Nordeste", "Oeste", "Sudeste", "Sul", "Sudoeste", "Leste", "Noroeste"]
13   direcao = direcaoAll[dir-1]
14
15   maiormenor = random.choice(["maior", "menor"])
16   m, n = random.randrange(7, 9, 1), random.randrange(15, 21, 1)
17   Linhas, Colunas = m, n
18
19   #direcoes 0     1     2     3     4     5     6     7     8
20   offset = [[0,0], [-1,0], [-1,1], [0,1], [1,1], [1,0], [1,-1], [0,-1], [-1,-1]]
21   (dir1,dir0,dir2) = direcoes(offset,dir)
22   direcao_viz = "$a_" + str(dir1) + "$, $a_" + str(dir0) + "$, $a_" + str(dir2) + "$"
23   #<<< END
24
25   A = (np.random.random((m, n)) * 10).astype(int)
26   B = (np.zeros(A.shape) - 1).astype(int)
27   for i in range(1, m - 1):
28     for j in range(1, n - 1):
29       if maiormenor == "menor":
30         if (A[i + offset[dir1][0], j + offset[dir1][1]] > A[i, j]
31             and A[i + offset[dir0][0], j + offset[dir0][1]] > A[i, j]
32             and A[i + offset[dir2][0], j + offset[dir2][1]] > A[i, j]):
33           B[i, j] = A[i, j]
34         else:
35           if (A[i + offset[dir1][0], j + offset[dir1][1]] < A[i, j]
36               and A[i + offset[dir0][0], j + offset[dir0][1]] < A[i, j]
37               and A[i + offset[dir2][0], j + offset[dir2][1]] < A[i, j]):
38             B[i, j] = A[i, j]
39
40   import hashlib
41   s = ''.join([str(i) for i in A.flatten()]) # 2d to 1d to str
42   h = hashlib.md5(s.encode()) # create hash - arquivo único
43   pathGrafico = 'figRosa(' + direcao + ')' + (maiormenor + ')(' + str(h.hexdigest()) + ')'
44   pathGraficoA, pathGraficoB = pathGrafico + 'a.png', pathGrafico + 'b.png'
45
46   drawMatrix(A, pathGraficoA)
47   drawMatrix(B, pathGraficoB)
48 ]]

```

Código 12.7: Questão paramétrica com matriz – Parte 2: Bloco de código em Python.

MCTest

Topic: DE-matriz

Group:

Short Description: rosa dos ventos

Type: QT

Difficulty: 1

Bloom taxonomy: remember

Last update: 2023-07-18

Who created: fzampirolli@ufabc.edu.br

Parametric: YES

#2470 1. Um elemento $a_0 = (i, j)$ é de uma matriz dito ser **Nordeste maior** se seu valor é maior do que o valor de seus vizinhos nas direções a_1 , a_2 , a_3 , considerando a seguinte codificação para as possíveis direções:

a_8 = Noroeste	a_1 = Norte	a_2 = Nordeste
a_7 = Oeste	$a_0 = (i, j)$	a_3 = Leste
a_6 = Sudoeste	a_5 = Sul	a_4 = Sudeste

Faça um programa para criar e ler uma matriz de entrada E de inteiros. Seu programa deve criar também uma matriz de saída F e inicializa-la com o valor “-1”. Dizemos que um elemento (i, j) de F é *ativado* quando atribuímos a $F[i, j]$ o valor de $E[i, j]$, isto é, quando fazemos $F[i, j] \leftarrow E[i, j]$. O seu programa deve também imprimir a matriz F depois de ativar todos os seus elementos que são **Nordeste maior**. Veja a seguir um exemplo 7×17 com a entrada e a saída correspondente.

Obs.: considere $a_0 = (i, j)$ não pertencer a um pixel de borda, isto é, na matriz F todos os elementos de borda ficam com valor “-1”. O seu programa deve funcionar para matrizes de quaisquer dimensões.

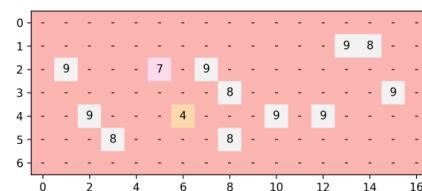
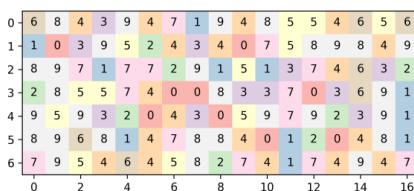


Figura 12.7: Recorte do PDF gerado para a questão paramétrica com matriz criada pelos Códigos 12.6 e 12.7.

12.2.2 Artigo descrevendo o MCTest em conjunto com *Google Forms* e *Google Sheets*

O artigo de Zampirolli, Batista, Iriarte e Junior (2020) aborda a adaptação do MCTest para solucionar o problema de avaliação a distância durante a pandemia. A solução permite criar questões paramétricas com Python e L^AT_EX. A correção automática é feita com o *Google Forms* e *Google Sheets*, constituindo uma contribuição original. A solução adaptada foi aplicada com sucesso em uma turma de Cálculo 1 com 100 estudantes.

Método

O MCTest permite a geração eficiente de QTs ou de QMs que podem ser modificadas parametricamente, produzindo muitas variações dessas questões. Isso é realizado por meio de várias bibliotecas da linguagem de programação Python, com as quais o MCTest calcula respostas automaticamente. O MCTest tem a vantagem de permitir questões paramétricas criadas com L^AT_EX e Python, com exemplos de uso apresentados no artigo e reproduzidos nesta seção.

O artigo descreve um desafio enfrentado na adaptação do MCTest para avaliações *online*

durante a pandemia, já que o sistema originalmente gerava exames em papel para serem aplicados em sala de aula. Para solucionar esse desafio, os autores propõem uma solução adaptada do MCTest que gera exames e os envia por e-mail aos estudantes em formato PDF, com submissões realizadas em uma planilha do *Google Forms* preenchida pelos estudantes e compilada ao professor em uma planilha do *Google Sheets*.

A planilha do *Google Sheets* é adaptada para a correção automática, conforme apresentado na Figura 12.8. Esta planilha também está disponível em github.com/fzampirolli/mctest, no arquivo `static/MCTest_Forms.ods`. A nota do estudante é calculada com base nas respostas enviadas, na chave de respostas armazenada na aba `templateAV1` e nas variações sorteadas para cada estudante na aba `variationsAV1`. A identificação, nome e e-mail de cada estudante são armazenados na aba `listStudents`. Todas essas abas são preenchidas com cópias dos arquivos enviados pelo MCTest, exemplificados a seguir. A pontuação é calculada para cada questão, incluindo uma QT com resposta exata, nas colunas H e I (Q4 e Q5, respectivamente). Os autores explicam em detalhes como a correção automática é realizada nas planilhas do *Google Sheets*, incluindo a correção manual pelo professor para avaliar a demonstração da resposta exata apresentada pelo estudante para essa QT.

Os autores estenderam o código do MCTest para permitir a correção automática de QTs, incluindo a possibilidade do professor receber dois arquivos CSV: um com as chaves de respostas para cada questão e outro com o número da variação enviada para cada estudante. Além disso, o código foi adaptado para permitir a criação de QTs com respostas paramétricas exatas, inseridas em um arquivo CSV e posteriormente na planilha do *Google Sheets*. Os autores apresentam exemplos de QMs e QTs, como a apresentada na Seção 12.2.1 – Equações paramétricas com equação e figura utilizando a biblioteca `sympy`, e explicam em detalhes como o MCTest lida com questões paramétricas e a correção automática de QTs. Um exemplo de QT exata foi apresentado no Código 8.1.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	Carimbo de data/hora	Student - ID	Student Name	Test	Q1	Q2	Q3	Q4	Q5	Grade Variation	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q5	
2	20/05/2020 16:41:49	1111	Student One	Test 1	C	B	B	12.81	https://	3	10	C	D	B	4.63	1,5	0	1,5	0	
3																				
4																				
5																				

+ = Responses to Form 1 templateAV1 variationsAV1 listStudents

Figura 12.8: Planilha mostrando o questionário preenchido de um estudante. Adaptado de Zampirolli, Batista, Iriarte e Junior (2020).

Arquivo CSV copiado na aba templateAV1, com os gabaritos. Ver explicações na Seção [8.3](#)

– [Detalhando os gabaritos no arquivo CSV](#) (as colunas são separadas por vírgula)

variation	Q1	Q2	Q3	Q4	Q5
1	C	E	D	3.47	
2	B	D	B	3.99	
3	A	B	C	4.07	
4	D	E	C	4.11	
5	E	B	A	4.29	
6	E	A	C	3.94	
7	B	B	D	3.56	
8	B	B	D	4.16	
9	B	E	B	3.73	
10	C	D	B	4.63	

Arquivo CSV copiado na aba variationsAV1, com os estudantes. Ver explicações na Seção [8.4 – Criando o PDF do exame com QR+QM+QT](#)

Room	ID	Name	Variation
Room1	1111	Student One	10
Room1	2222	Student Two	8

Arquivo CSV copiado na aba listStudents, com as variações. Ver explicações na Seção [3.3.1 – Criar turma com arquivo CSV](#)

1111	Student One	student1@email.com
2222	Student Two	student2@email.com

Experimentos

Os autores aplicaram o MCTest em uma turma da disciplina de Cálculo 1 na Universidade Federal do ABC, composta por 100 estudantes. A disciplina é parte do Bacharelado em Ciência e Tecnologia e possui alta taxa de reprovação. Durante sete semanas, os estudantes assistiram às aulas *online* e fizeram dois testes formativos, cada um contendo cinco questões paramétricas, duas QTs e três QMs. Os testes foram dados no mesmo formato do modelo apresentado na Figura [12.9](#), com questões e alternativas sorteadas. O Teste 1 foi aplicado *online* e os estudantes tiveram 24 horas para responder. O Teste 2, assim como o Exame 1, foram mais complexos e contaram para a nota final. Os estudantes tiveram dificuldades nas QTs, mas obtiveram bom desempenho nas QMs. A taxa de erros foi baixa nos testes e exames, e os estudantes não relataram problemas técnicos na submissão das respostas. Os autores suspeitam que a facilidade de acesso a recursos *online* possa ter influenciado no desempenho dos estudantes nas QTs. Para manter os estudantes engajados, o professor aplicou um terceiro exame contendo apenas QTs com submissão de imagens digitalizadas das respostas manuscritas.

Os autores abordam a questão do plágio em contextos *online* e propõem medidas preventivas,

 Universidade Federal do ABC Bacharelado em Ciência e Tecnologia Discipline: Funções de Uma Variável Classroom: fuv2020-class test Exam: Test 1	Room: s105 Date: 07-05-2020																		
Sig.: _____ Student: StudentOne ID/RA: 1																			
#139 - 2020-05-21 - 10:10:52																			
 <table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td>1</td> <td>2</td> <td>3</td> </tr> <tr> <td>A</td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> <tr> <td>B</td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> <tr> <td>C</td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> <tr> <td>D</td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> <tr> <td>E</td> <td><input type="radio"/></td> <td><input type="radio"/></td> </tr> </table>		1	2	3	A	<input type="radio"/>	<input type="radio"/>	B	<input type="radio"/>	<input type="radio"/>	C	<input type="radio"/>	<input type="radio"/>	D	<input type="radio"/>	<input type="radio"/>	E	<input type="radio"/>	<input type="radio"/>
1	2	3																	
A	<input type="radio"/>	<input type="radio"/>																	
B	<input type="radio"/>	<input type="radio"/>																	
C	<input type="radio"/>	<input type="radio"/>																	
D	<input type="radio"/>	<input type="radio"/>																	
E	<input type="radio"/>	<input type="radio"/>																	

Instructions:

- (a) Please complete this Test 1 on this link <https://forms.gle/PgFH7w9mo9zWteTX8>;
- (b) Form available until 08/05/2020 at 22h59m;
- (c) Only the first submission of the form will be accepted.

Multiple Choice Questions:

1. $y = \sqrt[4]{2 \cos(4\theta) + 3}$, find $\frac{dy}{d\theta}$:
 - A. $-\frac{6 \sin(4\theta)}{(2 \cos(4\theta) + 3)^{\frac{3}{4}}}$
 - B. $-\frac{4 \sin(4\theta)}{(2 \cos(4\theta) + 3)^{\frac{3}{4}}}$
 - C. $-\frac{8 \sin(4\theta)}{(2 \cos(4\theta) + 3)^{\frac{3}{4}}}$
 - D. $-\frac{2 \sin(4\theta)}{(2 \cos(4\theta) + 3)^{\frac{3}{4}}}$
 - E. $-\frac{2x \sin(4\theta)}{(2 \cos(4\theta) + 3)^{\frac{3}{4}}}$

Figura 12.9: Exemplo de exame utilizado em Cálculo 1. Adaptado de Zampirolli, Batista, Iriarte e Junior (2020).

incluindo a administração de exames individualizados, a imposição de restrições temporais e a supervisão por meio de *webcams*. Contudo, eles reconhecem que em circunstâncias extraordinárias, como a pandemia, torna-se necessário empregar ferramentas que viabilizem as atividades de aprendizado, como mencionado no artigo.

Os resultados dos experimentos mostraram ser importante elaborar questões *online* com enunciados melhor elaborados para os estudantes poderem raciocinar para achar as soluções e não achar respostas prontas na internet. Os autores destacam que o MCTest pode ser uma ferramenta valiosa para manter os estudantes engajados em atividades com QMs e sugerem a aplicação de vários exames desse tipo, com correção automática, para estimular a participação dos estudantes.

12.2.3 Artigo descrevendo a adoção de QMs com pesos diferentes nas alternativas

O artigo de [Zampirolli, Batista, Rodriguez, Rocha e Goya \(2021\)](#) apresenta uma adaptação do sistema MCTest que permite a avaliação automatizada de QMs com respostas ponderadas.

Método

O sistema MCTest original produz um arquivo CSV que inclui as respostas para cada QM, com as pontuações atribuídas a cada alternativa. A adaptação proposta neste artigo introduz uma coluna adicional no arquivo de respostas para cada questão, indicando o peso de cada alternativa. O sistema de avaliação automática calcula a nota do estudante ponderando a pontuação de cada alternativa de acordo com seu respectivo peso.

O processo de criação de questões e exames é o mesmo apresentado nos capítulos anteriores. A diferença está na forma de calcular a nota do estudante. Esta seção detalhará esse processo.

A planilha utilizada no artigo disponível no GitHub, em github.com/fzampirolli/mctest, no arquivo `static/GRADE.ods`. Essa planilha possui duas abas principais. A primeira é a aba BD, contendo as correções geradas pelo MCTest ao submeter os exames digitalizados, conforme visualizado na Figura 12.10. A segunda aba é para o cálculo da nota final, como apresentado na Figura 12.11.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	Pag	ID	Answ	Quest	Inv	Grad	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
2	1	2	4	10	0	8	D/C	D	A	A	A	C	D	A	C/A	A	11261203	11303210	11330213	11310132	11240231	11323102	11283120	11290321	11250231	11270312
3	2	5	4	10	1	9	C	B	D	B	B	B	A	A	O/B	D	11293102	11322013	11311320	11271023	11282031	11253012	11240321	11330213	11263021	11303210
4	3	4	4	10	0	8	C/D	B	C	C	B/D	C	C	A	C	C	11251320	11322013	11291203	11273102	11301320	11331302	11261302	11310321	11242301	11281203
5	4	1	4	10	0	3	A/C	D	C/D	C/B	C/B	C	A	C/B	A/D	B/D	11322301	11282130	11251320	11331023	11241032	11263102	11300213	11273012	11293120	11311230
6	5	2	4	10	0	10	C	D	A	A	A	C	D	A	A	A	11261203	11303210	11330213	11310132	11240231	11323102	11283120	11290321	11250231	11270312
7	6	4	4	10	0	9	D	B	C	C	D	B/C	C	A	C	C	11251320	11322013	11291203	11273102	11301320	11331302	11261302	11310321	11242301	11281203
8	7	1	4	10	0	10	C	D	D	B	B	C	A	B	D	D	11322301	11282130	11251320	11331023	11241032	11263102	11300213	11273012	11293120	11311230
9	8	2	4	10	0	10	C	D	A	A	A	C	D	A	A	A	11261203	11303210	11330213	11310132	11240231	11323102	11283120	11290321	11250231	11270312
10	9	5	4	10	0	8	D/C	B	A/D	B	B	B	A	A	B	D	11293102	11322013	11311320	11271023	11282031	11253012	11240321	11330213	11263021	11303210
11	10	3	4	10	0	9	C	C	D	B	B	D	A	D	B/A	C	11323201	11253201	11263120	11243021	11283012	11272310	11310321	11301320	11290321	11332301
12	11	4	4	10	2	8	D	B	O/C	C	D	C	C	A	O/C	C	11251320	11322013	11291203	11273102	11301320	11331302	11261302	11310321	11242301	11281203
13	12	3	4	10	0	9	C	C	D	B	B	D	C/A	D	A	C	11323201	11253201	11263120	11243021	11283012	11272310	11310321	11301320	11290321	11332301
14	13	3	4	10	0	9	C	C	D	B	B	D	A	D	D/A	C	11323201	11253201	11263120	11243021	11283012	11272310	11310321	11301320	11290321	11332301

← → + FinalGrade BD

Figura 12.10: Visualização do arquivo CSV com as correções. Adaptado de [Zampirolli, Batista, Rodriguez, Rocha e Goya \(2021\)](#).

Na Figura 12.10, as colunas Q a Z na planilha mostram o ID da questão no BD e a ordem das alternativas sorteadas para os estudantes. A célula Q2, por exemplo, possui o valor 11261203, em que 1126 é o ID da questão e 1203 indica a ordem das alternativas sorteadas (A=1, B=2, C=0, D=3). Com fórmulas simples, é possível calcular a nota final atribuindo pesos às alternativas. Na Figura 12.11, o peso da alternativa correta é 1, as parcialmente corretas têm peso 0,2 cada e a alternativa absurda tem peso 0. O campo amarelo na Figura 12.11 exibe a pontuação de cada questão, em que

	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE	AF	AG	AH	AI	AJ	AK	AL	AM	AN
1		A	B	C	D																										
2		0	1	2	3																										
3	Weight	1	0,2	0,2	0																										
4	Grade	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
5	8	0	1	1	1	1	1	1	0	1	D/C	D	A	A	A	C	D	A	C/A	A	1203	3210	0213	0132	0231	3102	0320	0321	0231	0312	
6	9	1	1	1	1	1	1	1	0	1	C	B	D	B	B	B	A	A	0/B	D	3102	2013	1320	1023	2031	3012	0321	0213	0301	3210	
7	8,2	0,2	1	1	1	0	1	1	1	1	C/D	B	C	C	B/D	C	C	A	C	C	1320	2013	1203	3102	1320	1302	0321	2301	1203		
8	4	0,2	1	0,2	0,2	0	1	1	0,2	0	0,2	A/C	D	C/D	C/B	C/B	C	A	C/B	A/D	B/D	2301	2130	1320	1023	3102	0213	3012	3120	1230	
9	10	1	1	1	1	1	1	1	1	1	C	D	A	A	A	C	D	A	A	A	1203	3210	0213	0132	0231	3102	0320	0321	0231	0312	
10	9	1	1	1	1	1	0	1	1	1	D	B	C	C	D	B/C	C	A	C	C	1320	2013	1203	3102	1320	1302	0321	2301	1203		
11	10	1	1	1	1	1	1	1	1	1	C	D	D	B	B	C	A	B	D	D	2301	2130	1320	1023	3102	0213	3012	3120	1230		
12	10	1	1	1	1	1	1	1	1	1	C	D	A	A	A	C	D	A	A	A	1203	3210	0213	0132	0231	3102	0320	0321	0231	0312	
13	8,4	0,2	1	0,2	1	1	1	1	1	1	D/C	B	A/D	B	B	B	A	A	B	D	3102	2013	1320	1023	2031	3012	0321	0213	3021	3210	
14	9	1	1	1	1	1	1	1	1	0	1	C	C	D	B	B	D	A	D	B/A	C	3201	3201	3120	3021	2310	0321	1320	0321	2301	
15	8	1	1	0	1	1	1	1	1	0	1	D	B	D	O/C	C	D	C	C	A	0/C	C	1320	2013	1203	3102	1320	1302	0321	2301	1203
16	9,2	1	1	1	1	1	1	0,2	1	1	1	C	C	D	B	B	D	C/A	D	A	C	3201	3201	3120	3021	3012	2310	0321	1320	0321	2301
17	9,2	1	1	1	1	1	1	1	0,2	1	C	C	D	B	B	D	A	D	D/A	C	3201	3201	3120	3021	3012	2310	0321	1320	0321	2301	
18																															

Figura 12.11: Cálculo final do exame. Adaptado de [Zampirolli, Batista, Rodriguez, Rocha e Goya \(2021\)](#).

cada linha corresponde ao QR de um estudante. A nota total do estudante para o exame é calculada somando as notas de cada questão.

O sistema MCTest pode ser usado para avaliar exames realizados em formato físico ou *online*. No caso de exames realizados em formato físico, os estudantes respondem às questões em um QR impresso, como apresentado na Figura 12.9. O QR impresso é então digitalizado e o sistema MCTest é usado para corrigir as respostas. No caso de exames realizados em formato *online*, os estudantes respondem às questões em um navegador da web, como descrito na Seção 12.2.2 – [Artigo descrevendo o MCTest em conjunto com Google Forms e Google Sheets](#).

Experimentos

O método deste artigo foi aplicado a exames realizados em formato físico ou *online* e validado com um total de 607 estudantes de três diferentes disciplinas: Redes e Comunicações, Natureza da Informação e Compiladores. Os resultados mostraram que o sistema de correção automática é preciso e confiável. O sistema foi bem avaliado pelos estudantes, que o consideraram uma ferramenta útil para a avaliação de seus conhecimentos.

Artigo comparando diferentes processos seletivos na EPUFABC

O artigo de [Zampirolli, Batista, Josko, Steil e Trevisan \(2021\)](#) analisa o processo de seleção para a Escola Preparatória da UFABC (EPUFABC) aplicado a milhares de candidatos de escolas públicas da região. A EPUFABC oferece um curso gratuito voltado exclusivamente para estudantes de baixa renda que se inscreverão em exames de entrada na universidade. O processo de seleção emprega a Teoria Clássica ao Item (TCI), mas há dúvidas sobre a imparcialidade dos critérios de desempate. Por isso, os autores contrastam a abordagem atual do processo com a Teoria da Resposta ao Item (TRI) ([BIRNBAUM, 1968](#)). Os resultados mostram que o método TRI pode melhorar substancialmente o processo de seleção do EPUFABC.

O artigo menciona que o exame de entrada da Universidade de São Paulo (USP) é um exemplo de seleção desafiadora, com inúmeros candidatos para poucas vagas. Para exames com alta taxa de

12.2. MCTEST-5: A VERSÃO WEB

inscrição, lidar com a nota de corte, na qual muitos candidatos provavelmente terão pontuação igual, é um desafio.

Para resolver esse problema, o Exame Nacional do Ensino Médio (ENEM) no Brasil passou a utilizar a Teoria da Resposta ao Item (TRI) desde 2009. A TRI também é empregada em outros países, como nos Estados Unidos, em exames como o TOEFL.

Método

Os autores foram inspirados pela metodologia do ENEM e avaliaram o processo de seleção utilizado pela EPUFABC, que atende a milhares de candidatos desde 2010. O exame de admissão da EPUFABC é composto por 10 QMs para cada área de conhecimento, além de um exame de Gramática Portuguesa, que substitui a redação exigida pelo ENEM. As áreas avaliadas são: Ciências Humanas, Ciências da Natureza, Línguas e Códigos (Inglês-Espanhol-Português) e Matemática.

A avaliação comparou o Processo de Seleção do EPUFABC aplicado de 2019 a 2021 com o ENEM, que adota o modelo logístico de três parâmetros da TRI. Foram desenvolvidos três *scripts* em Python e R no *Google Colab* para realizar as análises. Cada *script* passa por três etapas: pré-processamento, geração da matriz de respostas (em Python) e análise estatística (em R). Todos os dados utilizados no artigo podem ser baixados em vision.ufabc.edu.br/MCTest/public/IRT2021, que inclui todos os arquivos CSV utilizados pelos Notebooks, mas sem a identificação dos candidatos.

Para comparações, foram utilizados também os microdados do ENEM de 2019 disponibilizados pelo Ministério da Educação. Os dados do ENEM requerem um pré-processamento complexo devido ao seu tamanho, envolvendo segmentação, extração de colunas relevantes e geração de matrizes de respostas.

Experimentos

Este estudo baseia-se em três conjuntos de dados: (1) MCTest (2019 e 2020), (2) Moodle (2021) e (3) ENEM (2019). Tanto a Teoria Clássica ao Item (TCI) quanto a Teoria da Resposta ao Item (TRI) foram utilizadas para analisar e comparar os resultados.

Os exames TCI de 2019 e 2020 foram realizados presencialmente em um sábado, com 50 QMs nas cinco áreas de conhecimento referidas anteriormente. Os exames foram corrigidos automaticamente pelo sistema MCTest, conforme explicado no Capítulo 7 – [Exames com Quadro de Respostas \(QR\)](#). Em 2021, devido à pandemia de COVID-19, o processo de seleção foi adaptado para exames *online*, realizados ao longo de cinco dias. Os candidatos acessaram material de ensino (videoaulas) e QMs na plataforma Moodle e tiveram 24 horas para responder. A pontuação final considerou o número de respostas corretas, presença nos cinco dias de exame e idade do candidato.

Em 2019, houve 2.033 candidatos no TCI, enquanto em 2020 houve 2.043 candidatos. Em ambos os anos, 633 candidatos foram selecionados para a EPUFABC, sendo metade para aulas no período da tarde e a outra metade no período da noite.

No exame de 2021, os candidatos realizaram o exame *online* na plataforma Moodle, totalizando 2.768 participantes. A EPUFABC ofereceu 380 vagas para o ano de 2021, sendo 190 vagas para cada período, e as aulas foram disponibilizadas remotamente na plataforma Moodle.

Foram calculadas médias e desvios padrão para os exames de todas as áreas de conhecimento dos processos TCI2019, TCI2020 e ENEM2019. As análises também incluíram Curvas Características do Item (CCI) da TRI. A comparação entre TCI e TRI mostrou que o método TRI pode ser útil como critério de desempate em processos de seleção, já que considera a habilidade do candidato em responder corretamente às questões.

O estudo também destacou a melhora significativa nas notas dos candidatos no TCI2021, que foi realizado *online* com mais tempo disponível para responder às questões, em comparação com os processos presenciais anteriores. As análises mostraram que as habilidades dos candidatos em TCI2021, representadas pelas curvas CCI, estavam mais concentradas no lado direito da escala, indicando que candidatos com habilidades menores conseguiram acertar mais questões devido ao tempo extra para responder cada exame.

Após a continuidade deste trabalho, outro artigo de Zampirolli, Junior, Steil e Jr. (2021) foi publicado, descrevendo o desenvolvimento de um sistema interativo chamado ENEM interativo e disponível em mctest.ufabc.edu.br:8000/ENEM. Esse sistema permite que os estudantes utilizem exames antigos do ENEM e os microdados disponibilizados pelo INEP para estudo. No ENEM Interativo, os exames em formato PDF são convertidos para HTML e, automaticamente, botões de resposta são incorporados para cada questão. Ao finalizar cada exame, o botão de estatísticas abre uma nova página que apresenta informações como o gabarito, as respostas dos estudantes, as habilidades (valores fornecidos pelo INEP) e gráficos. O sistema proporciona uma plataforma mais interativa e acessível para os estudantes explorarem exames anteriores do ENEM e analisarem seu desempenho em detalhes, incluindo informações estatísticas relevantes para a compreensão de seu progresso e habilidades.

12.3 Considerações finais

Neste capítulo, foram discutidas as versões 4 e 5 do MCTest, assim como os artigos que as acompanharam. Ao longo do tempo, o MCTest passou por uma significativa evolução, saindo de suas primeiras versões focadas em exames impressos e se tornando uma versão web atual, integrada também ao Moodle. Essa transformação trouxe diversos benefícios e avanços para o sistema.

Os experimentos e casos de uso apresentados nos artigos destacaram a aplicabilidade bem-sucedida do MCTest em diversas situações reais de ensino. Isso reforça a eficiência e versatilidade do sistema em diferentes contextos, abrangendo exames presenciais, *online*, individualizados e integração com plataformas como o *Google Forms*. Esse recurso mostrou-se especialmente relevante durante a pandemia de COVID-19, permitindo a continuidade da avaliação educacional mesmo em cenários remotos.

O MCTest-4 é um software desenvolvido para a correção de exames com QMs. Ele automatiza o processo de correção, gerando resultados precisos e poupando tempo dos educadores. Já o MCTest-4.G foi criado para a geração de exames com QMs, permitindo a criação de exames distintos e aleatórios para cada estudante, proporcionando maior segurança na aplicação dos exames.

A versão mais recente, o MCTest-5, é considerada um software web altamente funcional.

12.3. CONSIDERAÇÕES FINAIS

Além de possuir as funcionalidades do MCTest-4, o MCTest-5 permite a criação, correção e compartilhamento de questões, apresentando recursos avançados e integração com outras plataformas educacionais, como o Moodle. No próximo capítulo, serão destacadas especialmente as especificidades das questões que envolvem EPs integrados ao *plugin* VPL do Moodle.

Por fim, é relevante ressaltar que a maioria das publicações efetuadas nesta versão web do MCTest foram subsidiadas pelo projeto FAPESP, sob o processo [2018/23561-1](#), cujo título é “Um sistema universal para a geração e correção automática de questões parametrizadas”.

Capítulo 13

MCTest+Moodle+VPL

Conteúdo

13.1 MCTest-5, com Moodle	238
13.2 MCTest-5, com Moodle+VPL	238
13.2.1 Artigo com a primeira integração entre MCTest+Moodle+VPL	238
13.2.2 Artigo aplicando a integração MCTest+Moodle+VPL em CS0	240
13.2.3 Artigo sobre o uso de múltiplas linguagens de programação em CS1	242
13.2.4 Artigo aplicando a integração MCTest+Moodle+VPL em CS1	243
13.2.5 Artigo explorando a integração de jogos no ensino de CS0	244
13.3 Aplicação do MCTest-5 com Moodle+VPL no Ensino de PDI	245
13.3.1 Método	246
13.3.2 Experimentos	247
13.4 MCTest-5 com Feedback Inteligente baseado em LLMs	248
13.4.1 Método	248
13.4.2 Experimentos e Resultados	249
13.5 Considerações finais	249

Neste capítulo, serão apresentados resumos dos artigos que relatam os métodos e experimentos de integração entre o MCTest e o Moodle, com foco especial na utilização do *plugin* VPL. A maioria desses artigos já foi detalhada em capítulos anteriores deste livro e, portanto, não será refeita a apresentação completa. Os experimentos foram realizados principalmente nas disciplinas de Bases Computacionais da Ciência (CS0) e Processamento da Informação (CS1) do Bacharelado em Ciência e Tecnologia da Universidade Federal do ABC (UFABC).

13.1 MCTest-5, com Moodle

Artigo sobre a exportação de questões paramétricas para o Moodle usando o MCTest

O artigo de [Zampirolli, Batista, Pimentel e Braga \(2021\)](#) aborda a exportação de questões paramétricas para o Moodle através do MCTest, apresentado em detalhes no Capítulo 9 – [Variações de exames com QM+QT para o Moodle](#). Neste capítulo, é explorada uma funcionalidade interessante do MCTest que permite exportar variações de exames nos formatos Aiken e XML, os quais podem ser utilizados pelo Moodle para criar um banco de questões.

Enquanto o Moodle suporta a criação de questões parametrizadas usando valores curinga, essas questões possuem limitações em relação às funções disponíveis na linguagem PHP. Ao utilizar o MCTest para criar o banco de questões, obtém-se uma vantagem, uma vez que ele suporta diversos tipos de questões parametrizadas. Por exemplo, é possível criar um exame contendo QMs e QTs com respostas exatas. Em seguida, o exame é configurado para gerar variações, salvando-o nos formatos Aiken ou XML. Posteriormente, esses arquivos podem ser importados para o banco de questões do Moodle, permitindo a criação de uma atividade com as questões recém-importadas. Dessa forma, é possível criar atividades muito mais personalizadas, superando as limitações dos curingas oferecidos apenas pelo Moodle.

O capítulo detalha a criação de um exame composto por três QMs paramétricas e uma QT com resposta exata. Para cada questão, são fornecidos exemplos de implementação e explicação sobre como o MCTest pode ser utilizado para gerar diferentes variações.

Além disso, são apresentados detalhes sobre o conteúdo dos arquivos nos formatos Aiken e XML, utilizados para armazenar QMs e QTs com respostas exatas. É importante ressaltar a necessidade de remover os comentários adicionados pelo MCTest nos arquivos Aiken antes de importá-los para o Moodle. No caso dos arquivos XML, é enfatizada a complexidade da sintaxe e a importância de remover os comentários em L^AT_EX presentes em cada questão no MCTest. Deve-se também atentar para o formato utilizado, como fórmulas matemáticas, que podem se tornar inválidas no Moodle devido à sua limitação na formatação de questões.

13.2 MCTest-5, com Moodle+VPL

Durante a pandemia de COVID-19, o MCTest desempenhou um papel crucial ao permitir a avaliação remota de milhares de estudantes, principalmente em disciplinas que envolvem exercícios de programação (EP). Foram implementadas e validadas diversas funcionalidades em turmas para viabilizar esse processo, com destaque para as principais melhorias apresentadas nesta seção.

13.2.1 Artigo com a primeira integração entre MCTest+Moodle+VPL

Enquanto a Seção 12.1.3 – [Artigo definindo a parametrização do MCTest após a versão 4](#) resumiu a primeira publicação sobre a parte paramétrica, a Seção 12.2.1 – [Artigo apresentando a](#)

primeira versão web do MCTest utilizou essa parametrização na versão web do MCTest. Nesta primeira integração, as correções dos EPs eram realizadas manualmente. Isso envolvia baixar todas as submissões dos estudantes, executar *scripts* para testar os códigos submetidos e, posteriormente, atribuir as notas manualmente por parte dos professores. Nos resultados das próximas seções, somente a criação de novas questões foi realizada de forma manual, conforme será detalhado a seguir.

O artigo de [Zampirolli, Pisani, Josko, Kobayashi, Silva, Goya e Savegnago \(2020\)](#), resumido nesta seção, apresenta uma solução para simplificar a criação e correção de questões paramétricas em uma disciplina de Processamento da Informação (ou Introdução à Programação, IP, ou também CS1) com abordagem de ensino híbrido (IP-BL – *blended learning*). A solução integra a ferramenta de código aberto MCTest com o Moodle e o plugin VPL (*Virtual Programming Lab*) ([PINO, 2012](#)), permitindo a geração e avaliação automática de questões paramétricas relacionadas à linguagem de programação. Essa abordagem foi aplicada em duas turmas da disciplina, totalizando 171 estudantes matriculados, e utilizou a linguagem de programação Java.

Método

A primeira parte do artigo aborda a parametrização de QTs destinadas aos estudantes para serem respondidas em folha de papel, com correção manual. A questão apresentada é uma variação do teste de mesa, conforme demonstrado nos Códigos [5.7](#) e [5.8](#). Esse tipo de questão foi incluído no Exame 1 da disciplina.

A segunda parte deste artigo apresenta uma questão parametrizada que pode ser submetida pelos estudantes como atividade VPL no Moodle. Essa questão foi aplicada no Exame 2 e requer a resolução utilizando uma linguagem de programação. Embora o IP-BL tenha adotado Java, qualquer linguagem de programação suportada pelo VPL do Moodle pode ser utilizada para resolver a questão. O exemplo de questão apresentado no artigo envolve um operador e dois vetores. O operador e o tamanho dos vetores são parâmetros que podem ser modificados para obter vários modelos da mesma questão.

A principal contribuição deste artigo é o uso de questões paramétricas com avaliação automatizada usando o VPL. O MCTest produz questões paramétricas, conforme descrito nos capítulos anteriores. No entanto, uma grande dificuldade para os professores em disciplinas de programação com muitos estudantes é a correção e o *feedback*. O VPL veio para fornecer essa parte, mas até então, sem abordar a parte paramétrica. Para resolver isso, o artigo apresentou um método para automatizar o EP parametrizado, sendo a principal contribuição para o estado da arte da Tecnologia da Informação e Comunicação (TIC) na educação apresentada neste artigo.

O sistema padrão do VPL aceitava apenas uma única versão de uma questão, o que exigiria criar uma avaliação prática de EP para cada versão de cada questão em um exame. Por exemplo, em um exame com três questões, cada uma com seis variações, o professor precisaria criar e configurar 18 avaliações práticas diferentes. Além disso, os estudantes precisam escolher as versões corretas segundo as avaliações exclusivas, o que pode levar a confusões e erros.

Utilizando o sistema modificado, o professor precisa criar três questões e 6 casos de teste. Em seguida, o estudante precisa enviar um segundo arquivo TXT contendo uma única linha como

“**MODELO: A**”, que representa o modelo da questão, definida no PDF do exame. Esse arquivo de texto é analisado utilizando uma expressão regular para isolar o código de modelo para aquele estudante, e em seguida, é usado para escolher o modelo correto dos arquivos de casos de teste utilizados pelo professor. Cada modelo tem seu próprio arquivo de casos de teste, que substitui o arquivo padrão do VPL.

Após a criação de uma atividade do VPL no Moodle, em “Arquivos de execução”, o professor precisa adicionar seis casos de teste e outros arquivos disponíveis em github.com/fzampirolli/mctest, na pasta `VPL_modification/V1-select_using_second_file`. Esses arquivos foram criados pelo Heitor Rodrigues Savegnago e seu orientador Prof. Dr. Paulo Henrique Pisani.

Como o método apresentado no artigo já está ultrapassado, ele não será replicado neste livro. No entanto, um método mais atualizado e aprimorado foi apresentado no Capítulo 10 – [Exames com MCTest+Moodle+VPL](#), eliminando a necessidade desses modelos.

Experimentos

Nos primeiros quatro meses de 2019, a disciplina CS1 recebeu um total de 1.437 matrículas de estudantes distribuídos em 46 turmas presenciais ou *blended learning* (IP-BL). As turmas presenciais adotaram as linguagens de programação Java ($36/46=78\%$ das turmas) ou Python. Em contraste, as turmas de IP-BL aplicaram uma combinação de pseudocódigo e Java ($2/46=4\%$), ou seja, os estudantes desenvolveram soluções em pseudocódigo (através da ferramenta [Portugol Studio](#)) e as traduziram para código Java. O foco deste artigo é nas turmas de IP-BL, nas quais a proposta foi aplicada em 2019. Analisando as ofertas anteriores, foi observado uma melhora geral na oferta no primeiro período de 2019, quando o método do artigo foi aplicado. No entanto, os autores não puderam afirmar que essa é a razão para o melhor desempenho.

13.2.2 Artigo aplicando a integração MCTest+Moodle+VPL em CS0

O artigo de [Zampirolli, Sato, Savegnago, Batista e Kobayashi \(2021\)](#) apresenta o contexto e desafios da educação a distância devido à pandemia, destacando a adaptação de sistemas de código aberto, como o Moodle e o MCTest. As principais contribuições são a melhoria das capacidades de criação de atividades individualizadas e a integração do MCTest com o *plugin* VPL do Moodle.

Neste artigo, é apresentado um método para criar e corrigir EPs parametrizados. A contribuição principal consiste em gerar listas unificadas de exercícios paramétricos, para as quais são produzidos casos de teste automaticamente. Esse método foi validado na disciplina de Bases Computacionais da Ciência (CS0) durante a pandemia de COVID-19. As listas de exercícios individualizados foram enviadas semanalmente por e-mail para os estudantes por uma adaptação do sistema MCTest. A correção automática foi realizada por meio de uma adaptação no *plugin* VPL do Moodle. Os recursos propostos foram utilizados por 28 turmas, totalizando 1.407 estudantes. Um questionário enviado aos estudantes revelou que cerca de 82% deles aprovaram o método.

Método

Os Códigos 5.21 e 5.22 introduziram as QTs paramétricas com a integração MCTest+Moodle+VPL, e o Capítulo 10 – Exames com MCTest+Moodle+VPL apresentou o processo de criação de um exame com essa integração. O artigo apresenta o primeiro experimento utilizando esse processo na disciplina de Bases Computacionais da Ciência (CS0) na UFABC em 2020, que será resumido nesta seção.

O artigo também apresenta exemplos de questões paramétricas e explica como compilar o conteúdo das questões utilizando o MCTest. Foram realizadas melhorias em relação à versão anterior do método, como a geração automática de questões agrupadas em uma lista semanal em PDF enviada por e-mail para cada estudante, evitando a repetição de exercícios, e a eliminação da necessidade de criação manual de casos de teste para cada modelo.

Em resumo, o método utilizado para produzir as listas semanais foi o seguinte:

1. Elaborar questões conforme o modelo apresentado;
2. Montar as questões em um Exame, especificando também as turmas;
3. Escolher o formato “Json” ao selecionar “Criar-Variações” na página de Exame. Os casos de teste serão enviados por e-mail para o professor em um arquivo `*_linker.json`;
4. Clicar no botão “Criar-PDF” do exame, escolhendo antes “Sim” para *feedback* ao estudante. O MCTest enviará ao professor o arquivo `*_students_variations.csv` que especifica qual variação foi selecionada para cada estudante;
5. Mover ambos os arquivos para “Arquivos de Execução” em uma atividade VPL do Moodle, após renomeá-los como `linker.json` e `students_variations.csv`, juntamente com os arquivos de adaptação do VPL disponíveis em github.com/fzampirolli/mctest, na pasta `VPL_modification/V10-new_selector`;
6. Esses arquivos incluem `makefile`, `vpl_evaluate.sh` e `zip.tar.gz.b64`.

Esses arquivos disponibilizados no GitHub também foram criados pelo Heitor Rodrigues Savegnago e seu orientador Prof. Dr. Paulo Henrique Pisani.

Experimentos

Devido à diversidade de critérios de avaliação adotados pelos vários professores, não foi possível unificar a análise das 43 turmas de CS0 em 2020.2 em relação às notas finais.

Cada professor aplicou seu próprio critério para calcular as notas finais dos estudantes, seguindo as regras da universidade, que incluíam dar pelo menos 72 horas para os estudantes enviarem suas soluções de exames, o que ajudava em caso de limitações de acesso à internet. Alguns professores optaram por utilizar apenas as listas unificadas para calcular as notas finais dos estudantes, enquanto outros combinaram as listas com projetos em grupo, e alguns ainda incluíram exames tradicionais.

Apesar da dificuldade de comparar os desempenhos, constatou-se que: (1) Os 505 estudantes das 15 turmas que não adotaram nenhum dos recursos obtiveram uma taxa de aprovação de 80%; (2) A mesma taxa foi alcançada pelos 137 estudantes das quatro turmas que adotaram apenas o material unificado; (3) Utilizando material e listas unificados, tratadas pelo MCTest+Moodle+VPL, 765 estudantes restantes das 24 turmas obtiveram uma taxa de aprovação de 81%. Esse número contrasta com a média histórica anterior de 75%, calculada para 23.461 estudantes desde 2009. É importante destacar que a disciplina CS0 em 2020.2 foi realizada 100% a distância pela primeira vez. Para contornar a imprecisão devido aos diversos critérios de avaliação, no final de 2020.2, um questionário foi enviado aos professores, tutores e estudantes para avaliar a metodologia proposta. Embora apenas quatro professores e quatro tutores tenham retornado com seus comentários, foram obtidas 443 respostas dos estudantes, o que foi bastante satisfatório, como apresentado no artigo.

13.2.3 Artigo sobre o uso de múltiplas linguagens de programação em CS1

No artigo de [Zampirolli, Teubl, Kobayashi, Neves, Rozante e Batista \(2021\)](#), foi aplicado o mesmo método apresentado anteriormente, na disciplina de Processamento da Informação (ou Introdução à Programação, IP, ou também CS1) da UFABC, durante a pandemia. A principal contribuição deste novo artigo foi disponibilizar material didático e avaliativo em seis linguagens de programação para correção automática utilizando a integração MCTest+Moodle+VPL.

Método

O método apresentado no artigo permite que cada estudante escolha sua linguagem de programação preferida para aprender lógica de programação. O material didático é disponibilizado em várias linguagens por meio de *notebooks* no Colab. Além disso, os estudantes podem optar por diferentes linguagens de programação para praticar com exercícios e enviar suas soluções como códigos de programação. As soluções são individualizadas devido ao uso de questões paramétricas geradas com a integração MCTest+Moodle+VPL.

A geração automática do material didático para seis linguagens de programação é detalhada no artigo, utilizando o conteúdo do livro “Processando a Informação” de [Neves e Zampirolli \(2017\)](#). O livro possui sete capítulos que abordam conceitos fundamentais, estruturação de código, condicionais, laços, vetores, matrizes e tópicos avançados. Para cada um dos seis primeiros capítulos, foram criados arquivos no formato `ipynb` contendo as notas de aula e exemplos em várias linguagens. Além disso, foram criados *notebooks* para as listas de exercícios práticos, todos armazenados na pasta “all” (ver Figura 13.1) do Google Drive disponibilizado pela Editora da UFABC, no seguinte endereço:

<https://editora.ufabc.edu.br/matematica-e-ciencias-da-computacao/58-processando-a-informacao>

Em cada arquivo `ipynb` da pasta “all” da Figura 13.1, a primeira linha de cada célula, seja de texto ou código, pode conter a(s) linguagem(ns) de programação escolhida(s) para o conteúdo daquela célula. Isso segue a sintaxe `# [tipo1]# [tipo2]#...#[tipon]#`, em que `tipoi` é a extensão do arquivo do código na linguagem escolhida.

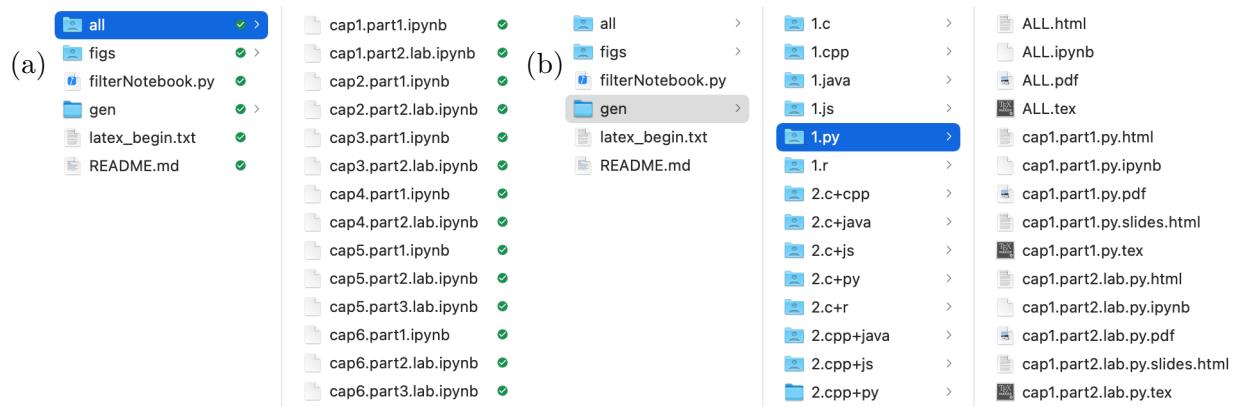


Figura 13.1: (a) estrutura de arquivos e pastas; (b) recorte da estrutura de arquivos e pastas gerados automaticamente, detalhando a pasta 1.py para Python. Fonte: [Zampirolli, Teubl, Kobayashi, Neves, Rozante e Batista \(2021\)](#).

O filtro `filterNotebook.py`, disponível em github.com/fzampirolli/filterNotebook, permite filtrar *notebooks* com base na linguagem de programação e convertê-los para os formatos HTML, LaTeX e PDF. Essa ferramenta suporta várias linguagens de programação, e o número de combinações possíveis depende do número de linguagens definidas.

Experimentos

O método foi validado em 5 turmas remotas, totalizando 223 estudantes, com uma taxa média de aprovação de 90%. Ele se mostrou facilmente adaptável e oferece flexibilidade aos estudantes, possibilitando uma abordagem mais personalizada no processo de aprendizagem de programação.

Os estudantes da disciplina CS1 foram divididos em dois grupos. O Grupo 1 teve aulas síncronas (teóricas e práticas) e os estudantes foram obrigados a gravar vídeos explicativos para as atividades. O Grupo 2 teve aulas assíncronas e os estudantes não foram obrigados a gravar vídeos explicativos. Os resultados do questionário aplicado aos estudantes mostraram que o Grupo 1 teve uma avaliação mais positiva da disciplina do que o Grupo 2. Em particular, os estudantes do Grupo 1 ficaram mais satisfeitos com as aulas síncronas, com o material didático e com as avaliações. A análise estatística também mostrou que o método aplicado ao Grupo 1 foi estatisticamente melhor do que o método aplicado ao Grupo 2. Os estudantes dos dois grupos concordaram que a disciplina foi importante para aprender sobre programação e o conhecimento de várias linguagens de programação é útil para a carreira profissional.

Os resultados deste estudo sugerem que as aulas síncronas podem ser uma forma eficaz de ensinar programação. No entanto, é importante que as aulas síncronas sejam bem planejadas e os estudantes sejam motivados a participar.

13.2.4 Artigo aplicando a integração MCTest+Moodle+VPL em CS1

O artigo de [Zampirolli, Josko, Venero, Kobayashi, Silva, Goya e Savegnago \(2021\)](#), publicado na revista *Computer Applications in Engineering Education* e resumido nesta seção, descreve um

experimento significativo envolvendo a implementação do método apresentado na Seção 13.2.1 – Artigo com a primeira integração entre MCTest+Moodle+VPL (ZAMPIROLI, 2020). Esse estudo complementa um artigo anterior publicado na mesma revista, resumido na Seção 12.1.2 – Artigo comparando as modalidades semipresencial e presencial CS1 com base na utilidade do MCTest-4 (ZAMPIROLI, 2018).

Nesse novo estudo, a integração MCTest+Moodle+VPL foi incorporada como parte do processo de ensino e avaliação na disciplina CS1. Inicialmente, tentou-se publicar tanto o método quanto o experimento em um único artigo nesta revista. No entanto, os revisores consideraram o conteúdo muito denso, o que levou à separação em dois artigos distintos: Zampirolli, Pisani, Josko, Kobayashi, Silva, Goya e Savegnago (2020) e Zampirolli, Josko, Venero, Kobayashi, Silva, Goya e Savegnago (2021).

Método

A principal característica da abordagem consiste na implementação da avaliação automatizada (AA) em uma disciplina CS1 com várias turmas. Esse método engloba a formulação de questões individuais para cada estudante e a disponibilização de avaliação e *feedback* automáticos.

Para a realização da avaliação unificada, foram empregadas questões paramétricas e respostas (nos casos de QMs) através do MCTest. Essa estratégia possibilitou a combinação de distintos tipos de questões para gerar um exame personalizado para cada estudante. Já para as QTs, foi utilizada a integração MCTest+Moodle+VPL, na qual os estudantes tiveram que submeter EPs em atividades VPL do Moodle.

Importante destacar que todo o processo avaliativo utilizado neste artigo foi previamente detalhado e exemplificado em capítulos anteriores deste livro. Além disso, é possível criar várias turmas pelo coordenador de disciplina, conforme detalhado na Seção 3.2.2 – Criar várias turmas com arquivo CSV. Da mesma forma, é explicado no Capítulo 6 – Visão geral dos exames como enviar um exame também para várias turmas.

Experimentos

A principal característica do experimento realizado é a capacidade de aplicar AA e exames unificados parametrizados em diferentes linguagens de programação, utilizando as ferramentas de código aberto MCTest e Moodle (com o plugin VPL). No primeiro trimestre de 2019, 19 das 44 turmas (presenciais) adotaram a abordagem, alcançando uma taxa de aprovação mais alta (67,5%) em comparação com as turmas que não utilizaram a solução de AA (59,1%), mantendo uma dispersão padrão semelhante. Resultados preliminares também indicaram uma forte correlação linear entre a taxa de aprovação e a média das notas nas AA. Nas turmas de aprendizagem mista (*blended learning*), duas turmas utilizaram o método nos exames unificados, com uma taxa de aprovação de 70,4%. Esses resultados corroboram estudos anteriores, indicando que a avaliação contínua combinada com *feedback* imediato pode contribuir para o processo de aprendizagem dos estudantes.

13.2.5 Artigo explorando a integração de jogos no ensino de CS0

O artigo de [Josko e Zampirolli \(2023\)](#) discute uma intervenção pedagógica que emprega elementos de gamificação para a introdução de conceitos de lógica de programação em estudantes. No estudo, os autores realizaram um estudo de caso envolvendo 48 estudantes matriculados no curso introdutório de programação (CS0) intitulado Bases Computacionais da Ciência, parte do programa de Bacharelado em Ciência e Tecnologia da UFABC. Importante ressaltar que o curso foi ministrado *online* devido à pandemia.

Método

A intervenção pedagógica consistiu em construir gradualmente um jogo simples utilizando a biblioteca Pygame ([pygame.org](#)) do Python no Google Colab ([colab.google](#)). Conceitos como instruções sequenciais, bibliotecas, estruturas de decisão e de repetição, além de simulações, foram introduzidos à medida que novas funcionalidades eram incluídas no jogo. Além disso, foram utilizadas avaliações automáticas e *feedback* para os estudantes, conforme relatado em outro artigo descrito na Seção [13.2.2 – Artigo aplicando a integração MCTest+Moodle+VPL em CS0](#).

Experimentos

Os autores aplicaram um questionário para capturar as perspectivas dos estudantes sobre a abordagem. As análises sugeriram que os elementos de gamificação melhoraram a motivação em relação aos tópicos do curso. Adicionalmente, a taxa de reprovação do grupo estudado foi menor do que a média de outras turmas no período da pandemia.

Assim, a intervenção recebeu *feedback* positivo dos estudantes. Entretanto, os autores afirmam que estudos controlados com maiores amostras são necessários para avaliar plenamente a efetividade da abordagem.

13.3 Aplicação do MCTest-5 com Moodle+VPL no Ensino de PDI

Ao contrário dos artigos apresentados na Seção [13.2](#), que descrevem experimentos em cursos introdutórios de lógica de programação (CS0 e CS1), esta seção relata a aplicação dessa integração no curso de Processamento Digital de Imagens (PDI).

O trabalho “A Practical Digital Image Processing Course with `morph.py`” de [Zampirolli, Josko, Teubl e Kurashima \(2024\)](#) foi publicado no Simpósio Brasileiro de Educação em Computação e **recebeu o primeiro lugar na categoria de Recursos e Ambientes Educacionais** entre os trabalhos apresentados no evento.

Isso marca o encerramento de um ciclo virtuoso iniciado em 2012, com a primeira publicação de [Zampirolli, Quilici-Gonzalez e Neves \(2013\)](#) relacionada ao PDI na primeira versão do MCTest, ainda em Matlab, introduzido no primeiro capítulo deste livro, na Seção [1.2 – Breve histórico do MCTest](#).

O ensino de PDI é desafiador devido à sua complexidade matemática e algorítmica. Este artigo apresenta um curso prático que utiliza a biblioteca Python `morph.py` para apoiar o ensino. O curso interativo emprega exemplos ilustrativos e exercícios práticos para facilitar a compreensão de conceitos e operadores fundamentais de PDI, desde noções básicas até tópicos avançados. Um estudo de caso exploratório com um grupo de 15 participantes confirmou a eficácia do método de ensino baseado na biblioteca `morph.py`, abordando as dificuldades de ensinar PDI a iniciantes.

13.3.1 Método

Nesta seção, é resumido o método apresentado no artigo, contextualizando o curso de PDI e detalhando a intervenção pedagógica, com ênfase na utilização da biblioteca `morph.py` como um fator motivacional. Por fim, foi discutido os ambientes de desenvolvimento.

Contexto

O curso de PDI é uma disciplina eletiva oferecida a estudantes de diversos programas na universidade, como o Bacharelado em Ciência da Computação e vários cursos de Engenharia. Com duração de 12 semanas e quatro horas de aula por semana, o curso é ministrado por meio de aulas laboratoriais síncronas, nas quais os professores apresentam Colabs contendo conceitos, exemplos e exercícios no Sistema de Gerenciamento de Aprendizado Moodle. Em 2023, o curso foi oferecido para 25 estudantes de fevereiro a maio, incluindo aulas gravadas produzidas durante a pandemia de Covid-19.

Intervenção Pedagógica

O curso aborda uma ampla gama de tópicos em PDI, com as primeiras seis semanas dedicadas a equipar os estudantes com habilidades essenciais para desenvolver operadores de PDI, como limiares, histogramas, convolução, erosão, dilatação, filtros, *watershed*, rotulagem e transformações de distância, seguindo o livro-texto de Gonzalez e Woods ([GONZALEZ; WOODS, 2009](#)). Na sétima semana, os estudantes fazem o Exame 1, enquanto as semanas subsequentes (oito a dez) focam na aplicação desses operadores de PDI para resolver problemas reais de visão computacional. Em suma, a primeira parte do curso visa fornecer aos estudantes as habilidades necessárias para desenvolver operadores de PDI, enquanto a segunda parte constrói sobre essa base, aplicando esses operadores para resolver vários problemas de visão computacional. A semana onze é reservada para revisão, e o curso é concluído com um exame final na semana doze.

A avaliação do curso consiste em quatro tarefas individualizadas e parametrizadas (15% da nota final), relacionadas ao último tópico abordado em sala de aula. Essas tarefas são fornecidas aos emails dos estudantes antes de cada aula e são avaliadas automaticamente utilizando a integração do MCTest, Moodle e VPL ([ZAMPIROLI, 2020](#); [ZAMPIROLI, 2021](#)). Além dessas tarefas, o Exame 1 (30% da nota final) consiste em exercícios semelhantes. Para o projeto individual final (15% da nota final), os estudantes devem desenvolver um Colab para resolver um problema de PDI. O exame

final (40% da nota final) segue uma estrutura similar ao projeto. Tanto o primeiro quanto o último exame devem ser concluídos dentro do período de duas horas de aula.

Ambientes de Desenvolvimento

A Figura 13.2 fornece uma visão geral da intervenção pedagógica empregada no curso de PDI, utilizando a biblioteca `morph.py`, que oferece 71 métodos, incluindo operações (como mínimo, máximo e negação) e operadores (como erosões e dilatações). Embora o curso ia870p3¹ (LOTUFO, 2019) tenha abordado alguns algoritmos da caixa de ferramentas `mmorph` (DOUGHERTY; LOTUFO, 2003), nem esse curso nem o curso apresentado no artigo cobriram toda a gama de funcionalidades oferecidas pela caixa de ferramentas.

O material do curso consistiu em seis *notebooks* conceituais durante as seis primeiras semanas e oito documentos focados em operadores morfológicos. Esses recursos foram utilizados até o Exame 1, marcando um marco significativo na progressão do curso. Após o Exame 1, o curso incorporou um conjunto de 27 demonstrações de *notebooks* que combinavam teoria com prática.

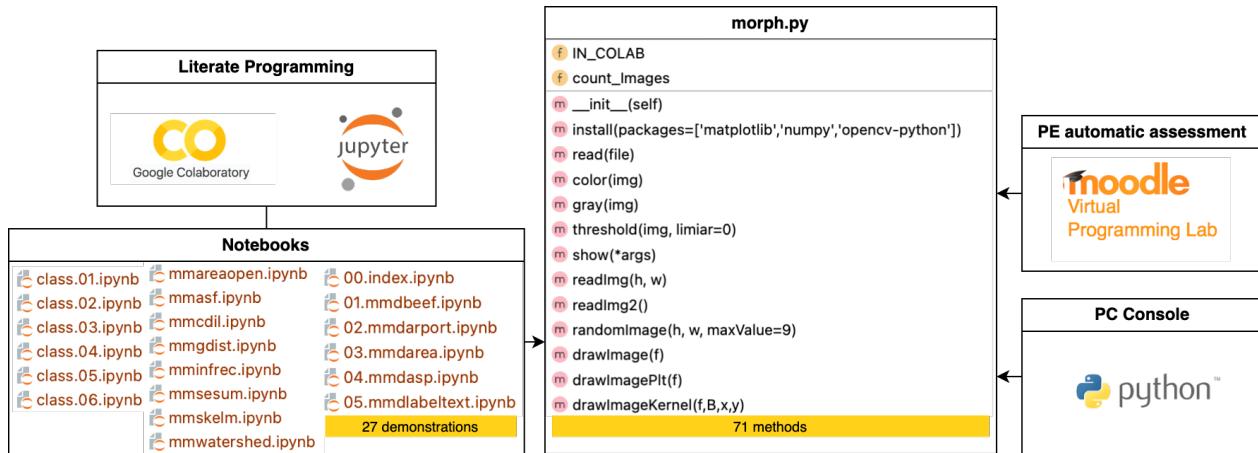


Figura 13.2: Visão geral da intervenção pedagógica empregada no curso de PDI. Fonte: (ZAMPIROLI, 2024).

Agora, o sistema de VPL fornece uma abordagem automatizada para avaliar as soluções dos estudantes. Por meio da biblioteca `morph.py`, é possível verificar diretamente as soluções dos estudantes e compará-las com as respostas corretas, agilizando o processo de avaliação. Além disso, a combinação de *notebooks* conceituais com *notebooks* de demonstração permite aos estudantes uma compreensão completa dos conceitos, seguida pela aplicação prática desses conceitos em exemplos específicos. Essa abordagem integrada ajuda a fortalecer a compreensão dos estudantes e a solidificar suas habilidades práticas em PDI.

13.3.2 Experimentos

Os autores realizaram uma pesquisa voluntária para coletar dados dos estudantes em nosso curso de PDI. A pesquisa foi administrada após o exame final e incluiu 5 perguntas baseadas em

¹Disponível em <http://github.com/robertoalotufo/ia870p3>.

Likert para obter *feedback* sobre o conteúdo e as estratégias do curso.

Dos estudantes que concluíram o curso, foram 15 respostas (aproximadamente 60%). Os autores analisaram essas respostas e descobriram que duas perguntas se assemelham a uma distribuição normal. Portanto, foi utilizado um método paramétrico (*teste t*) para essas duas perguntas e um método não paramétrico equivalente (*Wilcoxon Signed-Rank*) para as restantes. Além disso, foi aplicado o Alpha de Cronbach para avaliar a consistência interna dos dados Likert, obtendo um valor de 0,90. Finalmente, foi testada a hipótese de que a biblioteca `morph.py` teve um efeito neutro na aprendizagem dos estudantes.

Os resultados mostram diferenças significativas entre todas as perguntas e a hipótese nula. Por exemplo, a percepção positiva dos estudantes sobre a biblioteca indica que ela facilitou a exploração dos diferentes algoritmos pré-implementados sem a necessidade de implementá-los do zero. Como resultado, eles puderam se concentrar mais em entender os conceitos subjacentes e as técnicas por trás dos algoritmos, reduzindo a curva de aprendizado inicial. Finalmente, dos 25 estudantes que concluíram o curso, apenas 4 não obtiveram aprovação, resultando em uma taxa de aprovação de 84%.

Desafios à validade incluem a natureza eletiva do curso de PDI na universidade, o que pode afetar a representatividade dos resultados. Além disso, a realização de experimentos envolvendo grupos de teste e controle requer aprovação do comitê de ética e apresenta desafios logísticos. Assim, este estudo abre caminho para pesquisas futuras explorarem novas formas de experimentação.

13.4 MCTest-5 com Feedback Inteligente baseado em LLMs

Esta seção detalha o estudo publicado em [Zampirolli, Teubl, Pisani e Silva \(2026\)](#) no periódico *Computer Applications in Engineering Education*. O trabalho investiga como modelos de linguagem de larga escala (LLMs) podem ser integrados ao ecossistema MCTest+Moodle+VPL para fornecer feedback pedagógico imediato e personalizado em cursos de programação.

O desafio central abordado é a "lacuna de feedback" em turmas grandes, onde professores não conseguem oferecer orientações detalhadas para cada erro de lógica dos estudantes. O artigo propõe e valida uma arquitetura onde o código submetido pelo aluno é analisado por IA para gerar não apenas um veredito de "certo ou errado", mas uma explicação qualitativa que auxilie na correção autônoma do erro.

13.4.1 Método

O método baseia-se em uma abordagem experimental quantitativa e qualitativa, utilizando uma arquitetura de software projetada para integrar a execução de testes unitários com chamadas de API de diferentes provedores de LLM.

Arquitetura do Sistema

A arquitetura do feedback inteligente é composta por quatro camadas principais que operam de forma sequencial após a submissão do estudante no Moodle/VPL:

13.5. CONSIDERAÇÕES FINAIS

1. **Camada de Execução (VPL):** O código do estudante é executado contra um conjunto de casos de teste (testes unitários). Se o código falha, o sistema captura as mensagens de erro do compilador ou interpretador e os resultados dos testes que falharam;
2. **Camada de Contextualização (Prompt Engineering):** O sistema recupera o código submetido e os logs de erro. Esses dados são estruturados em um *prompt* técnico projetado para evitar que a IA forneça a resposta pronta, focando em "dicas" e explicações conceituais;
3. **Camada de Processamento (LLM Gateway):** O *prompt* é enviado para um dos sete modelos LLMs. A arquitetura permite avaliar a precisão técnica e o tom pedagógico de cada modelo;
4. **Camada de Entrega (Interface Moodle):** O feedback gerado pela IA é retornado ao console do VPL, aparecendo imediatamente para o estudante após a submissão, permitindo ciclos rápidos de tentativa e erro.

Procedimento Experimental

O experimento foi conduzido em cursos de Introdução à Programação (CS1), analisando milhares de submissões. Os autores avaliaram a qualidade do feedback retornado pelaIA e a eficácia no tempo de resposta que os alunos levam para corrigir erros de lógica complexos.

13.4.2 Experimentos e Resultados

Os resultados indicam que a integração de LLMs via VPL é eficaz para erros de sintaxe e lógica. A análise quantitativa, baseada em testes de Wilcoxon e no tamanho do efeito (d de Cohen), demonstrou uma percepção altamente positiva dos estudantes quanto à clareza e utilidade do feedback. O estudo observou que o sistema atuou como um "tutor socrático", reduzindo a dependência do aluno em relação ao professor para dúvidas procedurais básicas.

13.5 Considerações finais

Neste capítulo, foram apresentados vários artigos que relatam experiências importantes com a integração do MCTest com o Moodle e o *plugin* VPL. Essas integrações permitiram a avaliação automatizada de EPs em diversas disciplinas, com destaque para:

- Criação de QTs paramétricas e de QMs corrigidas manualmente e automaticamente;
- Exportação de questões do MCTest para o banco de questões do Moodle, superando limitações;
- Geração de listas unificadas de EPs paramétricos com casos de teste automatizados;
- Disponibilização de material didático e avaliativo em várias linguagens de programação;
- Uso de avaliação contínua e *feedback* imediato, resultando em maior taxa de aprovação.

Os artigos apresentam a evolução e aprimoramentos dessas integrações ao longo do tempo. Além disso, foi observado que as turmas que utilizaram os métodos propostos tiveram um desempenho melhor. No entanto, é difícil afirmar que essa foi a única razão para o melhor desempenho.

Portanto, é possível concluir que as abordagens propostas neste capítulo se mostraram eficazes e práticas, especialmente no contexto de ensino remoto durante a pandemia COVID-19. Essas integrações abrem novas possibilidades para oferecer avaliações personalizadas, automatizadas e flexíveis em diversas disciplinas, como as de programação.

Capítulo 14

Conclusões

Conteúdo

14.1 Novas possibilidades de aprimoramento	252
14.2 Novas publicações	253
14.3 Novas edições deste livro	254

O MCTest, sistema apresentado neste livro, proporciona aos professores uma ferramenta poderosa para aprimorar a avaliação das habilidades dos estudantes. Ao utilizar o MCTest, os docentes conseguem reduzir o esforço de criação e correção de exames e exercícios envolvendo questões de múltipla escolha (QMs) ou dissertativas (QTs). As QTs podem conter exercícios de programação (EPs), permitindo que os estudantes submetam os códigos em atividades VPL do Moodle.

Ao longo do tempo, o MCTest evoluiu por meio de suas diversas versões, incorporando novas tecnologias, metodologias e objetivos. Uma contribuição significativa do MCTest é sua habilidade em criar questões parametrizadas, que combinam descrições em L^AT_EX com códigos em Python.

O MCTest foi empregado com sucesso em processos seletivos da Especialização em Tecnologias e Sistemas de Informação e na Escola Preparatória da UFABC, bem como em exames de disciplinas ministradas na graduação e na pós-graduação. Essa eficácia foi comprovada na avaliação de disciplinas de computação e outras áreas.

A natureza de código aberto do MCTest, disponível no [GitHub](#), permite que ele seja utilizado e modificado sob a licença [AGPL3](#). No entanto, cada instituição deve definir seus próprios critérios em relação aos direitos autorais do conteúdo disponibilizado no MCTest.

Este livro foi organizado em quatro partes principais, focadas em diferentes aspectos do MCTest, abrangendo desde os fundamentos até experimentos. Na Parte I – Fundamentos, são abordados a introdução, componentes básicos e funcionalidades essenciais do MCTest. A Parte II – Questões no MCTest, trata dos diferentes tipos de questões disponíveis, como QMs e QTs com código. A Parte III – Exames, aborda o processo de criação e gerenciamento de exames no MCTest, incluindo

a integração com Moodle e VPL. Por fim, a Parte IV – Experimentos de Uso do MCTest, apresenta estudos de caso e exemplos práticos do uso do sistema em ambientes educacionais, publicados em artigos científicos, envolvendo quadro de respostas (QR), QMs e/ou QTs e a integração com Moodle e VPL. Essa estrutura possibilita uma abordagem abrangente sobre o sistema, atendendo a diferentes necessidades dos usuários.

Durante a jornada desta obra, foram exploradas as funcionalidades do MCTest, apresentando-se suas principais características e recursos. Espera-se que esta obra tenha sido proveitosa aos leitores, fornecendo uma visão detalhada de como o MCTest pode ser utilizado de maneira eficaz em contextos educacionais.

O MCTest continua em constante evolução e aprimoramento, incentivando os usuários a contribuírem com suas experiências e *feedbacks* para torná-lo ainda mais eficiente e adequado às demandas educacionais em evolução.

14.1 Novas possibilidades de aprimoramento

O MCTest oferece oportunidades para o desenvolvimento de novas funcionalidades, por meio de orientações de trabalho de conclusão de curso e mestrado. Alguns trabalhos em andamento de melhorias incluem:

- Agrupamento de questões utilizando processamento de linguagem natural: possibilitando a organização e categorização mais eficiente das questões com base em técnicas de processamento de linguagem natural, tornando a busca e seleção de questões mais precisa e relevante;
- Melhorias no empacotamento do MCTest para futuras implantações: visando facilitar a instalação e utilização do sistema em diferentes ambientes, aprimorando sua usabilidade para os usuários;
- Implementação de aprendizado adaptável: utilizando dados de desempenho dos estudantes para criar avaliações personalizadas no MCTest. Isso permitiria que o sistema se adapte ao progresso individual de cada estudante, proporcionando uma experiência de aprendizado mais eficaz e direcionada, de acordo com suas necessidades específicas;
- Facilitação da correção e automação de exames: buscando uma integração mais estreita entre o MCTest e a plataforma Moodle. Essa integração permitiria que os resultados das avaliações fossem automaticamente registrados no ambiente do Moodle, simplificando o processo de correção e oferecendo um *feedback* imediato e detalhado aos estudantes sobre seu desempenho;
- Desenvolvimento de um sistema distribuído em Python para geração e correção segura de exames, com a capacidade de executar questões paramétricas em servidores externos, visando aumentar a segurança e reduzir a vulnerabilidade do servidor do MCTest.

Essas potenciais melhorias agregam ainda mais utilidade e eficiência ao MCTest. Além disso, proporcionam oportunidades de pesquisa e desenvolvimento para estudantes interessados em contri-

14.2. NOVAS PUBLICAÇÕES

buir para o aprimoramento contínuo do sistema, promovendo sua evolução para atender às necessidades educacionais em constante mudança.

Além das mencionadas, existem diversas outras possibilidades de melhorias para o sistema. Algumas delas incluem:

- Implementar a funcionalidade de salvar e recuperar o banco de questões de um professor em formato JSON, permitindo uma gestão mais eficiente e organizada das questões;
- Utilizar recursos de segurança para proteger as comunicações entre o cliente e o servidor, como HTTPS, evitando potenciais ataques de interceptação de dados;
- Implementar a validação de e-mails dos usuários para garantir que apenas usuários autorizados possam acessar determinadas funcionalidades do sistema;
- Ampliar a variedade de tipos de questões, além das QMs e QTs, para possibilitar a criação de avaliações mais diversificadas e ricas em conteúdo, conforme apresentado nos artigos de [Teubl, Batista e Zampirolli \(2021\)](#) e [Teubl, Batista e Zampirolli \(2022\)](#);
- Aprimorar a interface do usuário, tornando-a mais intuitiva e amigável, a fim de facilitar a navegação e o uso do sistema;
- Implementar as avaliações das habilidades e competências dos estudantes em atividades envolvendo EPs em atividades VPL do Moodle, além do teste de mesa introduzido no artigo de [Teubl e Zampirolli \(2023\)](#)¹;
- Incorporar funcionalidades de análise de dados e estatísticas para os educadores poderem obter visões sobre o progresso dos estudantes e a eficácia das questões propostas, por exemplo, utilizando a Teoria de Resposta ao Item.

Essas melhorias são apenas algumas das possibilidades que podem tornar o sistema mais completo, seguro e eficiente, atendendo melhor às necessidades dos usuários e proporcionando uma experiência de aprendizado mais enriquecedora.

14.2 Novas publicações

Atualmente, estão em fase de análise mais novos artigos diretamente relacionados ao MCTest. Um deles trata do uso bem-sucedido do MCTest na disciplina de Processamento Digital de Imagens. Esse estudo examina como o MCTest pode ser aplicado de maneira eficaz para avaliar os conhecimentos adquiridos pelos estudantes nessa disciplina específica, contribuindo para uma melhor compreensão dos resultados e progresso dos estudantes.

¹Optou-se por não incluir o artigo de [Teubl e Zampirolli \(2023\)](#) no capítulo anterior de experimentos que utilizou a integração MCTest+Moodle+VPL. Essa decisão decorre do fato de que esta versão deverá ser incorporada em trabalho futuro, que abrangerá a avaliação de uma ampla gama de habilidades e competências dos estudantes, incluindo a habilidade de solucionar testes de mesa no VPL.

Além desse, a intenção é escrever mais artigos à medida que novas funcionalidades relevantes forem incorporadas ao MCTest. A evolução contínua do sistema possibilita a exploração de diversas áreas de pesquisa e a identificação de novas oportunidades para aprimorar o processo de avaliação das habilidades dos estudantes.

Outros pesquisadores da área também têm a oportunidade de adotar este sistema de código aberto para implementar suas próprias contribuições, desde que as compartilhem publicamente, permitindo que outros também se beneficiem das melhorias realizadas.

Com essas publicações, busca-se disseminar o conhecimento e compartilhar experiências sobre o uso do MCTest em diferentes contextos acadêmicos. A produção científica nessa área é fundamental para promover a eficácia do sistema e incentivar o desenvolvimento de novas práticas pedagógicas e tecnológicas no ensino em geral. As novas publicações contribuirão para a expansão do conhecimento sobre o MCTest e sua aplicabilidade em diversas disciplinas e cenários educacionais.

14.3 Novas edições deste livro

Como perspectiva para futuras edições deste livro, estão previstas inclusões de novas partes que expandam o escopo das informações apresentadas. Dentre as possibilidades, destacam-se as seguintes partes:

- **Integração com o Moodle:** Uma parte dedicada à integração entre o MCTest e o Moodle, uma plataforma popular de aprendizado a distância, permitiria explorar diversos tipos de questões. Seriam abordadas questões clássicas, questões de código (VPL) e a sinergia do MCTest com o Moodle e o VPL, proporcionando ao leitor uma visão mais abrangente das possibilidades de integração entre essas ferramentas;
- **Tópicos Avançados:** Nessa seção, os aspectos técnicos do MCTest seriam aprofundados, abordando temas como bibliotecas e instalações, arquitetura de software, gerenciamento de banco de dados, visão computacional e segurança. Essa abordagem detalhada permitiria ao leitor compreender melhor os fundamentos e desafios técnicos do sistema.

Essas adições seriam de grande valia para enriquecer ainda mais o conteúdo do livro, fornecendo aos leitores informações detalhadas sobre a integração do MCTest com outras plataformas e aspectos técnicos avançados. Dessa forma, o livro se consolidaria como uma referência abrangente e atualizada sobre essa importante ferramenta de avaliação de habilidades dos estudantes. A ampliação do escopo do livro permitiria acompanhar o progresso contínuo do MCTest e atender às demandas dos leitores por informações relevantes e atualizadas sobre a área.

Apêndice A

Acompanhamento de acesso de estudantes no Moodle

Conteúdo

A.1	Contexto da oferta	256
A.2	Intervenção pedagógica	256
A.3	O Serviço LabMoodle	257
A.4	Resultados e discussões	259
A.4.1	Arquivo <code>data.json</code>	261
A.4.2	Arquivo <code>dias_notas_*.csv</code>	262
A.4.3	Arquivo <code>faltas_notas_*.csv</code>	262
A.4.4	Arquivo <code>presenca_notas_*.csv</code>	263
A.4.5	Acessos no Moodle	263
A.5	Acessos às atividades do Moodle	263
A.6	Tabelas comparativas - turma A vs. turma B	264
A.6.1	Comparações antes da prova de recuperação	264
A.6.2	Comparações após a prova de recuperação - resultado final	264
A.7	Histórico de CS0 e CS1 na UFABC	265
A.8	Considerações finais	267

Este apêndice relata uma experiência realizada na disciplina de Processamento da Informação (CS1) na UFABC no primeiro quadrimestre de 2024, em duas turmas. A motivação foi substituir a lista de presença por um processo automático que verifica o acesso dos estudantes no ambiente Moodle durante as aulas realizadas no laboratório. Para isso, foi desenvolvido o serviço chamado **LabMoodle**, que acompanha as atividades dos estudantes cadastrados em uma disciplina no Sistema de Gerenciamento de Aprendizagem Moodle, disponibilizado em [\(http://mctest.ufabc.edu.br:8000/\)](http://mctest.ufabc.edu.br:8000/)

[LabMoodle](#)).

Apesar de este conteúdo do apêndice não estar diretamente relacionado ao MCTest, há uma relação indireta, pois foram oferecidos 80 Exercícios de Programação (EPs) estáticos com correção automática pelo VPL (*Virtual Programming Lab for Moodle*) ([PINO, 2012](#)). Os exames foram gerados pelo MCTest, com correção automática também utilizando o VPL, conforme relatado no Capítulo [10 – Exames com MCTest+Moodle+VPL](#).

A.1 Contexto da oferta

A disciplina de Processamento da Informação é obrigatória para estudantes em diversos programas de nossa universidade, com uma média de 2000 matrículas por ano. A partir de 2023, a disciplina é oferecida integralmente por um único professor em um laboratório de informática. O curso tem duração de 12 semanas, com quatro horas de aula por semana. No início de 2024, foi realizado um experimento em duas turmas, A e B, com 38 e 35 estudantes matriculados, respectivamente, de fevereiro a maio. O professor ministrou aulas laboratoriais síncronas aos estudantes por meio de um projetor, apresentando [Google Colabs](#) contendo conceitos, exemplos e exercícios, conforme relatado em [Zampirolli, Teubl, Kobayashi, Neves, Rozante e Batista \(2021\)](#). Além disso, o professor disponibilizou 80 exercícios com correção automática detalhados a seguir. Também foram disponibilizadas aulas gravadas, criadas durante a pandemia de Covid-19.

A.2 Intervenção pedagógica

A disciplina aborda os conceitos fundamentais de Lógica de Programação, tal como apresentados no livro-texto de [Neves e Zampirolli \(2017\)](#). A primeira parte do curso abrange os quatro primeiros capítulos: introdução e programas sequenciais, organização de código, condicionais e repetição. Na quinta semana, foi realizada uma revisão com um simulado utilizando uma prova antiga e o Exame 1. Nas semanas seis a dez, os capítulos cinco e seis do livro foram abordados, tratando de vetores e matrizes, respectivamente. A semana onze foi dedicada à revisão, também com um simulado utilizando uma prova antiga, e em seguida um exame final, seguido pelo exame de recuperação na semana doze. Todos esses exames são elaborados no MCTest e compostos por duas questões cada, com correção automática utilizando o *plugin* VPL do Moodle.

A estratégia de ensino foi implementada com uma abordagem conceitual no Google Colab no início de cada aula, seguida pela apresentação dos Exercícios Programados (EPs), permitindo tempo para os estudantes tentarem resolvê-los. O professor então discutiu algumas soluções dos estudantes (com autorização prévia). Foram distribuídos 80 EPs ao longo dos seis capítulos do livro, com ênfase em vetores (25 EPs) e matrizes (23 EPs). Além disso, foram realizados seis testes adaptativos, correspondentes aos capítulos do livro, contendo cinco questões de múltipla escolha em papel, com correção automática, também administrados no MCTest, conforme detalhado na Seção [8.6 – Exames adaptativos](#). Os EPs não foram pontuados, enquanto os testes concederam um bônus de 5% na nota final pela participação, independentemente do desempenho obtido.

A estratégia de avaliação para este curso consistiu no Exame 1 (40% da nota final), contendo exercícios semelhantes aos EPs. O exame final (60% da nota final) abrangeu os tópicos da segunda parte da disciplina. Ambos os exames, inicial e final, tiveram duração de duas horas e foram realizados utilizando o navegador *Safe Exam Browser* ([SEB](#)), sem consulta externa. Cada questão do exame foi elaborada de forma parametrizada, com 10 casos de testes gerados automaticamente pelo MCTest. Foram geradas 40 variações para cada exame, garantindo que cada estudante recebesse uma versão única, conforme detalhado no Capítulo 10 – [Exames com MCTest+Moodle+VPL](#).

A.3 O Serviço LabMoodle

O serviço disponibilizado em [\(<http://mctest.ufabc.edu.br:8000/LabMoodle>\)](http://mctest.ufabc.edu.br:8000/LabMoodle) para acompanhamento das atividades realizadas pelos estudantes no Moodle foi elaborado em PHP para a interface do professor, enquanto a lógica no servidor para a geração dos relatórios foi realizada na linguagem Python, utilizando principalmente as bibliotecas [Pandas](#) e [Seaborn](#).

O acompanhamento das atividades dos estudantes no Moodle é facilitado por meio de arquivos gerados nesse serviço. Um exemplo pode ser visto na pasta “[report](#)”. Para obter esses arquivos, primeiramente é necessário fazer *login* no Moodle e baixar o *log* da disciplina em formato CSV. Em seguida, no [SIGAA](#), acessa-se o portal docente, seleciona-se a turma desejada e, dentro do menu de estudantes, é possível lançar notas e faltas e exportar uma planilha em formato XLS contendo a lista dos estudantes, ver Figura A.1.

Para configurar corretamente as informações da disciplina, é necessário definir a data de início do curso como 05/02/2024 e a data de término como 07/05/2024. Além disso, é preciso filtrar o *log* do Moodle pelo componente, ou deixar o valor padrão “Tudo”, indicando que não será realizado filtro do componente. Estabelecer o número mínimo de faltas após o término do curso como 14. Para os estudantes reprovados por falta, com o conceito “O”, deve ser marcado o *checkbox* correspondente. Existe também um *checkbox* para ser marcado se desejar omitir dados de estudantes. Finalmente, os dados de cada aula devem ser preenchidos: como o dia da semana, o horário de início, a duração de 2 horas, e o prefixo dos IPs do laboratório, como “172.17.14”, conforme apresentado na Figura A.2.

Finalmente, no serviço fornecido, há um formulário para realizar o *upload* de arquivos a fim de gerar relatórios. Os professores são instruídos a selecionar os arquivos nos formatos CSV e XLS obtidos nos passos anteriores. Em seguida, podem escolher o botão “Enviar” para iniciar o processo de *upload* e geração de relatório. Além disso, é fornecido um *link* para visitar o projeto no [GitHub](#), conforme apresentado na Figura A.3.

O processo de geração de relatório é demorado e depende da conexão de internet. Por exemplo, demorou 2 minutos para um arquivo de *log* de 45MB e gerou uma página, cujo início é apresentado na Figura A.4. Os dados dos estudantes foram omitidos nas figuras apresentadas neste apêndice. O restante da página apresenta seis imagens e será detalhado na próxima seção de resultados e discussões.

Conforme ilustrado na Figura A.4, após o envio bem-sucedido dos arquivos, o professor é informado sobre a conclusão do processo e são apresentadas algumas informações relevantes. Ele

Acompanhamento das Atividades dos Estudantes no Moodle

Ver modelos de arquivos gerados na pasta [report](#). Os próximos passos ensinam como obter esses arquivos. Ao final desta página é possível gerar esses arquivos e um relatório semelhante às imagens [ex01a](#), [ex01b](#), [ex01c](#) e [ex01d](#) de forma automática.

1) Log no Moodle

Arquivo: logs.CSV

Fazer download do log retirado da disciplina do Moodle.

1. Ir em Disciplina +
2. Engrenagem + Mais +
3. Relatórios (Logs) +
4. Obter estes logs +
5. Download do csv (final da página)

2) Faltas dos Estudantes no SIGAA

Arquivo: faltas.XLS

1. Acesse o [SIGAA](#) e vá para o portal docente
2. Portal docente + Escolher a Turma + Alunos + Lançar Notas +
3. Exportar planilha.

Figura A.1: Tela do serviço - Parte 1 - informações gerais.

Configurar corretamente informações da disciplina:

Data de Início do Curso: Data de Término do Curso:

Filtrar o Log do Moodle pelo componente:

Número mínimo de faltas após o término do curso:

Atribuir conceito "O" para reprovados por falta: Omitir dados de estudantes:

Aula 1

Dia: Horário: Duração:
Prefixo dos IPs do Laboratório:

Aula 2

Dia: Horário: Duração:
Prefixo dos IPs do Laboratório:

Figura A.2: Tela do serviço - Parte 2: Configurações dos relatórios e dos dados da disciplina.

Upload de arquivos para gerar os relatórios

Selecione os arquivos nos formatos CSV e XLS obtidos nos dois passos anteriores.
Em seguida, escolha enviar:

Arquivo **logs.CSV**:
 Nenhum arquivo escolhido

Arquivo **faltas.xls**:
 Nenhum arquivo escolhido

Enviar sugestões de melhoria para fzampirolli@ufabc.edu.br

Visite o projeto no GitHub: <https://github.com/fzampirolli/LabMoodle>

Figura A.3: Tela do serviço - Parte 3: envio dos arquivos e geração do relatório.

tem a opção de baixar o relatório compactado, voltar para a página anterior ou visualizar os dados gerados. Além disso, é destacado que todos os arquivos enviados ou gerados são apagados do servidor a cada 5 minutos. Os professores podem observar detalhes sobre os arquivos enviados, como seus nomes e caminhos, bem como informações sobre o filtro escolhido e o número de linhas no *log* após o filtro. Também são exibidas algumas imagens geradas, como gráficos relacionados aos acessos durante as aulas de laboratório, aos estudantes reprovados por falta e aos acessos sem filtros por nome ou RA de estudante, evidenciando a presença ou ausência nas aulas.

Destaco na Figura A.4 que nas linhas 9 e 10, dois estudantes apresentam a informação “<<< Não está no log” abaixo da coluna “Acessos”. Tal indicação sugere que esses estudantes estão registrados no sistema acadêmico SIGAA, porém não participaram de nenhuma das aulas de laboratório.

A.4 Resultados e discussões

Esta seção complementa a parte inicial apresentada na Figura A.4, descrevendo os arquivos adicionais que compõem o conjunto de informações recebidas pelo professor. Junto com o relatório inicial, o professor também recebe um arquivo compactado contendo diversos outros arquivos gerados durante o processo de análise e geração de dados. Nesta seção, serão detalhados esses arquivos.

As imagens *_Nome.png na listagem a seguir apresentam os nomes dos estudantes no eixo *x* do gráfico, enquanto as imagens *_RA.png mostram os números de matrícula (RA) dos estudantes. Caso o professor deseje enviar aos estudantes seus desempenhos em comparação com a turma, os gráficos com os RAs são mais apropriados. Sem redundância, nesta seção serão apresentados apenas os gráficos com RA contidos nos arquivos alunos_Acessos_Sem_Filtros_RA.png e alunos_Total_Presencias_RA.png, pois alunos_Total_Acessos_RA.png pode ser visualizado no

Arquivos enviados com sucesso!

[Download do relatório compactado](#)

[Voltar](#)

Todos os arquivos enviados/gerados são apagados do servidor a cada 5 minutos!

Alguns dados gerados

Arquivo 1: ./tmp/2024-04-20_20-46-11/uploads/logs_PI-2024.1-Zampirolli_20240414-2259.csv

Arquivo 2: ./tmp/2024-04-20_20-46-11/uploads/notas_BCM0505-22_TDB2BCM0505-22SA_20241-OK2.xls

Filtro escolhido: Laboratório de Programação Virtual

Número de linhas no log após filtro: 73341

Arquivo gerado: ./tmp/2024-04-20_20-46-11/report/faltas_notas_BCM0505-22_TDB2BCM0505-22SA_20241-OK2.xls.csv

ACESSOS DURANTE AS AULAS DE LABORATÓRIO:

num	RA	Nome	Email	Acessos
1	11202		@aluno.ufabc.edu.br	49 40 28 34 30 54 36 84 33 24 19 23 30 32 30 62
2	1120		@aluno.ufabc.edu.br	35 27 30 46 30 14 46 47 30 38 38 58 26 4
3	1120		@aluno.ufabc.edu.br	56 110 53 112 50 80 112 31 51 45 94 93 50
4	1120		@aluno.ufabc.edu.br	26 39 24 16 46 46 30 77 19 3 47
5	1120		@aluno.ufabc.edu.br	89 62 43 109 137 99
6	1120		@aluno.ufabc.edu.br	38 54 27 60 130 22 147 52 52 112 74 2 12 14 47
7	1120		@aluno.ufabc.edu.br	55 59 79 64 27 49 61 63 101 44 101 76 123 103 39
8	1120		@aluno.ufabc.edu.br	34 31 54 3 88 4
9	1120			<<< Não está no log
10	1120		@aluno.ufabc.edu.br	<<< Não está no log
11	1120		@aluno.ufabc.edu.br	60 50 57 15 47 66 24 11 5 7 25 50 16 59 9 2
12	1120		@aluno.ufabc.edu.br	23 169 25 50 51 62 33 103 14 40 66 122 43 93
13	1120		@aluno.ufabc.edu.br	53 82 90 151 199 168 97 142 68 59 1 152 136 57 45 70
14	1120		@aluno.ufabc.edu.br	41 38 52 25 28 80 49 35 30 59 37 31 29 34 47 55 42 17
15	1120		@aluno.ufabc.edu.br	53 31 34 63 13 32 44 31 28 66 40
16	1120		@aluno.ufabc.edu.br	52 10 59 38 35 127 24 33 17 31 16 18 25 19 30 41 16
17	1120		@aluno.ufabc.edu.br	47 38 82 62 46 28 68 70 40 46 65 46 63 56 70 95 48
18	1120		@aluno.ufabc.edu.br	78 22 205 77 73 104 32 9 41 37 55 67 37 27 32 37 11 17
19	1120		@aluno.ufabc.edu.br	62 30 51 38 54 30 54 84 55 53 101 47 40 38 11
20	1120		@aluno.ufabc.edu.br	26 99 43 40 40 101 135 52 35 31 32 22 67
21	1120		@aluno.ufabc.edu.br	35 46 33 21 26 37 22 54 40 58 42 40 51 53 65 26
22	1120		@aluno.ufabc.edu.br	26 15 52 29 52 99 18 19 41 27 102 67 9
23	1120		@aluno.ufabc.edu.br	88 43 150 108 49 39 189 51 76 108 124 12 63 100 46 23
24	1120		@aluno.ufabc.edu.br	84 65 81 118 44 62 17 71 33 45 71 45 22 69 62 138 106 43
25	1120		@aluno.ufabc.edu.br	195 36 155 49 51 4 129 51 17 39 58 42 24 64 105 53
26	1120		@aluno.ufabc.edu.br	97 64 43 94
27	1120		@aluno.ufabc.edu.br	2 144 90 259 35 82 53 118 125 170 152 92 98 148 126
28	1120		@aluno.ufabc.edu.br	2 44 77 60 59 51 72 14 42 15 30
29	1120		@aluno.ufabc.edu.br	101 46 82 31 66 101 133 21 69 66 90 22 56 70 75 56 140
30	1120		@aluno.ufabc.edu.br	12 30 8 26 40 14
31	1120		@aluno.ufabc.edu.br	58 17 45 92 78 93 60 71 52 122 52 70 90 87 142 102
32	1120		@aluno.ufabc.edu.br	61 55 63 39 56 50 164 88 40 35 74 16 18 22 38 118 53 188
33	1120		@aluno.ufabc.edu.br	31 105 23
34	1120		@aluno.ufabc.edu.br	27 15 34 44 25 45 101 219 45 99 83 97 65 86 40 101 35
35	1120		@aluno.ufabc.edu.br	32 25 94 70 78 60 34 125 50 103 47 37 49 48 103 70

Arquivo gerado: ./tmp/2024-04-20_20-46-11/report/alunos_Total_Presencas_Nome.png
 Arquivo gerado: ./tmp/2024-04-20_20-46-11/report/alunos_Total_Presencas_RA.png
 Arquivo gerado: ./tmp/2024-04-20_20-46-11/report/alunos_Total_Acessos_Nome.png
 Arquivo gerado: ./tmp/2024-04-20_20-46-11/report/alunos_Total_Acessos_RA.png
 Acessos Sem Filtros por Nome de Aluno (filtro: Laboratório de Programação Virtual) – Hatchura indica acessos fora da aula
 Arquivo gerado: ./tmp/2024-04-20_20-46-11/report/alunos_Acessos_Nome.png
 Acessos Sem Filtros por RA de Aluno (filtro: Laboratório de Programação Virtual) – Hatchura indica acessos fora da aula
 Arquivo gerado: ./tmp/2024-04-20_20-46-11/report/alunos_Acessos_RA.png

ALUNOS REPROVADOS POR FALTA:				
num	RA	Nome	Email	Conceito Faltas
5	11202		@aluno.ufabc.edu.br	F 24
8	1120		@aluno.ufabc.edu.br	F 24
9	1120		@aluno.ufabc.edu.br	F 36
10	1120		@aluno.ufabc.edu.br	F 36
18	1120		@aluno.ufabc.edu.br	F 28
26	1120		@aluno.ufabc.edu.br	F 24
30	1120		@aluno.ufabc.edu.br	F 24
33	1120		J.S@aluno.ufabc.edu.br	F 30

omitido

Figura A.4: Tela do serviço - Parte inicial do relatório gerado.

A.4. RESULTADOS E DISCUSSÕES

primeiro caso.

Arquivos descompactados

```
alunos_Acessos_Sem_Filtros_Nome.png
alunos_Acessos_Sem_Filtros_RA.png
alunos_Total_Acessos_Nome.png
alunos_Total_Acessos_RA.png
alunos_Total_Presencias_Nome.png
alunos_Total_Presencias_RA.png
data.json
dias_notas_BCM0505-B-22SA_20241-OK2.xls.csv
faltas_notas_BCM0505-B-22SA_20241-OK2.xls.csv
presenca_notas_BCM0505-B-22SA_20241-OK2.xls.csv
```

A.4.1 Arquivo data.json

Nessa lista de arquivos possui também texto nos formatos JSON e CSV. O promeiro `data.json` é apresentado um exemplo a seguir. Esse arquivo contém informações sobre o período de análise (de 5 de fevereiro a 7 de maio de 2024), o filtro aplicado (Laboratório de Programação Virtual), e detalhes sobre as aulas, incluindo dias, horas e duração. Também são fornecidos caminhos para os diretórios de *upload* e relatório, bem como os arquivos CSV e XLS utilizados no processo.

Arquivo data.json

```
{  
  "startDate": "2024-02-05",  
  "endDate": "2024-05-07",  
  "filter_field": "Laboratório de Programação Virtual",  
  "min_absences": "14",  
  "assign_0": "off",  
  "omit_data": "off",  
  "uploadDir": ".\tmp\2024-04-20_20-30-22\uploads\",  
  "reportDir": ".\tmp\2024-04-20_20-30-22\report\",  
  "csvPath": ".\tmp\2024-04-20_20-30-22\uploads\logs_PI-2024.1-Zampirolli.csv",  
  "xlsPath": ".\tmp\2024-04-20_20-30-22\uploads\notas_BCM0505-B-22SA_20241-OK2.xls",  
  "classes": [{"  
    "day": "1",  
    "hour": "10:00",  
    "duration": "2:00",  
    "ipPrefix": "172.17.14"}, {"  
    "day": "3",  
    "hour": "8:00",  
    "duration": "2:00",  
    "ipPrefix": "172.17.14"}]}
```

A.4.2 Arquivo dias_notas_*.csv

O arquivo `dias_notas_*.csv` contém informações sobre o início e o fim de cada sessão de aula, juntamente com um prefixo de endereço IP correspondente. Cada linha representa uma sessão de aula, com as colunas representando a data e horário de início, data e horário de fim, e o prefixo do endereço IP do laboratório em que a aula ocorreu.

Arquivo dias_notas_*.csv

Inicio	Fim	IP		
06/02/2024	10:00	06/02/2024	12:00	172.17.14
08/02/2024	08:00	08/02/2024	10:00	172.17.14
13/02/2024	10:00	13/02/2024	12:00	172.17.14
...				

A.4.3 Arquivo faltas_notas_*.csv

O arquivo `faltas_notas_*.csv` contém informações sobre os estudantes, como suas matrículas, nomes, e-mails, resultados obtidos, quantidade de faltas e situação atual. Esses dados podem ser utilizados pelo professor para atualizar o arquivo original XLS exportado do SIGAA, permitindo a

A.5. ACESSOS ÀS ATIVIDADES DO MOODLE

atualização das colunas de Resultado e Faltas. Posteriormente, o professor pode importar essas atualizações para o sistema acadêmico, garantindo a precisão dos conceitos dos estudantes registrados.

Arquivo faltas_notas_*.csv

Matrícula	Nome	E-mail	Resultado	Faltas	Sit.
omitido	omitido	omitido@aluno.ufabc.edu.br	B	4	APR
omitido	omitido	omitido@aluno.ufabc.edu.br	A	8	APR
omitido	omitido	omitido@aluno.ufabc.edu.br	D	10	APR
...					

A.4.4 Arquivo presenca_notas_*.csv

O arquivo `presenca_notas_*.csv` mantém as três primeiras colunas para matrículas, nomes e e-mails. Em seguida, cada coluna representa um dia de aula conforme definido pelo arquivo `dias_notas_*.csv`. Além disso, o arquivo inclui as colunas “Acessos_Sem_Filtros”, “Total_Presencas”, “Faltas” e “Conceito”. Esses dados são obtidos cruzando as informações dos arquivos CSV e XLS enviados pelo professor, além dos dados informados no formulário HTML antes do envio. Com base nesse arquivo `presenca_notas_*.csv`, foram geradas seis imagens PNG, como detalhado na próxima seção.

Arquivo presenca_notas_*.csv

...	01-06/02	02-08/02	03-13/02	...	Acessos_Sem_Filtros	Total_Presencas	Faltas	Conceito
40	40			909	32		4	B
35	27	30		767	28		8	A
				1433	26		10	D
...								

A.4.5 Acessos no Moodle

A Figura A.5 apresenta a imagem `alunos_Acessos_Sem_Filtros_RA.png` contendo informações bastante relevantes, destacadas na área sombreada em vermelho, que correspondem ao número de acessos dos estudantes fora do horário das aulas na turma A. Esses dados sugerem uma possível falta de dedicação por parte dos estudantes nesta disciplina ao longo do período regular até a prova final. No topo de cada barra, são exibidos os totais de acessos, juntamente com o conceito atribuído ao estudante antes do exame final (sendo permitido realizar o exame final apenas para os estudantes com conceitos D e F). Um gráfico semelhante para a turma B é apresentado na Figura A.6.

A.5 Acessos às atividades do Moodle

O gráfico de barras da Figura A.7 apresenta o número de estudantes que tentaram resolver cada um dos 80 EPs, distribuídos ao longo do eixo x . No eixo y , está representado o número de

estudantes que tentaram realizar cada EP. Cada barra no gráfico corresponde a um EP específico, como “EP1_1”, “EP1_2”, ..., “EP6_23”. Observa-se uma variação considerável no número de acessos entre os diferentes EPs, indicando diferentes níveis de interação dos estudantes com o material disponibilizado. Além disso, o gráfico inclui uma linha de tendência exponencial com um coeficiente de determinação $R^2 = 0.449$, indicando que a linha é uma aproximação moderada para os dados. Esta linha ajuda a visualizar a tendência geral de decrescimento dos acessos ao longo dos EPs. Isso pode indicar o baixo desempenho dos estudantes na prova final, como relatado na próxima seção. A cor azul tem os EPs relacionados à primeira prova. A cor verde é associada aos EPs relacionados a vetores, enquanto a cor salmão representa os EPs relacionados a matrizes, ambas abordando o conteúdo da prova final. Finalmente, vale enfatizar ainda que aproximadamente 60 destes EPs foram resolvidos em sala de aula pelo professor, com é possível observar os EPs que se aproximam da linha de tendência.

A.6 Tabelas comparativas - turma A vs. turma B

Esta seção destaca a comparação do desempenho entre as turmas A e B, tanto antes quanto após a prova de recuperação. Antes do exame, a turma B exibia uma correlação mais pronunciada entre acessos e conceitos, com uma maior proporção de estudantes alcançando o conceito A.

A.6.1 Comparações antes da prova de recuperação

Após a análise da tabela comparativa antes da prova de recuperação (Tabela A.1), observa-se que a turma B apresentou uma correlação ligeiramente maior entre acessos (durante a aula e fora também) e conceitos em comparação com a turma A. No entanto, a turma B obteve um número maior de estudantes com conceito A, indicando um desempenho global melhor em relação à turma B antes da prova de recuperação. Por outro lado, a turma B teve uma proporção maior de estudantes com conceitos D e F, sugerindo um desempenho mais baixo nesse período. Esses valores foram alterados após a prova de recuperação, como apresentado na próxima seção.

Tabela A.1: Tabela comparativa - antes da prova de recuperação

Turma	Correlações (desempenho vs)		Conceitos					Reprovação %
	Acessos	Presenças	A	B	C	D	F	
A	0.498	0.389	3	3	2	4	26	68.42
B	0.558	0.526	7	1	1	7	19	54.29

A.6.2 Comparações após a prova de recuperação - resultado final

Após a realização da prova de recuperação e a obtenção dos resultados finais, conforme apresentado na tabela comparativa (Tabela A.2), notamos que houve uma melhoria no desempenho da turma A, com um aumento no número de estudantes com conceitos A, C e D e uma redução na proporção de estudantes com conceito F. Por outro lado, na turma B, embora não tenha alterado no número de estudantes com conceito A e B, houve também um aumento no número de estudantes com

conceitos C e D. Porém, o destaque continua sendo a porcentagem de reprovações elevada na turma A (55.26%) comparada à turma B (32.35%). Nessa tabela não foram apresentadas as correlações entre desempenho e acessos (além de frequência), pois não foi mais contabilizada a presença após a prova 2, visto que alguns estudantes que já foram aprovados não participaram mais das aulas.

Tabela A.2: Tabela comparativa - após da prova de recuperação

Turma	Conceitos					Reprovação %
	A	B	C	D	F	
A	4	2	5	6	21	55.26
B	7	1	2	13	11	32.35

Para contextualizar o desempenho das turmas A e B em comparação com todas as turmas de CS1, a próxima seção apresenta o histórico de todas as turmas de CS1 ofertadas na UFABC. Esse levantamento possibilita uma melhor comparação entre as turmas no período pré-pandemia, durante a pandemia e após a pandemia. Observa-se que o desempenho da Turma B está próximo à média histórica de 29% de reprovação no período ideal. Para mais detalhes, ver a próxima seção.

A.7 Histórico de CS0 e CS1 na UFABC

O artigo de [Zampirolli, Goya, Pimentel e Kobayashi \(2018\)](#) apresenta um processo de avaliação desenvolvido para o curso de Processamento da Informação (CS1) da UFABC. Nesse estudo, o desempenho de turmas presenciais é comparado ao de turmas que utilizaram aprendizagem híbrida (*Blended Learning*). Para isso, foram aplicadas ferramentas digitais que padronizam a avaliação. É importante notar que, nas turmas híbridas, apenas as provas foram realizadas presencialmente. Os dados de nove anos de turmas, até o segundo semestre de 2017, foram analisados para validar o processo de avaliação proposto. Observou-se que o nível de variância dos conceitos nas turmas híbridas foi menor em comparação com as turmas presenciais. Além disso, a média de reprovações nas turmas híbridas foi ligeiramente inferior (32,3%) em relação às turmas presenciais (34,1%). Esse artigo também foi resumido na Seção [12.1.2 – Artigo comparando as modalidades semipresencial e presencial CS1 com base na utilidade do MCTest-4](#).

O artigo de [Zampirolli, Josko, Venero, Kobayashi, Silva, Goya e Savegnago \(2021\)](#) descreve a experiência de aplicação de Avaliação Automatizada (AA) no curso de Bases Computacionais (CS0) da UFABC. Durante a pandemia de 2020, a AA se mostrou essencial ao proporcionar *feedback* imediato em exercícios de programação, o que contribuiu para a redução da taxa de reprovação nas turmas que adotaram essa metodologia (32,5%) em comparação às turmas que não a utilizaram (41,9%). Observou-se, ainda, uma forte correlação linear entre a taxa de aprovação e a média das notas nos exercícios de AA, o que reforça a eficácia da avaliação contínua e do *feedback* imediato no processo de aprendizagem dos estudantes. Além disso, o artigo compara o desempenho dos estudantes que utilizaram as linguagens Java e Python. Este artigo também foi resumido na Seção [13.2.4 – Artigo aplicando a integração MCTest+Moodle+VPL em CS1](#).

A Tabela [A.3](#) consolida os dados das disciplinas CS0 (código 0005) e CS1 (código 0505) no

período de 2009 até 2024.1. Nela, é possível extrair várias informações relevantes.

Por exemplo, o período ideal para cursar cada disciplina, destacado com fundo rosa, apresenta uma média de aprovações superior em comparação com os períodos não ideais. Isso sugere que, quando os estudantes cursam as disciplinas no momento recomendado, existem menos estudantes com reprovações e o desempenho tende a ser melhor.

Outro ponto a ser destacado é a diminuição das reprovações durante a pandemia (de 2020.1 a 2022.2), o que pode ser atribuído à impossibilidade em realizar avaliações presenciais, além disso, os estudantes tinham um prazo de 72 horas para concluir as provas. Esse cenário, embora tenha contribuído para uma redução nas taxas de reprovação, pode não refletir uma melhora real na aprendizagem.

Após a pandemia, a partir de 2023.1, observa-se que as taxas de reprovação permaneceram abaixo da média geral. Uma possível explicação para esse fenômeno é que os professores possam ter reduzido as exigências nas avaliações, algo que poderia ser confirmado com a aplicação de questionários.

Além disso, a partir de 2023.1, a disciplina CS1 passou por uma mudança na carga horária: antes eram 5 horas semanais, sendo 3 horas teóricas com um professor para 90 estudantes (3 turmas), além de 2 horas em laboratório com o apoio de 3 professores, um por turma. Agora, a disciplina conta com 4 horas de laboratório sob a supervisão de um único professor.

Um detalhe sobre CS1 em 2022.3 é que as 9 turmas tiveram um único professor na parte teórica, sendo ministrada remotamente, com avaliações semelhantes às ocorridas durante a pandemia, e foi utilizado teste de mesa ([TEUBL; ZAMPIROLI, 2023](#)). Porém, a parte prática foi realizada presencialmente nos laboratórios, também com avaliação presencial.

Detalhando o histórico de CS1 na UFABC

O momento ideal para cursar esta disciplina é durante o primeiro quadrimestre letivo do ano, o que normalmente corresponde ao terceiro quadrimestre do estudante no curso. A Tabela A.4 apresenta o histórico de oferecimentos da disciplina entre os quadrimestres 2009.3 (setembro a dezembro) e 2024.1 (fevereiro a maio).

A cada ano, são atribuídos coordenadores para a disciplina, os quais podem sugerir materiais didáticos e Exercícios de Programação (EPs) por meio de um curso dedicado no Moodle. Esses recursos ficam disponíveis para todos os docentes, mas seu uso não é obrigatório. Cada docente possui autonomia para adotar ou complementar esses materiais conforme sua preferência. Consequentemente, não é possível determinar quais abordagens de ensino foram utilizadas em cada turma. No entanto, é possível identificar padrões de desempenho com base nas taxas históricas de reprovação, como discutido a seguir.

A Tabela A.4 apresenta as taxas médias de reprovação por quadrimestre na disciplina *Lógica de Programação* (CS1) do Bacharelado em Ciência e Tecnologia. Os dados estão organizados por turno: Diurno (D) e Noturno (E).

As marcações em rosa indicam quadrimestres realizados em *períodos acadêmicos ideais*, enquanto as marcações em laranja correspondem a quadrimestres afetados pela pandemia de COVID-19.

A.8. CONSIDERAÇÕES FINAIS

Essas distinções visuais permitem observar variações nas taxas de reprovação sob diferentes condições de oferta.

Os dados agregados nas últimas linhas da Tabela A.4 revelam tendências no desempenho estudantil. Em todas as turmas oferecidas entre 2009 e 2024, a taxa média geral de reprovação foi de 36%, com desvio padrão de 15%, considerando 834 turmas e mais de 26.500 estudantes matriculados.

Ao separar os dados por turno, as turmas Diurnas (D) apresentaram média de reprovação de 32%, enquanto as turmas Noturnas (E) apresentaram média de 40%. Esse padrão sugere dificuldades persistentes entre os estudantes do período noturno, possivelmente associadas a fatores externos, como jornada de trabalho, cansaço ou menor acesso a suporte acadêmico.

Durante os *períodos acadêmicos ideais*—definidos como quadrimestres regulares, sem interrupções, normalmente no primeiro quadrimestre do ano—, a média de reprovação foi de 24% nas turmas Diurnas e 35% nas turmas Noturnas. Esses dados indicam que, sob condições estáveis de oferta, há melhora no desempenho, especialmente nas turmas Diurnas.

Por outro lado, em *períodos não ideais*—marcados por repetição da disciplina, irregularidades na oferta ou sobreposição de compromissos—, as taxas de reprovação foram mais elevadas: 41% nas turmas Diurnas e 47% nas turmas Noturnas. Esses são os maiores índices médios observados, reforçando a influência do momento de oferta e da estrutura do curso sobre os resultados acadêmicos, especialmente no período noturno.

Outro aspecto observado refere-se ao período da pandemia. Durante a oferta totalmente remota da disciplina, mas ainda em período acadêmico ideal, a taxa média de reprovação foi de 34%. Nos dois quadrimestres seguintes à pandemia (2023.2 e 2024.1), essa taxa caiu para 25%, valor inferior à média geral de 29% para períodos ideais. A partir de 2023.2, houve uma mudança na *estrutura da disciplina*: em vez de 5 horas semanais (sendo 3 horas de aula expositiva com um docente para 90 estudantes e atividades de laboratório com três docentes distintos), passou-se a adotar um formato de 4 horas semanais inteiramente dedicadas ao laboratório, ministrado por um único docente. Não é possível determinar se essa alteração estrutural ou o fim do ensino remoto foi o principal fator responsável pela redução observada nas taxas de retenção. Para isso, são necessárias investigações adicionais, incluindo análise das práticas de avaliação e comparação com outras disciplinas.

Como exemplo, a disciplina introdutória CS0, oferecida no início do curso de bacharelado interdisciplinar, manteve uma taxa média de retenção estável em torno de 21% durante todos os períodos acadêmicos ideais, inclusive durante e após a pandemia, ver Tabela A.3. Essa estabilidade indica que as mudanças observadas em CS1 podem não se aplicar a todas as disciplinas de programação.

A.8 Considerações finais

O experimento realizado nas turmas A e B da disciplina de Processamento da Informação no primeiro quadrimestre de 2024 proporcionou informações valiosas sobre o engajamento e desempenho dos estudantes. A análise dos dados de acesso ao Moodle, combinada com os conceitos obtidos, permitiu avaliar a correlação entre essas métricas e identificar áreas de melhoria.

Os resultados revelaram que a turma B apresentou uma correlação um pouco mais forte entre acessos, presenças e desempenho, com uma proporção maior de estudantes obtendo conceitos mais altos em geral. Isso sugere que, mesmo recebendo o mesmo conteúdo e avaliações semelhantes, uma turma pode ter um desempenho superior por razões que vão além do escopo deste estudo. Após a prova de recuperação, observou-se uma melhoria geral no desempenho das turmas A e B, com mais estudantes alcançando conceitos superiores.

O serviço LabMoodle, desenvolvido para acompanhar as atividades dos estudantes no Moodle, demonstrou ser uma ferramenta valiosa para os professores. Ao fornecer informações detalhadas sobre acessos, presenças e desempenho, o serviço permitiu identificar pontos de atenção e orientar estratégias pedagógicas mais eficazes.

É importante ressaltar que os dados de acesso ao Moodle e as presenças em aula representam apenas uma parte do processo de aprendizagem dos estudantes. Outros fatores, como a dedicação individual fora do horário de aula, as estratégias de estudo e o engajamento com o conteúdo, também desempenham um papel crucial no desempenho acadêmico.

Uma limitação deste método é que alguns estudantes podem realizar seus estudos fora do Moodle, tanto durante a aula quanto fora dela. Além disso, não foi avaliado o conhecimento prévio dos estudantes. Assim, estudantes com experiência em programação podem não participar das atividades e ainda assim apresentar um bom desempenho, como observado nos gráficos apresentados. Finalmente, presume-se que os IPs de um laboratório devem ter um prefixo, como “172.17.14”, conforme apresentado na Figura A.2. O professor pode também considerar todos os IPs, substituindo o prefixo por “172” (conforme a rede local) ou simplesmente “.”.

Como sugestões para trabalhos futuros, considerando que os estudantes adquirem conhecimentos básicos de lógica de programação no seu ingresso na graduação, durante a disciplina de Bases Computacionais da Ciência (CS0), geralmente ministrada utilizando a linguagem Python, propõe-se algumas modificações nas avaliações da disciplina de Processamento da Informação. Uma sugestão seria realizar a primeira prova abrangendo conceitos até laços de repetição ao final da semana 4, funcionando como uma revisão dos conteúdos abordados no curso CS0. Na semana 7, poderia ser aplicada uma prova sobre vetores, e na semana 11, uma prova sobre matrizes, distribuídas com pesos de avaliação de 30%, 35% e 35%, respectivamente. Outra sugestão é substituir os testes adaptativos de múltipla escolha em papel por uma implementação similar no próprio ambiente Moodle. Ou ainda realizar simulados antes de cada avaliação, valendo um bônus de 5% em cada um. Finalmente, um estudante sugeriu resolver mais EPs passo-a-passo pelo professor, além de mostrar soluções prontas e explicar.

As informações obtidas neste experimento podem ser utilizadas para aprimorar o planejamento e a condução de futuras ofertas da disciplina. Ajustes nas abordagens pedagógicas, incentivos ao engajamento e um acompanhamento mais próximo dos estudantes com dificuldades podem ser implementados com base nas lições aprendidas.

A.8. CONSIDERAÇÕES FINAIS

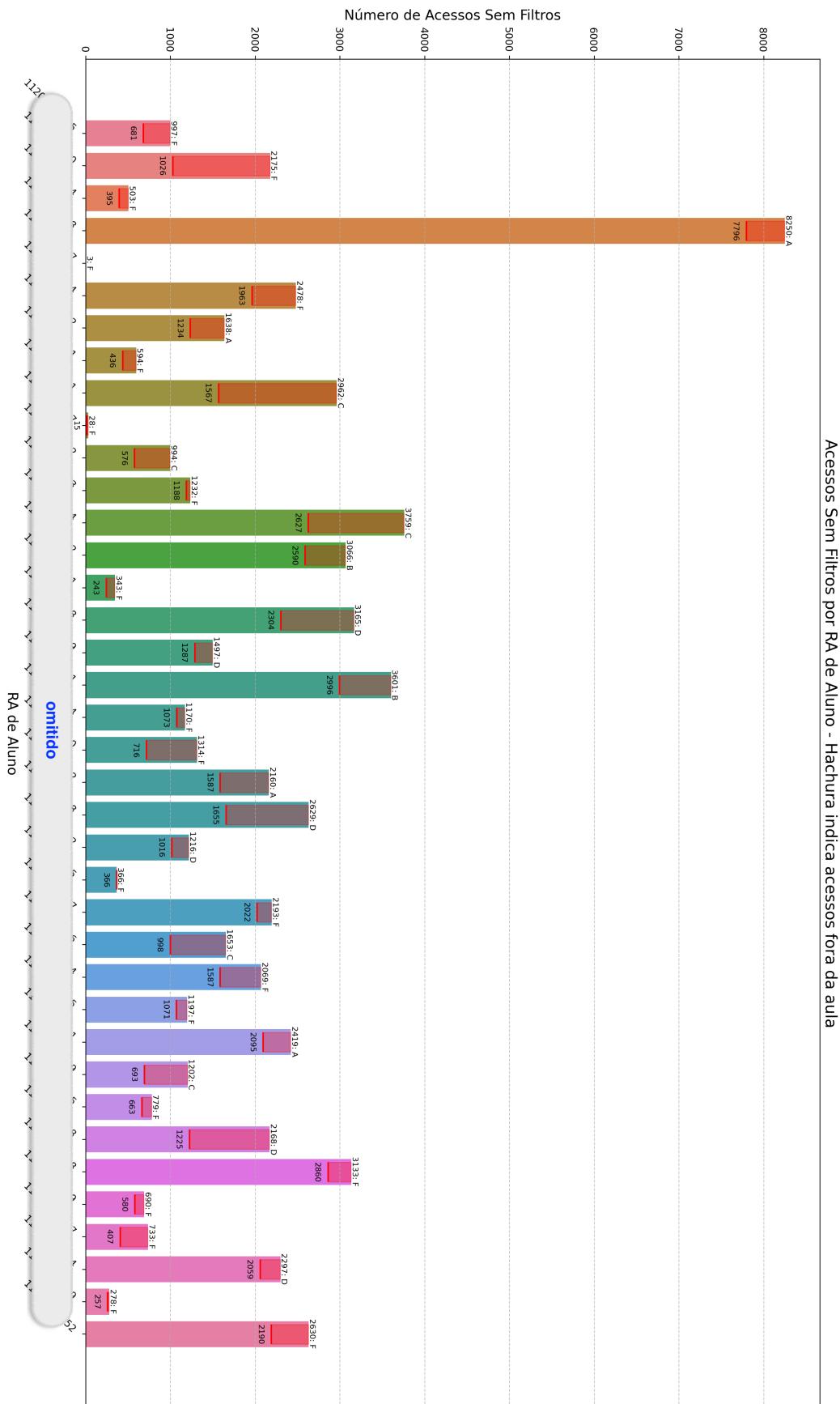


Figura A.5: Gráfico contendo os acessos totais dos estudantes da turma A, com destaque para o baixo acesso fora da aula.

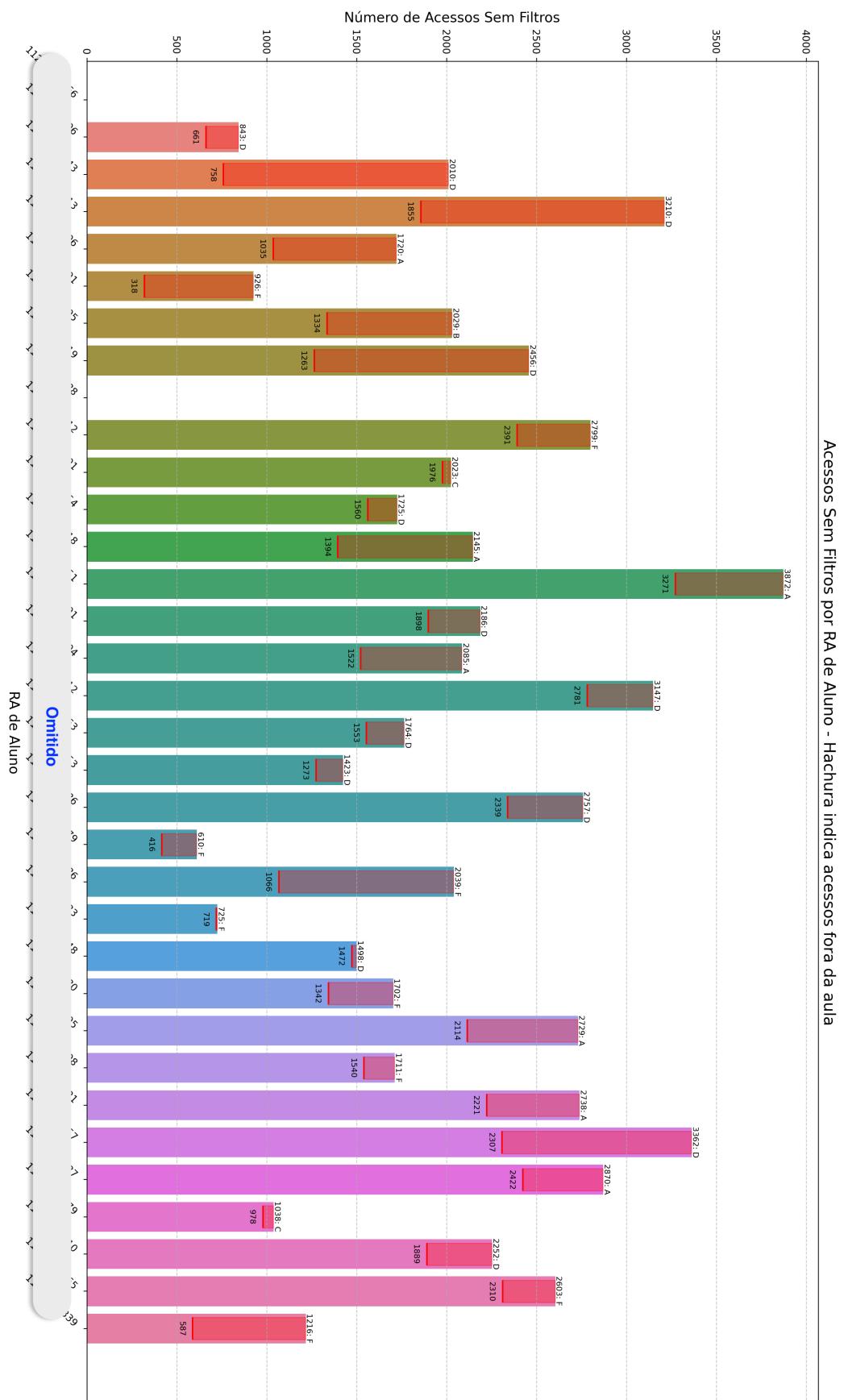


Figura A.6: Gráfico contendo os acessos totais dos estudantes da turma B, com destaque para o baixo acesso fora da aula.

A.8. CONSIDERAÇÕES FINAIS

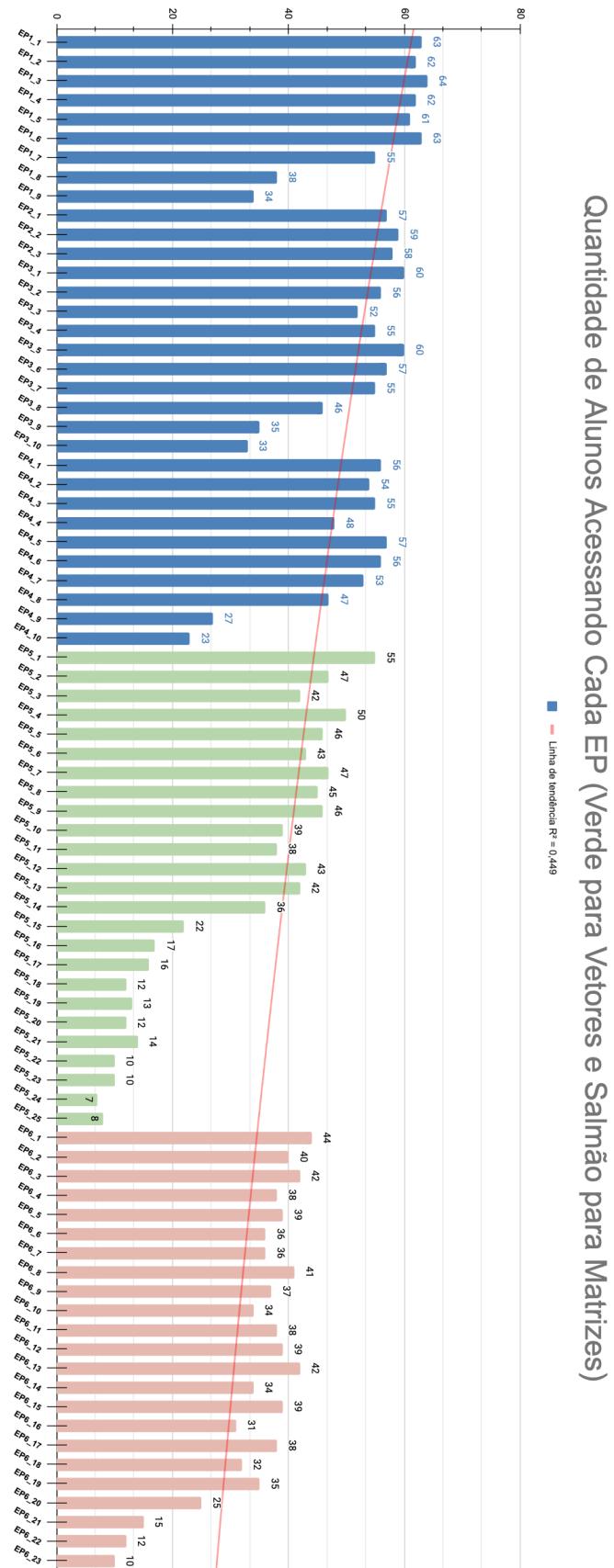


Figura A.7: Quantidade de estudantes que tentaram resolver os EP (verde para EPs sobre vetores e salmão para matrizes).

Tabela A.3: Histórico de desempenhos das turmas CS0 (0005) e CS1 (0505). A tabela apresenta a média e o desvio padrão (STD) das reprovações. O fundo rosa indica o período ideal para cursar as disciplinas, e o fundo laranja representa as turmas durante a pandemia.

Período	0005				0505			
	Média	STD	Turma	Alunos	Média	STD	Turma	Alunos
2009.1	9%	7%	47	1272				
2009.2	45%	23%	6	119				
2009.3					25%	15%	34	856
2010.1					31%	15%	4	107
2010.2	26%	12%	64	1687	37%	16%	9	217
2011.1					29%	22%	47	1216
2011.2	24%	11%	61	1730				
2011.3					42%	14%	11	262
2012.1	42%	15%	4	87	31%	12%	45	1206
2012.2	28%	17%	60	1814				
2012.3	45%	12%	8	162	57%	25%	11	214
2013.1					30%	16%	39	1082
2013.2	26%	14%	60	1853	56%	29%	12	215
2013.3	72%	9%	4	82				
2014.1	62%	15%	6	160				
2014.2	23%	16%	63	1767	31%	21%	37	999
2014.3	39%	13%	4	117	36%	18%	10	340
2015.1	35%	19%	7	192	24%	14%	33	1085
2015.2	23%	14%	52	1586	34%	12%	16	401
2015.3	44%	30%	6	147	45%	11%	6	145
2016.1	25%	18%	12	358	21%	20%	38	1104
2016.2	20%	14%	52	1608	36%	15%	11	402
2016.3	48%	19%	8	221	43%	3%	2	243
2017.1	30%	20%	13	480	24%	16%	42	1298
2017.2	22%	14%	53	1630				
2017.3	41%	17%	4	131	58%	2%	6	522
2018.1	25%	18%	12	428	26%	16%	69	2365
2018.2	16%	10%	50	1546	52%	17%	18	466
2018.3	37%	16%	6	204	43%	10%	4	282
2019.1	27%	18%	12	458	37%	21%	84	2472
2019.2	27%	18%	54	1666	43%	11%	3	122
2019.3	39%	23%	6	205	48%	19%	8	384
2020.1	45%	30%	12	345	39%	25%	75	2838
2020.2	20%	10%	43	1413				
2021.1	42%	15%	12	387				
2021.3	23%	11%	44	2073	36%	11%	34	1528
2022.2	48%	23%	16	436	24%	12%	44	1161
2022.3	27%	14%	68	2012	42%	23%	9	229
2023.1	41%	12%	4	109				
2023.2	18%	9%	55	1921	23%	16%	32	1247
2023.3					21%	8%	6	252
2024.1					26%	20%	34	1287
Total geral	33%	16%	988	30406	36%	16%	833	26547
Ideal	22%	13%	826	25578	29%	17%	687	21744
Não Ideal	42%	18%	162	4828	43%	14%	146	4803

A.8. CONSIDERAÇÕES FINAIS

Tabela A.4: Taxas médias de reprovação em PI por quadrimestre (2009–2024).

Period	Shift	Average	STD	Classes	Students	Period	Shift	Average	STD	Classes	Students
2009.3	D	24%	16%	23	643	2017.1	D	19%	12%	23	766
	E	28%	12%	11	213		E	31%	18%	19	532
2010.1	D	18%	3%	2	54	2017.3	D	58%	2%	6	522
	E	43%	7%	2	53		E				
2010.2	D	37%	20%	6	147	2018.1	D	24%	15%	41	1506
	E	37%	7%	3	70		E	30%	17%	28	859
2011.1	D	24%	21%	24	634	2018.2	D	45%	12%	13	350
	E	34%	22%	23	582		E	72%	13%	5	116
2011.3	D	35%	8%	7	185	2018.3	D	43%	10%	4	282
	E	55%	12%	4	77		E				
2012.1	D	28%	11%	21	539	2019.1	D	29%	17%	45	1363
	E	33%	13%	24	667		E	47%	21%	39	1109
2012.3	D	61%	27%	4	73	2019.2	D				
	E	54%	26%	7	141		E	43%	11%	3	122
2013.1	D	23%	13%	16	477	2019.3	D	31%	1%	4	282
	E	35%	17%	23	605		E	65%	8%	4	102
2013.2	D	50%	28%	7	138	2020.1	D	30%	17%	43	2078
	E	65%	32%	5	77		E	50%	29%	32	760
2014.2	D	25%	21%	20	527	2021.3	D	35%	10%	17	753
	E	38%	20%	17	472		E	38%	12%	17	775
2014.3	D	33%	21%	7	272	2022.2	D	22%	13%	22	536
	E	44%	2%	3	68		E	27%	10%	22	625
2015.1	D	20%	12%	23	803	2022.3	D	47%	23%	5	106
	E	34%	13%	10	282		E	36%	24%	4	123
2015.2	D	36%	10%	11	255	2023.2	D	17%	17%	16	626
	E	29%	15%	5	146		E	29%	14%	16	621
2015.3	D	45%	11%	6	145	2023.3	D				
	E						E	18%	11%	7	252
2016.1	D	17%	13%	25	784	2024.1	D	17%	16%	17	637
	E	30%	28%	13	320		E	35%	20%	17	650
2016.2	D	27%	9%	6	279	Grand Total		36%	15%	834	26547
	E	47%	15%	5	123		Total	D	32%	14%	466
2016.3	D	43%	3%	2	243		E	40%	16%	368	10542
	E					Ideal	D	24%	15%	25	845
						E	35%	18%	21	605	
						Not Ideal	D	41%	13%	6	222
						E	47%	14%	4	113	

Pink highlights indicate ideal periods

Orange refer to COVID-19 pandemic periods

Apêndice B

Exames Adaptativos na disciplina CS1

Conteúdo

B.1	Contextualização dos exames adaptativos	276
B.2	Teste 1: Sequencial – Aleatório	276
B.2.1	Criação dos testes adaptativos	277
B.2.2	Correção dos testes adaptativos	277
B.2.3	Descrição das questões paramétricas utilizadas	278
B.3	Teste 2: Método – SAT	279
B.3.1	Criação dos testes adaptativos	280
B.3.2	Correção dos testes adaptativos	281
B.4	Teste 3: Condisional – WPC	282
B.4.1	Criação dos testes adaptativos	282
B.4.2	Correção dos testes adaptativos	282
B.5	Teste 4: Repetição – WPC	283
B.6	Teste 5: Vetor – MLE-v0	283
B.7	Teste 6: Matriz – MLE-v1	283
B.7.1	Descrição das questões utilizadas	284
B.7.2	Calibração das questões	286
B.8	Questionário avaliativo	293
B.8.1	Questões aplicadas	293
B.8.2	ANÁLISE ESTATÍSTICA DOS RESULTADOS	297
B.9	Considerações finais	298

Este apêndice descreve uma experiência conduzida na disciplina de Processamento da Informação (CS1) na UFABC durante o primeiro quadrimestre de 2024, em duas turmas. O objetivo foi realizar seis testes, sendo cinco adaptativos ao longo do curso, como avaliações formativas, com

peso de 5% no conceito final, com bônus proporcional ao número de testes realizados. Os estudantes foram informados sobre esses testes adaptativos como uma estratégia motivacional aplicada na disciplina. Este experimento complementa a Seção 8.6 – Exames adaptativos. Os detalhes sobre o método de ensino-aprendizagem-avaliação nas duas turmas foram apresentados no Apêndice A – Acompanhamento de acesso de estudantes no Moodle, e não serão repetidos neste apêndice, focando apenas na parte dos exames adaptativos.

O conteúdo deste apêndice é uma adaptação do Trabalho de Conclusão de Curso do estudante Lucas Montagnani Calil Elias intitulado “Aprendizado Adaptável Aproveitando os Dados de Desempenho do Aluno para Fazer Avaliações Personalizadas no MCTest”, desenvolvido no âmbito do Bacharelado em Ciência da Computação na Universidade Federal do ABC, no período de 2023-2024.

B.1 Contextualização dos exames adaptativos

A Tabela B.1 apresenta uma lista de seis testes utilizados para avaliar o desempenho e a compreensão dos estudantes em áreas específicas da programação. Geralmente, esses testes são administrados na semana seguinte à apresentação do respectivo tópico. O primeiro teste é sobre Sequencial, sendo do tipo Aleatório, indicando que envolve a execução sequencial de instruções com perguntas apresentadas de forma aleatória. O segundo teste aborda Método e é do tipo SAT, avaliando a capacidade dos estudantes em trabalhar com métodos e lógica. Os Testes 3 e 4 tratam de Condicional e Repetição, respectivamente, sendo do tipo WPC, focando na compreensão e implementação de estruturas condicionais e de repetição. Após esses testes, houve a primeira prova avaliativa (valendo 40% do conceito final), realizada na semana 5.

O Teste 5 aborda o tema de vetores e é do tipo MLE-v0, destinado a avaliar conhecimentos sobre vetores e operações relacionadas. O Teste 6 versa sobre matrizes e é classificado como MLE-v1, apresentando um nível de dificuldade mais elevado. Cada teste consiste em 200 variações sorteadas entre os estudantes das duas turmas. Todos os testes possuem 5 questões de múltipla escolha, cada uma com 5 alternativas, sendo apenas uma correta. Todas as questões foram criadas pelo professor e atribuídas a uma das taxonomias de Bloom, priorizando os três primeiros níveis: Lembrar, Entender e Aplicar, devido ao curso ser introdutório de Lógica de Programação (CS1). Na semana 11, houve a segunda prova avaliativa (valendo 60%), e na semana 12, a prova de recuperação. Todas as provas avaliativas foram realizadas integrando as ferramentas MCTest, Moodle e VPL. Essas avaliações tiveram duração de duas horas e foram realizadas utilizando o navegador *Safe Exam Browser* (SEB), sem consulta externa à atividade avaliativa no Moodle. Esses seis testes serão detalhados nas próximas seções.

B.2 Teste 1: Sequencial – Aleatório

Na primeira semana, foi apresentada uma introdução à programação, o uso do ambiente Google Colab e atividades VPL no Moodle com correção automática, abordando programas sequenciais. Este conteúdo foi avaliado na semana seguinte com o primeiro teste, que não foi adaptativo, pois os

Tabela B.1: Testes e seus respectivos tópicos e tipos.

Teste	Tópico	Tipo	Descrição
1	Sequencial	Aleatório	Questões sequenciais apresentadas aleatoriamente
2	Método	SAT	<i>Semi-Adaptive Testing</i>
3	Condisional	WPC	<i>Weighted Probability of Correctness</i>
4	Repetição	WPC	<i>Weighted Probability of Correctness</i>
5	Vetor	MLE-v0	<i>Maximum Likelihood Estimation-v0</i>
6	Matriz	MLE-v1	<i>Maximum Likelihood Estimation-v1</i>

estudantes ainda não tinham realizado testes anteriores para tentar identificar suas habilidades.

B.2.1 Criação dos testes adaptativos

Foram criadas sete questões, sendo sorteadas cinco para cada teste, como apresentado na Seção B.2.3. Dessa, uma era de Lembrar, quatro de Entender e duas de Aplicar, de acordo com a Taxonomia de Bloom. Todas as questões foram paramétricas, ou seja, com possibilidade de variação de dados para cada estudante. O sorteio das questões ocorreu no nível de Entender, selecionando uma questão de cada um dos dois Grupos 2 e 3, que possuíam duas questões cada, conforme mostrado na Tabela B.2.

Após criar um PDF para cada turma com o botão “Cria-PDF” na tela de exame, o teste foi impresso e aplicado nas duas turmas. Em seguida, esses testes foram digitalizados e corrigidos clicando o botão “Upload-PDF”, também na tela de exame. Dessa forma, as questões passaram pela primeira calibração, conforme descrito na Seção 8.6 – Exames adaptativos. Esse mesmo processo se repetiu nos 5 testes seguintes, porém utilizando testes adaptativos diferentes, conforme será relatado nas próximas seções.

B.2.2 Correção dos testes adaptativos

A Tabela B.2 apresenta a porcentagem de acertos e o número de questões respondidas para cada chave do banco de dados, agrupadas por Taxonomia de Bloom. Na taxonomia Lembrar, a questão com chave 2637 obteve 68% de acertos, com um total de 57 respostas. Na taxonomia Entender, as questões do Grupo 2 (chaves 89 e 103) alcançaram 97% e 87% de acertos, respectivamente, com 34 e 23 respostas. As questões do Grupo 3 (chaves 2639 e 2641) obtiveram 64% e 69% de acertos, respectivamente, com 28 e 29 respostas. Na taxonomia Aplicar, as questões com chaves 87 e 2783 obtiveram 72% e 70% de acertos, respectivamente, com 57 respostas cada.

É possível observar que a aleatoriedade nos sorteios das questões dos Grupos 2 e 3, assim como as porcentagens de acertos, estão adequadas. Além disso, é importante notar que a taxonomia não reflete necessariamente na dificuldade de acerto das questões. Por exemplo, as questões de Lembrar, Entender (Grupo 3) e Aplicar apresentaram porcentagens de acertos próximas.

Tabela B.2: Porcentagem de acertos e número de questões respondidas para cada chave do banco de dados no Teste 1 – Sequencial, agrupadas por Taxonomia de Bloom.

	Lembrar	Entender			Aplicar	
Grupos	1	2	3	4	5	
Chaves	2637	89	103	2639	2641	87 2783
Número Respostas	57	34	23	28	29	57 57
Acertos %	68	97	87	64	69	72 70
			média de acertos %		75	
			desvio padrão		11	

B.2.3 Descrição das questões paramétricas utilizadas

Na Figura B.1, é apresentada a questão da taxonomia Lembrar, com chave 2637. Nas Figuras B.2 e B.3, são apresentadas as questões da taxonomia Entender, pertencentes ao Grupo 1, com chaves 89 e 103, respectivamente. Nas Figuras B.4 e B.5, são apresentadas as questões da taxonomia Entender, pertencentes ao Grupo 2, com chaves 2639 e 2641, respectivamente. Nas Figuras B.6 e B.7, são apresentadas as questões da taxonomia Aplicar, com chaves 87 e 2783, respectivamente. Cada figura apresenta o enunciado da questão, a sua respectiva chave e as alternativas, nas quais o destaque em azul representa a alternativa correta (marcada como #0). Lembrando que todas essas questões são paramétricas e diferentes para cada uma das 200 variações do Teste 1 geradas.

#2637 1. Qual(is) da(s) seguinte(s) alternativa(s) conta apenas com operadores relacionais?

- (I) or, <, >=, <=
- (II) <, >, !=, ==
- (III) >, <=, ==, =
- (IV) not, or, >, <
- (V) <=, >=, <, >

A.^{*}3 I, IV B.^{*}1 I, V C.^{#0}II, V D.^{*}4 II, III E.^{*}2 I, III

Figura B.1: Questão da taxonomia Lembrar, com chave 2637.

#0089 1. Dado o programa a seguir, assinale a alternativa que o descreve corretamente a saída:

```
inteiro idade, anoNascimento = 1977, anoAtual = 2015
idade = anoAtual - anoNascimento
escreva(idade)
```

A.^{*}212 B.^{#0}38 C.^{*}25 D.^{*}14 E.^{*}18 F.^{*}432

Figura B.2: Questão da taxonomia Entender (Grupo 2), com chave 89.

#0103 1. Dado o programa a seguir, assinale a alternativa que o descreve corretamente a saída:

```
inteiro distancia = 20, KM_Inicial = 1971, KM_Final = 2011
distancia = KM_Final - KM_Inicial
escreva(distancia)
```

A.^{*}12 B.^{*}10 C.^{*}44 D.^{*}25 E.^{#0}40 F.^{*}424

Figura B.3: Questão da taxonomia Entender (Grupo 2), com chave 103.

B.3. TESTE 2: MÉTODO – SAT

#2639 1. Considerando $A = 20$, $B = 9$ e $C = 9$, assinale a opção correta relacionada à lógica de programação.

- (I) $A + B > 8C \wedge A - C < -8A + B$
- (II) $9B \geq 7A \wedge A \leq 9$
- (III) $A + 6 > B + C$
- (IV) $4C \leq 8C + 8$
- (V) $A + C < 3B \wedge 3B + C < 9A$

A._{•3}T, F, F, T, F B._{•4}F, T, T, T, T C._{•2}T, T, F, T, F D._{•1}T, F, T, T, F E._{•0}F, F, T, T, F

Figura B.4: Questão da taxonomia Entender (Grupo 3), com chave 2639.

#2641 1. Considerando a expressão $A + B < A - B$, assinale a opção correta relacionada à expressão correspondente simplificada.

A._{•0}B < 0 B._{•2}A + B < A/B C._{•3}False D._{•4}A*B < A - B E._{•1}B > 0

Figura B.5: Questão da taxonomia Entender (Grupo 3), com chave 2641.

#0087 1. Considerando o programa a seguir, selecione a opção que descreve corretamente a saída do mesmo:

```
1 X, Y = 1, 2
2 aux = X
3 aux *= 2 + X
4 X = aux
5 print(X)
```

A._{•6}3 B._{•4}-3 C._{•5}-1 D._{•2}0 E._{•1}5 F._{•3}-4

Figura B.6: Questão da taxonomia Aplicar, com chave 87.

#2783 1. Qual é o valor de B após a execução do código abaixo?

```
1 B = 8
2 F = 7
3 D = 2
4 B = F // 2
5 D = D * 2 % 3
6 B = D
```

A._{•3}3 B._{•1}4 C._{•2}9 D._{•0}1 E._{•4}6

Figura B.7: Questão da taxonomia Aplicar, com chave 2783.

B.3 Teste 2: Método – SAT

Na Seção 8.6.1 – Exemplificando o uso do método SAT, o método SAT (*Semi-Adaptive Testing*) foi apresentado detalhadamente por meio de um exemplo fictício. Agora, será detalhado um exemplo real de como esse método foi aplicado no curso CS1 em 2024. As identificações dos estudantes foram omitidas e, por questões de espaço, foram apresentados apenas 4 exemplos de variações do teste, bem como 4 estudantes.

Lembrando que todas as questões usadas em cada teste ainda não foram aplicadas, ou seja, ainda não existiu uma calibração baseada nas respostas dos estudantes. A única calibração foi realizada pelo professor, de forma subjetiva, classificando cada questão pelos seis níveis da Taxonomia de Bloom.

No segundo teste, abordando o tópico de método e com apenas instruções sequenciais, o teste adaptativo foi conduzido analisando os acertos dos estudantes no Teste 1, ponderado pela Taxonomia de Bloom estabelecida pelo professor.

B.3.1 Criação dos testes adaptativos

As 200 variações do Teste 2 foram ordenadas pela média das taxonomias de Bloom (b_i), que foram normalizadas entre -2 e 3. O nível Lembrar recebeu -2 e o nível Criar recebeu 3. A seguir, são apresentadas as duas variações com as menores taxonomias médias e as duas com as maiores taxonomias dentre essas 200 variações aleatórias geradas ao clicar no botão “Criar-Variações” na tela de exame, conforme descrito na Seção 8.6.1.

Arquivo *_adaptive_test_variations

Variation	ID	MeanAbilities	STD	a1	a2	a3	a4	a5	b1	b2	b3	b4	b5	k1	k2	k3	k4	k5
197	75943	-1.800	0.400	1	1	1	1	1	-2	-2	-2	-1	-2	2761	2644	2728	2649	2764
31	75777	-1.600	0.490	1	1	1	1	1	-1	-2	-1	-2	-2	2650	2710	2711	2761	2644
...																		
83	75829	-0.400	0.490	1	1	1	1	1	0	-1	-1	0	0	2718	2646	2729	2715	2766
173	75919	-0.400	0.490	1	1	1	1	1	-1	-1	0	0	0	2720	2711	2766	2721	2763

Na Seção 8.6.1, foram exemplificadas as habilidades anteriores de cada estudante. A seguir, são apresentadas os dois com piores e melhores habilidades. A habilidade -5 é atribuída ao estudante que faltou ao teste.

Arquivo *_adaptive_test.csv

RoomID	Code	Name	Email	2024-02-15:575:pi-Teste01-seq	MeanPreviousAbilities
749	A	stu01	stu01@omitido.br	-5	-5
749	A	stu02	stu02@omitido.br	-5	-5
...					
749	A	stu71	stu71@omitido.br	0	0
750	B	stu72	stu72@omitido.br	0	0

Finalmente, na Seção 8.6.1, é apresentada a variação de cada estudante. A seguir, são mostrados apenas os estudantes com as duas habilidades mais baixas e as duas mais altas. Para aumentar o número de variações atribuídas aos estudantes, foi realizado um sorteio entre as habilidades `MeanPreviousAbilities-0.05` e `MeanPreviousAbilities+0.05`, como observado na variação 173, atribuída aos dois estudantes com as melhores habilidades.

Arquivo *_variations.csv

Room	Name	Email	Variation	MeanPreviousAbilities
A	stu01	stu01@omitido.br	197	-5
A	stu02	stu02@omitido.br	197	-5
...				
A	stu71	stu71@omitido.br	173	0
B	stu72	stu72@omitido.br	173	0

B.3. TESTE 2: MÉTODO – SAT

É importante destacar que a distribuição das variações para cada estudante ocorre selecionando o estudante com a menor habilidade e atribuindo a variação com a menor média (no intervalo +/- 0.05). O mesmo procedimento é aplicado ao estudante com a maior habilidade. As demais variações são atribuídas aos estudantes de forma linear, com base em suas médias de habilidade. Para mais detalhes, consulte o método `getHashAdaptive()` no arquivo `UtilsLatex.py`¹.

B.3.2 Correção dos testes adaptativos

Neste segundo teste, foram registradas 58 respostas, conforme sumarizado na Tabela B.3. As chaves das questões não estão incluídas nesta tabela, pois consistiram em 30 questões agrupadas em 14 categorias (Grupos), das quais 5 questões (uma por grupo) foram selecionadas aleatoriamente. Para todos os testes, foram geradas 200 variações. Essas questões foram adaptadas do Relatório Técnico de Araújo, Caceffo, Garcia e Azevedo (2020).

É possível observar na linha de Respostas da Tabela B.3 um baixo número de respostas, devido à quantidade excessiva de questões a serem sorteadas. Isso prejudica a qualidade da calibração de cada questão. Por exemplo, a questão 14 teve apenas uma resposta correta, enquanto a questão 21 teve 15 respostas corretas, evidenciando que esta última está muito fácil e provavelmente precisa ser revisada, pois pode não estar discriminando adequadamente entre os estudantes que sabem o conteúdo e os que não sabem.

Tabela B.3: Porcentagem de acertos e número de questões respondidas no Teste 2: Método – WPC.

Grupos	Lembrar								
	1	2	3	4	5	6	7	8	9
Questões	1	2	3	4	5	6	7	8	9
Repostas	3	5	4	4	22	7	20	18	11
Acertos %	0	20	75	0	9	43	30	83	36

Grupos	Entender															
	5	6	7	8	9											
Questões	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Repostas	8	3	2	18	1	1	2	8	3	12	6	15	8	18	8	3
Acertos %	50	67	50	39	100	0	50	100	0	33	50	100	50	56	63	100

Grupos	Aplicar					
	10	11	12	13	14	15
Questões	26	27	28	29	30	
Repostas	13	7	24	17	19	
Acertos %	31	29	46	41	58	
média de acertos %						47
desvio padrão						30

¹ (<https://github.com/fzampirolli/mctest/blob/master/exam/UtilsLatex.py>)

B.4 Teste 3: Condisional – WPC

O Teste 3 sobre questões envolvendo condicionais utiliza o método *Weighted Probability of Correctness* (WPC) ou Porcentagem Ponderada de Acertos para avaliar as habilidades dos estudantes e será detalhado a seguir sua utilização nas duas turmas.

B.4.1 Criação dos testes adaptativos

Como as 18 questões usadas neste teste são novas, a Taxonomia de Bloom também é considerada para definir os pesos das variações. O arquivo `*_adaptive_test_variations.csv` é similar em todos os 5 testes adaptativos aplicados. No arquivo `*_adaptive_test.csv` gerado ao clicar em “Criar-PDF”, é considerada a porcentagem de acerto de cada questão, multiplicada por 1 se o estudante acertou ou 0 caso contrário. Esses valores são somados e então normalizados entre -5 e 5. As primeiras colunas com identificação da turma e do estudante foram omitidas a seguir. As questões foram adaptadas do Relatório Técnico de Araújo, Caceffo, Garcia e Azevedo (2020).

Arquivo `*_adaptive_test.csv`

...	2024-2-15:575:pi-Teste01-seq	2024-2-22:576:pi-Teste02-método	MeanPreviousAbilities
...	-5	-5	-5
...	-5	-5	-5
...			
...	2.092	2.158	2.125
...	2.128	2.325	2.226

B.4.2 Correção dos testes adaptativos

A Tabela B.4 apresenta a porcentagem de acertos e o número de questões respondidas no Teste 3, agrupadas de acordo com a Taxonomia de Bloom. Cada coluna representa uma categoria da taxonomia. Os grupos numerados de 1 a 6 indicam os grupos de questões, lembrando que são sorteadas 5 questões em cada teste e uma questão por grupo. Neste teste, foram 56 respondentes. O número de respostas indica quantos estudantes responderam a cada questão, enquanto a porcentagem de acertos reflete a taxa de sucesso dos estudantes ao responderem cada questão.

Tabela B.4: Porcentagem de acertos e número de questões respondidas no Teste 3: Condisional – WPC.

	Lembrar	Entender		Aplicar						Analizar		Avaliar		Criar				
Grupos	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Questões	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Respostas	17	23	9	15	11	17	8	10	12	10	4	12	14	29	26	20	26	17
Acertos %	94	30	78	87	73	94	88	90	58	100	100	25	50	72	81	20	31	65
															média de acertos % 69			
															desvio padrão 26			

B.5 Teste 4: Repetição – WPC

O Teste 4, centrado em questões relacionadas à repetição, também utiliza o método de avaliação WPC (*Weighted Probability of Correctness*) para determinar as habilidades dos estudantes, seguindo uma abordagem semelhante à apresentada na seção anterior. Nesta seção, são apresentados exclusivamente os resultados da correção deste teste aplicado às duas turmas. A Tabela B.5 não inclui uma linha de grupos devido à ausência de grupos com mais de uma questão. Assim, cada questão é tratada como um grupo individual e possui a mesma probabilidade de ser selecionada entre as 5 questões do teste. Nesta aplicação, houve um total de 50 respondentes.

Tabela B.5: Porcentagem de acertos e número de questões respondidas no Teste 4: Repetição – WPC.

Questões	Lembrar								Entender				Aplicar			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Repostas	10	14	4	9	11	11	10	14	6	11	10	10	9	13	17	11
Acertos %	30	57	100	89	64	64	60	50	33	55	60	10	22	23	71	55
																média de acertos % 53
																desvio padrão 23

B.6 Teste 5: Vetor – MLE-v0

O Teste 5, que trata de vetores, utiliza o teste adaptativo MLE (*Maximum Likelihood Estimation*) em sua primeira versão. Nesse método, a média do produto da habilidade da questão b_i é calculada, sendo multiplicada por 1 se o estudante acertou a questão ou 0 caso contrário. Essa abordagem segue os mesmos princípios aplicados nos tipos SAT e WPC, conforme demonstrado anteriormente. Neles, a habilidade média do estudante nos quatro testes anteriores é usada para determinar variações proporcionais à dificuldade das variações, considerando uma distribuição linear. Por exemplo, o estudante com menor habilidade média recebe uma variação com a menor média dos b_i , e assim por diante. Na próxima seção, será feita uma comparação com a forma clássica de *Test Information* (BAKER, 2017).

A Tabela B.6 exibe a porcentagem de acertos e o número de questões respondidas no Teste 5, agrupadas por Taxonomia de Bloom. As questões estão distribuídas entre as categorias Lembrar, Entender, Aplicar e Analisar. As colunas correspondem às diferentes categorias de Bloom e as respectivas questões. Os valores de Repostas indicam o número de respostas para cada questão, e os valores de Acertos mostram a porcentagem de acertos para cada questão. Neste teste foram 53 respondentes.

B.7 Teste 6: Matriz – MLE-v1

No Teste 6, que aborda matrizes, foi empregado uma segunda versão do método adaptativo MLE (*Maximum Likelihood Estimation*). Nessa abordagem, o conceito de *Test Information* (TI) foi

Tabela B.6: Porcentagem de acertos e número de questões respondidas no Teste 5: Vetor – MLE.

	Lembrar			Entender				Aplicar								Analisar		
Questões	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Repostas	8	22	31	32	16	9	8	25	9	27	13	13	12	11	5	6	9	9
Acertos %	50	73	35	41	44	33	50	24	56	33	54	54	67	27	0	67	33	11
																		média de acertos % 42
																		desvio padrão 19

utilizado para atribuir a melhor variação ao estudante.

A Tabela B.7 apresenta a porcentagem de acertos e o número de questões respondidas no Teste 6, agrupadas por Taxonomia de Bloom. As respostas estão distribuídas em três categorias: Lembrar, Entender e Aplicar. Cada categoria abrange um conjunto específico de chaves de questão, com seus respectivos números de respostas e percentuais de acertos. Vale ressaltar que, devido ao número limitado de variações atribuídas aos estudantes, as questões com as chaves 3077 e 268 não foram sorteadas. Foram 55 respondentes neste Teste 6.

Para obter mais detalhes sobre as correções realizadas e o método utilizado, é possível consultar a função `getHashVariationByCat()` no arquivo `UtilsLatex.py`, disponível no GitHub². Atualmente, em caso de empate nos valores de TI, uma variação é selecionada aleatoriamente dentro do intervalo +/- 0.05 do valor de TI.

Tabela B.7: Porcentagem de acertos e número de questões respondidas no Teste 6: Matriz – MLE.

	Lembrar			Entender				Aplicar				
Chaves	264	3078	3079	3077	3081	3083	3084	3085	268	3086	3087	
Repostas	43	47	50		16	9	5	44		46	15	
Acertos %	93	85	34		31	33	60	43		48	60	
												média de acertos % 54
												desvio padrão 21

A fim de aumentar a variedade de variações sorteadas (na Tabela B.8, versão 0, sempre foi atribuída a primeira variação com o mesmo TI), porém, agora quando ocorre um empate no TI, é selecionada aleatoriamente uma variação dentro de um intervalo de TI, conforme mencionado anteriormente. Como mostrado na Tabela B.8, no primeiro caso foram geradas 6 variações e no segundo caso 15 variações. Em comparação com o método WPC aplicado ao Teste 6, foram utilizadas 54 variações distintas distribuídas linearmente entre os estudantes. Nas duas turmas foram 72 estudantes matriculados. Ou seja, a probabilidade de dois estudantes receberem a mesma variação e sentarem próximos é baixa na correção do MLE-v1 e também nos métodos adaptativos WPC e SAT.

B.7.1 Descrição das questões utilizadas

Esta seção apresenta uma descrição das questões utilizadas no Teste 6, organizadas de acordo com a Taxonomia de Bloom. As Figuras B.8, B.9 e B.10 correspondem a questões da taxonomia Lembrar, com chaves 264, 3078 e 3079, respectivamente. Já as Figuras B.11, B.12, B.13 e B.14

²(<https://github.com/fzampirolli/mctest/blob/master/exam/UtilsLatex.py>).

Tabela B.8: Comparação entre as frequências das variações.

Versão 0	Frequência	Versão 1	Frequência
8	2	21	1
15	1	24	1
37	54	37	5
81	8	44	13
193	6	65	2
199	3	81	4
		108	5
		121	6
		179	1
		184	14
		186	1
		188	4
		193	1
		194	4
		199	12

representam questões da taxonomia Entender, com chaves 3081, 3083, 3084 e 3085, respectivamente. Por fim, as Figuras B.15 e B.16 correspondem a questões da taxonomia Aplicar, com chaves 3086 e 3087, respectivamente. Cada figura apresenta o enunciado da questão, a sua respectiva chave e as alternativas, em que o destaque em azul representa a alternativa correta (marcada como #0).

Diferentemente das provas avaliativas, essas questões são simples e devem ser respondidas em até 30 minutos. Mesmo assim, é possível observar, exceto nas duas primeiras questões, que muitos estudantes erram as respostas, o que pode indicar uma falta de dedicação aos estudos, conforme também apontado no Apêndice A – Acompanhamento de acesso de estudantes no Moodle. Vale lembrar que as questões utilizadas nos testes foram aplicadas pela primeira vez e melhorias deverão ser realizadas, analisando, por exemplo, a Curva Característica do Item. Além disso, a questão com chave 3078 está mal formulada, pois não existe estrutura de dados matriz em Python, mas sim lista (no curso CS1 não foi abordado bibliotecas, como Numpy).

#0264 1. Selecione qual das seguintes configurações de linhas e colunas de matriz possui maior número de elementos:

- A. #09 linhas e 9 colunas B. #28 linhas e 6 colunas C. #28 linhas e 6 colunas D. #49 linhas e 7 colunas E. #18 linhas e 5 colunas

Figura B.8: Questão da taxonomia Lembrar, com chave 264.

#3078 1. Qual é a estrutura de dados bidimensional usada para armazenar elementos em linhas e colunas em Python?

- A. #4Tupla B. #3Set (conjunto) C. #Lista D. #2Dicionário E. #0Matriz

Figura B.9: Questão da taxonomia Lembrar, com chave 3078.

#3079 1. Como acessar o elemento na segunda linha, e terceira coluna de uma matriz em Python?

- A. #m(2)(3) B. #m[1,2] C. #m[2][3] D. #m[1][2] E. #m[2,3]

Figura B.10: Questão da taxonomia Lembrar, com chave 3079.

#3081 1. Qual é a função da seguinte expressão em Python: `len(matriz)`?

- A.*₂Calcula a soma de todos os elementos da matriz B.*₄Calcula a média dos elementos da matriz C.*₆Calcula o número de linhas da matriz D.*₁Calcula o número de colunas da matriz E.*₃Calcula o número total de elementos na matriz

Figura B.11: Questão da taxonomia Entender, com chave 3081.

#3083 1. Qual é a função da seguinte expressão em Python: `len(matriz[0])`?

- A.*₄Calcula a média dos elementos da matriz B.*₂Calcula a soma de todos os elementos da matriz C.*₆Calcula o número de colunas da matriz D.*₁Calcula o número de linhas da matriz E.*₃Calcula o número total de elementos na matriz

Figura B.12: Questão da taxonomia Entender, com chave 3083.

#3084 1. Qual é a saída do seguinte código Python?

```

1  matriz = [
2    [1, 2, 3],
3    [4, 5, 6],
4    [7, 8, 9]
5  ]
6  print(matriz[1][2])

```

- A.*₃8 B.*₂7 C.*₆0 D.*₁9 E.*₃3

Figura B.13: Questão da taxonomia Entender, com chave 3084.

#3085 1. Como podemos imprimir todos os elementos de uma matriz em Python, linha por linha?

- A.*₀Usando um loop `for` para iterar sobre as linhas da matriz B.*₃Usando a função `len(matriz)` C.*₁Usando um loop `for` para iterar sobre as colunas da matriz D.*₄Usando a função `matriz.all()` E.*₂Usando a função `print(matriz)`

Figura B.14: Questão da taxonomia Entender, com chave 3085.

#3086 1. Qual é a diferença entre `matriz = [0] * 3` e `matriz = [[0] * 3]` em Python?

- A.*₄A primeira gera um erro de sintaxe, a segunda cria uma matriz 1x3 com todos os elementos iguais a 0 B.*₆O A primeira cria uma lista com 3 elementos iguais a 0, a segunda cria uma matriz 1x3 com todos os elementos iguais a 0
C.*₂A primeira cria uma matriz 1x3 com todos os elementos iguais a 0, a segunda cria uma lista com 3 elementos iguais a [0]
D.*₃A primeira cria uma lista com 3 elementos iguais a 0, a segunda cria uma matriz 3x1 com todos os elementos iguais a 0
E.*₁Não há diferença, ambas criam a mesma matriz

Figura B.15: Questão da taxonomia Aplicar, com chave 3086.

#3087 1. Qual é a saída do seguinte código Python?

```

1  matriz=[
2    [1,2,3],
3    [4,5,6],
4    [7,8,9]
5  ]
6  soma = 0
7  for i in range(len(matriz)):
8    for j in range(len(matriz[0])):
9      soma += matriz[i][j]
10 print(soma)

```

- A.*₁54 B.*₀45 C.*₄9 D.*₃15 E.*₂36

Figura B.16: Questão da taxonomia Aplicar, com chave 3087.

B.7.2 Calibração das questões

A estimativa dos parâmetros da Teoria de Resposta ao Item (TRI), ou calibração, foi introduzida na Seção 8.6.2 – Estimativa dos parâmetros do modelo TRI, utilizando o método `minimize`

da biblioteca `scipy.optimize`. Essa abordagem foi adotada devido à conveniência de ter o MCTest implementado em Python. Entretanto, nesta seção, serão apresentadas análises utilizando a linguagem R, pois essa possui diversos recursos gráficos que facilitam a visualização e interpretação dos resultados.

A Tabela B.9 contém os parâmetros obtidos a partir da calibração dos itens utilizando diferentes modelos de ajuste. Os modelos foram ajustados aos dados empregando a função `mirt()` do pacote R chamado `mirt`.

O modelo de 1 parâmetro (1PL) foi ajustado aos dados, utilizando o modelo de Rasch. Os parâmetros estimados para cada item no modelo 1PL são apresentados na coluna 1PL da tabela.

Em seguida, o modelo de 2 parâmetros (2PL) foi ajustado aos dados. Esse modelo leva em consideração tanto a habilidade dos participantes quanto a dificuldade dos itens. Os parâmetros estimados para cada item no modelo 2PL são apresentados nas colunas 2PL da tabela, com os valores correspondentes para os parâmetros de discriminação (representados por a) e dificuldade (representados por b).

Por fim, o modelo de 3 parâmetros (3PL) foi ajustado aos dados. Esse modelo considera a habilidade dos participantes, a dificuldade dos itens e a chance de acerto ao acaso. Os parâmetros estimados para cada item no modelo 3PL são apresentados nas colunas 3PL da tabela, com os valores correspondentes para os parâmetros de discriminação (representados por a), dificuldade (representados por b) e chance de acerto ao acaso (representados por c).

Alguns pontos importantes a destacar nesta tabela:

- Nos modelos 2PL e 3PL, os valores de dificuldade (parâmetro b) encontram-se fora do intervalo recomendado de -5 a 5 implementado no MCTest. Isso indica que alguns itens podem estar muito fáceis ou muito difíceis para a população avaliada;
- Os valores de discriminação (parâmetro a) também não se encontram dentro do intervalo ideal de 0,5 a 1,5. Estes valores foram estimados empiricamente após alguns testes no Colab³. Alguns itens apresentam baixa capacidade de discriminar entre indivíduos com diferentes níveis de proficiência;
- Essa falta de adequação dos parâmetros aos intervalos esperados sugere que as questões não foram bem calibradas, provavelmente devido ao baixo número de respondentes utilizado nessa análise inicial.

Portanto, serão necessários ajustes adicionais no futuro, como coletar mais dados e refinar o processo de calibração dos itens, para melhorar a qualidade psicométrica desses instrumentos. Apenas dessa forma será possível obter uma avaliação mais confiável do construto de interesse.

A Tabela B.9 foi gerada a partir do Código B.1. Para poder executar instruções em R em uma célula no Colab, é necessário antes ativar o suporte ao R com o comando `%load_ext rpy2.ipython`. Em seguida, no início de uma próxima célula, deve-se incluir a diretiva `%%R` para indicar que o conteúdo da célula será interpretado como código R. É necessário também instalar a biblioteca `mirt`

³ (https://colab.research.google.com/drive/1ka7_SR_QB4G7ZPVvH3p_E0bZEbOH1vhK).

Tabela B.9: Parâmetros da TRI nos Modelos 1PL, 2PL e 3PL, Além de Média e Desvio Padrão.

	Chaves	1PL		2PL		3PL		Estatísticas	
		b	a	b	a	b	c	Média	DP
Lembrar	264	-3.43	1.10	-3.16	1.02	-3.17	0.16	0.95	0.22
	3078	-2.22	1.00	-2.15	1.07	-1.90	0.13	0.85	0.36
	3079	0.80	1.13	0.73	1.11	0.79	0.02	0.33	0.47
Entender	3081	0.87	2.91	0.32	3.05	0.30	0.01	0.27	0.46
	3083	0.41	2.03	0.06	2.38	0.05	0.03	0.33	0.50
	3084	-1.31	-1.53	-0.56	-1.24	-0.73	0.12	0.60	0.55
	3085	0.17	0.37	0.63	0.64	1.97	0.27	0.43	0.50
Aplicar	3086	0.04	0.96	0.07	1.00	0.14	0.04	0.47	0.51
	3087	-0.82	1.18	-0.81	1.30	-0.61	0.11	0.57	0.51
RMSE		2.92		2.94		2.95			

antes de utilizá-la. Isso pode ser feito com o comando `install.packages("mirt")`. O arquivo `teste.csv` contém as respostas dos 55 estudantes nas linhas para as 9 questões nas colunas (1 para acerto e 0 para erro).

Na Tabela B.9, os valores de RMSE (*Root Mean Squared Error* - Erro Médio Quadrático) para os modelos 1PL, 2PL e 3PL são 2.92, 2.94 e 2.95, respectivamente. RMSE mede a diferença entre os valores observados e os valores previstos. Esses valores indicam que todos os modelos têm desempenho preditivo semelhante.

O Código B.1 ajusta três modelos de Teoria de Resposta ao Item (TRI) - Rasch (1 parâmetro), 2PL (2 parâmetros) e 3PL (3 parâmetros) - aos dados contidos no arquivo `teste.csv`. Para cada modelo, são extraídos os coeficientes estimados. Além disso, são calculadas a média e o desvio padrão dos acertos por questão. Todos esses resultados são então combinados em um único *dataframe* e salvos em um arquivo CSV com o nome `teste.csv_models.csv`. O resultado desse código permite comparar, na Tabela B.9, os parâmetros estimados pelos diferentes modelos de TRI, bem como as estatísticas descritivas das respostas dos participantes.

Atenção:

É importante salientar que as porcentagens de acertos das questões apresentadas nas Tabelas B.7 e B.9 são diferentes por motivos metodológicos. Na Tabela B.7, são consideradas apenas as 5 questões sorteadas para cada estudante entre as 9 possíveis. Já na Tabela B.9, para as questões não sorteadas para um determinado estudante, o valor 0 é atribuído, assumindo-se que o estudante errou a questão. Apesar dessa diferença metodológica, as duas tabelas apresentam alta correlação de acertos (aproximadamente 0,76). Isso se deve ao fato de que cada questão no Teste 6 tem a mesma probabilidade de ser sorteada para cada estudante. É importante ressaltar que os testes foram adaptativos, o que significa que a seleção das questões para cada estudante leva em consideração as habilidades demonstradas nos 5 testes anteriores. Em outras palavras, se um estudante apresenta baixo desempenho geral, as questões com menor nível de taxonomia de Bloom serão priorizadas. Portanto, ao comparar os valores apresentados nas duas tabelas, é fundamental considerar as diferenças metodológicas e o caráter adaptativo dos testes.

Curvas Características do Item

A Figura B.17 ilustra o ICC para os 9 itens de teste. O painel esquerdo mostra o ICC no modelo 1PL, o painel central mostra o ICC no modelo 2PL e o painel direito mostra o ICC no modelo 3PL. Para fins de comparação, a Figura 4.7 ilustra CCI utilizando apenas a linguagem Python. Essa figura foi obtida a partir do Colab⁴.

Além de configurar a parte do R no Colab, como visto na seção anterior, é necessário instalar a biblioteca `mirt` antes de utilizá-la. O Código B.2 gera a imagem apresentada na Figura B.17.

Ao examinar essas figuras, podemos observar diferenças distintas nas formas das curvas. No modelo 1PL, as curvas são suaves e aumentam monotonicamente, refletindo um parâmetro de discriminação uniforme em todos os itens. O modelo 2PL introduz variabilidade nas inclinações das curvas devido a diferentes parâmetros de discriminação para cada item, permitindo uma compreensão mais matizada da dificuldade do item. O modelo 3PL complica ainda mais o formato das curvas ao incluir um parâmetro de adivinhação, que leva em consideração a possibilidade de adivinhação aleatória em itens mais fáceis. Essa adição ajuda a modelar melhor a probabilidade de uma resposta correta, especialmente para estudantes com habilidades mais baixas.

A análise das CCI é útil para avaliar a qualidade dos itens do teste. Itens com curvas mais íngremes indicam maior discriminação, ou seja, são mais eficazes para diferenciar entre indivíduos com habilidades diferentes. Por outro lado, itens com curvas mais planas podem indicar menor discriminação e menor sensibilidade para avaliar as diferenças nas habilidades dos indivíduos. Portanto, a análise das CCI auxilia na identificação de itens que podem precisar de revisão ou substituição no teste.

⁴ (https://colab.research.google.com/drive/1ka7_SR_QB4G7ZPVvH3p_E0bZEbOH1vhK).

Célula do Colab:

```

1  %%R
2  # Carregar o pacote mirt, se ainda não foi carregado
3  if(!require(mirt)){install.packages("mirt")}
4
5  # Carregar os pacotes mirt e data.table
6  library(mirt)
7  library(data.table)
8
9  # Ajuste os modelos usando a função mirt()
10 f <- "teste.csv"
11 data <- read.delim(f, header = TRUE, sep = ",")
12
13 # Definir as chaves das questões
14 item_keys <- c(264, 3078, 3079, 3081, 3083, 3084, 3085, 3086, 3087)
15
16 # Substituir nomes das variáveis por chaves dos itens
17 names(data)[1:ncol(data)] <- item_keys
18
19 # Modelo de 1 parâmetro (Rasch)
20 model_1pl <- mirt(data, 1, itemtype = "Rasch", SE=TRUE, TOL='1e-10')
21 coef_1pl <- as.data.frame(coef(model_1pl, simplify = TRUE, IRTpars=TRUE)$items)
22
23 # Modelo de 2 parâmetros
24 model_2pl <- mirt(data, 1, itemtype = "2PL", SE=TRUE, TOL='1e-10')
25 coef_2pl <- as.data.frame(coef(model_2pl, simplify = TRUE, IRTpars=TRUE)$items)
26
27 # Modelo de 3 parâmetros
28 model_3pl <- mirt(data, 1, itemtype = "3PL", SE=TRUE, TOL='1e-10')
29 coef_3pl <- as.data.frame(coef(model_3pl, simplify = TRUE, IRTpars=TRUE)$items)
30
31 # Calcular a média e o desvio padrão dos acertos ignorando NAs
32 mean_score <- colMeans(data, na.rm = TRUE)
33 sd_score <- apply(data, 2, sd, na.rm = TRUE)
34
35 # Combinar os coeficientes dos modelos em um único data frame
36 all_coef <- cbind(coef_1pl, coef_2pl, coef_3pl, Mean = mean_score, SD = sd_score)
37
38 # Salvar os resultados em um único arquivo CSV
39 write.csv(all_coef, file = paste0(f, "_models.csv"), row.names = TRUE)

```

Código B.1: Exemplo de código R em uma célula do Colab para gerar a Tabela B.9.

Curvas de Informação ao Item

A Figura B.18 apresenta as Curvas de Informação ao Item (CII) para os modelos 2PL e 3PL, respectivamente, utilizando a função `mirt()` do pacote R chamado `mirt`. Nos modelos 2PL e 3PL, os itens 3081 e 3083 apresentam picos altos de informação, sugerindo que esses itens fornecem uma quantidade substancial de informação sobre a habilidade dos indivíduos em uma região específica do traço latente. Esse comportamento é típico de itens com alta capacidade discriminativa, porém, pode indicar que esses itens são mais adequados para diferenciar indivíduos com habilidades

Célula do Colab:

```

1  %%R
2  # Carregar o pacote mirt
3  library(mirt)
4
5  # Ajuste os modelos usando a função mirt()
6  f <- "teste.csv"
7  data <- read.delim(f, header = TRUE, sep = ",")
8
9  # Definir as chaves das questões
10 item_keys <- c(264, 3078, 3079, 3081, 3083, 3084, 3085, 3086, 3087)
11
12 # Substituir nomes das variáveis por chaves dos itens
13 names(data)[1:ncol(data)] <- item_keys
14
15 # Modelo de 3 parâmetros
16 model_3pl <- mirt(data, 1, itemtype = "3PL", SE=TRUE, TOL='1e-10')
17 coef_3pl <- as.data.frame(coef(model_3pl, simplify = TRUE, IRTpars=TRUE)$items)
18
19 # Configurar o gráfico para salvar como PNG
20 png("plot_3pl_ICC.png")
21
22 # Obter informações dos itens
23 theta <- seq(-4, 4, length.out = 100)
24 icc_data <- lapply(1:ncol(data), function(i) {
25   # Ajustando para obter a probabilidade correta
26   prob <- probtrace(extract.item(model_3pl, i), Theta = theta)[, 2]
27   data.frame(Theta = theta, Prob = prob)
28 })
29
30 # Plotar as curvas características dos itens
31 colors <- rainbow(ncol(data)) # Usar cores diferentes
32 plot(NULL, xlim = c(-4, 4), ylim = c(0, 1), xlab = "Theta",
33       ylab = "Probabilidade de Acerto",
34       main = "Curvas Características dos Itens para o modelo 3PL")
35
36 for (i in 1:ncol(data)) {
37   lines(icc_data[[i]]$Theta, icc_data[[i]]$Prob, col = colors[i], lwd = 2)
38 }
39
40 # Definir as chaves dos itens como legendas
41 item_keys <- colnames(data)
42 legend("topright", legend = item_keys, col = colors, lty = 1, lwd = 2)
43
44 # Fechar o dispositivo gráfico
45 dev.off()

```

Código B.2: Exemplo de código R em uma célula do Colab para gerar o gráfico CCI no modelo 3PL, ver Figura B.17.

próximas ao ponto de máxima informação do item, sendo menos eficientes em distinguir indivíduos com habilidades distantes desse ponto.

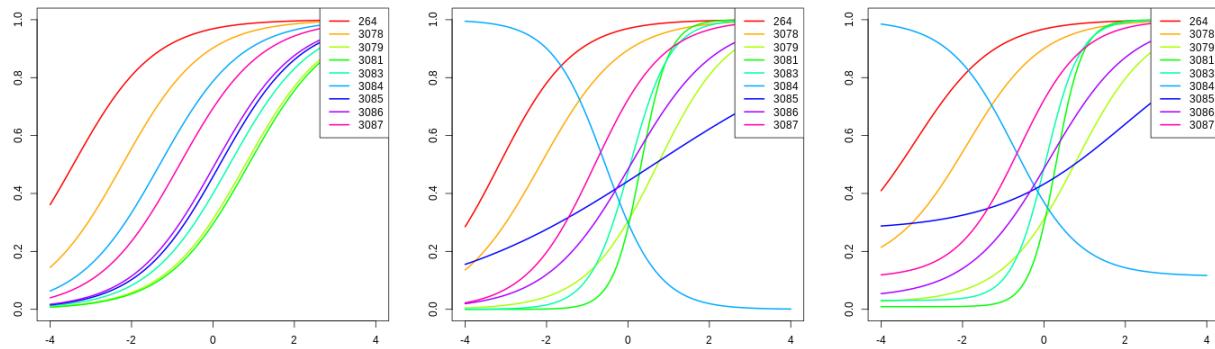


Figura B.17: Curvas Características dos Itens nos modelos 1PL (esquerda), 2PL (centro) e 3PL (direita).

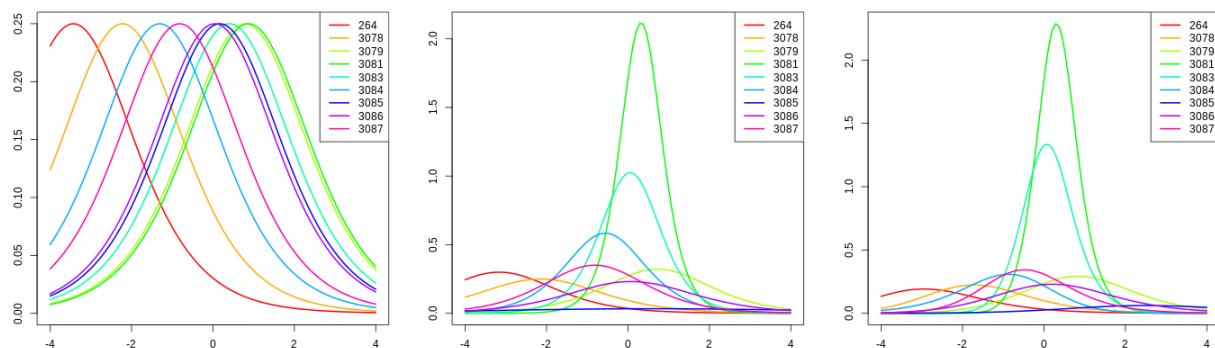


Figura B.18: Curvas de Informação ao Item nos modelos 1PL (esquerda), 2PL (centro) e 3PL (direita).

Curvas de Informação do Teste

A Figura B.19 apresenta as Curvas de Informação do Teste (CIT) para os modelos 1PL, 2PL e 3PL. Observa-se que as CITs nestes modelos atinge picos próximos da habilidade $\theta = 0$, com valores máximos de informação entre 1 e 5. As CITs se concentram na região entre $\theta = -2$ e $\theta = 2$, indicando que o teste possui maior precisão de medição em torno dessa faixa de habilidade.

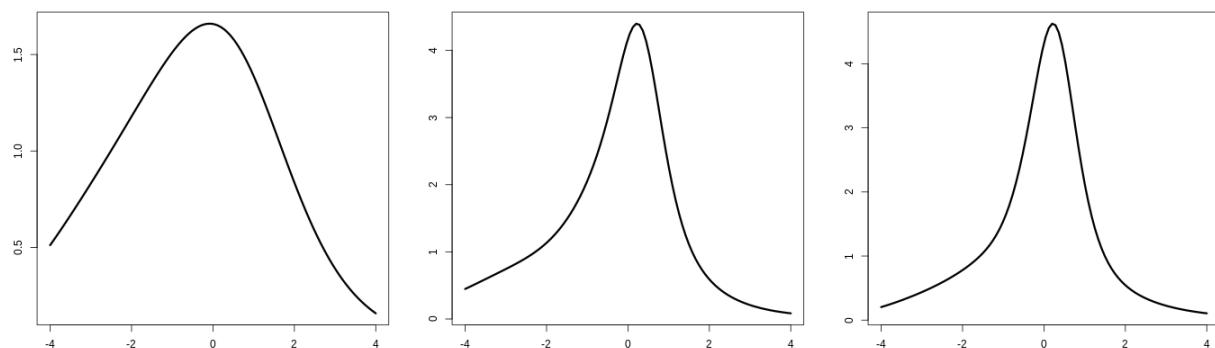


Figura B.19: Curvas de Informação do Teste nos modelos 1PL (esquerda), 2PL (centro) e 3PL (direita).

Curvas de Informação vs Erro Padrão do Teste

A Figura B.20 exibe o TIC e SEM para os modelos 1PL, 2PL e 3PL. A análise comparativa desses modelos revela características interessantes, com máxima informação em torno de $\theta = 0$, além de interseções entre as curvas do TIC e SEM perto de -2 e 2. Além disso, o TIC em forma de sino apresenta quedas mais acentuadas para 2PL e 3PL à medida que se afasta de $\theta = 0$, enquanto o SEM aumenta em torno do ponto mínimo em $\theta = 0$, indicando maior precisão de medição perto do ponto mínimo e menor precisão em traços latentes extremos.

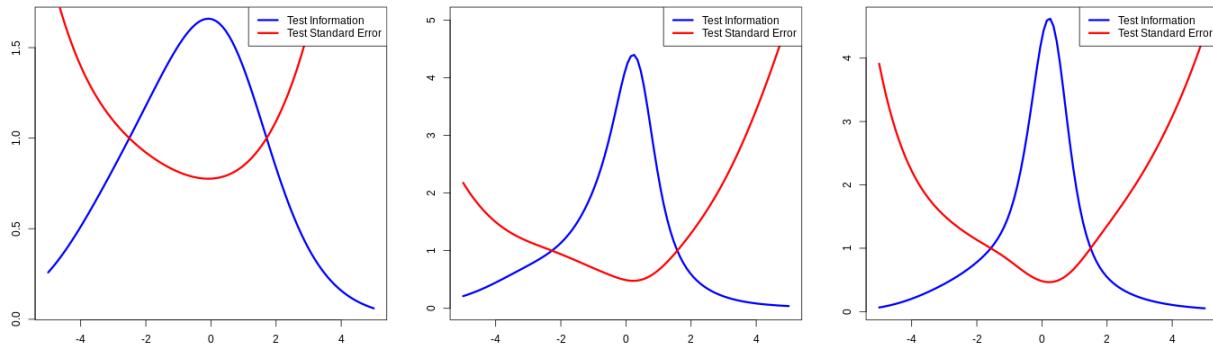


Figura B.20: Curvas de Informação vs Erro Padrão do Teste nos modelos 1PL (esquerda), 2PL (centro) e 3PL (direita).

B.8 Questionário avaliativo

Conforme detalhado no Apêndice A – Acompanhamento de acesso de estudantes no Moodle, houve um total de 72 estudantes matriculados em duas turmas: A, com 38 estudantes, e B, com 34 estudantes. A taxa de reprovação na turma A foi de 55,26%, enquanto na turma B foi de 32,35%. Isso resultou em 17 estudantes aprovados na turma A e 23 na turma B, totalizando 40 estudantes aprovados no curso. Após a semana 11 do curso, um questionário foi disponibilizado para todos os estudantes matriculados; entretanto, apenas 17 deles (23.6%) responderam, sendo 6 na turma A e 11 na turma B.

B.8.1 Questões aplicadas

A seguir, são apresentadas as questões do questionário e um resumo das respostas dos estudantes. Foi utilizada a escala Likert com os seguintes valores: 1 - Discordo totalmente; 2 - Discordo; 3 - Indiferente; 4 - Concordo; 5 - Concorde totalmente.

Questões gerais

Esta seção aborda questões gerais relacionadas ao conhecimento prévio dos estudantes sobre linguagens de programação, suas percepções sobre o aprendizado e expectativas em relação à disciplina. A Figura B.21 mostra BoxPlot (TUKEY, 1977) das respostas dos estudantes a essas questões.

É possível destacar que, nas questões Q3 e Q4, com média de 4,3, os estudantes demonstraram uma alta percepção da importância de conhecer linguagens de programação, mesmo com a presença de dois *outliers*.

- Q1. O meu conhecimento de Linguagem de Programação (LP) ANTES de cursar PI já era muito bom;
- Q2. O meu conhecimento de LP APÓS o final de PI melhorou muito;
- Q3. Conhecer LPs pode ajudar na minha ATUAÇÃO ACADÊMICA, em OUTRAS DISCIPLINAS;
- Q4. Conhecer LPs pode ajudar na minha ATUAÇÃO PROFISSIONAL;
- Q5. Aula PRESENCIAL na TEORIA (no Laboratório) é melhor para o aprendizado;
- Q6. Aula PRESENCIAL na PRÁTICA (no Laboratório) é melhor para o aprendizado.

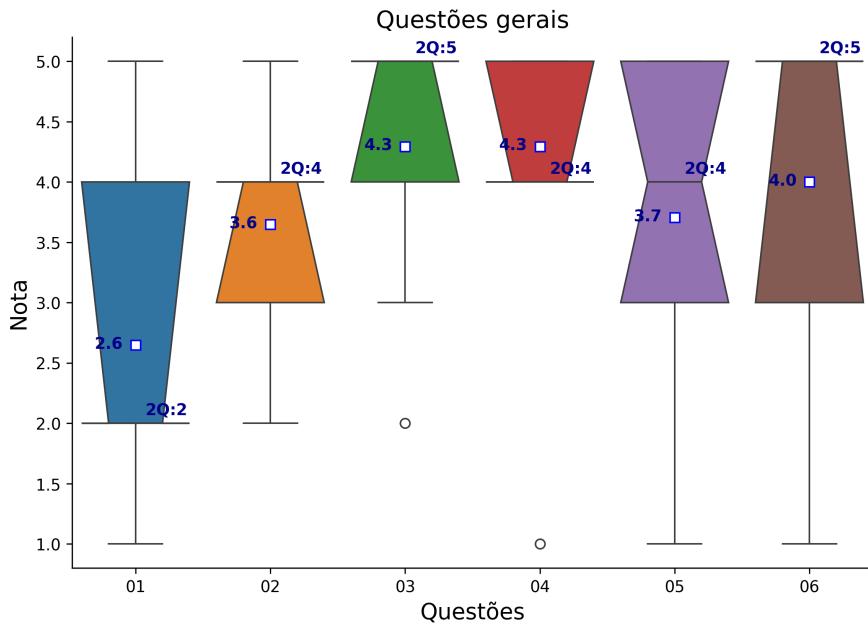


Figura B.21: Questionário com questões gerais.

Questões sobre material de ensino

A Figura B.22 apresenta questões relacionadas ao material didático utilizado na disciplina, incluindo a importância da diversidade de linguagens de programação e o *feedback* sobre o material oferecido, com destaque para a Q8 sobre o uso do Colab, que recebeu uma média de 4.3. Segue a descrição completa dessas questões:

- Q7. Acho importante que MATERIAL DIDÁTICO seja oferecido em várias LPs, a escolha do estudante;
- Q8. É importante o uso do COLAB como MATERIAL DIDÁTICO em PI;
- Q9. Acho importante os vídeos das aulas sobre o COLAB disponíveis em <<https://sites.google.com/site/fzampirolli/pi-2024-1>>;
- Q10. Eu recomendaria aos colegas essa disciplina, ofertando MATERIAL DIDÁTICO com várias LPs;
- Q11. O MATERIAL DIDÁTICO com várias LPs que foi oferecido na minha turma é muito bom.

Questões sobre avaliações

Na Figura B.23, são abordadas questões sobre as atividades e avaliações realizadas na disciplina, incluindo a opinião dos estudantes sobre o uso de plataformas de avaliação *online*. O destaque

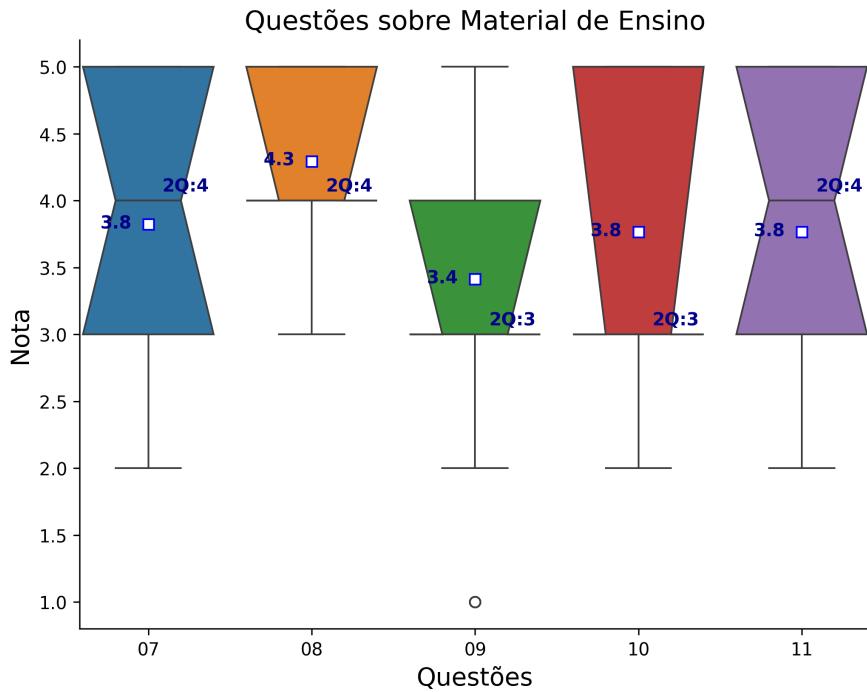


Figura B.22: Questionário com questões sobre material de ensino.

nessa parte foi que a média em todos os itens ficou próxima de 4. Segue a descrição completa dessas questões:

- Q12. Acho importante oferecer ATIVIDADES em várias LPs, a escolha do estudante;
- Q13. É muito interessante o uso do Moodle+VPL nas ATIVIDADES (EPs e Provas) com correção automática e feedback imediato;
- Q14. A AVALIAÇÃO INDIVIDUAL ajuda a diminuir plágios, logo ajuda no aprendizado;
- Q15. Eu recomendaria aos meus colegas essa disciplina com a oferta de EPs com várias LPs;
- Q16. Os EPs com várias LPs que foram oferecidos na minha turma são muito bons.

Questões sobre teste adaptativo

Por fim, na Figura B.24, são exploradas as percepções dos estudantes sobre os testes adaptativos, incluindo sua utilidade, desafios e influência nos hábitos de estudo. As respostas indicam que os estudantes consideram os Testes Semanais Individuais importantes (Q17 com média de 3.8), relatam uma melhora na confiança e compreensão dos conceitos de lógica de programação (Q18 com média de 3.5), percebem os testes adaptativos como desafiadores (Q19 com média de 4.0), afirmam que os testes adaptativos influenciaram seus hábitos de estudo (Q20 com média de 3.3) e concordam que os testes adaptativos proporcionaram desafios personalizados que atendiam às suas necessidades de aprendizagem (Q21 com média de 3.4). Destaca-se que as médias das respostas variaram entre 3.3 e 4.0, refletindo uma avaliação positiva geral em relação a esses aspectos.

- Q17. Acho importante os Testes Semanal Individual;
- Q18. Houve melhora na confiança e compreensão dos conceitos de lógica de programação;
- Q19. Os testes adaptativos foram desafiadores;
- Q20. Os testes adaptativos influenciaram nos hábitos de estudo;

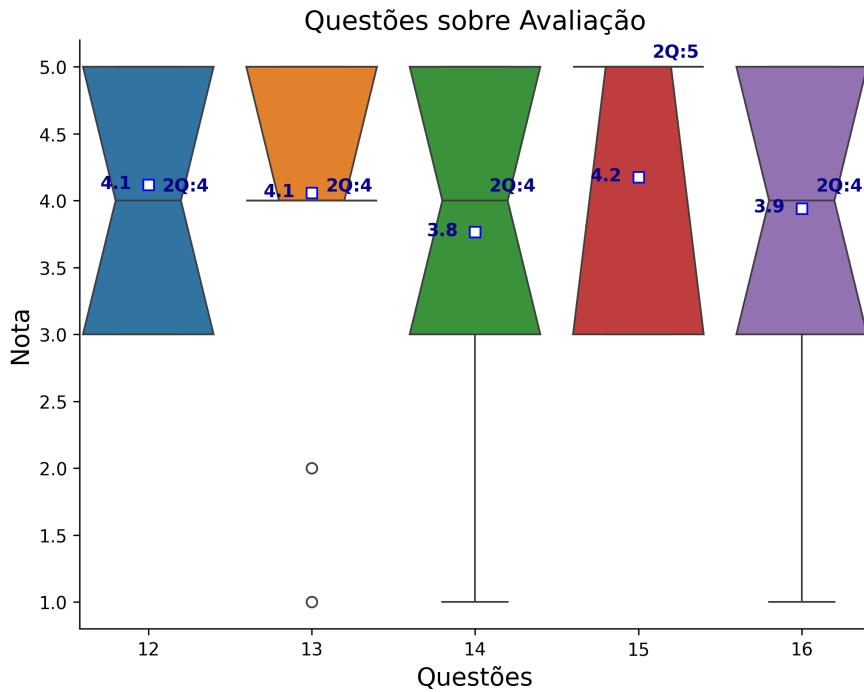


Figura B.23: Questionário com questões sobre avaliações.

Q21. Os testes adaptativos proporcionaram desafios personalizados que atendiam às necessidades de aprendizagem.

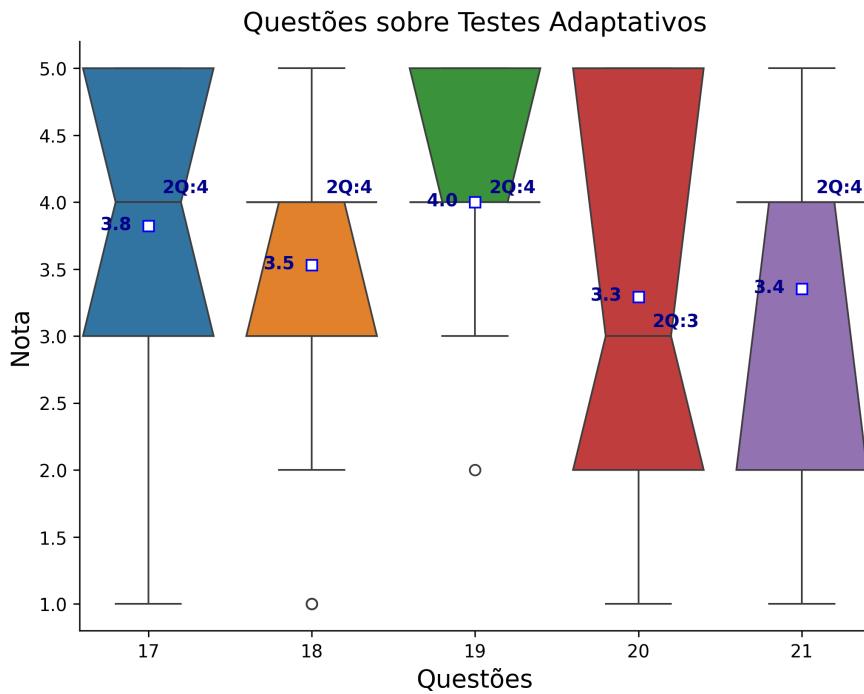


Figura B.24: Questionário com questões sobre teste adaptativo.

A Figura B.25 apresenta diversas medidas estatísticas da questão Q17, que não foram mostradas nas figuras anteriores, utilizando o modelo *Violin Plots* (HINTZE; NELSON, 1998).

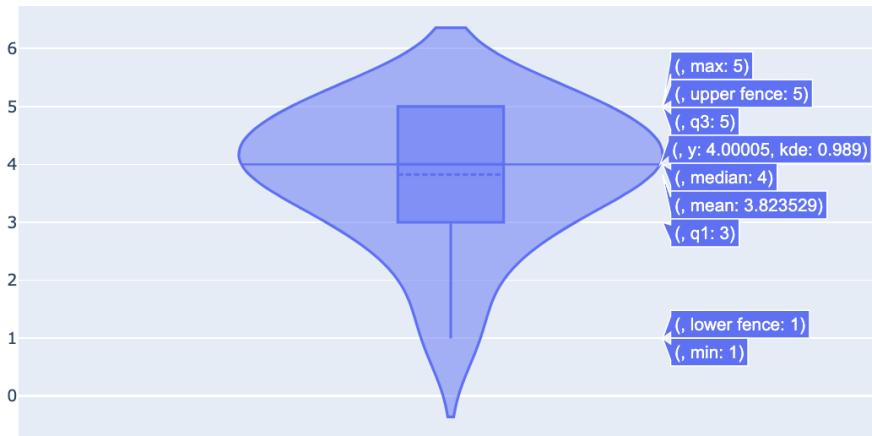


Figura B.25: Destaque para a Q17. Acho importante os Testes Semanal Individual.

B.8.2 Análise estatística dos resultados

Complementando os resultados da análise descritiva apresentada na seção anterior, a Tabela B.10 apresenta os resultados de um teste estatístico realizado em um conjunto de dados com as respostas de 17 estudantes às 20 questões de múltipla escolha e duas questões abertas. O objetivo do teste é investigar se a média das respostas dos estudantes para cada questão é significativamente maior do que 3, indicando um impacto positivo da metodologia no aprendizado dos participantes. Para essa análise, foi utilizado o teste t de uma amostra para uma média (\bar{x}), com um nível de significância de 5% (LOWRY, 2014). As hipóteses para o teste foram definidas da seguinte maneira:

H_0 : A metodologia teve um efeito neutro no aprendizado dos estudantes, ou seja, $\bar{x} \leq 3$.

H_1 : A metodologia teve um impacto positivo no aprendizado dos estudantes, ou seja, $\bar{x} > 3$.

Para decidir se rejeita ou não a hipótese nula (H_0), é comparado o p-Value resultante do teste t com o nível de significância. Se o p-Value for menor que o nível de significância (0.05), então rejeita-se a hipótese nula. Isso indica que há evidências estatísticas de que a média das respostas é significativamente maior do que 3. Na última coluna da tabela, um “X” denota a não rejeição da hipótese nula. Ou seja, as questões marcadas não apresentam significância estatística para afirmar que a metodologia teve um impacto positivo no aprendizado.

A tabela também fornece outras estatísticas relevantes, como média, mediana, p-Value, Efeito e Poder. O p-Value é a probabilidade de obter um resultado tão extremo quanto (ou mais extremo do que) o observado, assumindo que a hipótese nula é verdadeira. O Efeito mede a magnitude da diferença entre a média observada e o valor de referência (3 neste caso), expresso em unidades de desvio padrão. O Poder do teste é a probabilidade de rejeitar a hipótese nula quando ela é falsa, ou seja, a capacidade do teste de detectar um efeito significativo, se houver um.

É importante lembrar que, embora um p-Value menor que o nível de significância indique que a média das respostas é significativamente maior do que 3, isso não implica necessariamente que a metodologia teve um impacto positivo no aprendizado dos estudantes. Pode haver outros fatores não considerados no estudo que também podem influenciar as respostas dos estudantes. Além disso, o poder do teste depende do tamanho do efeito e do tamanho da amostra. Portanto, um alto poder do

teste não garante que a hipótese nula será rejeitada, mas sim que o teste tem uma alta probabilidade de detectar um efeito significativo, se houver um.

Os resultados da análise indicam que a metodologia de ensino adotada teve um efeito positivo no aprendizado dos estudantes. A maioria das questões do questionário apresentou significância estatística, com médias de respostas acima de 3. Isso sugere que os estudantes que participaram do estudo consideraram a metodologia útil e eficaz para o seu aprendizado.

Embora esses resultados sejam promissores, é importante ressaltar que novas pesquisas são necessárias para confirmar a efetividade da metodologia em larga escala. Estudos futuros, com amostras maiores e mais diversificadas, incluindo grupos de teste e controle, permitiriam análises mais robustas e generalizações mais confiáveis sobre o impacto da metodologia no aprendizado de estudantes em diferentes contextos.

Tabela B.10: Tabela com os resultados estatísticos

Questão	Média	Mediana	p-Value	Efeito	Poder	H0
Q1	2.65	2.0	0.29	-0.27	1.07	X
Q2	3.65	4.0	0.01	0.69	0.82	
Q3	4.29	5.0	0.00	1.41	0.64	
Q4	4.29	4.0	0.00	1.31	0.66	
Q5	3.71	4.0	0.05	0.52	0.87	
Q6	4.00	5.0	0.01	0.69	0.82	
Q7	3.82	4.0	0.00	0.87	0.78	
Q8	4.29	4.0	0.00	1.68	0.57	
Q9	3.41	3.0	0.20	0.32	0.92	X
Q10	3.76	3.0	0.01	0.74	0.81	
Q11	3.76	4.0	0.01	0.67	0.83	
Q12	4.12	4.0	0.00	1.30	0.66	
Q13	4.06	4.0	0.00	0.97	0.75	
Q14	3.76	4.0	0.02	0.61	0.84	
Q15	4.18	5.0	0.00	1.24	0.68	
Q16	3.94	4.0	0.01	0.75	0.81	
Q17	3.82	4.0	0.01	0.73	0.81	
Q18	3.53	4.0	0.12	0.40	0.90	X
Q19	4.00	4.0	0.00	1.15	0.70	
Q20	3.29	3.0	0.40	0.21	0.95	X
Q21	3.35	4.0	0.30	0.26	0.93	X

B.9 Considerações finais

Este apêndice descreve uma experiência conduzida na disciplina de Processamento da Informação (CS1) na UFABC durante o primeiro quadrimestre de 2024, em duas turmas. O objetivo foi realizar seis testes, sendo cinco adaptativos ao longo do curso, como avaliações formativas, com peso de 5% no conceito final, com bônus proporcional ao número de testes realizados. Os estudantes foram informados sobre esses testes adaptativos como uma estratégia motivacional aplicada na disciplina.

Os testes adaptativos utilizaram diferentes métodos, como SAT (*Semi-Adaptive Testing*), WPC (*Weighted Probability of Correctness*) e MLE (*Maximum Likelihood Estimation*), com o objetivo de avaliar e acompanhar o progresso dos estudantes de forma personalizada. A análise dos

B.9. CONSIDERAÇÕES FINAIS

resultados dos testes mostrou que os métodos adaptativos foram capazes de fornecer uma avaliação mais precisa e individualizada do aprendizado dos estudantes, permitindo que fossem propostos desafios personalizados de acordo com o nível de habilidade de cada um.

O questionário aplicado no final do curso indicou que a maioria dos estudantes considerou os testes adaptativos importantes, desafiadores e benéficos para o seu aprendizado, com impacto positivo na confiança e compreensão dos conceitos de lógica de programação. A análise estatística das respostas ao questionário revelou que a metodologia de ensino-aprendizagem-avaliação adotada, incluindo os testes adaptativos, teve um efeito positivo no aprendizado dos estudantes, com a maioria das questões apresentando significância estatística.

Embora os resultados sejam promissores, novos estudos com amostras maiores e mais diversificadas serão necessários para confirmar a efetividade da metodologia em larga escala e obter generalizações mais confiáveis. Esta experiência apresentou uma implementação bem-sucedida de testes adaptativos em uma disciplina introdutória de programação, com impactos positivos no aprendizado e engajamento dos estudantes, demonstrando ser uma estratégia motivacional eficaz que merece ser explorada em outros contextos educacionais.

Apêndice C

Sobre o SEB

Conteúdo

C.1 Configurando uma atividade no SEB	301
C.2 Considerações finais	302

Para restringir o acesso a uma atividade VPL no Moodle, foi utilizado o SEB (*Safe Exam Browser*). O arquivo de configuração do SEB está disponível para Windows. Foi utilizado o Windows 10 e a versão do SEB usada neste trabalho foi a 3.7.1.704 (safeexambrowser.org). Essa versão deve ser instalada em todas as máquinas do laboratório. Após gerar o arquivo de configuração do SEB e configurar a atividade VPL conforme descrito, basta clicar duas vezes nesse arquivo e o SEB será executado automaticamente, direcionando para a atividade VPL específica.

C.1 Configurando uma atividade no SEB

Na aba “General”, inclua a URL da atividade VPL do Moodle no campo “*Start URL*”. Se utilizar mais de uma atividade VPL do Moodle, inclua apenas a URL do curso e, na aba “*Browser*”, marque a opção “*Allow navigation back/forward in exam*”. Na aba “*Network*”, inclua a URL alternativa para a atividade, como mostrado na Figura C.1. Preste muita atenção à “*Expression*” no final da URL /vp1/*?id=12481*, pois esse número corresponde à chave da atividade VPL no Moodle. Na aba “*Exam*”, marque a opção “*Use Browser Exam Key and Configuration Key*” e copie a chave no campo “*Browser Exam Key*” somente após salvar o arquivo SEB descrito a seguir. Na aba “*Security*”, defina em “*Maximum allowed number of connected displays*” um valor maior que um, caso esteja utilizando um projetor além do monitor principal. Por fim, na aba “*Config File*”, clique em “*Save Settings As...*” para salvar um arquivo no formato SEB¹.

A expressão de filtro foi criada pelo Prof. Paulo Henrique Pisani, da UFABC, a quem o autor

¹Para mais detalhes sobre o SEB, veja https://safeexambrowser.org/windows/win_usermanual_en.html.

agradece.

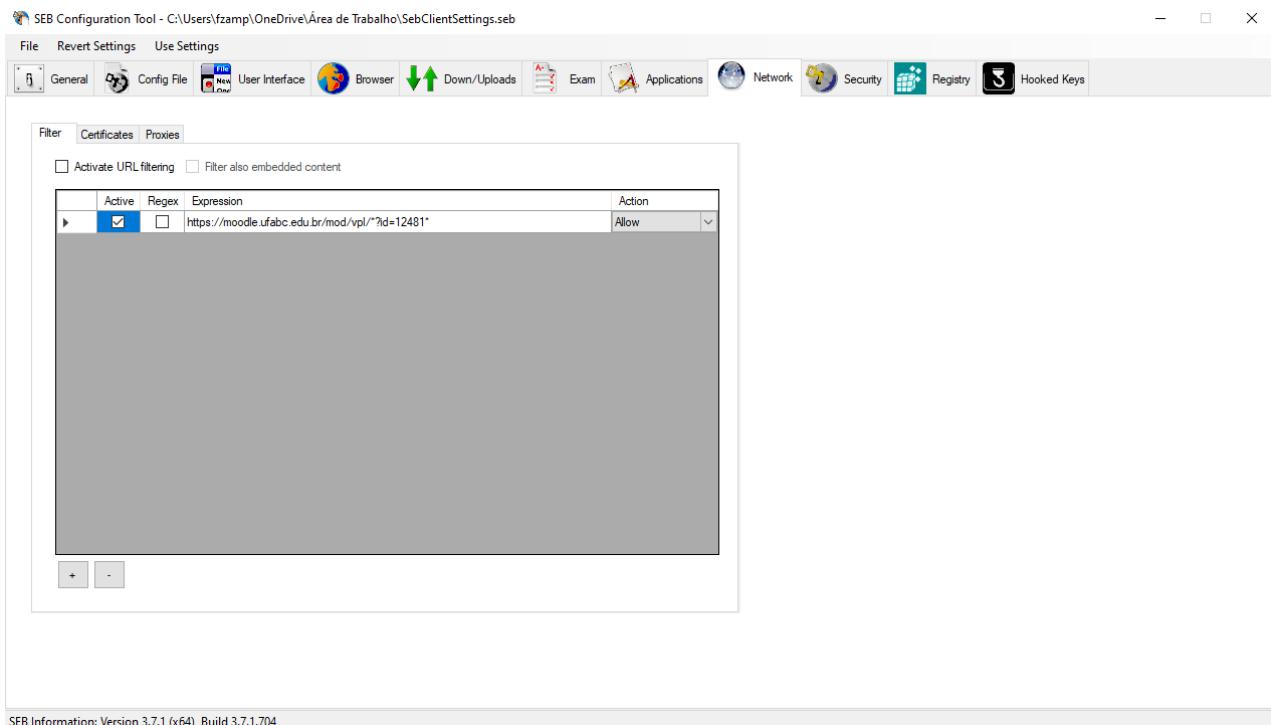


Figura C.1: Captura de tela do SEB mostrando a aba “Rede” com restrições de acesso

Essa chave deve ser copiada para a atividade VPL do Moodle. No ícone de engrenagem na disciplina do Moodle, vá para “*Configuration*”, então “*Submission restrictions*”, “*Show more*”, defina “*SEB browser required*” para SIM e cole a chave em “*SEB exam Key/s*”, conforme mostrado na Figura C.2.

Se os IPs do laboratório estiverem em uma faixa consecutiva, é possível restringir o acesso à atividade VPL apenas às máquinas do laboratório, incluindo a faixa no campo “*Allowed submission from net*” (veja a Figura C.2). Por exemplo, a faixa 111.11.11.1–62 permite todos os IPs entre 111.11.11.1 e 111.11.11.62. Caso contrário, é necessário incluir cada IP separado por vírgula ².

C.2 Considerações finais

Este apêndice descreve o uso do SEB para restringir o acesso a uma atividade VPL no Moodle. O arquivo de configuração do SEB foi gerado e aplicado à atividade VPL, garantindo que os estudantes possam acessar a atividade apenas por meio do ambiente seguro do SEB. Isso ajuda a evitar o acesso não autorizado e garante a integridade do processo de avaliação. O uso do SEB, combinado com as restrições de acesso na atividade VPL pelos IP’s, fornece uma solução robusta e segura para avaliações de programação online.

²Para mais detalhes sobre a configuração de IPs na atividade VPL, veja a documentação em vpl.dis.ulpgc.es.

C.2. CONSIDERAÇÕES FINAIS

▼ Submission restrictions

Maximum number of files

Type of work

Disable external file upload,
paste and drop external
content [Show less...](#)

This activity acts as example

Maximum upload file size

Password
[Click to enter text](#)  

Allowed submission from net

SEB browser required 

SEB exam Key/s 

Figura C.2: Captura de tela de uma atividade VPL no Moodle mostrando a chave do SEB

Apêndice D

Sobre validação de código

Conteúdo

D.1	Instalações e bibliotecas usadas	306
D.1.1	Instalação do Bandit	306
D.1.2	Instalação do Go	306
D.2	Criar um projeto simples para troca de mensagens	306
D.3	Exemplo simples de uso do Bandit	310
D.3.1	Criação do arquivo <code>example.py</code>	310
D.3.2	Instalação do Bandit	310
D.3.3	Análise do arquivo com o Bandit	310
D.3.4	Interpretação do relatório	310
D.4	Projeto servidor <code>mctest-validator</code>	312
D.4.1	Ajuste no MCTest	312
D.4.2	Comunicação entre MCTest e <code>mctest-validator</code>	314
D.5	Considerações finais	316

Este apêndice trata da validação de código Python em uma questão paramétrica antes de ser salvo no banco de dados. Esta é uma adaptação do Trabalho de Conclusão de Curso do Programa de Graduação em Ciéncia da Computação (área de concentração: Segurança da Informação), como parte dos requisitos necessários para a obtenção do título de Bacharel em Ciéncia da Computação do estudante Gabriel Tavares Frota de Azevedo. O trabalho foi desenvolvido no curso da Universidade Federal do ABC, no período de 2023-2024.

Este apêndice começa apresentando a estrutura inicial de uma validação realizada na linguagem de programação Go, também conhecida como GoLang. Em seguida, apresenta a solução proposta pelo estudante Gabriel. Por fim, o apêndice apresenta um resumo do trabalho desenvolvido.

D.1 Instalações e bibliotecas usadas

D.1.1 Instalação do Bandit

O Bandit é uma ferramenta essencial para analisar vulnerabilidades em código Python. Para instalá-lo, utilize o seguinte comando:

```
pip install bandit
```

D.1.2 Instalação do Go

O Go é uma linguagem de programação desenvolvida pelo Google. Para instalá-lo, siga os seguintes passos:

1. **Baixar o Go:** Baixe a versão 1.23.3 do Go (se necessário, utilize uma versão mais recente) em go.dev;
2. **Extrair o Arquivo:** Extraia o arquivo baixado para o diretório /usr/local;
3. **Atualizar o PATH:** Adicione o Go ao PATH do sistema para que os comandos Go estejam disponíveis no terminal;
4. **Carregar as Alterações:** Atualize o PATH no terminal atual.

```
wget https://golang.org/dl/go1.23.3.linux-amd64.tar.gz
sudo tar -C /usr/local -xzf go1.23.3.linux-amd64.tar.gz
sudo sed -i '$ a export PATH=$PATH:/usr/local/go/bin' ~/.bashrc
source ~/.bashrc
```

D.2 Criar um projeto simples para troca de mensagens

Para criar um projeto Go simples que aplica requisições REST usando o método POST, seguir os passos abaixo. Este exemplo ilustrará como configurar um servidor básico que aceita dados em formato JSON via POST e retorna uma resposta.

Estrutura do projeto

Criar uma estrutura de diretório e arquivos como apresentado na Figura D.1.

Passo 1: Inicializar o projeto

1. Criar um novo diretório para o projeto:

```
mkdir simple-rest-api
cd simple-rest-api
```

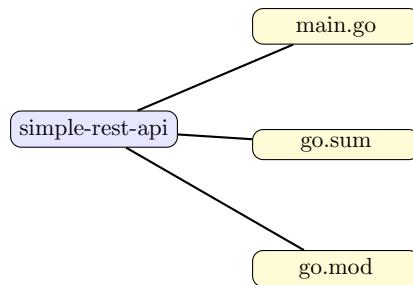


Figura D.1: Diretório e arquivos de `simple-rest-api`.

2. Inicializar um novo módulo Go:

```
go mod init simple-rest-api
```

3. Adicionar a dependência do Gin:

```
go get github.com/gin-gonic/gin
```

Esse comando baixa e integra a biblioteca no ambiente de desenvolvimento, tornando possível utilizar as funcionalidades do Gin, como roteamento, manipulação de requisições HTTP e construção de APIs.

Passo 2: Criar o arquivo `main.go`

O Código D.1 (arquivo `main.go`) define uma API simples utilizando a biblioteca Gin. A estrutura `Message` representa o formato JSON esperado para a requisição. O servidor disponibiliza uma rota POST em `/message`. Quando a rota é acessada, o código decodifica o corpo da requisição para a estrutura `Message`. Caso a validação do JSON falhe (campo `text` ausente ou estrutura inválida), o servidor retorna um erro (*HTTP Status Code 400 Bad Request*) com detalhes do erro. Se a validação for bem-sucedida, o servidor processa a mensagem e retorna uma resposta (*HTTP Status Code 200 OK*) confirmando o recebimento do texto.

Passo 3: Executar o servidor

Para executar o servidor, usar o seguinte comando:

```
go run main.go
```

Isso iniciará um servidor na porta 8080.

Código Go:

```

1 package main
2 import (
3     "net/http"
4     "github.com/gin-gonic/gin"
5 )
6 type Message struct {
7     Text string `json:"text" binding:"required"`
8 }
9 func main() {
10    // Criar um roteador Gin
11    r := gin.Default()
12    // Definir a rota para o método POST
13    r.POST("/message", func(c *gin.Context) {
14        var msg Message
15        // Bind JSON recebido à estrutura Message
16        if err := c.ShouldBindJSON(&msg); err != nil {
17            c.JSON(http.StatusBadRequest, gin.H{"error": err.Error()})
18            return
19        }
20        // Retornar uma resposta com a mensagem recebida
21        c.JSON(http.StatusOK, gin.H{"received": msg.Text})
22    })
23    // Executar o servidor na porta 8080
24    r.Run(":8080")
25 }
```

Código D.1: Exemplo de código Go para `main.go`.**Passo 4: Testar a API**

Testar a API usando `curl`. A seguir está um exemplo usando `curl`:

```
curl -X POST http://localhost:8080/message \
-H "Content-Type: application/json" \
-d '{"text": "Hello, World!"}'
```

Resposta esperada

Se tudo estiver configurado corretamente, a resposta deverá ser esta:

```
{
    "received": "Hello, World!"}
```

O diagrama apresentado na Figura D.2 ilustra a interação entre um cliente e um servidor de API durante uma requisição POST para a rota final `/message`. O cliente envia um JSON contendo a mensagem `"text": "Hello, World!"`. O servidor valida o formato JSON recebido. Se ocorrer um erro de validação, o servidor responde com um erro 400 (*Bad Request*) informando que o campo

D.2. CRIAR UM PROJETO SIMPLES PARA TROCA DE MENSAGENS

`text` é obrigatório. Caso a validação seja bem-sucedida, o servidor processa a mensagem e retorna uma resposta 200 (*OK*) com a mensagem recebida, confirmando o processamento. Veja a seguir uma captura de saída do servidor com a primeira requisição POST correta e duas requisições com erros:

Saída do servidor:

```
[GIN] 2024/11/15 - 10:09:12 | 200 | 2.730988ms | ::1 | POST  "/message"
[GIN] 2024/11/15 - 10:09:51 | 400 | 101.305µs | ::1 | POST  "/message"
[GIN] 2024/11/15 - 10:10:16 | 400 | 98.401µs | ::1 | POST  "/message"
```

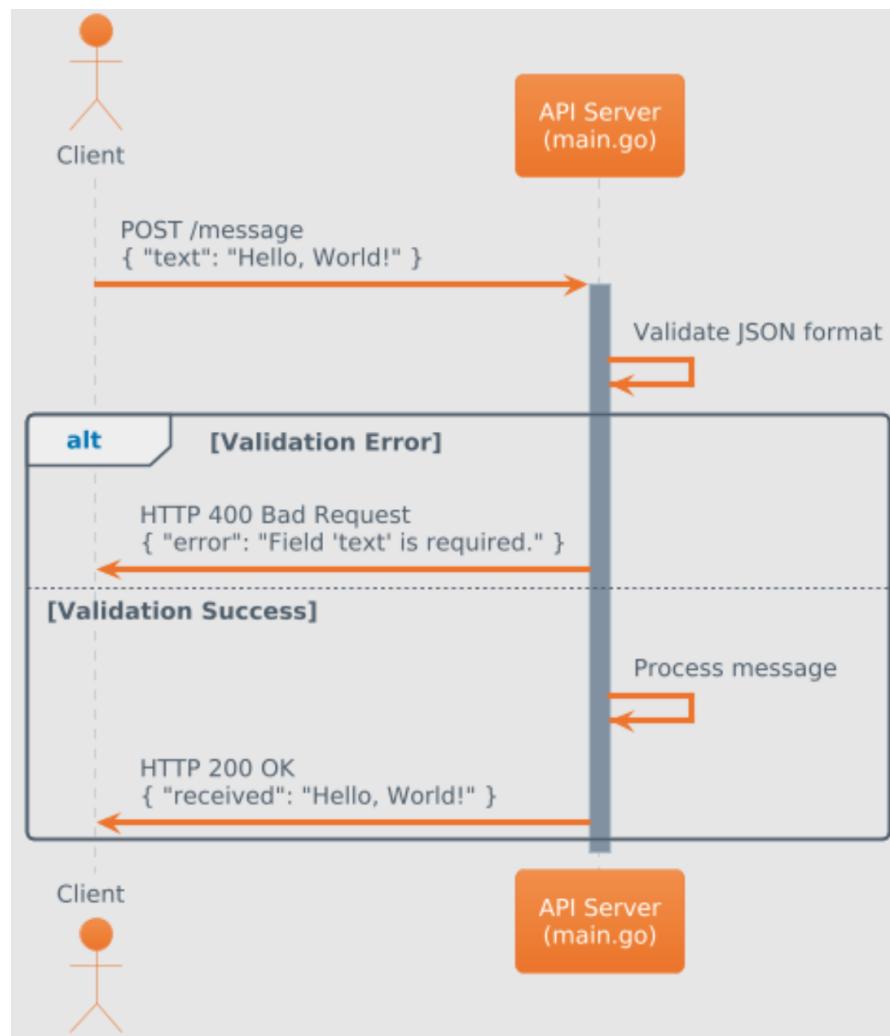


Figura D.2: Diagrama de sequência ilustrando a interação cliente-servidor para criar mensagens.

Este exemplo é básico e pode ser expandido para incluir mais rotas, métodos HTTP, manipulação de erros, e integração com um banco de dados, como será apresentado na Seção D.4. Antes, será apresentado como utilizar a biblioteca Bandit para analisar segurança de código em Python.

D.3 Exemplo simples de uso do Bandit

Nesta seção, será demonstrado um exemplo básico de utilização do Bandit para análise de segurança de código em Python. O Bandit é uma ferramenta que examina *scripts* Python em busca de potenciais vulnerabilidades.

D.3.1 Criação do arquivo example.py

Primeiro, crie um arquivo chamado `example.py` com o conteúdo apresentado no Código D.2.

Código Python:

```
1 import subprocess
2
3 def unsafe_method():
4     subprocess.call('ls') # Potencialmente inseguro
```

Código D.2: Exemplo de código Python com potencial vulnerabilidade.

O código acima contém um método `unsafe_method` que utiliza o comando `subprocess.call` para executar um comando externo. Essa abordagem pode ser insegura, especialmente se as entradas do comando forem manipuladas por usuários externos.

D.3.2 Instalação do Bandit

Para instalar o Bandit, utilize o comando a seguir:

```
pip install bandit
```

D.3.3 Análise do arquivo com o Bandit

Após a instalação, execute o seguinte comando para realizar a análise de segurança no arquivo `example.py`:

```
bandit -f json -r example.py
```

Esse comando gera um relatório no formato JSON, identificando potenciais vulnerabilidades no código. O relatório apresenta detalhes, incluindo o trecho de código afetado, número da linha, links para documentação relevante, e recomendações para mitigar os riscos. O Código D.3 apresenta trecho desse JSON.

D.3.4 Interpretação do relatório

O relatório gerado pelo Bandit aplicado no arquivo `example.py`, conforme mostrado no Código D.3, indica que o uso do método `subprocess.call` no método `unsafe_method` pode introduzir vulnerabilidades no sistema. Isso ocorre porque comandos externos podem ser executados de

Código JSON:

```

1   {
2     "errors": [],
3     "generated_at": "2024-11-15T14:37:04Z",
4     "metrics": {
5       "./example.py": {
6         "CONFIDENCE.HIGH": 3,
7         "CONFIDENCE.LOW": 0,
8         "CONFIDENCE.MEDIUM": 0,
9         "CONFIDENCE.UNDEFINED": 0,
10        "SEVERITY.HIGH": 0,
11        "SEVERITY.LOW": 3,
12        "SEVERITY.MEDIUM": 0,
13        "SEVERITY.UNDEFINED": 0,
14        "loc": 3,
15        "nosec": 0,
16        "skipped_tests": 0
17      },
18      ...
19    },
20    "results": [
21      {
22        "code": "1 import subprocess\n2 def unsafe_method():\n3     subprocess.call('ls') # Potencialmente inseguro\n",
23        ...
24        ...
25        "issue_confidence": "HIGH",
26        "issue_cwe": {
27          "id": 78,
28          "link": "https://cwe.mitre.org/data/definitions/78.html"
29        },
30        "issue_severity": "LOW",
31        "issue_text": "Consider possible security implications associated with \
32                      the subprocess module.",
33        "line_number": 1,
34        "line_range": [
35          1
36        ],
37        "test_id": "B404",
38        "test_name": "blacklist"
39      },
40      ...
41    ]
42  }

```

Código D.3: Trechos do conteúdo JSON destacando potencial vulnerabilidade no Código D.2.

forma incontrolada, especialmente quando as entradas não são validadas adequadamente. A análise realizada pelo comando `bandit -f json -r example.py`, conforme o Código D.2, identificou três vulnerabilidades de baixa gravidade (`SEVERITY.LOW`), todas com alta confiança (`CONFIDENCE.HIGH`). Os alertas estão associados ao uso do módulo `subprocess` no método `unsafe_method`, conforme descrito abaixo:

- **Teste B404:** Destaca implicações de segurança relacionadas ao uso do módulo `subprocess`,

conforme detalhado no link da CWE [CWE-78](#).

- **Teste B607:** Aponta riscos no uso de caminhos parciais para executáveis, o que pode causar falhas ou comportamentos inesperados.
- **Teste B603:** Identifica a possibilidade de execução de entradas não confiáveis ao utilizar `subprocess.call`, reforçando a necessidade de medidas preventivas.

D.4 Projeto servidor mctest-validator

Este projeto foi desenvolvido pelo estudante Gabriel Tavares Frota de Azevedo e está disponível no GitHub: [mctest-validator-api](#).

A sequência de instruções apresentada no Código D.4 pode ser utilizada para instalar o projeto `mctest-validator-api` do GitHub do MCTest. A sequência instala a biblioteca `bandit`, baixa a linguagem de programação Go e baixa o projeto `mctest-validator-api` do GitHub, configura o Go e inicia o servidor.

Código Bash:

```

1 pip install bandit
2 # Instalar Go usando PPA - https://go.dev/wiki/Ubuntu
3 wget https://golang.org/dl/go1.23.3.linux-amd64.tar.gz
4 sudo tar -C /usr/local -xzf go1.23.3.linux-amd64.tar.gz
5 sudo sed -i '$ a export PATH=$PATH:/usr/local/go/bin' ~/.bashrc
6 source ~/.bashrc
7 git clone https://github.com/leirbagseravat/mctest-validator-api.git
8 cd mctest-validator-api
9 go mod tidy # Baixar e instalar as dependências
10 go run main.go > logfile.txt 2>&1 & # Executar a aplicação mctest-validator-api

```

Código D.4: Trecho de instruções para a instalação do serviço `mctest-validator-api`.

Esse conjunto de instruções apresentado no Código D.4 foi incluído no *script* de instalação `_setup-all.sh`, disponível no GitHub do projeto do MCTest. Para mais detalhes, ver a Seção 1.4 – [Instalando o MCTest](#).

D.4.1 Ajuste no MCTest

Os ajustes realizados no MCTest focaram nos serviços de criação e atualização de questões, envolvendo modificações nos arquivos `views.py` e `UtilsMCTest4.py`, localizados na pasta `topic` do projeto MCTest no GitHub github.com/fzampirolli/mctest. Durante o salvamento de uma questão, esses serviços validam os campos do formulário para assegurar que estão de acordo com os requisitos definidos. O objetivo dessas modificações é garantir que as questões estejam em conformidade com critérios de segurança e formatação.

Uma das alterações principais foi a criação do método `generateCode`, apresentada no Código D.5, que foi incorporada ao arquivo `UtilsMCTest4.py`. Esse método utiliza os parâmetros `request`, `question` e `pk`, processando o texto da questão ao dividi-lo em linhas e eliminando espaços em branco desnecessários. Cada linha tratada é armazenada em uma lista, que é posteriormente concatenada na *string* `mystr`, conforme ilustrado nas linhas 2 a 10 do código.

Código Python:

```
1  def generateCode(request, question, pk):
2      AllLines = question.splitlines()
3      for i in range(len(AllLines)):
4          if AllLines[i].replace(" ", "") == "":
5              AllLines[i] = AllLines[i][1:] + '\n'
6          else:
7              AllLines[i] = AllLines[i] + '\n'
8
9      # Concatena as linhas em uma única string sem caracteres de retorno
10     mystr = ''.join(AllLines).replace("\r", "")
11     myDef = UtilsMC.get_code(mystr, 'def')
12
13     if myDef is not None:
14         import requests
15         url = f'http://localhost:8080/mctest/validator/question/{pk}'
16         files = {'file': '\n'.join(myDef)}
17         r = requests.post(url, files=files)
18         res = r.json()
19         print(res)
20
21     # Em caso de erro, formata e exibe a resposta JSON na página de erro
22     if 'ERROR' == res['result']['status']:
23         error_message = json.dumps(res, indent=4, sort_keys=True)
24         return render(request, 'exam/exam_errors.html',
25                         {'error_json': error_message})
```

Código D.5: Método que filtra código em Python da questão e valida em `mctest-validator`.

A *string* resultante é passada para o método `get_code` (linha 11), responsável por identificar definições de funções no texto da questão. Caso sejam detectadas definições de código Python delimitadas por “[`def`:” e “”], essas linhas são enviadas para validação por meio de uma requisição HTTP POST. A URL da API é construída dinamicamente utilizando o identificador `pk`. O Código D.5 apresenta os detalhes dessa operação.

A API de validação retorna um relatório no formato JSON, que é analisado para identificar possíveis erros. Caso algum erro seja detectado, uma mensagem detalhada é exibida ao usuário em uma página de erro. O método `generateCode`, ao identificar problemas, interrompe o fluxo de execução e retorna a validação. Caso contrário, o método `formset.save()` é acionado, e o usuário é redirecionado para a URL especificada. O arquivo `exam_errors.html`, localizado na pasta `exam/template/exam`, foi modificado para incluir o tratamento do erro, caso existente, conforme apresentado no Código D.6, o qual processa o conteúdo JSON retornado pela API.

Código HTML:

```

1   ...
2   <!-- Display JSON formatted error details -->
3   {% if error_json %}
4       <!-- Actions buttons -->
5       <div class="text-right mt-4">
6           <button onclick="window.history.back()" class="btn btn-outline-primary">
7               {% trans "Back" %}
8           </button>
9           <hr>
10      </div>
11      <h4>
12          {% trans "Code between [[def: and ]] contains instructions not permitted" %}
13      </h4>
14      <h4>{% trans "Report:" %}</h4>
15      <pre>{{ error_json }}</pre>
16      ...
17  {% endif %}

```

Código D.6: Trecho de código para tratar o conteúdo JSON retornado pelo `requests` do método `generateCode`.

O método `generateCode` é utilizada nas `views QuestionCreate e UpdateQuestion` no arquivo `views.py`, antes de as alterações serem salvas no banco de dados. O Código D.7 exemplifica sua utilização em uma atualização de questão, com destaque para as linhas adicionadas.

Código Python:

```

1   ...
2   # Método criado por Gabriel Tavares Frota de Azevedo para o TCC do BCC/UFABC.
3   validation = UtilsMC.generateCode(request, question_inst.question_text, pk)
4   if validation is not None:
5       return validation
6
7   question_inst.save()
8   ...

```

Código D.7: Trecho de código incluído na `view` de atualização de questão para chamar o método `generateCode`

O propósito do método `generateCode` é garantir a segurança e integridade das questões geradas, implementando um sistema eficiente de validação automatizada.

D.4.2 Comunicação entre MCTest e mctest-validator

A interação entre o MCTest e o `mctest-validator` utiliza a biblioteca `Requests`, uma ferramenta Python que facilita a manipulação de solicitações HTTP. Após a análise sintática da questão, o MCTest realiza uma requisição REST via método POST, enviando o código Python no corpo da

solicitação para uma URL que inclui o identificador da questão.

O JSON retornado pelo `mctest-validator` contém um campo de *status*, que determina se a questão passou na validação. Em caso de sucesso, a questão é salva no MCTest. Se falhar, o usuário é redirecionado para uma página de erro que detalha as razões da falha. Este processo assegura que apenas questões válidas sejam aceitas, mantendo a qualidade e integridade do sistema.

A Figura D.3 ilustra a estrutura de arquivos e pastas do projeto `mctest-validator-api`. Esta estrutura demonstra a organização do código em diferentes diretórios, cada um responsável por uma parte específica da aplicação, como configuração, controladores, mapeadores e serviços. Essa divisão facilita a manutenção e evolução do projeto.

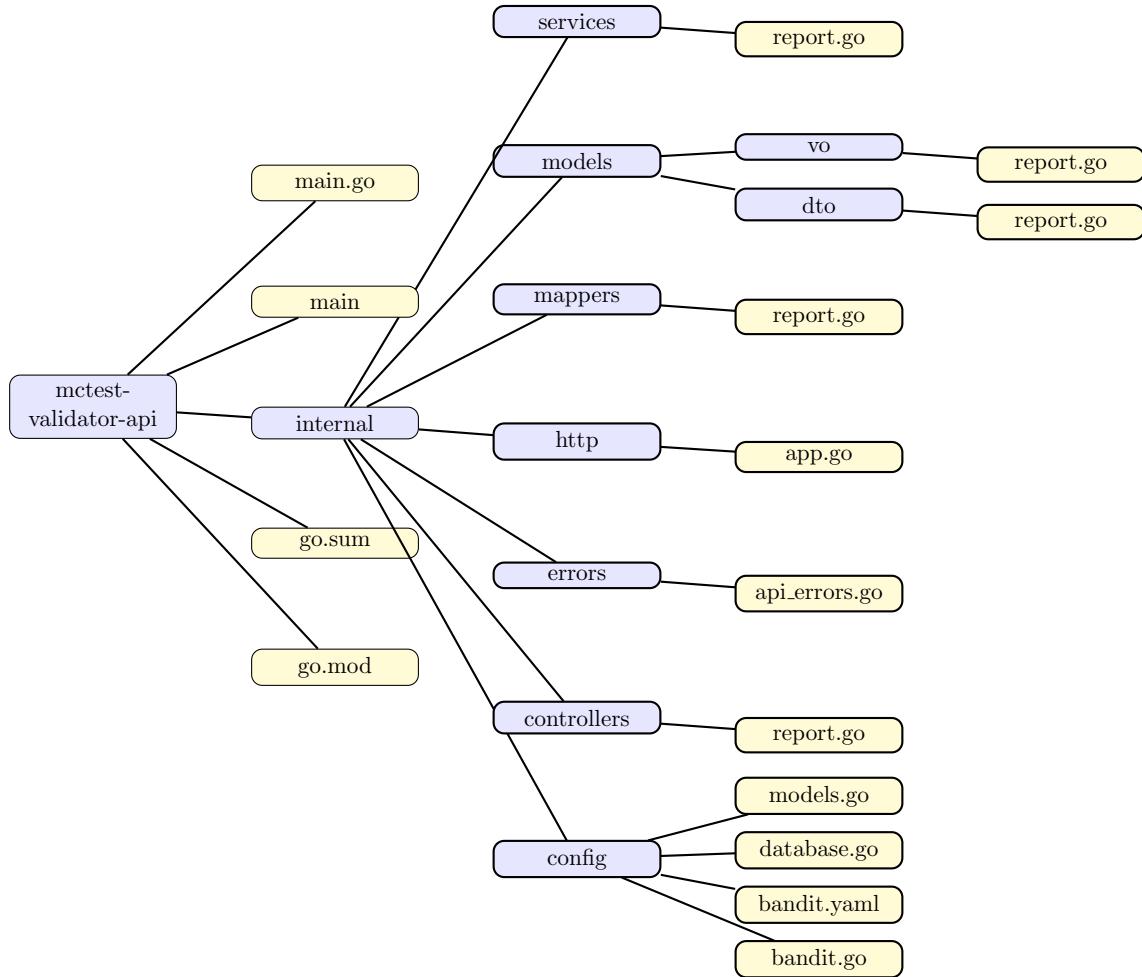


Figura D.3: Diretórios e arquivos de `mctest-validator`.

O diagrama UML apresentado na Figura D.4 mostra a sequência de interações entre diferentes componentes da aplicação `mctest-validator-api`. Ele é dividido em três camadas:

- ***Application Layer*** (Camada de Aplicação): Inclui os participantes principais `main.go`, `http/app.go` e `controllers/report.go`.
- ***Domain Layer*** (Camada de Domínio): Contém a lógica de negócios e inclui `services/report.go`, `mappers/report.go`, `models/dto/report.go` e `models/vo/report.go`.

- **Infrastructure Layer** (Camada de Infraestrutura): Responsável pelas operações de banco de dados com `config/database.go`.

O fluxo começa com `main.go` iniciando a aplicação, configurando o roteador e registrando rotas. Em seguida, a aplicação processa requisições HTTP, executa a lógica de negócios e interage com o banco de dados, retornando a resposta apropriada ao cliente.

D.5 Considerações finais

Este apêndice apresentou uma solução para a validação de código Python em um contexto educacional, priorizando a segurança e a integridade dos dados. A integração do sistema de validação ao MCTest demonstrou a viabilidade de aplicar técnicas de análise de código estático em ambientes reais de ensino.

Como desdobramentos futuros, propõe-se: expandir o sistema para incorporar o uso de diferentes ferramentas de análise estática, com o objetivo de ampliar o espectro de vulnerabilidades detectáveis; aplicar técnicas de aprendizado de máquina para aumentar a precisão na detecção de vulnerabilidades e adaptar a análise ao contexto específico de cada questão; desenvolver mecanismos capazes de fornecer *feedback* detalhado aos usuários em casos de erros; além de realizar testes de código estático, executar esse código e retornar as variáveis a serem utilizadas na questão, como os casos de teste utilizados na integração com o Moodle; e realizar testes de desempenho rigorosos, visando otimizar a eficiência e identificar possíveis gargalos no funcionamento do sistema.

D.5. CONSIDERAÇÕES FINAIS

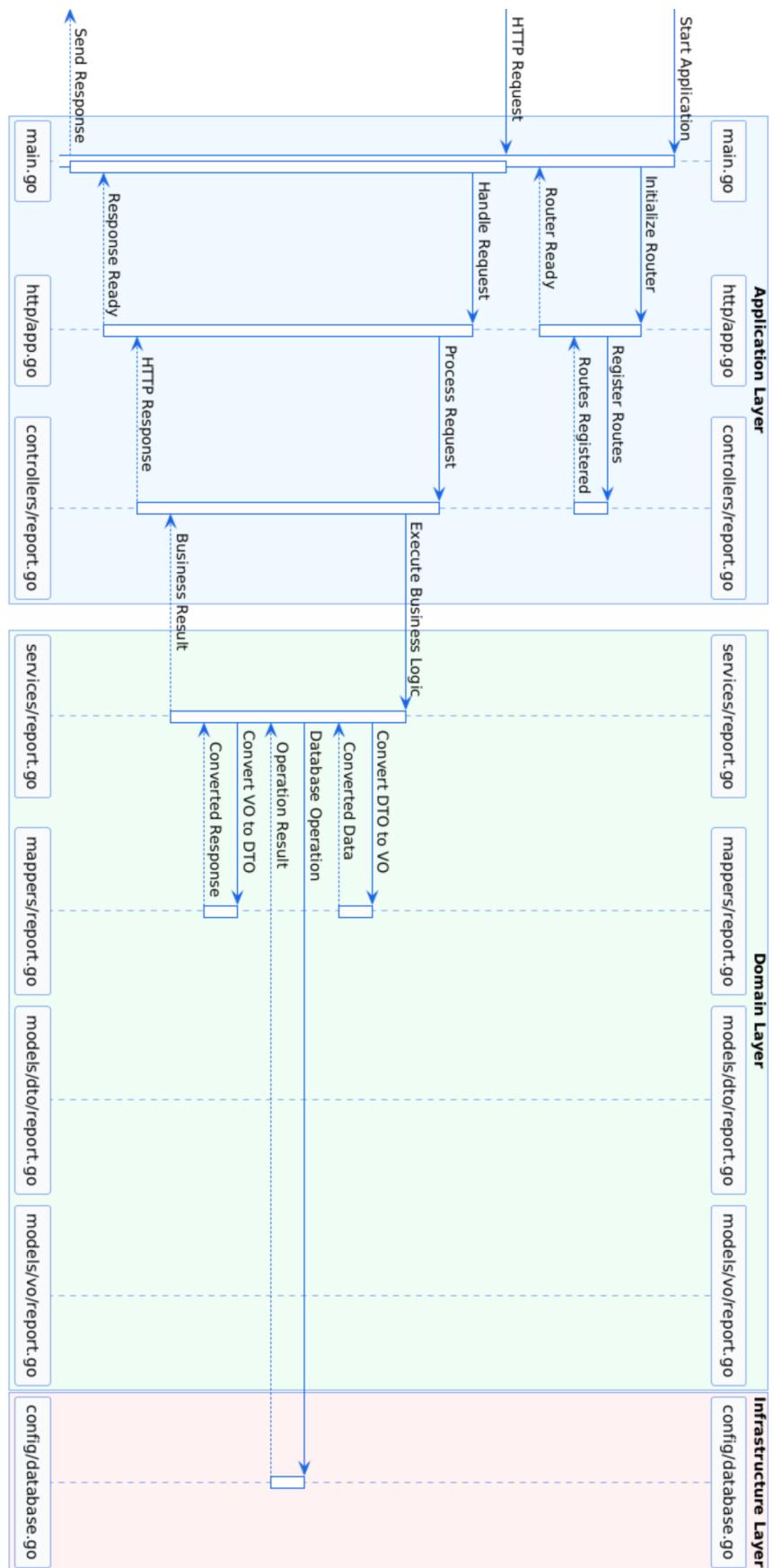


Figura D.4: Mensagens entre os diversos arquivos de `mctest-validator`.

Bibliografia

- ARAÚJO, G; CACEFFO, R; GARCIA, I; AZEVEDO, R. Estudo da taxonomia de bloom e criação de um instrumento de medição do nível de bloom no contexto de mc102. UNICAMP-IC, 2020.
- BAKER, Frank B; KIM, Seock-Ho et al. **The basics of item response theory using R.** [S.l.]: Springer, 2017. v. 969.
- BIRNBAUM, A.L. Some latent trait models and their use in inferring an examinee's ability. **Statistical theories of mental test scores**, Addison-Wesley, 1968.
- CALSAVARA, Alcides; SERRA, Ana Paula Gonçalves; ZAMPIROLLI, Francisco de Assis; CARVALHO, Leandro Silva Galvão de; JONATHAN, Miguel; CORREIA, Ronaldo Celso Messias. Método baseado nos referenciais de formação da sbc para reestruturação de descritivos de disciplinas de ciência da computação em conformidade com as DCN de 2016. In: **Workshop sobre Educação em Computação**. [S.l.: s.n.], 2018. <[doi](#)>.
- CHINA, Rodrigo Teiske; ZAMPIROLLI, Francisco de Assis. Correção automática de testes de múltipla escolha utilizando processamento de imagem e vídeo no android. In: **SICUFABC**. [S.l.: s.n.], 2014. <[url](#)>.
- CHINA, Rodrigo Teiske; ZAMPIROLLI, Francisco de Assis; NEVES, Rogério; QUILICI-GONZALEZ, José Artur. An application for automatic multiple-choice test grading on android. **REVISTA BRASILEIRA DE INICIAÇÃO CIENTÍFICA**, v. 3, p. 1–22, 2016. ISSN 2359232X.
- CORREIA, Ronaldo Celso Messias; CALSAVARA, Alcides; SERRA, Ana Paula Gonçalves; ZAMPIROLLI, F. A.; JONATHAN, Miguel. Ciência da computação: A história e as características dos cursos de ciência da computação no brasil. p. 12–16, 2018. <[url](#)>.
- DOUGHERTY, Edward R.; LOTUFO, Roberto de Alencar. **Hands-on morphological image processing**. [S.l.]: SPIE press, 2003. v. 59.
- GONZALEZ, Rafael C; WOODS, Richard C. **Processamento digital de imagens**. [S.l.]: Pearson Educación, 2009.
- HINTZE, J.L.; NELSON, R.D. Violin plots: a box plot-density trace synergism. **The American Statistician**, Taylor & Francis, v. 52, n. 2, p. 181–184, 1998.
- HOGAN, Emma; LI, Ruoxuan; RAJ, Adalbert Gerald Soosai. CS0 vs. CS1: Understanding fears and confidence amongst non-majors in introductory CS courses. In: **Proceedings of the 54th ACM Technical Symposium on Computer Science Education V.** 1. [S.l.: s.n.], 2023. p. 25–31.
- JOSKO, João Marcelo; ZAMPIROLLI, Francisco de Assis. Introducing programming logic consistently through gamification. In: **Simpósio Brasileiro de Informática na Educação**. [S.l.: s.n.], 2023.

- JW, T. Exploratory data analysis. **Reading: Addison-Wesley**, 1977.
- KOBAYASHI, Guiou; ZAMPIROLLI, Francisco de Assis; QUILICI-GONZALEZ, José Artur. Report of a distance learning course of specialization in information technology in a brazilian public university. In: **IEEE Frontiers in Education Conference**. [S.l.: s.n.], 2016. <[doi](#)>.
- LAMPORT, Leslie. Latex—a document preparation system addison—wesley. **Reading, MA**, 1985.
- LOTUFO, Roberto de Alencar. **Set of functions used in the course IA870 - Mathematical Morphology**. 2019. <<https://github.com/robertoalotufo/ia870p3>>. GitHub repository.
- LOWRY, Richard. **Concepts and applications of inferential statistics**. 2014. <<http://vassarstats.net/textbook>>. [Online; accessed 1912-Mai-2024].
- MIN, Shangchao; ARYADOUST, Vahid. A systematic review of item response theory in language assessment: Implications for the dimensionality of language ability. **Studies in Educational Evaluation**, Elsevier, v. 68, p. 100963, 2021.
- NEVES, Rogério; ZAMPIROLLI, Francisco de Assis. **Processando a informação: um livro prático de programação independente de linguagem**. São Bernardo do Campo: EdUFABC, 2017. ISBN 9788568576717.
- PINO, Juan Carlos Rodríguez-del; ROYO, Enrique Rubio; FIGUEROA, Zenón Hernández. A virtual programming lab for moodle with automatic assessment and anti-plagiarism features. 2012.
- PRESEDO, Concepción; ARMÉNDARIZ, Ana J; LÓPEZ-CUADRADO, Javier; PÉREZ, Tomás A. Calibración de ítems vía expertos utilizando moodle. **Revista Ibero-americana de Educação**, v. 69, n. 1, p. 117–132, 2015.
- SOUZA, Fábio Rezande de; ZAMPIROLLI, Francisco de Assis; KOBAYASHI, Guiou. Convolutional neural network applied to code assignment grading. In: **International Conference on Computer Supported Education**. [S.l.: s.n.], 2019. ISBN 9789897583674. <[doi](#)>.
- TEUBL, Fernando; BATISTA, Valério Ramos; ZAMPIROLLI, Francisco de Assis. Maketests: Generate and correct individualized questions with many styles. In: **International Conference on Computer Supported Education**. [S.l.: s.n.], 2021. <[doi](#)>.
- TEUBL, Fernando; BATISTA, Valério Ramos; ZAMPIROLLI, Francisco de Assis. Maketests: A flexible generator and corrector for hardcopy exams. In: **Computer Supported Education**. 1. ed. [S.l.: s.n.], 2022. p. 293–315. ISBN 9783031147562. <[doi](#)>.
- TEUBL, Fernando; ZAMPIROLLI, Francisco de Assis. Automated correction for trace tables in a CS1 course. In: **Simpósio Brasileiro de Informática na Educação**. [S.l.: s.n.], 2023.
- TUKEY, John W. Box-and-whisker plots. **Exploratory data analysis**, Addison-Wesley Reading, MA, USA, p. 39–43, 1977.
- ZAMPIROLLI, Francisco de Assis. **MCTest: como criar e corrigir exames parametrizados automaticamente**. Santo André, SP: Edição do Autor, 2023. xxiv, 244 p. (CDD 22 ed. – 005.4). 1a Edição. ISBN 978-65-00-79086-3.
- ZAMPIROLLI, Francisco de Assis; BATISTA, Valério Ramos; IRIARTE, Edson Alex Arrázola; JUNIOR, Irineu Antunes. Online assessments with parametric questions and automatic corrections: an improvement for MCTest using google forms and sheets. In: **Simpósio Brasileiro de Informática na Educação**. [S.l.: s.n.], 2020. <[doi](#)>.

ZAMPIROLLI, Francisco de Assis; BATISTA, Valério Ramos; JOSKO, João Marcelo; STEIL, Leonardo José; TREVISAN, Sandra Cristina. Experiments in selection processes of students for a crammer. In: **Latin American Conference on Learning Objects and Technologies**. [S.l.: s.n.], 2021. [<doi>](#).

ZAMPIROLLI, Francisco de Assis; BATISTA, Valério Ramos; PIMENTEL, Edson Pinheiro; BRAGA, Juliana. Facilitating the generation of parametric questions and their export to moodle. In: **IEEE Frontiers in Education Conference**. [S.l.: s.n.], 2021. [<url>](#).

ZAMPIROLLI, Francisco de Assis; BATISTA, Valério Ramos; QUILICI-GONZALEZ, José Artur. An automatic generator and corrector of multiple choice tests with random answer keys. In: **IEEE Frontiers in Education Conference**. [S.l.: s.n.], 2016. [<doi>](#).

ZAMPIROLLI, Francisco de Assis; BATISTA, Valério Ramos; RODRIGUEZ, Carla Lopes; ROCHA, Rafaela Vilela da; GOYA, Denise. Automated assessment with multiple-choice questions using weighted answers. In: **13th International Conference on Computer Supported Education**. [S.l.: s.n.], 2021. [<url>](#).

ZAMPIROLLI, Francisco de Assis; CHINA, Rodrigo Teiske; NEVES, Rogério; QUILICI-GONZALEZ, José Artur. Automatic correction of multiple-choice tests on android devices. In: **Workshop de Visão Computacional**. [S.l.: s.n.], 2015. [<url>](#).

ZAMPIROLLI, Francisco de Assis; GOYA, Denise; PIMENTEL, Edson Pinheiro; KOBAYASHI, Guiou. Evaluation process for an introductory programming course using blended learning in engineering education. **COMPUTER APPLICATIONS IN ENGINEERING EDUCATION**, v. 2018, p. 1–13, 2018. ISSN 10613773.

ZAMPIROLLI, Francisco de Assis; JOSKO, João Marcelo; TEUBL, Fernando; KURASHIMA, Celso Setsuo. A practical digital image processing course with morph.py. In: **Anais do IV Simpósio Brasileiro de Educação em Computação**. Porto Alegre, RS, Brasil: SBC, 2024. p. 304–313. Disponível em: <https://sol.sbc.org.br/index.php/educomp/article/view/28199>.

ZAMPIROLLI, Francisco de Assis; JOSKO, João Marcelo; TEUBL, Fernando; KURASHIMA, Celso Setsuo; LOPES, Renato de Avila. Teaching hands-on digital image processing with morph.py: Methods and comprehensive results. **Revista Brasileira de Informática na Educação - RBIE**, Porto Alegre, RS, Brasil, 2025. ISSN 1414-5685.

ZAMPIROLLI, Francisco de Assis; JOSKO, João Marcelo; VENERO, Mirtha Lina Fernández; KOBAYASHI, Guiou; SILVA, Francisco José Fraga da; GOYA, Denise; SAVEGNAGO, Heitor Rodrigues. An experience of automated assessment in a large-scale introduction programming course. **COMPUTER APPLICATIONS IN ENGINEERING EDUCATION**, p. 1284–1299, 2021. ISSN 10613773.

ZAMPIROLLI, Francisco de Assis; JUNIOR, Irineu Antunes; STEIL, Leonardo José; JR., Leandro Teodoro. Interactive ENEM: exams with statistics and free access. In: **Latin American Conference on Learning Objects and Technologies**. [S.l.: s.n.], 2021. [<doi>](#).

ZAMPIROLLI, Francisco de Assis; PISANI, Paulo Henrique; JOSKO, João Marcelo; KOBAYASHI, Guiou; SILVA, Francisco José Fraga da; GOYA, Denise; SAVEGNAGO, Heitor Rodrigues. Parameterized and automated assessment on an introductory programming course. In: **Simpósio Brasileiro de Informática na Educação**. [S.l.: s.n.], 2020. [<doi>](#).

ZAMPIROLLI, Francisco de Assis; QUILICI-GONZALEZ, José Artur; NEVES, Rogério. Automatic correction of multiple-choice tests using digital cameras and image processing. In: **Workshop de Visão Computacional**. [S.l.: s.n.], 2013. [<url>](#).

ZAMPIROLLI, Francisco de Assis; SATO, Cristiane M.; SAVENAGO, Heitor Rodrigues; BATISTA, Valério Ramos; KOBAYASHI, Guiou. Automated assessment of parametric programming in a large-scale course. In: **Latin American Conference on Learning Objects and Technologies**. [S.l.: s.n.], 2021. <[doi](#)>.

ZAMPIROLLI, Francisco de Assis; TEUBL, Fernando; BATISTA, Valério Ramos. A generator and corrector of parametric questions in hard copy. In: **International Conference on Information Technology : New Generations**. [S.l.: s.n.], 2019. <[url](#)>.

ZAMPIROLLI, Francisco de Assis; TEUBL, Fernando; BATISTA, Valério Ramos. Online generator and corrector of parametric questions in hard copy useful for the elaboration of thousands of individualized exams. In: **International Conference on Computer Supported Education**. [S.l.]: SCITEPRESS - Science and Technology Publications, Lda, 2019. ISBN 9789897583674. <[doi](#)>.

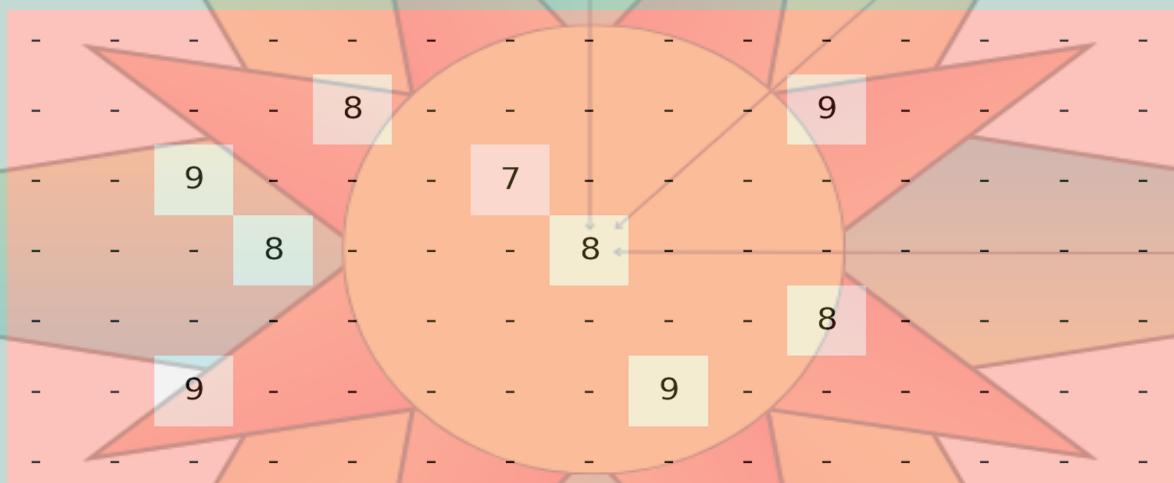
ZAMPIROLLI, Francisco de Assis; TEUBL, Fernando; KOBAYASHI, Guiou; NEVES, Rogério; ROZANTE, Luiz; BATISTA, Valério Ramos. Introductory computer science course by adopting many programming languages. In: **Simpósio Brasileiro de Informática na Educação**. [S.l.: s.n.], 2021. <[doi](#)>.

ZAMPIROLLI, Francisco de Assis; TEUBL, Fernando; PISANI, Paulo Henrique; SILVA, Thiago Alexandre Paiares e. Intelligent feedback for individualized introductory programming exercises. **Computer Applications in Engineering Education**, v. 34, n. 1, p. e70132, 2026. Disponível em: <<https://onlinelibrary.wiley.com/doi/abs/10.1002/cae.70132>>.

N

Este livro ensina como criar e avaliar exames com questões paramétricas, um tipo especial de questão que incorpora valores aleatórios em seu enunciado. O livro também relata as melhores experiências dos últimos 12 anos em avaliações automatizadas, que proporcionaram benefícios para milhares de estudantes da UFABC.

Exemplo: Na área de Lógica de Programação, o livro apresenta um programa que retorna a matriz "nordeste maior" a partir da matriz de entrada, assim como ilustrado neste exemplo na capa. Já em Processamento Digital de Imagens, a solução é alcançada por meio da adaptação de erosão ou dilatação com um elemento estruturante de tamanho 3x3.



Obs.: Esta questão possui 16 variações, sendo 8 relacionadas às direções cardeais e as opções de maior ou menor. Além disso, é possível variar as dimensões da matriz e seus respectivos valores. Para disciplinas mais avançadas, é possível aumentar a vizinhança, por exemplo, para 5x5 ou 7x7, criando ainda mais variações.



github.com/fzampirolli/mctest



ISBN 978-65-00-79086-3

S

