UNIVERSIDADE FEDERAL RURAL DO SEMIÁRIDO
UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE
MESTRADO EM CIÊNCIA DA COMPUTAÇÃO

ABRAHÃO CHRISTOPHE BEZERRA LOPES

DESENVOLVIMENTO DE UM JOGO DIDÁTICO PARA ENSINO DE PROGRAMAÇÃO ORIENTADA A OBJETOS E SUA APLICAÇÃO EM CURSOS TÉCNICOS DE COMPUTAÇÃO

MOSSORÓ – RN
2012

ABRAHÃO CHRISTOPHE BEZERRA LOPES

DESENVOLVIMENTO DE UM JOGO DIDÁTICO PARA ENSINO DE PROGRAMAÇÃO ORIENTADA A OBJETOS E SUA APLICAÇÃO EM CURSOS TÉCNICOS DE COMPUTAÇÃO

Dissertação apresentada ao Mestrado de Ciência da Computação – associação ampla entre a Universidade do Estado do Rio Grande do Norte e a Universidade Federal Rural do Semi-árido, para a obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. José Patrocínio da Silva – UFERSA.

MOSSORÓ – RN

2012

Ficha catalográfica preparada pelo setor de classificação e catalogação da Biblioteca "Orlando Teixeira" da UFERSA

L864d Lopes, Abrahão Christophe Bezerra.
Desenvolvimento de um jogo didático para o ensino de programação orientada a objetos e sua aplicação em cursos técnicos de computação. / Abrahão Christophe Bezerra Lopes. -- Mossoró, 2012.
124 f.: il.

Dissertação (Mestrado em Ciência da Computação) Área de concentração: Sistemas computacionais – Universidade Federal Rural do Semi-Árido.

Orientador: Dr. José Patrocínio da Silva.

1.Programação orientada a objetos. 2. Metodologia de ensino. 3. Jogos didáticos. I.Título.

CDD: 005.117

Bibliotecária: Vanessa de Oliveira Pessoa
CRB15/453

ABRAHÃO CHRISTOPHE BEZERRA LOPES

DESENVOLVIMENTO DE UM JOGO DIDÁTICO PARA ENSINO DE PROGRAMAÇÃO ORIENTADA A OBJETOS E SUA APLICAÇÃO EM CURSOS TÉCNICOS DE COMPUTAÇÃO

Dissertação apresentada ao Mestrado em Ciência da Computação para a obtenção do título de Mestre em Ciência da Computação.

APROVADA EM: 30 / 03 / 13 .

BANCA EXAMINADORA

Prof. Dr. José Patrocínio da Silva – UFERSA
Orientador – Presidente da Banca

Prof. Dra. Fabiana Karla de Oliveira Martins Varella – UFERSA
Primeiro Membro

Prof. Dra. Heliana Bezerra Soares – UFRN
Segundo Membro

Dedico à minha filha, Flayra Mayanne. Que seu futuro seja brilhante! À minha esposa Silvia Kaline, por estar sempre ao meu lado. E aos meus pais, Edvaldo e Francisca, que me ensinaram que os estudos são a chave para o sucesso.

AGRADECIMENTOS

À Deus, por mais esta conquista;

Ao meu orientador, Prof. Dr. José Patrocínio da Silva, por toda ajuda e tempo dedicado;

Aos membros da banca, Prof. Dra. Fabiana Karla de Oliveira Martins Varella e Prof. Dra. Heliana Bezerra Soares, por suas valiosas contribuições;

Aos professores do mestrado, por compartilhar seu conhecimento;

Aos colegas do mestrado, pela ajuda nos desafios das disciplinas e pelos momentos de alegria a cada etapa vencida;

À Prof. M.Sc. Alba Sandyra Bezerra Lopes por suas sugestões e incentivo.

E a todos aqueles que me incentivaram a prosseguir até o fim.

RESUMO

Muitos alunos de cursos de computação apresentam grandes dificuldades em disciplinas que envolvem programação de computadores, o que causa baixo rendimento ou desistências. O presente trabalho propõe uma metodologia de ensino para a disciplina Programação Orientada a Objetos (POO). Tal disciplina abrange uma variedade de conceitos amplamente diferentes da programação estruturada, que em geral é o primeiro paradigma ensinado a novos programadores. Se tais conceitos não forem bem compreendidos, o aluno não conseguirá desenvolver os programas e solucionar os problemas propostos pelo professor, podendo, inclusive, resultar em reprovação. A metodologia apresentada neste trabalho propõe o uso de jogos didáticos para auxiliar o aluno a compreender os conceitos básicos da orientação a objetos. Foi desenvolvido um conjunto de cartas, semelhante a um baralho, e vários jogos didáticos que usam estas cartas e possuem regras que refletem os principais conceitos da POO como Classe, Objeto e Herança, entre outros. A metodologia foi aplicada entre alunos que cursam o nível médio no curso Técnico em Informática do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte – IFRN – Campus Mossoró. Os resultados das experiências realizadas comprovam que a metodologia é eficaz na transmissão e fixação dos conceitos de POO. Os alunos participantes afirmaram ter compreendido os conceitos apresentados e demonstraram aumento em sua motivação concernente às disciplinas de programação.

Palavras-Chave: Metodologia de Ensino, Jogos didáticos, Programação Orientada a Objetos

ABSTRACT

Many students in computer courses have great difficulties in disciplines that involve computer programming, which causes low grades or class abandonment. This paper proposes a methodology for teaching Object Oriented Programming (OOP) discipline. This course covers a variety of widely differing concepts of structured programming, which is usually the first paradigm taught to new programmers. If these concepts are not well understood, the student will not be able to develop programs and resolve the problems proposed by the teacher, and may even result in failure. The methodology presented in this paper proposes the use of educational games to help the student understand the basic concepts of object orientation. A set of cards has been developed, and several educational games that use these cards and have rules that reflect the concepts of OOP as Class, Object and Inheritance, among others. The methodology was applied to students who attend the high school course in Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte - IFRN - Campus Mossoró. The results of experiments show that the method is effective in transmission and fixation of the concepts of OOP. Participating students said they understood the concepts presented and demonstrated an increase in their motivation regarding the programming disciplines.

Keywords: Teaching Methodology, Educative Games, Object Oriented Programming

LISTA DE FIGURAS

Figura 1: Jogo de cartas que ensina sobre biomas das árvores brasileiras 16
Figura 2: Jogo de cartas para ensinar tabela periódica 16
Figura 3: Jogo de cartas Trunfo Químico 17
Figura 4: Ludo Químico 18
Figura 5: Tabuleiro do jogo “Conhecendo a física” 18
Figura 6: Jogo de tabuleiro para ensino de biologia molecular 19
Figura 7: Jogo do DNA 19
Figura 8: Scratch: Ambiente para o ensino de programação 20
Figura 9: TutorICC:ambiente para ensino da linguagem Pascal 21
Figura 10: ProGame: um jogo para estimular o raciocínio lógico 22
Figura 11: PyGonometria: jogo desenvolvido por alunos em linguagem Python 22
Figura 12: Jogo didático para ensino de algoritmos de busca 23
Figura 13: Jogo de tabuleiro sobre Teoria dos Grafos 24
Figura 14: Blocos de montar para o ensino de Algoritmos 25
Figura 15: Jogo de cartas que ensina sobre polimorfismo e padrões de projeto 26
Figura 16: Jogo de cartas que ensina programação de microcontroladores 26
Figura 17: Cartas colecionáveis de Magic: The Gathering 28
Figura 18: Versão eletrônica do jogo Magic The Gathering 28
Figura 19: Tela do jogo de cartas on-line SHADOW ERA 29
Figura 20: Cartas colecionáveis YU-GI-OH! e POKÉMON 30
Figura 21: Jogos de cartas WAR® e SOLETRANDO® 31
Figura 22: Jogos de cartas distribuídos por ESTRELA 31
Figura 23: Jogos de cartas produzidos por HASBRO 32
Figura 24: Jogos de cartas da MATTEL 32
Figura 25: Jogos de cartas produzidos pela TOYSTER 33
Figura 26: Jogos de cartas produzidos por COPAG 34
Figura 27: Ambientação do jogo 36
Figura 28: Descrição das áreas de uma carta 38
Figura 29: Carta de classe e sua representação em UML 40
Figura 30: Exemplo de uma classe com variáveis e métodos e a representação em UML 40
Figura 31: Dois objetos criados a partir de uma classe definida pelo jogador 41

Figura 32: BlueJ: ambiente para ensino de orientação a objetos em Java 42
Figura 33: Carta que representa uma variável 44
Figura 34: Representação de valores lógicos 44
Figura 35: Carta de valor constante 45
Figura 36: Carta para modificação de valores 46
Figura 37: Mensagem de erro ao tentar modificar o valor de uma constante 46
Figura 38: Exemplo de variáveis globais e locais 48
Figura 39: Carta de método 49
Figura 40: Carta de método com parâmetro e retorno 50
Figura 41: Exemplo da troca de mensagens entre objetos usando parâmetros e retorno 51
Figura 42: Exemplo de sobrecarga de métodos 53
Figura 43: Classe com método estático 54
Figura 44: Carta de Herança 55
Figura 45: Exemplo de herança durante uma partida 56
Figura 46: Diagrama de herança em UML a partir de um conjunto de cartas 57
Figura 47: Exemplo de sobreposição de métodos e variáveis 59
Figura 48: Classe abstrata e Classe final 60
Figura 49: Cartas de edição 61
Figura 50: Cartas Objetivo 71
Figura 51: Alunos durante aplicação da experiência 74

LISTA DE QUADROS

Quadro 1: Código para criação de um objeto 42
Quadro 2: Código para declaração de uma variável em Java 43
Quadro 3: Código para declaração de uma variável em Java 45
Quadro 4: Exemplo de variáveis globais e variáveis locais 47
Quadro 5: Código de declaração de um método 50
Quadro 6: Código de um método com parâmetro e retorno 52
Quadro 7: Implementação de herança entre classes 56
Quadro 8: Exemplo de sobreposição 58

LISTA DE GRÁFICOS

Gráfico 1: Experiências anteriores com jogos didáticos 76
Gráfico 2: Formato de apresentação do jogo 77
Gráfico 3: Principal elemento do jogo 78
Gráfico 4: Nível de conhecimento sobre P.O.O. 79
Gráfico 5: Conhecimento sobre conceitos específicos 80
Gráfico 6: Conhecimento sobre P.O.O. após aplicação da metodologia 83
Gráfico 7: Conhecimentos sobre conceitos específicos após aplicação da metodologia 84
Gráfico 8: Conhecimentos sobre conceitos específicos 1 88
Gráfico 9: Conhecimentos sobre conceitos específicos 2 88
Gráfico 10: Conhecimentos sobre conceitos específicos após aplicação da metodologia 91
Gráfico 11: Alunos do 4º ano: Experiências anteriores com jogos didáticos 92
Gráfico 12: Conhecimento prévio sobre Orientação a Objetos 94
Gráfico 13: Conhecimentos sobre conceitos específicos 95
Gráfico 14: Conhecimento sobre P.O.O. após a experiência 97
Gráfico 15: Conhecimentos específicos após a experiência 98

SUMÁRIO

1 INTRODUÇÃO 13
2 MÉTODOS DIDÁTICOS PARA O ENSINO DE CIÊNCIAS 15
2.1 O estado da arte 15
2.2 Jogos e atividades eletrônicas para ensino de computação 20
2.3 Jogos e atividades não eletrônicas para ensino de computação 23
2.4 O mercado dos jogos de cartas 27
3 DESENVOLVIMENTO DO JOGO DIDÁTICO 35
3.1 Ambientação do jogo 36
3.2 Visão geral das cartas do jogo 37
3.3 Classes e objetos 39
3.4 Variáveis e constantes 43
3.5 Métodos 48
3.6 Herança 54
3.7 Cartas opcionais 60
3.8 Regras do Jogo 61
3.8.1 Regras gerais 62
3.8.2 Regras sobre cartas de classes e objetos 63
3.8.3 Regras sobre cartas de variáveis e constantes 63
3.8.4 Regras sobre cartas de métodos 64
3.8.5 Regras sobre relações de herança 64
3.8.6 Modificação de regras 65
3.9 Variações do jogo 65
3.9.1 Jogo 1: Classes, variáveis e métodos 66
3.9.2 Jogo 2: Criação de objetos 67
3.9.3 Jogo 3: Relações de herança 69
3.9.4 Jogo 4: Todos os conceitos reunidos 70
3.9.5 Jogo 5: Baseado em objetivos 71
3.9.6 Jogo 6: Reprodução de diagramas 71
3.9.7 Outros jogos 72
4 APLICAÇÃO DO “P.O.O. – O JOGO” 73
4.1 Alunos do 2º Ano 74

4.1.1 Resultados do questionário 1: conhecimentos prévios 75
4.1.2 Resultados do questionário 2: avaliação da experiência 81
4.2 Alunos do 3º Ano 85
4.2.1 Resultados do questionário 1: conhecimentos prévios 85
4.2.2 Resultados do questionário 2: avaliação da experiência 89
4.3 Alunos do 4º Ano 91
4.3.1 Resultados do questionário 1: conhecimentos prévios 92
4.3.2 Resultados do questionário 2: avaliação da experiência. 95
4.4 Sugestões e comentários dos participantes 98
5 CONCLUSÕES 100
5.1 Contribuições acadêmicas 101
5.2 Recomendações para trabalhos futuros 102
REFERÊNCIAS 103
ANEXOS 107

CAPÍTULO 1

1 INTRODUÇÃO

Nas escolas de ensino médio que oferecem cursos técnicos de informática, os alunos com idades entre 13 e 17 anos, em média, têm o primeiro contato com disciplinas de programação.

Sabe-se que um bom raciocínio lógico e matemático é essencial para o sucesso em disciplinas que envolvem programação de computadores. Infelizmente, muitos alunos nesta faixa etária, principalmente, aqueles que cursaram o ensino fundamental em escolas públicas, não conseguem desenvolver adequadamente esta habilidade.

Diante de tal questão, as disciplinas de programação de computadores se tornam de difícil compreensão para estes alunos, desmotivando-os em seus estudos e levando-os à desistência ou a resultados insatisfatórios.

Diversas pesquisas sugerem técnicas pedagógicas que auxiliam o ensino da lógica necessária para a programação de computadores, entre elas o uso de ambientes de programação especialmente desenvolvidos para o ensino, o uso de jogos eletrônicos didáticos, ou ainda através do desenvolvimento de jogos pelo próprio aluno.

Apesar de tais técnicas pedagógicas apresentarem resultados satisfatórios com alguns estudantes, em geral são utilizadas no estágio inicial da aprendizagem, onde, normalmente, trabalha-se com a programação estruturada. Se o aluno não conseguir compreender adequadamente tais conceitos, seu desempenho em disciplinas posteriores será deficiente.

Em linhas gerais, após aprender a programação estruturada, o aluno é apresentado à orientação a objetos, que traz conceitos bem diferentes daqueles já estudados e, que causam uma recepção negativa por quem está acostumado a pensar de forma estruturada.

Este trabalho tem como objetivo propor uma nova maneira de introduzir, gradualmente, os conceitos da programação orientada a objetos, de forma a minimizar o impacto da transição entre o pensamento estruturado e o pensamento orientado a objeto, através do uso de jogos com cartas.

Para alcançar tal objetivo, foram realizadas pesquisas na literatura a fim de identificar ferramentas didáticas lúdicas; desenvolveu-se um método de ensino para os conceitos de

orientação a objetos; aplicou-se a metodologia entre alunos do curso técnico em informática; foi realizada a avaliação da metodologia através de questionários e, finalmente, os dados dos questionários foram analisados.

No Capítulo 2 é apresentado o estado da arte dos jogos educativos para o ensino de ciências e de computação, sejam utilizando o meio eletrônico ou outras mídias, e uma visão do mercado dos jogos de cartas.

No Capítulo 3 são descritas as etapas do desenvolvimento de um jogo didático para o ensino de programação orientada a objetos, sua ambientação, componentes, mecânica, regras e como este pode ser aplicado em sala de aula.

No Capítulo 4 são discutidos os resultados de experiências conduzidas entre alunos com diferentes níveis de conhecimento de programação de computadores e o impacto que os jogos propostos exerceram sobre o aprendizado.

Por fim, no Capítulo 5 são apresentadas as conclusões obtidas, contribuições acadêmicas e recomendações para trabalhos futuros.

CAPÍTULO 2

2 MÉTODOS DIDÁTICOS PARA O ENSINO DE CIÊNCIAS

Diversas abordagens didáticas para o ensino de disciplinas de ciências e da área da computação, como as disciplinas de programação, por exemplo, tem sido pesquisadas, desenvolvidas e aplicadas entre os alunos.

Esta seção enumera algumas dessas metodologias e atividades, além de apresentar um breve estudo a cerca do mercado dos jogos de cartas no mundo e no Brasil.

2.1 O ESTADO DA ARTE

Segundo CORREIA e DOMINGUES (2010), o aluno assimila melhor o que é ensinado ao utilizar diversos sentidos (visão, tato, audição, etc.), ao ser incentivado a refletir, a agir e a seguir regras. É fato que aulas com atividades práticas, além de motivarem o aluno, também, o ajudam a compreender melhor a teoria por aplicá-la no mundo real.

Diversos pesquisadores se dedicam ao desenvolvimento de atividades e ferramentas educativas que promovam o lúdico entre os alunos. Aprender brincando é uma forma estimulante de se obter conhecimento.

Por exemplo, para ensinar sobre biomas das árvores brasileiras, na disciplina de biologia, CANTO e ZACARIAS (2009) utilizaram uma versão personalizada do popular jogo de cartas Super Trunfo® (GROW, 2011). As cartas deste jogo (Figura 1) trazem imagens de árvores brasileiras, seu nome científico e informações numéricas sobre a altura, diâmetro, densidade, e tempo de germinação. O jogo pode ser jogado por dois ou mais participantes. Em cada turno, o jogador escolhe uma das propriedades numéricas da carta e compara com a dos adversários. Aquele que tiver o maior valor numérico vence o turno e fica com as cartas do adversário. O jogo procede até que um jogador obtenha todas as cartas.

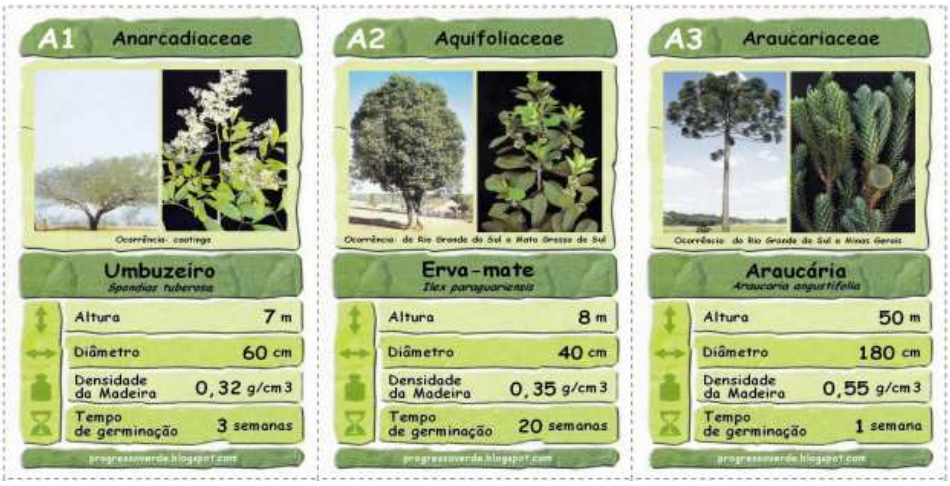


Figura 1: Jogo de cartas que ensina sobre biomas das árvores brasileiras
Fonte: CANTO e ZACARIAS (2009)

Segundo os autores, a experiência com este jogo facilitou o aprendizado sobre o assunto e despertou nos alunos o interesse por conhecer mais sobre as árvores brasileiras. Além disso, estimulou a busca por jogos em outros assuntos e disciplinas.

Outra experiência similar foi realizada por GODOI, OLIVEIRA e CODOGNOTO (2009) que desenvolveram um jogo de cartas no estilo Super Trunfo® para ensinar sobre elementos químicos da tabela periódica (Figura 2).

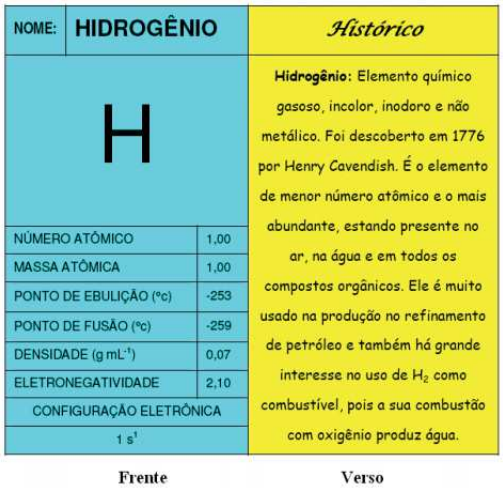


Figura 2: Jogo de cartas para ensinar tabela periódica
Fonte: GODOI, OLIVEIRA e CODOGNOTO (2009)

Segundo os autores, os alunos viam o assunto de tabela periódica como algo pouco interessante, que precisava ser decorado apenas para obter uma boa nota na prova. O jogo auxiliou os alunos a compreenderem sobre como utilizar melhor a tabela periódica, compreendendo as informações nela contidas, sua organização e propriedades, inclusive fazendo correlações entre algumas delas sem serem induzidos pelos professores.

Um jogo similar chamado Trunfo Químico (MORENO, 2010) foi adotado pela rede de ensino público estadual do Rio de Janeiro como material didático no currículo da disciplina (VENTURA, 2010). Na experiência realizada por ROCHA e FIGUEIREDO (2009), os alunos participantes tiveram rendimento superior em questões aplicadas depois do uso do jogo e afirmaram ser mais fácil aprender brincando.

Figura 3: Jogo de cartas Trunfo Químico

Fonte: VENTURA (2010)

Outro jogo na área da Química foi o desenvolvido por ZANON, GUERREIRO e OLIVEIRA (2008). O “Ludo Químico” é um jogo de corrida em que os participantes devem, ao longo do percurso, responder perguntas sobre nomenclatura de compostos orgânicos. Utiliza-se um tabuleiro, cartas (Figura 4), peões e dados. Vence o jogador ou a equipe que conseguir chegar primeiro ao fim do tabuleiro. A atividade realizada com 100 alunos teve resultados satisfatórios e os motivou a aprenderem as nomenclaturas corretas dos compostos orgânicos.

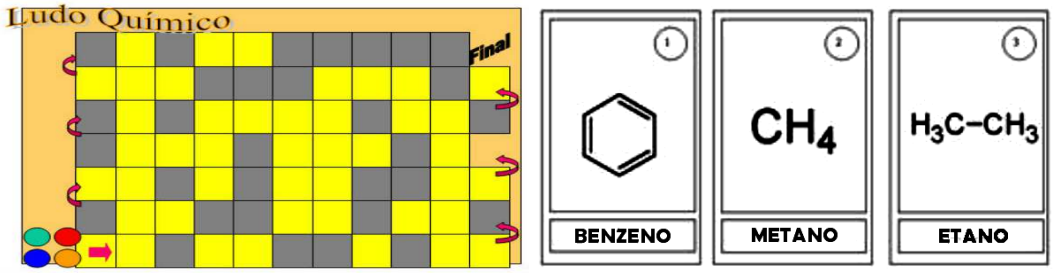


Figura 4: Ludo Químico

Fonte: ZANON, GUERREIRO e OLIVEIRA (2008)

PEREIRA, FUSINATO e NEVES (2011) desenvolveram um jogo de tabuleiro similar chamado “Conhecendo a física” (Figura 5), que traz cartas com perguntas que devem ser respondidas sempre que o jogador parar sobre o ponto de interrogação. As cartas de perguntas possuem níveis de dificuldade variados e abordam temas como hidrostática, termodinâmica, eletricidade, mecânica, óptica, magnetismo, etc.

Figura 5: Tabuleiro do jogo “Conhecendo a física”

Fonte: PEREIRA, FUSINATO e NEVES (2011)

Na área da biologia molecular existe um jogo chamado “Célula Adentro”, desenvolvido por SPIEGEL, *et al. (*2001) que utiliza um jogo de tabuleiro (Figura 6), e regras semelhantes ao popular jogo Detetive® (ESTRELA, 2011), para ensinar conceitos sobre: membrana plasmática, organelas citoplasmáticas, bactérias e protozoários, DNA, etc. No

início da partida é escolhido um caso (tema) e ao longo da partida são recebidas pistas orientam como o caso pode ser resolvido. É possível jogar de forma competitiva ou cooperativa, permitindo diversas formas de aprendizado.

Figura 6: Jogo de tabuleiro para ensino de biologia molecular
Fonte: SPIEGEL, *et al. (*2001)

O Jogo do DNA (JANN e LEITE, 2010) tem o objetivo de ensinar sobre biologia celular, molecular e genética. Utiliza peças feitas com o material emborrachado Etil Vinil Acetato (E.V.A.) para representar os componentes que formam uma molécula de DNA (Figura 7). A experiência foi aplicada entre alunos do 3º ano do ensino médio, que precisaram montar corretamente as moléculas, recebendo pistas conforme a necessidade. O jogo ajudou a desenvolver a cooperação entre os alunos, que formavam grupos, e aprimorou o conhecimento sobre o DNA.

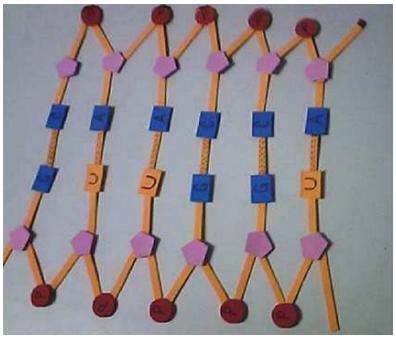


Figura 7: Jogo do DNA
Fonte: JANN e LEITE (2010)

Ao analisar os trabalhos anteriormente citados, observa-se que é possível desenvolver ferramentas de ensino nos mais diversos formatos e abordando uma grande variedade de disciplinas e assuntos.

2.2 JOGOS E ATIVIDADES ELETRÔNICAS PARA ENSINO DE COMPUTAÇÃO

Na literatura, encontram-se diversos exemplos de ambientes computacionais desenvolvidos como intuito de conduzir o aluno à compreensão Dos conceitos, por exemplo, das linguagens de programação.

Um exemplo é o *software SCRATCH* desenvolvido por RESNICK, *et al.* (2009). O ambiente apresenta uma lista de comandos (*repeat*, *if-else*, *goto*, etc.) representados como peças de um quebra-cabeças, conforme exibido na Figura 8. Tais peças podem ser encaixadas umas nas outras com objetivo de representar sequências de ações que serão realizadas por imagens e outros tipos de mídias que podem ser utilizadas.

Utilizando o *Scratch*, desenvolvido por RESNICK, *et al.*(2009), o aluno pode visualizar com facilidade o resultado da lista de ações. A ferramenta permite a criação de vídeos, animações e jogos mesmo por alunos com pouco conhecimento de programação.

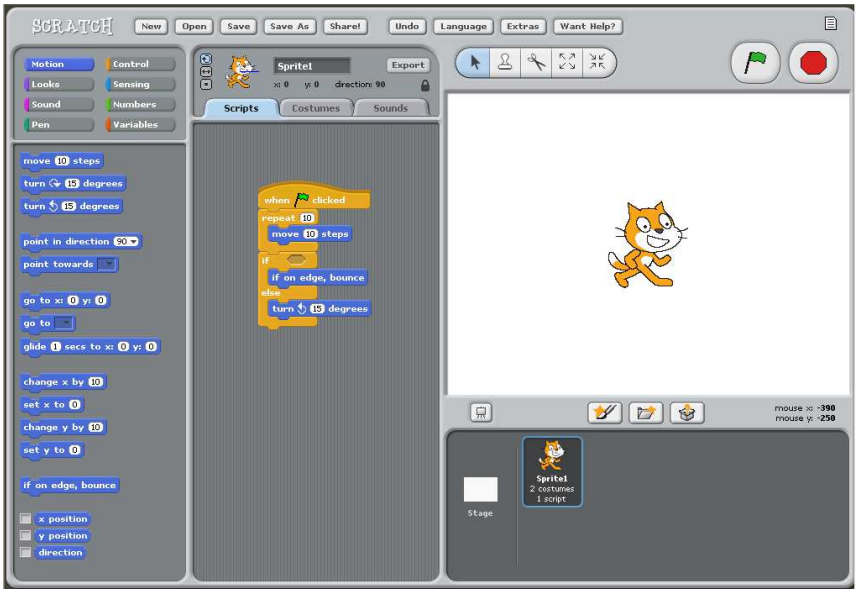


Figura 8: Scratch: Ambiente para o ensino de programação

Fonte: RESNICK, *et al.*(2009)

Outro tipo de *software* para ensino são os ambientes tutores, como o TutorICC, Figura 9, desenvolvido por PÍCCOLO, *et al.* (2010). Este tipo de ferramenta de certa forma substitui o professor, conduzindo os alunos por lições previamente definidas.

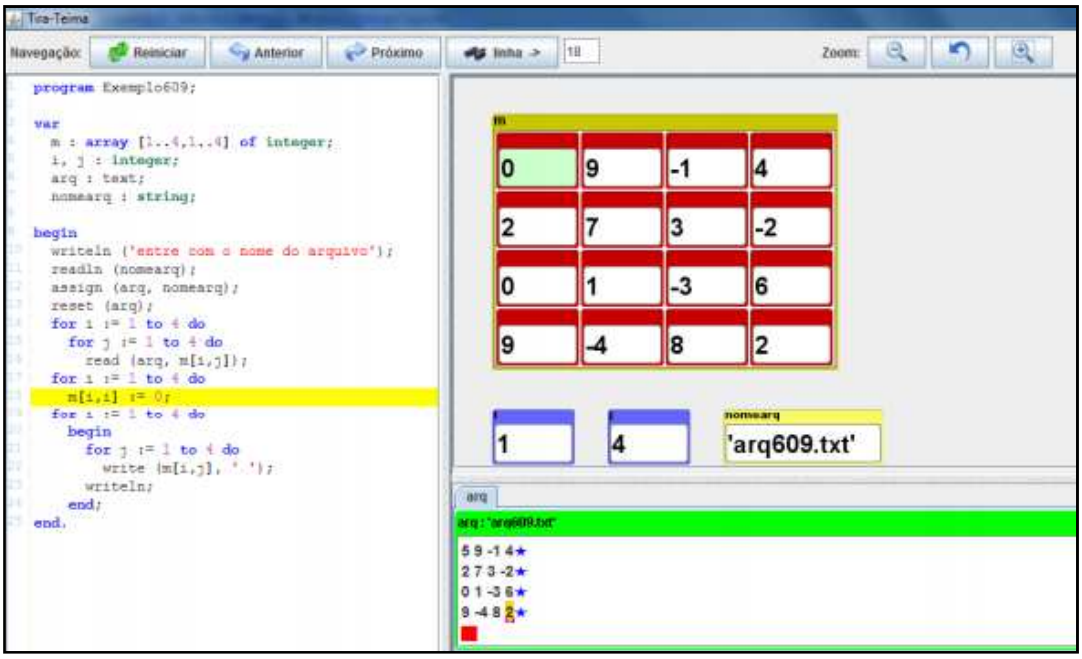


Figura 9: TutorICC:ambiente para ensino da linguagem Pascal

Fonte: PÍCCOLO, *et al.* (2010)

O ambiente é composto por diversas áreas de visualização que exibem o código-fonte, representação de variáveis, vetores e matrizes, e observações acerca do código.

Além de apresentar o conteúdo da disciplina, o TutorICC também oferece exercícios e testes para acompanhar o desenvolvimento do aluno e fazer recomendações de quais assuntos devem ser estudados em seguida.

Alguns pesquisadores utilizam jogos eletrônicos para estimular o raciocínio lógico, como SALES e DANTAS (2010) que desenvolveram o ProGame (Figura 10), um jogo eletrônico que envolve a montagem de algoritmos, compreensão de mensagens de erro e correção de código.

Figura 10: ProGame: um jogo para estimular o raciocínio lógico

Fonte: SALES e DANTAS (2010)

Conforme avança pelas fases do jogo, o aluno precisa resolver enigmas envolvendo códigos de programação que somente após solucionados permitirão que a fase seguinte seja alcançada.

Outra metodologia utilizada é o ensino de programação através do desenvolvimento de jogos pelos alunos. REBOUÇAS, *et al.* (2010) descrevem uma experiência utilizando a linguagem de programação *Python* para o desenvolvimento de jogos educativos simples, como o PyGonometria (Figura 11), cujo objetivo é selecionar o ângulo correto para completar a imagem.

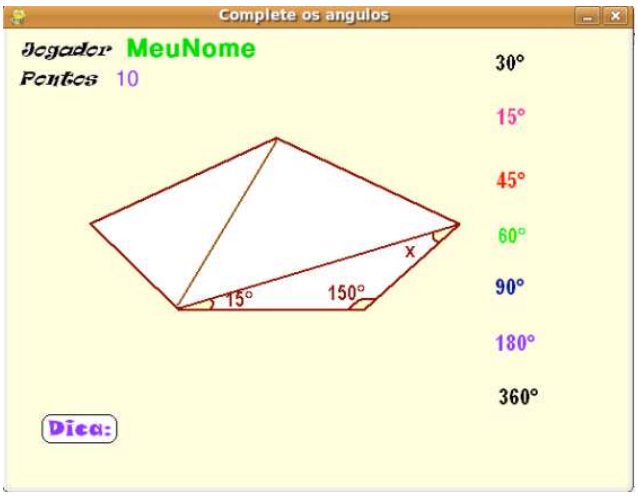


Figura 11: PyGonometria: jogo desenvolvido por alunos em linguagem Python

Fonte: REBOUÇAS, *et al.* (2010)

Todos os métodos descritos na seção 2.2 visam motivar o aluno para que obtenha melhor rendimento nas disciplinas de programação. Segundo os autores, bons resultados foram observados em cada uma das atividades.

A seção 2.3 apresenta atividades que poderão ser aplicadas entre os alunos sem a necessidade do uso de computadores.

2.3 JOGOS E ATIVIDADES NÃO ELETRÔNICAS PARA ENSINO DE COMPUTAÇÃO

Entre os trabalhos que visam o ensino de computação, destacam-se as atividades desenvolvidas por BELL, WITTEN e FELLOWS (2005). Os autores propõem atividades que ensinam números binários, representação de imagens, compressão de texto, roteamento em redes, máquinas de estados finitos, algoritmos de busca, algoritmos de ordenação, princípios de linguagens de programação, entre outros.

Utilizando materiais simples como cartões com gravuras impressas, tubos plásticos e um roteiro que descreve a atividade, este método simples permite que até crianças nas primeiras séries do ensino fundamental compreendam tais conceitos.

A Figura 12 apresenta um cartão inspirado no jogo Batalha Naval que é usado para ensinar sobre algoritmos de busca linear. Nesta variação do jogo, cada aluno recebe um cartão em que os navios possuem números em ordem aleatória. O jogador deve escolher um número e dizer a seu adversário. Em turnos alternados, cada jogador diz uma letra e o adversário revela o número do navio. Vence aquele que conseguir encontrar o navio escolhido pelo adversário primeiro.

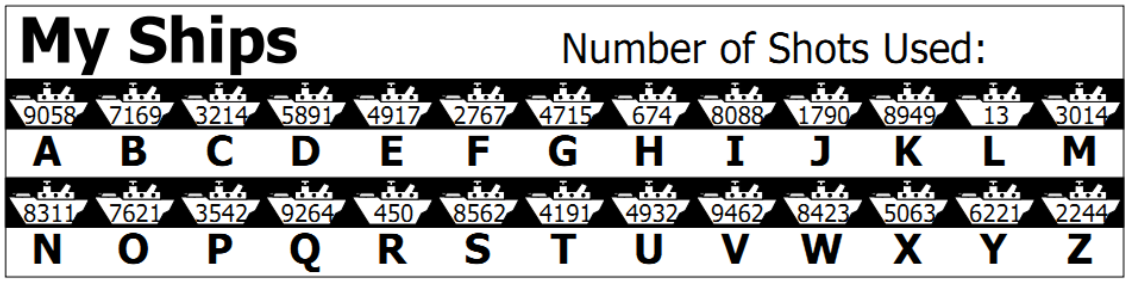


Figura 12: Jogo didático para ensino de algoritmos de busca

Fonte: BELL, WITTEN e FELLOWS (2005)

Na busca linear, os registros não estão ordenados, sendo necessário percorrer toda a lista até que se encontre o valor desejado. Após aplicar esta atividade, um novo cartão é entregue para cada participante, porém desta vez os números nos navios estarão classificados em ordem crescente, o que permite a realização de uma busca binária. Após jogar com o segundo cartão, os alunos perceberão que é mais fácil encontrar o navio escolhido, pois valores menores estarão à esquerda e valores maiores estarão à direita.

O trabalho de LIM (2007) apresenta um jogo de tabuleiro que explora conceitos da Teoria dos Grafos. O tabuleiro (Figura 13) é um mapa onde pontos representam cidades (vértices) e linhas que representam ferrovias (arestas) que interligam as cidades.

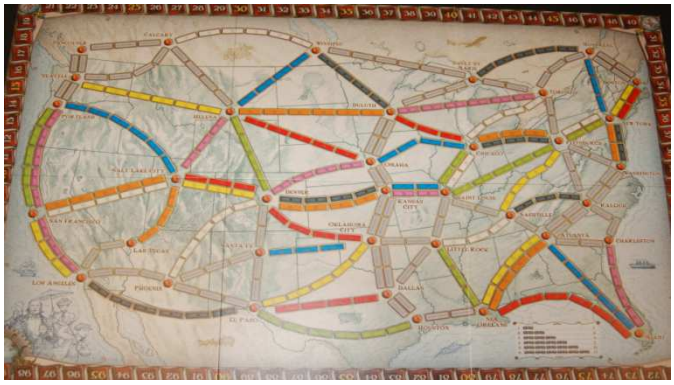


Figura 13: Jogo de tabuleiro sobre Teoria dos Grafos
Fonte: LIM (2007)

Além do tabuleiro também fazem parte do jogo: cartas com cores variadas e indicações numéricas; e marcadores usados para sobrepor as linhas do mapa. O jogador acumula pontos conforme avança por rotas que ainda não foram tomadas por nenhum adversário.

Com este jogo, o autor explora conceitos da Teoria dos Grafos como propriedades dos vértices (grau do vértice e adjacência), propriedades das arestas (peso, coloração), grafos simples e grafos múltiplos, sub-grafos, rotas, ciclos e árvores.

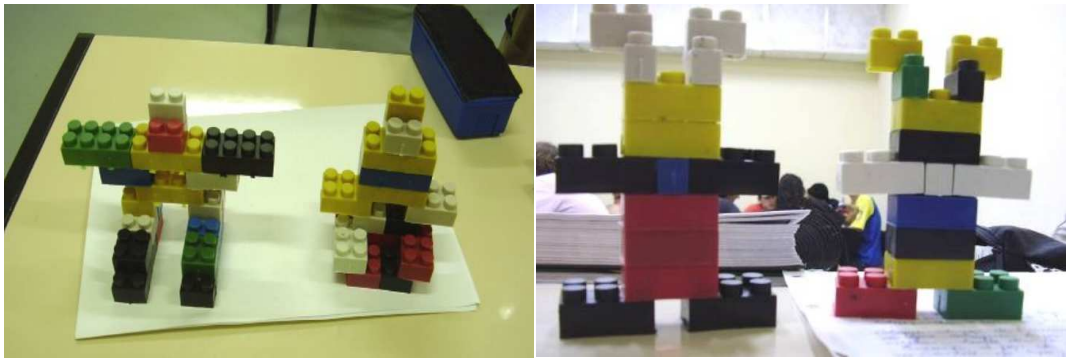
O autor propõe o jogo como introdução aos conceitos da Teoria dos Grafos e a posterior implementação de classes e algoritmos na linguagem *Java* a fim de associar o lúdico à prática.

Muitos alunos dos cursos de computação apresentam dificuldades nas disciplinas de programação por não terem obtido um bom aproveitamento em disciplinas base, como

Algoritmos e Lógica de Programação. CORREIA e DOMINGUES (2010) propõem uma atividade que utiliza blocos de montar, como LEGO®, por exemplo, para apresentar conceitos iniciais da programação.

Nesta atividade os alunos são divididos em grupos e recebem um conjunto de peças. O grupo deve montar uma figura e escrever em uma folha de papel os passos usados para reproduzi-la. As instruções são entregues a outro grupo, que sem conhecer o aspecto do objeto montado tentarão construí-lo.

A Figura 14a mostra o resultado de tentativas iniciais, quando os alunos ainda não conhecem conceitos de padronização da linguagem. Neste caso a figura reproduzida (à direita) difere completamente do original (à esquerda). Em tentativas posteriores, após aprender alguns conceitos básicos, os alunos conseguem reproduzir com perfeição qualquer figura construída pelo grupo adversário, como pode ser visto na Figura 14b. Segundo CORREIA e DOMINGUES (2010), o rendimento e interesse das turmas aumentaram após aplicação desta atividade.



(a) (b)

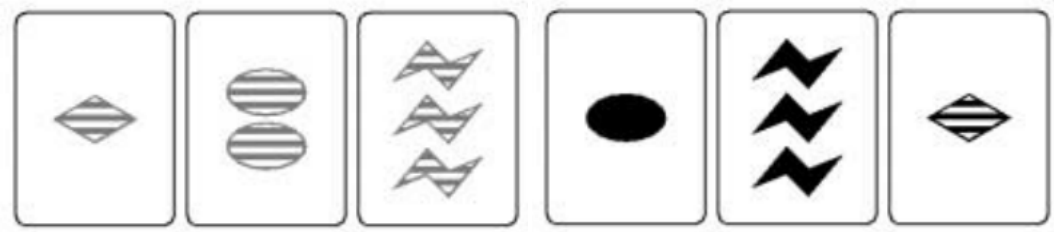
Figura 14: Blocos de montar para o ensino de Algoritmos
Fonte: CORREIA e DOMINGUES (2010)

Para ensinar conceitos de orientação a objetos e padrões de projeto, HANSEN (2004) desenvolveu o jogo *The Game of Set*, que consiste em um baralho com figuras geométricas que possuem, cores, quantidades e preenchimentos variados. O jogador tem como objetivo formar um conjunto de três cartas em que cada característica seja a mesma em todas ou que a característica seja diferente em todas.

A Figura 15a apresenta um conjunto válido, pois as três cartas escolhidas possuem o mesmo tipo de preenchimento (tracejado), e a mesma cor, além de ter quantidades diferentes e

formas geométricas diferentes. Já na Figura 15b encontra-se um conjunto inválido, pois apesar da cor ser a mesma em todas, as figuras serem diferentes em todas, o preenchimento combina apenas na primeira e segunda cartas da Figura 15b e a quantidade de símbolos apenas na primeira e terceira cartas da mesma figura.

Após conhecer o jogo, os alunos devem implementá-lo em linguagem *Java*, construindo as classes necessárias para representar as cartas e observar o polimorfismo existente. Também são introduzidos os padrões de projeto *Strategy*, *Factory* e *Flyweight*, além de utilizar o pacote gráfico AWT da linguagem *Java*.



(a) (b)

Figura 15: Jogo de cartas que ensina sobre polimorfismo e padrões de projeto
Fonte: HANSEN (2004)

COMPLUBOT (2011) desenvolveu um jogo de cartas (Figura 16) para ensinar a sintaxe utilizada na programação de micro controladores da plataforma ARDUINO®. Cada carta possui palavras chaves que representam funções, nomes de variáveis e valores. O objetivo do jogador é obter pontos ao formar comandos válidos.

Figura 16: Jogo de cartas que ensina programação de microcontroladores
Fonte: COMPLUBOT (2011)

Uma experiência foi conduzida com crianças e teve resultado positivo. O jogo despertou o interesse e facilitou discussões sobre a linguagem de programação.

BÖRSTLER e SCHULTE (2005), sugerem um jogo que usa cartões criados pelo jogador que descrevem variáveis e ações que podem ser realizadas por um objeto. O jogo funciona como uma peça teatral onde cada aluno representa um objeto e deve seguir o roteiro descrito no cartão enquanto interage com outros participantes. Este jogo requer a participação de um professor ou facilitador que possa transmitir os conceitos de orientação a objetos enquanto a partida segue seu curso. É indicado para grupos com muitos participantes.

KIM, *et al.* (2006), propuseram um jogo de cartas para ensinar a troca de mensagens entre objetos e outros conceitos da programação orientada a objetos. As cartas são construídas pelo jogador com perguntas e respostas sobre um tema escolhido previamente. Cada carta só pode realizar as perguntas nela escritas e responder perguntas para as quais possui resposta válida. Com esse jogo é possível ensinar os conceitos de classe, instância, abstração, herança, etc., sem que seja preciso usar termos técnicos que assustam os iniciantes.

A seção 2.4 apresenta um panorama de como se encontra o atual mercado de jogos de cartas no mundo e no Brasil, a fim de justificar a escolha deste tipo de mídia para o desenvolvimento deste trabalho.

2.4 O MERCADO DOS JOGOS DE CARTAS

Um dos mais populares *card game* da atualidade, *Magic: The Gathering* (WIZARDS OF THE COAST, 2011), foi publicado pela primeira vez em 1993 e deu origem ao conceito de jogos de cartas colecionáveis; uma indústria que somente nos Estados Unidos faturou cerca de 800 milhões de dólares em 2008 (DAVID-MARSHALL, VAN DREUNEN e WANG, 2010), sendo *Magic: The Gathering* responsável por 46,8% deste mercado.

Em 2011, *Magic: The Gathering* (Figura 17) possui cerca de 12 milhões de jogadores no mundo (ICV2, 2011), não se limitando apenas às cartas impressas, pois existem versões eletrônicas do jogo que proporcionam disputas *on-line* contra jogadores de qualquer lugar do mundo. No Brasil os produtos da linha *Magic: The Gathering* são distribuídos por Devir Livraria (DEVIR, 2011).

Figura 17: Cartas colecionáveis de *Magic: The Gathering*
Fonte: WIZARDS OF THE COAST (2011)

A versão digital de *Magic: The Gathering* (Figura 18), para a rede de jogos *on-line XBOX Live*®, ultrapassou no ano de 2011 a quantidade de quinhentas mil cópias vendidas (JOYSTIQ, 2011), sem contar as versões disponíveis para o console *Playstation 3*® da fabricante SONY e para computadores pessoais. O jogo eletrônico *on-line* há alguns anos já representava aproximadamente 40% do faturamento da empresa desenvolvedora (GAMESPY, 2007).

Figura 18: Versão eletrônica do jogo Magic The Gathering
Fonte: IGN (2011)

O comércio de cartas em formato eletrônico tem inspirado outras empresas a investirem neste mercado. Normalmente o jogador tem acesso gratuito a um conjunto limitado de cartas e para conseguir progredir no jogo, é necessário comprar novas cartas pagando através de cartão de crédito. Em alguns jogos também é possível realizar a troca de cartas virtuais com outros jogadores, fazer apostas de cartas, ou conquistar pontos para trocar por novas cartas. Além disso, são promovidos campeonatos mundiais que incluem prêmios em dinheiro.

Outro jogo on-line que adotou este modelo chama-se SHADOW ERA (WULVEN GAME STUDIOS, 2011), Figura 19, que além da versão para jogar *on-line*, através de um navegador *web*, também possui versões para WINDOWS, MAC, ANDROID e iOS.

Figura 19: Tela do jogo de cartas on-line SHADOW ERA
Fonte: WULVEN GAME STUDIOS (2011)

A partir de 1999 outras empresas começaram a apostar neste mercado, lançando produtos baseados em séries de animação, como é o caso dos jogos de cartas YU-GI-OH! (KONAMI, 2011) e POKÉMON TRADING CARD GAME (THE POKÉMON COMPANY, 2011), mostrados na Figura 20, que compartilham 23,6% e 9,8% do mercado de jogos de cartas Americano, respectivamente (DAVID-MARSHALL, VAN DREUNEN e WANG, 2010).

Figura 20: Cartas colecionáveis YU-GI-OH! e POKÉMON
Fonte: THE POKÉMON COMPANY (2011)

O modelo de negócios é similar entre tais jogos de cartas colecionáveis: o jogador compra um conjunto inicial de cartas e posteriormente adquire pacotes menores com novas cartas para aumentar sua coleção e opções de estratégia. É algo similar a colecionar álbuns de figurinhas, porém com a vantagem de serem cartas jogáveis.

Atualmente tem-se percebido que muitas empresas com atuação no Brasil têm investido na comercialização de jogos no formato *card games*. Seja através de jogos originais ou pela conversão de populares jogos de tabuleiro.

A empresa GROW (2011), desde a década de 1970, produz o jogo de cartas colecionáveis Super Trunfo®, que consiste em comparar informações entre duas cartas, vencendo o jogador que possuir o maior número. Existem dezenas de coleções diferentes do Super Trunfo® disponíveis para venda, entre elas coleções sobre animais, veículos, personagens de desenhos animados, times de futebol, etc.

A partir de 2010, a GROW passou a investir em outros jogos de cartas, como uma conversão do jogo de tabuleiro WAR® (Figura 21), fabricado e distribuído pela própria empresa, onde o jogador deve conduzir exércitos e conquistar territórios. A GROW também investiu um conjunto de cartas inspirado no programa televisivo SOLETRANDO® (Figura 21), em que é preciso soletrar corretamente cada letra de palavras que são sorteadas no decorrer do jogo.

Figura 21: Jogos de cartas WAR® e SOLETRANDO®
Fonte: GROW (2011)

A ESTRELA (2011), conhecida fabricante de brinquedos, também lançou alguns *card games* adaptados a partir de populares jogos de tabuleiro, entre eles estão BANCO IMOBILIÁRIO®, cujo objetivo é comprar terrenos e construir imóveis; JOGO DA VIDA®, onde o jogador deve escolher uma profissão e construir uma família; DETETIVE®, onde é preciso encontrar o assassino, a arma e o local onde o crime ocorreu; e COMBATE®, um jogo de estratégia militar. A Figura 22 apresenta a embalagem dos quatro jogos citados.

Figura 22: Jogos de cartas distribuídos por ESTRELA
Fonte: ESTRELA (2011)

A HASBRO (2011), grande fabricante mundial de brinquedos, também possui versões em cartas de alguns jogos de tabuleiro, entre eles: MONOPOLY®, jogo de compra e venda de terrenos, similar ao Banco Imobiliário®; CLUE®, um jogo de investigação semelhante ao Detetive®; Piktureka®, jogo cujo objetivo é encontrar figuras; e *The Game Of Life* Aventuras®, um jogo para desenvolver a narrativa onde é preciso contar histórias com os elementos que aparecem nas cartas sorteadas. Esses jogos são apresentados na Figura 23.

Figura 23: Jogos de cartas produzidos por HASBRO
Fonte: HASBRO (2011)

Os jogos de cartas lançados pela ESTRELA (2011), GROW (2011) e HASBRO (2011) possuem grande potencial de mercado, pois custa aproximadamente um décimo do valor da versão em tabuleiro e exploram franquias já conhecidas pelo público brasileiro. Em geral estão disponíveis em supermercados, grandes varejistas, farmácias, bancas de jornal, etc.

A MATTEL (2011) é distribuidora de um dos jogos de cartas mais conhecidos da atualidade: UNO®, cujo objetivo é ser o primeiro jogador a se desfazer de todas as cartas. Além do UNO®, ela distribui outros jogos como SKIP-BO®, onde é preciso ordenar cartas em seqüencias numéricas para marcar pontos; WHAC-A-MOLE®, onde o jogador precisa ser ágil para coletar cartas na mesa quando os jogadores formam pares; e BLINK®, um jogo de raciocínio rápido. A Figura 24 ilustra os jogos comercializados pela MATTEL (2011).

Figura 24: Jogos de cartas da MATTEL
Fonte: MATTEL (2011)

Outra empresa que também investe em jogos com cartas é a TOYSTER (2011). Entre eles encontram-se: ROCK N' ROLL®, um jogo de descarte chamado SUPER CARDS®,

semelhante ao UNO® (MATTEL, 2011) em que o jogador deve combinar figuras e descartar; e um jogo original chamado SUPERFÃ®, que traz curiosidades sobre um cantor e o jogador deve conquistar mais pontos para vencer. A TOYSTER (2011) também possui uma linha infantil que abrange jogos da memória, e jogos educacionais sobre o alfabeto, números, animais, construção de narrativas, entre outros. Alguns dos jogos são ilustrados na Figura 25.

Figura 25: Jogos de cartas produzidos pela TOYSTER
Fonte: TOYSTER (2011)

A COPAG (2011), empresa nacional com mais de 100 anos de existência e que detém cerca de 90% do mercado nacional de cartas (ISTOÉ DINHEIRO, 2011), diversificou seu catálogo ao adquirir direitos comerciais sobre marcas como DISNEY®, DC COMICS®, NICKELODEON®, STAR WARS®, etc., e passou a comercializar dezenas de jogos com personagens destas franquias. Os jogos produzidos pela COPAG incluem variações de dominó, Sudoku® e Batalha Naval, mas também apresenta idéias originais, como um jogo educativo para ensino da língua inglesa, o CROSS THE T®. Eles são encontrados à venda em lojas *on-line*, estabelecimentos comerciais diversos e também é ofertado como brinde em algumas redes de *fastfood*. A Figura 26 ilustra alguns dos jogos da COPAG.

A COPAG (2011) também é distribuidora oficial no Brasil do jogo de cartas colecionáveis POKÉMON® (THE POKÉMON COMPANY, 2011).

Figura 26: Jogos de cartas produzidos por COPAG
Fonte: COPAG (2011)

Também é possível encontrar *sites* na internet que oferecem jogos de cartas gratuitos que podem ser impressos e jogados. Entre esse sites estão CARD GAME FORGE (2011) que funciona como um repositório onde desenvolvedores independentes de jogos podem distribuir e divulgar suas criações; WTACTICS (2011) oferece um jogo de cartas similar a *Magic: The Gathering*® inspirado em personagens élficos (ainda é um trabalho em desenvolvimento); ORION'S BELL (2011) disponibiliza um conjunto de cartas que faz parte do jogo OZ®, inspirado no livro O MÁGICO DE OZ (BAUM, 1900).

Além destes, muitos outros *sites* são encontrados com facilidade através de ferramentas de busca utilizando *palavras-chave* como: *free card game*, *printable card game*, *downloadable card game*, etc.

No Capítulo 3 são apresentados detalhes sobre o desenvolvimento de um *card game* para o ensino de programação orientada a objetos.

CAPÍTULO 3

3 DESENVOLVIMENTO DO JOGO DIDÁTICO

Projetar um jogo vai além de criar conteúdo e regras. Para BRATHWAITE e SCHREIBER (2009) o projeto de um jogo é o processo de criar objetivos que façam os jogadores sentirem motivados a alcançá-los e criar regras que os ajudem a tomar decisões significativas em sua busca pelo objetivo.

Durante o processo de criação, o jogador deve ser o centro das atenções. O desenvolvedor deve colocar-se no lugar do jogador e buscar respostas para perguntas como: “Sobre o que é esse jogo?”, “Como eu jogo?”, “Como eu ganho?”, “O que eu preciso fazer?”, etc. Não há nada mais importante num jogo do que oferecer opções ao jogador para que ele faça escolhas que o ajude a progredir em direção ao seu objetivo.

BRATHWAITE e SCHREIBER (2009) enumeram alguns elementos importantes que devem ser observados durante o projeto de um jogo, entre eles:

- Ambientação do jogo – a criação de um tema para o jogo, uma história, um cenário, etc.
- Conteúdo do jogo – a criação de personagens, itens, enigmas e missões.
- Interface com o usuário – refere-se à como os jogadores recebem informação e interagem com o jogo. Os tabuleiros, cartas ou outros elementos devem apresentar informações que sejam facilmente compreendidas, usadas e interpretadas pelo jogador.
- Composição literária – são diálogos, textos, histórias no ambiente do jogo.
- Sistema do jogo – a criação de regras ou padrões matemáticos fundamentais.
- Projeto de níveis – inclui a criação de mapas, disposição de objetos e desafios deste mapa.

As seções deste capítulo apresentam a aplicação de alguns desses elementos durante a criação do jogo em questão.

3.1 AMBIENTAÇÃO DO JOGO

A primeira idéia de ambientação tinha como tema uma era medieval, com reis, cavaleiros e soldados empunhando espadas, lanças, etc., semelhante a um jogo de xadrez. Entretanto após cuidadosa análise, percebeu-se que os conceitos da orientação a objetos se tornariam confusos ao serem aplicados sobre personagens humanos, descartando então este cenário.

Optou-se então por elaborar a ambientação do jogo em um cenário futurista, envolvendo robôs, personagens que podem ser facilmente associados à tecnologia dos computadores atuais e possuem a capacidade de serem desmontados e remontados com novas peças que podem agregar novas habilidades.

Neste mundo imaginário, existe um torneio esportivo chamado Guerra de Robôs. Cada jogador tem à sua disposição um equipamento que é uma fábrica portátil capaz de montar robôs de batalha (os objetos). Porém, antes que os robôs sejam produzidos, é necessário definir projetos (as classes) que descrevem como os robôs irão funcionar.

Quando os robôs saem da fábrica, entram em batalha e podem atacar outros robôs do adversário ou, caso não haja nenhum inimigo, ataca diretamente a fonte de energia da fábrica até que esta seja desligada, impedindo o jogador adversário de continuar o jogo. A Figura 27 ilustra o esporte fictício.

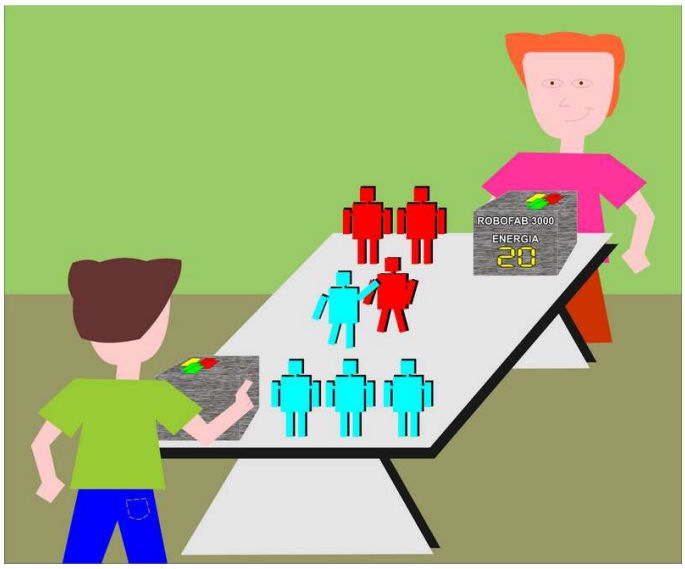


Figura 27: Ambientação do jogo

É nesta ambientação que acontecem as partidas do jogo proposto neste trabalho, que foi denominado como “P.O.O. - O Jogo”.

KIM, *et al.* (2006), em seu trabalho sobre ensino de programação com uso de jogos, optam por não utilizar um tema fixo, permitindo que um professor facilitador defina o que será abordado antes de cada partida. Apesar de tal característica criar variações, quando os alunos estiverem fora da sala de aula, poderão não conseguir desenvolver adequadamente seus próprios temas, causando problemas na aprendizagem devido a seus erros. Sendo assim, optou-se por um tema pré-definido para este jogo.

As seções 3.2 à 3.9 apresentam detalhes sobre o conteúdo desenvolvido para o jogo, mais especificamente o desenvolvimento das cartas que são usadas para jogar e ensinar os conceitos de orientação a objetos.

3.2 VISÃO GERAL DAS CARTAS DO JOGO

Para projetar as cartas do jogo, utilizou-se o programa *Magic Set Editor* (VAN LAARHOVEN, 2011), que possui licença livre e traz modelos de cartas de alguns jogos comerciais populares, entre eles *Magic: The Gathering*® (WIZARDS OF THE COAST, 2011), Yu-Gi-Oh!® (KONAMI, 2011), *Pokémon Trading Card Game*® (THE POKÉMON COMPANY, 2011), e outros.

Após analisar detalhadamente os componentes apresentados em cada um dos modelos de cartas disponíveis no programa, optou-se por utilizar o modelo *ModernStyle - After 8th edition* das cartas de *Magic: The Gathering*®, por apresentar áreas delimitadas e permitir a inserção de diferentes elementos em cada área sem causar sobreposição.

Apesar de o modelo escolhido oferecer um grande número de combinações entre os elementos disponíveis, definiu-se que as cartas de “P.O.O. - O Jogo” deveria conter poucos itens para descrever os conceitos a serem ensinados a fim de torná-lo um jogo simples. A Figura 28 apresenta alguns dos elementos que são comumente encontrados.

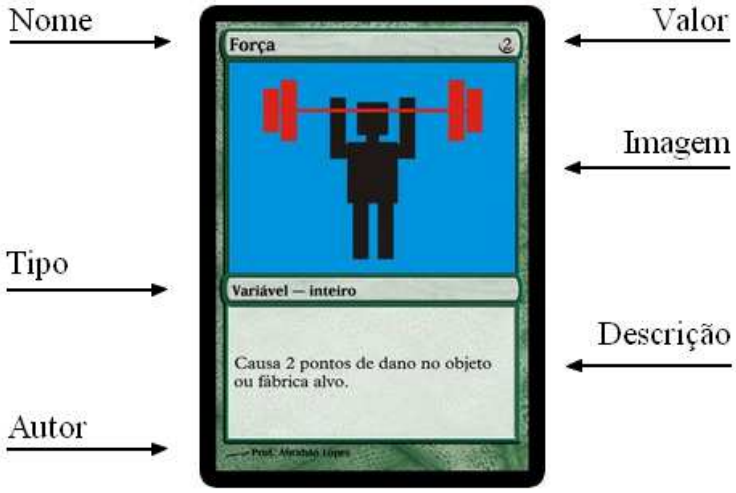


Figura 28: Descrição das áreas de uma carta

As características apresentadas são:

- Nome da carta: exibe o nome da variável, do método, da classe, etc.
- Valor da carta: usado apenas nas cartas de variáveis e constantes, determina o valor inicial atribuído.
- Imagem: permite ao jogador associar o conteúdo da carta (texto/valores) à ilustração, ajudando-o a memorizar seu conteúdo.
- Tipo da carta: agrupa as cartas em categorias como variáveis, classes, métodos, edição, etc.
- Cores: cada categoria de cartas possui uma cor específica que está relacionada a alguma característica da função por ela exercida no jogo.
- Descrição: um breve texto que explica como a carta deve ser usada no jogo.
- Autor da carta: nome de quem criou a carta.

Outros elementos foram inseridos nas cartas mediante a necessidade de ressaltar algum conceito da orientação a objetos ou destacar algo importante contido no texto descritivo da carta e serão explicados conforme forem apresentados durante o texto.

Nos tópicos 3.3 à 3.7 serão apresentados mais detalhes sobre as cartas desenvolvidas bem como sua aplicação aos conceitos da programação orientada a objetos.

3.3 CLASSES E OBJETOS

Entre os principais conceitos da orientação a objetos, estão as Classes e os Objetos. Apesar da convivência cotidiana com inúmeros objetos físicos, ao apresentar aos alunos o conceito de como representar computacionalmente tais elementos, certa dificuldade é percebida.

Na literatura são encontradas diversas sugestões sobre como ajudar os alunos a perceberem a diferença destes dois conceitos. SIERRA e BATES (2007) comparam uma classe com uma ficha de cadastro de endereços com campos em branco e os objetos com cada ficha preenchida com dados de diferentes pessoas. SANTOS (2003) exemplifica uma classe como sendo a planta baixa de um prédio, enquanto prédios construídos a partir desta planta são os objetos. BARNES e KÖLLING (2008) citam inúmeros carros e suas características específicas como sendo objetos enquanto a descrição geral de tais características como a classe. ARNOLD, GOSLING e HOLMES (2007) citam uma linha de produção de *walkmans* onde cada item produzido é um objeto enquanto o projeto é a classe.

Após verificar estas e outras sugestões, optou-se por utilizar o exemplo de linha de montagem, em que um projeto em *blueprint* representa a classe e os itens produzidos representam os objetos.

As cartas que representam as classes foram confeccionadas na cor dourada, para enfatizar a importância da classe em um programa orientado a objetos. As ilustrações das cartas desse tipo são inspiradas nas impressões *blueprint* comuns em projetos de máquinas e equipamentos mecânicos. A Figura 29 ilustra a representação de uma classe através de uma carta de jogo.

As classes são usadas para agrupar variáveis e métodos a fim de descrever as características dos objetos que serão criados bem como as ações que estes podem executar. Cartas para representar métodos e variáveis também foram desenvolvidas e são explicadas em detalhes na seção 3.4.

Em qualquer uma das variações do jogo, um dos objetivos é criar classes com a maior quantidade de métodos e variáveis, pois desta maneira o jogador terá vantagens sobre seu adversário. Quanto mais métodos a classe possuir, mais ações podem ser realizadas, criando variações de ataque ou defesa. Da mesma maneira, quanto mais variáveis a classe possuir, mais características úteis os objetos criados a partir desta classe possuirão, tornando-se um diferencial durante a partida.

Figura 29: Carta de classe e sua representação em UML

Nos jogos desenvolvidos neste trabalho as classes são formadas colocando-se inicialmente uma carta de classe na mesa e sobre ela quantas cartas de métodos e variáveis o jogador conseguir, obviamente respeitando algumas regras que serão descritas posteriormente na seção 3.8. A Figura 30 ilustra a disposição das cartas que formam uma classe e sua respectiva representação através de um diagrama UML.

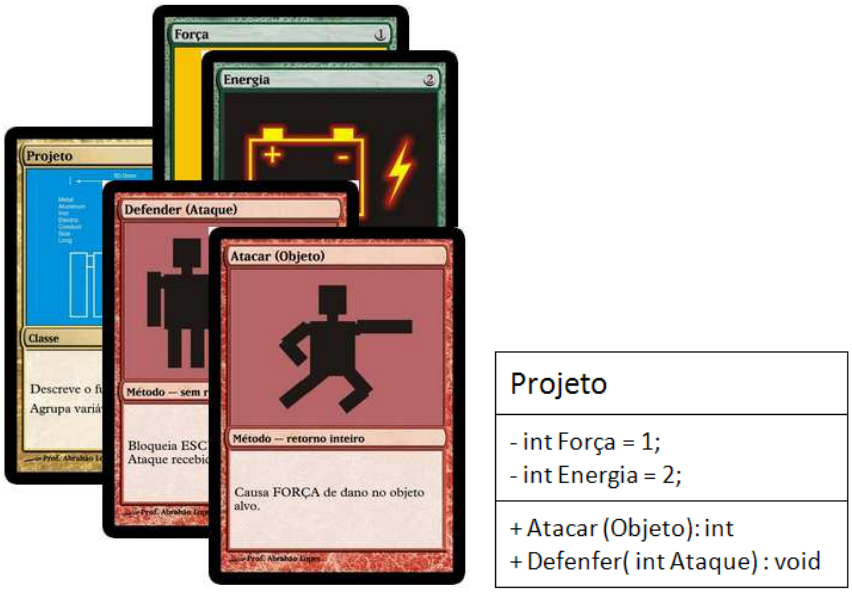


Figura 30: Exemplo de uma classe com variáveis e métodos e a representação em UML

Durante o ensino, os alunos são incentivados a desenhar diagramas de classe UML para representar as classes que foram montadas com as cartas do jogo a fim de que possam se familiarizar com a notação. O processo inverso também pode ser aplicado, ou seja, apresentar um diagrama de classe e fazê-los montar a mesma classe usando as cartas de jogo.

Após ensinar o conceito de classe, o próximo passo é apresentar o conceito de instâncias de classe, ou objetos. Para SANTOS (2003) um objeto é a materialização de uma classe, podendo ser usado para representar dados e executar operações.

Fazendo uma analogia, uma classe não pode atacar um adversário ou defender-se, pois não é mais que um monte de papel com projetos, desenhos, medições e cálculos. Quando os procedimentos nesses projetos são seguidos e as peças são unidas para formar o produto descrito, então haverá algo físico, sólido, um elemento que poderá realizar ações e sofrer reações.

As cartas que representam objetos foram definidas na cor cinza, em alusão ao metal necessário para fundição dos robôs. Para que uma carta de objeto possa entrar em jogo é necessário que haja uma classe previamente definida e o objeto poderá realizar todas as ações dos métodos desta classe e possuirá todas as variáveis nela definidas. Elas são dispostas ao lado da carta de classe, conforme demonstrado na Figura 31.

Figura 31: Dois objetos criados a partir de uma classe definida pelo jogador

O *BlueJ* (KÖLLING, 2011), ambiente de programação desenvolvido especialmente para o ensino de orientação a objetos em *Java*, permite demonstrar com facilidade a diferença entre uma classe e objetos instanciados a partir dela. Nele é possível representar as classes através de diagramas semelhantes aos da linguagem UML, e os objetos são representados através de retângulos vermelhos, conforme exibidos na Figura 32.

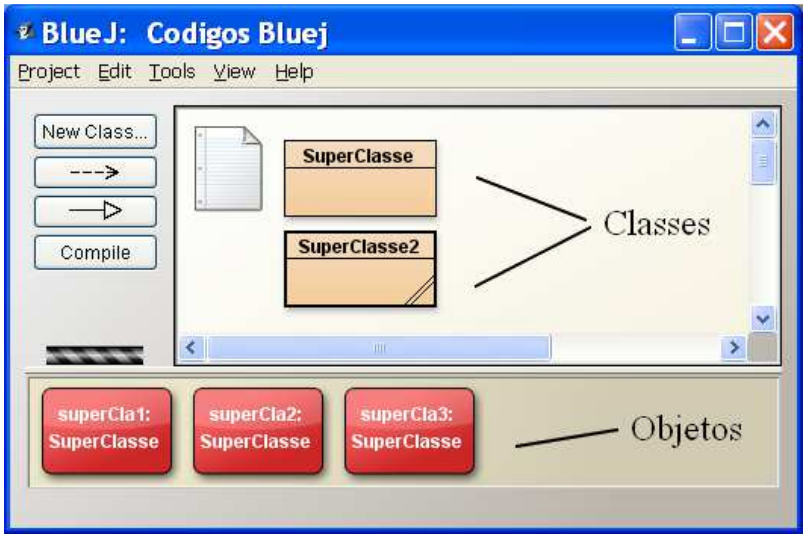


Figura 32: *BlueJ*: ambiente para ensino de orientação a objetos em *Java*

Para instanciar um objeto a partir de uma classe é necessário usar código semelhante ao exibido no Quadro 1. A palavra reservada *new* é responsável por informar ao compilador que um objeto deve ser criado. Em geral esse código é colocado numa classe à parte, mas no *BlueJ* é possível instanciar um objeto clicando com o botão direito do *mouse* sobre a classe e, no menu flutuante que surge, escolhendo-se a opção que contém a palavra *new*, sem a necessidade de escrever outras classes. Uma comodidade para os estudantes que estão iniciando seus estudos nesta linguagem.

```
Projeto p1 = new Projeto( );
```

Quadro 1: Código para criação de um objeto

As seções 3.4 e 3.5 explicam sobre os elementos que compõem uma classe.

3.4 VARIÁVEIS E CONSTANTES

Para JANDL JUNIOR (2007), uma variável é um nome definido pelo programador ao qual pode ser associado um valor pertencente a certo tipo de dados e armazenado em um conjunto de posições da memória do computador.

Em geral os programas de computador utilizam espaços reservados na memória do computador para armazenar dados pré-definidos no código fonte, calculados durante o funcionamento do programa ou que fornecidos pelo usuário através do teclado e caixas de diálogo das interfaces gráficas.

A maioria das linguagens de programação adota algumas regras para a definição de variáveis, como por exemplo, não usar caracteres especiais como asterisco, hífen, e outros, além de excluir algumas palavras reservadas, como *new*, *static*, *public*, etc.

Ao representar uma variável no código fonte é necessário informar: um tipo de dado, que pode ser inteiro, caractere, lógico, ou outro tipo definido pelo programador; um nome único, em geral que descreva o propósito desta variável; e opcionalmente pode-se informar um valor que será atribuído à variável em sua inicialização.

Um exemplo válido para a declaração de variáveis em *Java* é o código mostrado no Quadro 2.

```
int energia = 2;
```

Quadro 2: Código para declaração de uma variável em *Java*

As cartas que representam a declaração de variáveis foram desenhadas de maneira a apresentar todos os elementos existentes na linha de comando do Quadro 2.

Conforme se observa na Figura 33, no canto superior da carta, à esquerda, está registrado o nome da variável; no canto superior, à direita, no interior do círculo, está o valor armazenado por esta variável em sua inicialização, ou seja, quando a carta entrar em jogo, ela possuirá este valor; e, por fim, abaixo da imagem, à direita da palavra *variável*, o tipo de dado (inteiro) aparece em destaque.

A cor verde foi escolhida para as cartas de variáveis em alusão à permissão para

modificar o valor armazenado, pois durante o jogo, o valor pode aumentar ou diminuir em decorrência da ação de outras cartas, e o novo valor substitui o valor impresso na carta.

Figura 33: Carta que representa uma variável

Outro tipo de dado que foi representado nas cartas foi o tipo lógico (*boolean*), que armazena exclusivamente os valores “verdadeiro” ou “falso”. Cartas com este tipo de dado são usadas para representar uma habilidade que pode ser ativada ou desativada durante a partida. A Figura 34 ilustra como são representados os valores “verdadeiro” e “falso”.

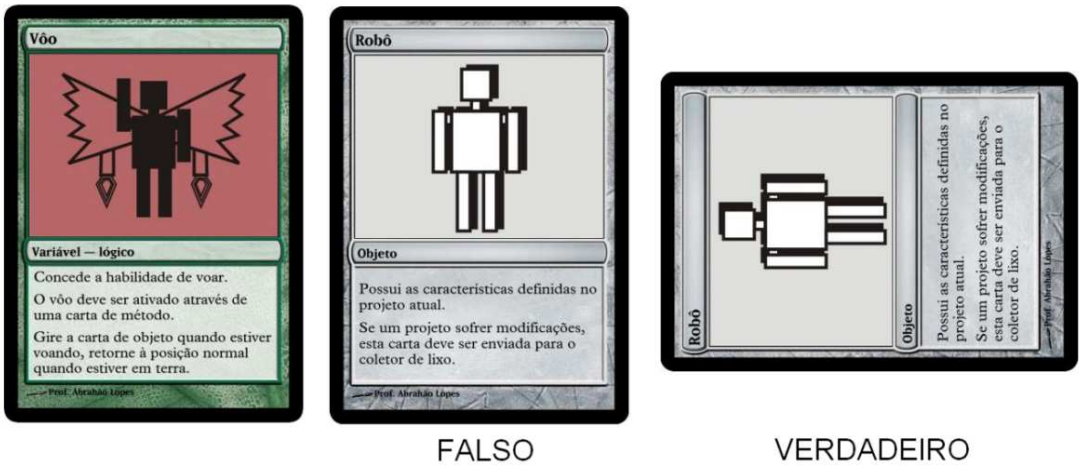


Figura 34: Representação de valores lógicos

Devido à representação do valor usar o posicionamento da carta na horizontal ou vertical, foi necessário limitar o uso de apenas uma habilidade em cada classe, pois de outra maneira seriam necessários elementos externos para representar os valores.

Tipos de dados como *String*, *Double*, *Float* e outros são introduzidos através de exemplos e exercícios fora do contexto deste jogo. O mesmo é feito para explicar sobre promoção e coerção de tipos (*typecasting*), que se refere à atribuição entre diferentes tipos de dados.

Em certos programas utilizam-se valores fixos que não podem ser alterados durante a execução, chamados de constantes. Em *Java* esses valores são conhecidos como variáveis finais. Algumas cartas foram identificadas como “constante” ao invés de “variável” e não podem sofrer ação de cartas que modificam o seu valor. A Figura 35 exibe uma carta constante.

Figura 35: Carta de valor constante

Para a declaração de constantes, utiliza-se o modificador *final* conforme mostrado no código do Quadro 3.

```
final int forca = 4;
```

Quadro 3: Código para declaração de uma variável em *Java*

Ao explicar sobre o modificador *final*, pode-se também introduzir outros modificadores como *static*, *public* e *private*, explicando ao aluno as funções realizadas quando aplicadas às variáveis.

Algumas cartas foram definidas com o propósito de modificar o valor das cartas de variáveis durante a partida. As cartas “Incremento” e “Decremento” trazem em seus textos uma restrição que determina que esta carta não pode modificar o valor de Constantes, conforme exibido pela Figura 36. Desta maneira o aluno pode perceber a diferença entre as duas formas de armazenamento de dados na memória.

Figura 36: Carta para modificação de valores

Na Figura 37 é possível visualizar a mensagem de erro que o compilador *Java* apresenta quando se faz uma tentativa de modificar o valor de uma constante. Ao implementar este código o aluno perceberá que a modificação não é permitida devido à diretiva “final” aplicada durante a declaração da variável.

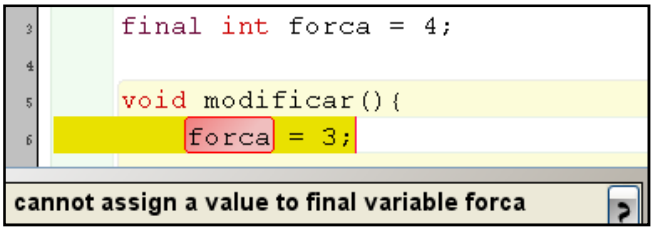


Figura 37: Mensagem de erro ao tentar modificar o valor de uma constante

As cartas de incremento e decremento também são úteis para introduzir os operadores aritméticos de incremento e decremento, que em *Java* são representados pelos operadores ++ (mais, mais) e pelo operador – – (menos, menos), respectivamente. Além destes, os demais operadores aritméticos também são apresentados, porém serão utilizados mais regularmente ao trabalhar com as cartas de métodos, que serão apresentadas na seção 3.5.

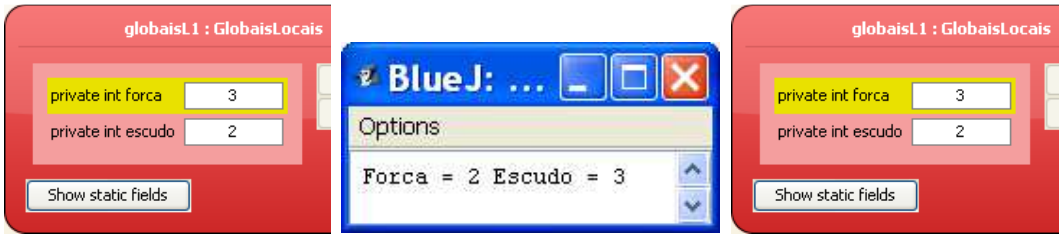
Para ensinar sobre o escopo de variáveis globais (de instância) e variáveis locais (temporárias), algumas cartas trazem em seu texto a restrição “até o final do turno”, significando que as modificações aplicadas aos valores não são permanentes, e desaparecem logo que o método encerra. Modificações que permanecem após o encerramento do turno estão relacionadas às variáveis globais enquanto as modificações descartadas ao final de uma jogada estão relacionadas às variáveis locais.

O Quadro 4 apresenta um código que ilustra o uso de variáveis globais e locais. Ao executar as linhas 6 e 7, serão impressos na tela os valores das variáveis globais (3 e 2, respectivamente). Nas linhas 9 e 10 são declaradas variáveis locais com o mesmo nome das variáveis globais e os valores são invertidos. Ao executar as linhas 12 e 13, que é o mesmo das linhas 6 e 7, serão impressos os valores das variáveis locais (2 e 3, respectivamente).

```
1  public class GlobaisLocais{
2
3      private int forca = 3, escudo = 2;
4
5      void inverter(){
6          System.out.println("Forca = " + forca +
7              " Escudo = " + escudo);
8
9          int escudo = this.forca;
10         int forca = this.escudo;
11
12         System.out.println("Forca = " + forca +
13             " Escudo = " + escudo);
14     }
15 }
```

Quadro 4: Exemplo de variáveis globais e variáveis locais

Com o auxílio do *software BlueJ* (KÖLLING, 2011), o aluno pode inspecionar o conteúdo de objetos instanciados e observar que ao criar um objeto, o valor inicial das variáveis força e escudo são 3 e 2, respectivamente (Figura 38a). Ao executar o método, as variáveis locais são criadas e os valores invertidos são apresentados (Figura 38b). Após a finalização do método, uma nova inspeção é realizada e observa-se que os valores permanecem como no início, pois as variáveis locais são temporárias e foram destruídas (Figura 38c).



(a) (b) (c)

Figura 38: Exemplo de variáveis globais e locais

Nas linhas 9 e 10 do Quadro 4 utiliza-se a referência *this* que permite ao objeto referenciar suas variáveis de instância, mesmo que possuam nomes idênticos aos das variáveis locais. Este conceito pode ser explicado aos alunos neste momento, se o professor julgar adequado.

A seção 3.5 trata das cartas de métodos e das ações por elas realizadas.

3.5 MÉTODOS

De acordo com JANDL JUNIOR (2007), os métodos, que também são chamados de operações ou funções-membro, são sub-rotinas associadas aos objetos, que possuem trechos de código que permitem realizar ações ou transformações sobre os valores dos atributos de tais objetos, modificando seu estado e proporcionando o comportamento desejado.

Os métodos são formados por duas partes: um cabeçalho, que é composto pelo nome do método e pode também conter parâmetros, tipos de retorno e modificadores como *public* ou *private*, por exemplo; e pelo código a ser executado, também chamado de corpo do

método.

As cartas desenvolvidas para representar os métodos possuem a cor vermelha, escolhida com intenção de representar ação, movimento.

A Figura 39 exibe uma carta de método que, assim como a carta de variável, foi desenhada de forma a conter todos os elementos necessários à declaração de um método válido em *Java*. Na parte superior da carta está o nome do método, que descreve brevemente o tipo de ação que ele realiza, seguido de parênteses, que identificam na linguagem os métodos ou funções.

Figura 39: Carta de método

Abaixo do nome encontra-se uma ilustração que representa graficamente a ação a ser realizada. Abaixo da imagem aparece a palavra Método, identificando o tipo da carta, seguido pelo tipo do retorno. Por fim é descrita a ação ou ações realizadas pelo método em um texto curto, porém claro o suficiente para que não haja erro na interpretação.

Uma declaração válida para a carta da Figura 39 encontra-se no Quadro 5. A linha 2 do código contém o cabeçalho do método, e as linhas 3 a 7 apresentam o corpo do método.

```
class Carta{
    void voar(){
        if (voo == true){
            voo = false;
        }else{
            voo = true;
        }
    }
}
```

Quadro 5: Código de declaração de um método

Os parâmetros que aparecem dentro dos parênteses que seguem o nome do método também estão presentes nas cartas, conforme a Figura 40. Os valores passados como parâmetros normalmente são usados em cálculos ou outras ações realizadas pelo método. Nesta mesma figura vê-se o tipo de retorno do método (inteiro).

Figura 40: Carta de método com parâmetro e retorno

Com o uso de parâmetros e retorno, pode-se realizar a troca de informações entre objetos, permitindo as ações realizadas pelos métodos sejam transmitidas entre as cartas em jogo.

Percebeu-se que muitos alunos possuem dificuldade para compreender o conceito de retorno. Para explicar visualmente este conceito, pode-se utilizar como exemplo a simulação apresentada na Figura 41.

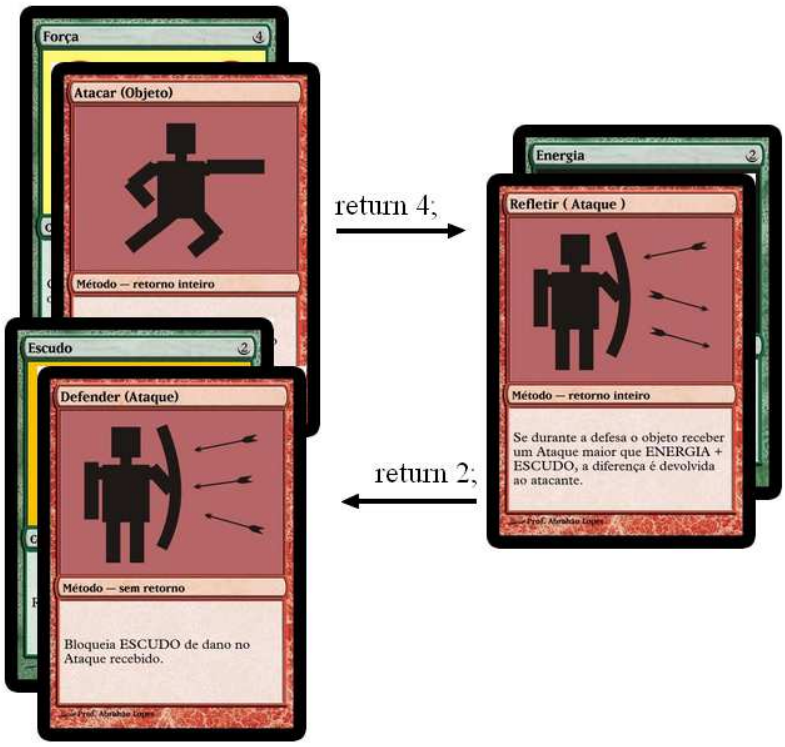


Figura 41: Exemplo da troca de mensagens entre objetos usando parâmetros e retorno

Observa-se que na Figura 41 uma carta de Ataque retorna o valor 4, referente à carta Força. A seguir a carta Refletir recebe o valor 4 como parâmetro, calcula o dano considerando a carta Energia que tem valor 2, e retorna o resultado da operação. A carta *Defender* recebe o valor 2 como parâmetro, calcula o dano e encerra o combate, pois esta não possui retorno, não podendo enviar o resultado à outro método.

O Quadro 6 exibe o código da carta *Refletir*, e mostra na linha 3 o tipo de retorno (*int*) antes do nome do método, e nas linhas 6 e 8 a diretiva *return*, responsável por retornar o valor indicado ao método solicitante.

Uma característica importante da orientação a objetos é a sobrecarga de métodos. JANDL JUNIOR (2007) explica que a sobrecarga de métodos permite a existência de dois ou mais métodos com o mesmo nome, desde que suas assinaturas (parâmetros e tipo de retorno) sejam diferentes. Com isso, é possível manter a mesma semântica (um mesmo nome) para

operações semelhantes que podem ser realizadas de modos diferentes, simplificando o uso da classe.

```
class MetodoComRetorno
{
    int refletir(int ataque){
        dano = ataque - (energia + escudo);
        if(resultado >= 0){
            return dano;
        }else{
            return 0;
        }
    }
}
```

Quadro 6: Código de um método com parâmetro e retorno

Também foram desenvolvidas algumas cartas a fim de explorar a sobrecarga de métodos, conforme exibido na Figura 42. A carta *Atacar* da Figura 42a apresenta um único parâmetro e a ação de retornar o valor da carta *Força*. A carta *Atacar* da Figura 42b apesar de possuir o mesmo nome, requer dois parâmetros para ser executado e sua ação é completamente diferente da carta da Figura 42a.

Durante uma partida, ambas as cartas podem fazer parte do mesmo projeto, permitindo ao jogador escolher qual das duas utilizar, ampliando assim suas opões de ações e estratégias.

Ao ensinar sobrecarga, há também a oportunidade de apresentar o conceito sobre métodos construtores.

(a) (b)

Figura 42: Exemplo de sobrecarga de métodos

Outra modificação que pode ser aplicada aos métodos é acionada pela palavra chave *static*. O uso desse modificador permite que o método de uma classe seja executado sem que haja a necessidade de criar uma instância da classe. Esses métodos são chamados de métodos estáticos.

Para ensinar sobre este conceito, utilizaram-se cartas de classe que apresentam em sua descrição a declaração e o funcionamento de um método, além de explicar ao jogador que não é preciso criar objetos desta classe para poder usá-lo, conforme ilustra a Figura 43.

A linguagem *Java* é repleta de classes com métodos estáticos, desde métodos para simples exibição de texto na tela, métodos para realizar conversão entre tipos de dados, e até mesmo o método principal *main( )* que inicia a execução de um programa.

O uso destas cartas de classes com métodos estáticos pode facilitar o entendimento do aluno e incentivá-lo a desenvolver esse tipo de método em suas aplicações.

Em cartas desse tipo foi adicionada uma restrição que limitará a sua freqüência de uso, exigindo certa quantidade de turnos para que ela possa ser usada novamente. A força do tiro e o tempo de recarga da arma foram representados por um conjunto de números no formato: força/tempo, como é visto na Figura 43, o conjunto 4/2, abaixo do texto descritivo.

Figura 43: Classe com método estático

3.6 HERANÇA

Linguagens de programação orientadas a objeto apresentam uma importante característica chamada Herança. JANDL JUNIOR (2007) conceitua herança como um mecanismo que possibilita a uma classe usar campos e métodos definidos em outra classe, o que significa o compartilhamento de membros entre classes. Tal característica permite ao programador reaproveitar grande parte do código escrito em outras classes.

A herança é baseada numa relação hierárquica, em que uma classe pai (ou *superclasse*, a classe que é usada como referência) transmite suas definições para as classes definidas como filhas (ou *subclasses*, a classe que herda de outra).

Algumas vantagens de se utilizar a herança são: redução da necessidade de escrever códigos similares; o fato de que o código reaproveitado já foi testado anteriormente, reduzindo assim a quantidade de erros; adição de novos métodos nas classes filhas em relação aos da classe pai; elimina a duplicidade de campos e métodos similares; e outras;

Uma carta de herança (Figura 44) foi definida para permitir o uso desta característica no jogo. E para incentivar o uso desse tipo de construção, foi definida uma regra para o jogo que determina a remoção de todas as cartas de objeto se houver qualquer alteração na classe. Assim, para não perder os objetos que estão em jogo, o jogador pode construir uma estrutura

de herança, criando uma nova classe que herda todas as características da classe atual e adiciona as novas características na subclasse.

Figura 44: Carta de Herança

A Figura 45 apresenta um exemplo de como a herança é utilizada durante uma partida. Considere-se que em determinado momento da partida o jogador possui a classe indicada na Figura 45a e dois objetos instanciados a partir dela. Se as cartas Vôo e Voar fossem adicionados a esta classe, os dois objetos precisariam ser removidos do jogo, por causa da regra apresentada no parágrafo anterior, deixando o jogador sem opções de ataque ou defesa.

Ao optar por uma estrutura de herança, uma nova classe contendo as cartas Vôo e Voar é definida e a carta Herança é usada para criar a relação pai-filho, fazendo com que objetos criados a partir da subclasse possuam as novas características e todas as características da superclasse, indicadas na Figura 45b.

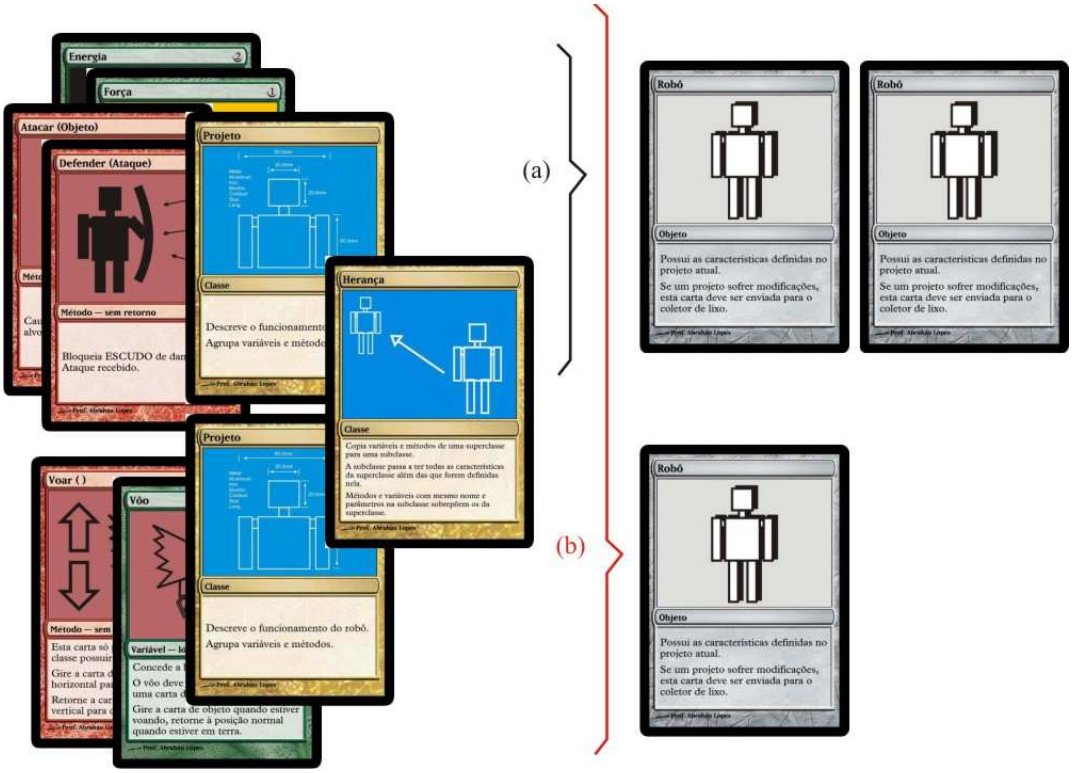


Figura 45: Exemplo de herança durante uma partida

O Quadro 7 exibe o código *Java* de como a herança da Figura 45 é implementada. A linha 7 apresenta a palavra reservada *extends* que é responsável por criar a estrutura hierárquica entre as classes.

```
1  class SuperClasse{
2      int forca = 2, energia = 1;
3      void defender(int ataque){ /* código */}
4      int atacar(Object alvo){/* código */}
5  }
6
7  class SubClasse extends SuperClasse{
8      boolean voo = false;
9      void voar(){ /* código */}
10 }
```

Quadro 7: Implementação de herança entre classes

Os alunos também devem ser incentivados a desenharem diagramas de classe em UML para representar as relações de herança, como demonstrado na seção 3.3, Figura 30,

pág. 40. A Figura 46 apresenta um exemplo de um diagrama criado a partir da herança formada pelas cartas ao lado.

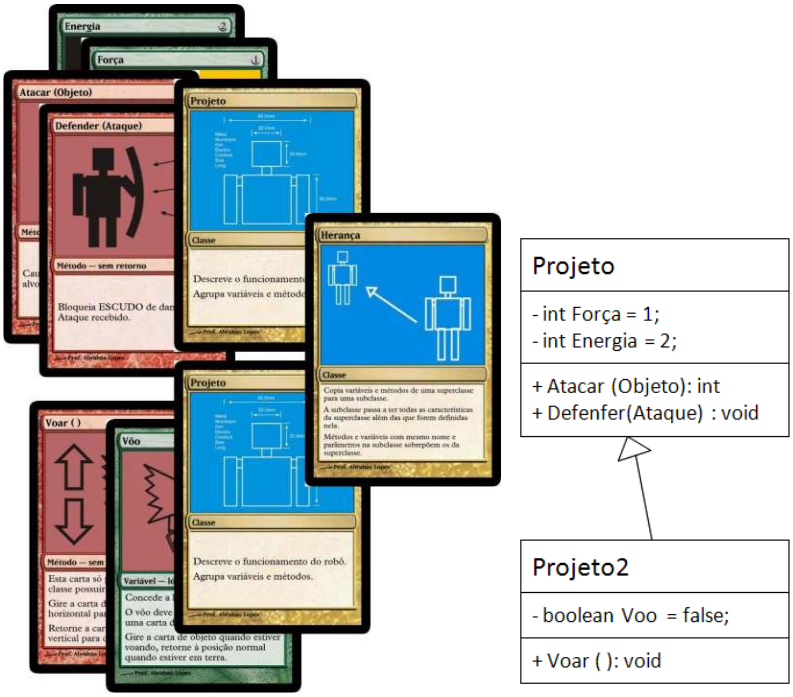


Figura 46: Diagrama de herança em UML a partir de um conjunto de cartas

Uma característica a cerca de métodos e variáveis definidos nas subclasses que se deve destacar é a sobreposição. Esta característica permite que uma subclasse declare variáveis ou métodos com o mesmo nome, tipo de retorno ou parâmetros, existente na superclasse, porém com valores e comportamentos diferentes.

O Quadro 8 apresenta um trecho de código que exemplifica o conceito de sobreposição de métodos. O método alerta( ) pertencente à superclasse e declarado na linha 3 exibe a mensagem “CUIDADO!”. Um método de mesmo nome é declarado na subclasse, linha 9, porém seu comportamento é diferente, exibindo a mensagem “ATENÇÃO! PERIGO!” quando chamado, ou seja, o código apresentado na linha 10 sobrepõe o código da linha 4 da classe pai. Apesar da sobreposição nesse método, os demais elementos da superclasse são herdados pela subclasse sem qualquer alteração, como é o caso da variável escudo (linha 2).

```
class SuperClasse2{
    int escudo = 2;
    void alerta(){
        System.out.println("CUIDADO!");
    }
}

class SubClasse2 extends SuperClasse2{
    void alerta(){
        System.out.println("ATENÇÃO! PERIGO!");
    }
}
```

Quadro 8: Exemplo de sobreposição

Durante o jogo a sobreposição pode acontecer em situações como as que são simuladas na Figura 47. A carta *Atacar (Objeto)* da superclasse (a) retorna o valor da carta de força, que é 2 neste caso. A subclasse (b) possui uma carta de *Força* com valor 4, que sobrepõe o valor 2 da superclasse.

Se o método *Atacar (Objeto)* usado fosse o da superclasse, o objeto da subclasse já teria uma vantagem em relação a um objeto da superclasse, pois o valor de ataque seria maior. Entretanto a subclasse possui uma carta *Atacar (Objeto)* que realiza uma ação diferente, retirando 2 pontos de energia para transformá-los em um ponto extra de força, causando um ataque com 5 de dano, neste exemplo.

Ao ensinar sobreposição aos alunos é importante comentar sobre outras propriedades deste conceito, como: o acesso ao método original da superclasse pode ser feito através da palavra reservada *super*; métodos de mesmo nome, mas com tipo de retorno diferente ou parâmetros diferentes causam sobreposição; e restrições causadas pelo uso de modificadores como *private*, *static*, *protected*, *final*, etc.

Outro modificador importante é palavra chave *abstract*, que em *Java* é usado para definir classes abstratas. Com o modificador abstract é possível criar superclasses que declaram variáveis e métodos para serem usados numa hierarquia de classes, mas que proíbe a instanciação de objetos a partir dela.

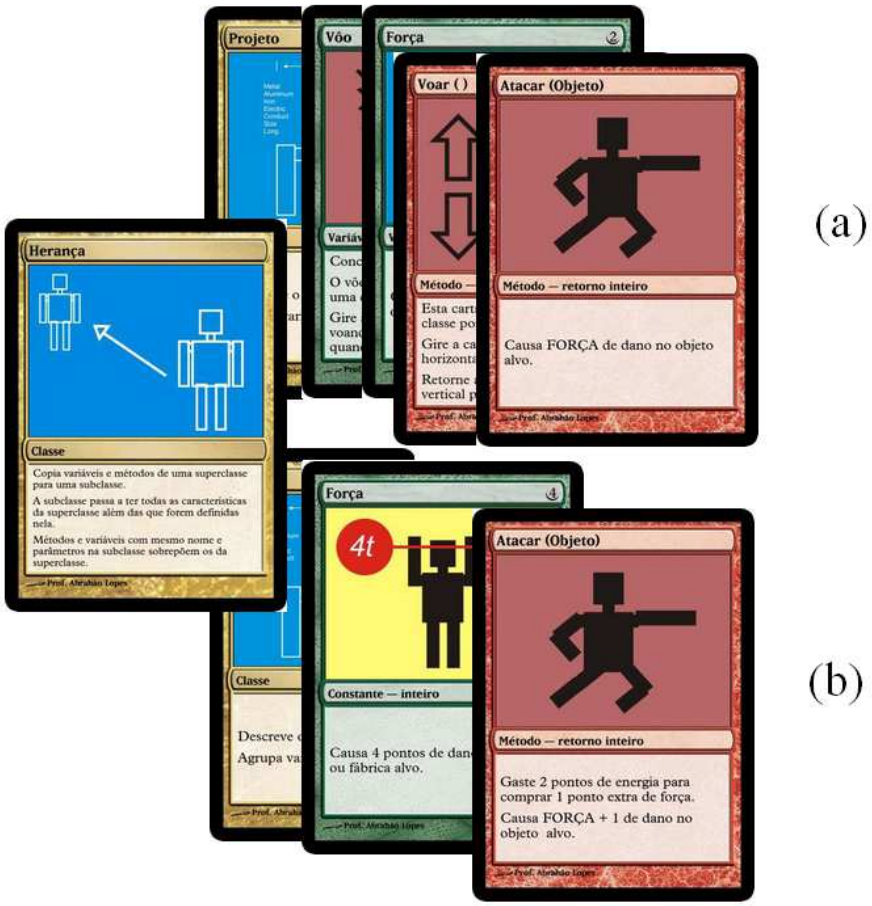


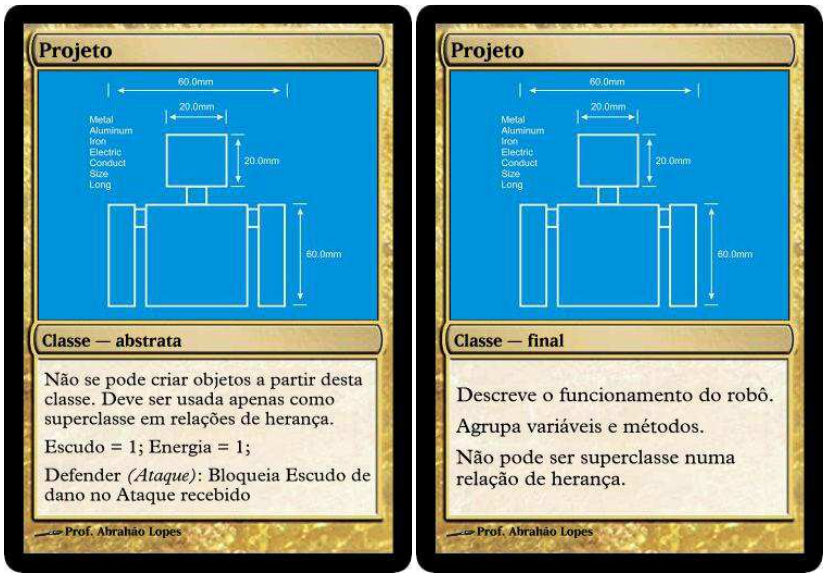
Figura 47: Exemplo de sobreposição de métodos e variáveis

Para representar classes abstratas foram criadas cartas como a que é mostrada na Figura 48a. Estas cartas trazem em sua descrição informações sobre métodos e variáveis definidos pela classe. Neste exemplo, a classe abstrata reúne as características das cartas *Escudo 1*, *Energia 1* e *Defender (Ataque)*.

De certa forma as cartas de classe abstrata concedem vantagem ao jogador, pois reduz a quantidade cartas necessárias para se ter um objeto funcional.

Ao ensinar sobre classes abstratas, pode-se também introduzir o aluno ao assunto de interfaces, ampliando as possibilidades do código.

O modificador *final* também foi representado em algumas classes, como na Figura 48b. Uma classe que apresente este modificador não permite que subclasses sejam criadas a partir dela, ou seja, ela não pode ser superclasse numa relação de herança. O jogador deverá pensar bem antes de iniciar sua primeira classe com esse tipo de carta, pois perderá a vantagem de criar uma relação de herança a partir dela.



(a) (b)

Figura 48: Classe abstrata e Classe final

A seção a 3.7 trata de outras cartas que não estão diretamente relacionadas aos conceitos de programação orientada a objetos.

3.7 CARTAS OPCIONAIS

Algumas cartas foram criadas com o propósito de adicionar mais possibilidades de estratégias durante as partidas e são chamadas de cartas de edição.

A Figura 49 apresenta três exemplos de cartas desse tipo.

(a) (b) (c)

Figura 49: Cartas de edição

A primeira delas é a carta *Adicionar Comentário* (Figura 49a) que permite desabilitar uma carta qualquer, seja do jogador ou do adversário, eliminando temporariamente uma ação ou habilidade que está em jogo; a segunda é a carta *Remover Comentário* (Figura 49b), que anula o efeito da carta *Adicionar Comentário*; e por fim a carta *Executar 2 métodos* (Figura 49c) que dá uma vantagem ao jogador ao permitir que duas ações sejam executadas em seqüência, sobrepondo a regra geral do jogo que diz que cada objeto só pode executar uma ação por turno.

Outras cartas desse tipo poderão ser adicionadas ao jogo conforme se julgue necessário, entretanto é importante que não haja grande quantidade delas no baralho para que o foco da partida seja os conceitos de orientação a objetos, e não a edição de código.

3.8 REGRAS DO JOGO

"P.O.O – O Jogo" foi planejado para evoluir de acordo com o nível de conhecimento dos alunos e possui diferentes conjuntos de regras, sendo cada conjunto relacionado a um assunto o ou grupo de assuntos da orientação a objetos.

Nesta seção são apresentadas as regras gerais, válidas em qualquer partida, e regras específicas, relacionadas aos conceitos ensinados. Para cada regra apresentada há também

uma breve explicação sobre o motivo pelo qual ela foi incluída no jogo.

3.8.1 Regras gerais

As regras desta seção são válidas para qualquer partida, sendo substituídas apenas quando indicado pelo texto de uma carta.

Regra 1: As cartas de todos os jogadores devem ser unidas em uma única pilha e misturadas.

Esta regra foi adota por se tratar de um jogo em que cada participante pode criar e imprimir suas próprias cartas. O jogo se tornaria injusto para um dos lados caso o jogador elaborasse um baralho com cartas que apresentassem valores altos de defesa e ataque ou outros elementos que o deixasse em vantagem. Estando todas as cartas em uma única pilha, todos os jogadores possuem a mesma chance de vitória.

Regra 2: Cada jogador inicia a partida com 7 cartas na mão e esse é o número máximo de cartas que podem ser mantidas na mão. Ao final do turno, se o jogador tiver mais que 7 cartas na mão deve descartar as cartas excedentes até ficar com apenas 7.

A quantidade de cartas foi definida com base nas regras de *Magic the Gathering*® (WIZARDS OF THE COAST, 2011). Apesar disso, após criteriosas observações, percebeu-se que este número de cartas é suficiente para montar uma classe completa mesmo nos turnos iniciais de uma partida.

Regra 3: No início de cada um de seus turnos, o jogador deve trazer a carta que está no topo da pilha para a sua mão.

Esta regra é usada para que novas cartas entrem em jogo e ofereça aos jogadores a possibilidade de desenvolver diferentes estratégias devido à ordem aleatória em que as cartas aparecem.

Regra 4: Cada jogador inicia a partida com 20 pontos de energia. Se os pontos de energia de um jogador chegar em zero, ele é derrotado e a partida termina.

Esta é a maneira de saber quem é o vencedor da partida. O número 20 também foi baseado nas regras de *Magic the Gathering*® (WIZARDS OF THE COAST, 2011).

Regra 5: Se as cartas da pilha acabarem, os jogadores deverão misturar as cartas que saíram do jogo e formar uma nova pilha.

Com este procedimento uma partida pode continuar por mais tempo, permitindo aos

jogadores desenvolver classes e heranças mais complexas.

3.8.2 Regras sobre cartas de classes e objetos

Regra 6:Cada jogador só pode criar uma única classe por turno. Esta classe não pode ser vazia.

Esta regra tem por objetivo incentivar o desenvolvimento de classes fortes, fazendo com que o jogador concentre o maior número de métodos e variáveis em uma mesma classe.

Regra 7: Um objeto só pode entrar em jogo se houver uma classe definida.

Assim como acontece em um programa real, não é possível criar objetos sem antes criar uma classe.

Regra 8: Se houver adição, edição ou remoção de métodos e variáveis em uma classe, todos os objetos instanciados a partir dessa classe devem ser retirados do jogo.

Reflete o que acontece ao modificar um programa no computador, pois será necessário compilar o código novamente, eliminando todos os objetos em execução.

Regra 9: O jogador não precisa criar objetos a partir de Classes de Armas que possuem métodos estáticos para executar a ação descrita no texto da carta.

Métodos estáticos podem ser executados sem a criação de objetos. Porém não há proibição de criar objetos a partir desse tipo de classe, desde que o jogador possua cartas de objeto do tipo Arma.

Regra 10: Ao editar uma classe, as cartas de variáveis e métodos não podem voltar para a mão do jogador, mas podem ser movidas para outra classe. Se não houver lugar onde colocá-las, as cartas devem sair do jogo.

Esse tipo de edição pode ser realizada desde que a Regra 8 seja respeitada.

3.8.3 Regras sobre cartas de variáveis e constantes

Regra 11: Em uma classe não pode haver duas ou mais cartas de variável ou constante com o mesmo nome.

Cada variável precisa ter um identificador único, portanto cartas de mesmo nome não

são permitidas. Se uma carta de mesmo nome, porém com maior valor for entrar em jogo, a carta de menor valor deve ser retirada o jogo.

Regra 12: Em uma classe ou relação de herança só pode haver uma única carta de variável do tipo lógico.

A representação dos valores “falso” e “verdadeiro” é feita com a disposição da carta na posição vertical e horizontal, respectivamente (ver a Figura 34, pág. 44), não sendo possível representar mais do que um valor lógico por vez.

3.8.4 Regras sobre cartas de métodos

Regra 13: Uma classe pode conter duas ou mais cartas de método com o mesmo nome, desde que os parâmetros (entre os parênteses) ou o tipo de retorno sejam diferentes.

Esta regra está de acordo com a sobrecarga de métodos. Se uma carta de mesmo nome, mesmos parâmetros e mesmo retorno for entrar em jogo, a carta anterior deve ser retirada o jogo.

Regra 14: Cada objeto pode executar um único método por turno.

Este limite é definido para que haja equilíbrio entre os jogadores e todos tenham a oportunidade de realizar jogadas em seu turno.

Regra 15: Durante uma ação de ataque, o jogador que sofre o ataque pode executar um método de defesa mesmo não sendo o seu turno.

Esta regra garante a utilidade dos métodos de defesa, pois não haveria lógica em o jogador do turno atual chamar um método de defesa quando nenhum dos outros jogadores pode atacar.

3.8.5 Regras sobre relações de herança

Regra 16: A carta Herança não conta como classe, podendo ser colocada em jogo no mesmo turno em que a subclasse for colocada na mesa.

Na prática, o código que cria a relação de herança, a palavra-chave *extends*, faz parte da subclasse, portanto neste caso a Regra 6 não se aplica.

Regra 17: Classes com a indicação "final" não podem ser superclasse numa relação de herança.

A regra reflete a utilização do modificador *final* nas relações de herança em *Java*. Na prática, usar *final* em uma classe garante que o código não será sobrescrito.

Regra 18: Não é permitido criar objetos a partir de classes com a indicação "abstract". Cartas desse tipo são para uso exclusivo em relações de herança.

Esta regra também reflete a utilização prática do modificador *abstract*.

3.8.6 Modificação de regras

Os jogadores podem alterar, remover ou acrescentar regras antes do início de uma partida, desde que:

a) não violem os conceitos da orientação a objetos;

b) sejam aceitas por todos os jogadores da partida;

c) seja analisada e aprovada pelo professor orientador (quando possível);

Algumas sugestões de modificação são:

- Aumentar ou reduzir os pontos de energia de cada jogador para prolongar ou antecipar o tempo de uma partida;
- Aumentar a quantidade de cartas trazidas para a mão no início de cada turno, tornando o jogo mais dinâmico;
- Encerrar o jogo quando as cartas da pilha acabarem.

Além de ensinar o jogo também tem objetivo de divertir, por isso está prevista a flexibilidade nas regras.

3.9 VARIAÇÕES DO JOGO

"P.O.O – O Jogo" não é apenas um jogo, e sim um conjunto de cartas que podem ser utilizadas em diversos jogos diferentes, assim como acontece com os quatro conjuntos do baralho tradicional (copas, espadas, paus e ouros).

Nesta seção são apresentadas sugestões de alguns jogos que utilizam as cartas

desenvolvidas neste trabalho. Cada jogo observa o nível de conhecimento do aluno e tem por objetivo explorar gradualmente os conceitos ensinados durante as aulas.

3.9.1 Jogo 1: Classes, variáveis e métodos

Objetivos Didáticos

- Fazer o aluno compreender que uma classe é composta por variáveis e métodos;
- Ensinar sobre restrições em nomes de variáveis e métodos;
- Ensinar o conceito de sobrecarga de métodos;

Objetivos do Jogo

- Criar a maior quantidade de classes com o maior número de variáveis e métodos que o jogador conseguir.

Cartas utilizadas

- Cartas de Classe (sem os modificadores *abstract*, *final*, e *com método estático*);
- Cartas de Variáveis e constantes;
- Cartas de Métodos;

Mecânica do Jogo

Dois ou mais jogadores podem participar deste jogo. Uma única pilha é formada com todas as cartas. Cada participante inicia o jogo com sete cartas tiradas do topo da pilha. Após um sorteio, define-se o jogador que começa e a ordem segue o sentido horário.

Em seu turno, o jogador realiza as seguintes ações:

1. Puxar uma carta da pilha (obrigatório);
2. Criar uma classe;
3. Adicionar variáveis e métodos as classes;
4. Editar classes (mover variáveis ou métodos de uma classe para outra);
5. Remover cartas da mesa para o lixo;
6. Encerrar o turno e passar a vez ao próximo jogador;

Restrições a serem observadas

- Uma classe não pode ter duas variáveis com mesmo nome;
- Uma classe pode ter dois ou mais métodos de mesmo nome, desde que os parâmetros ou tipos de retorno sejam diferentes (sobrecarga).

Duração da partida

Os jogadores podem acordar uma quantidade de turnos antes do início da partida, por exemplo, 10 turnos. Ou, os jogadores podem jogar até que não haja mais cartas na pilha.

Pontuação e vitória

Após encerrada a partida, para as cartas na mesa somam-se:

- 3 pontos para cada carta de classe;
- 2 pontos para cada carta de variável ou constante;
- 1 ponto para cada carta de método;

Vence o jogo aquele jogador que somar mais pontos ao final da partida.

3.9.2 Jogo 2: Criação de objetos

Objetivos Didáticos

- Fazer o aluno compreender que os objetos são criados a partir de uma classe;
- Explicar que são os objetos que executam as ações descritas pelos métodos.
- Cada objeto possui seus próprios valores de variáveis, apesar de executarem as mesmas ações.
- Os métodos modificam os valores das variáveis.
- Objetos comunicam-se através do envio de mensagens (parâmetros e retornos).

Objetivos do Jogo

- Construir classes, instanciar objetos (robôs), atacar os jogadores oponentes até tirá-los do jogo.

Cartas utilizadas

- Cartas de Classe (sem os modificadores *abstract*, *final*, e *com método estático*);

- Cartas de Variáveis e constantes;
- Cartas de Métodos;
- Cartas de Objeto (Robôs);

Mecânica do Jogo

Dois ou mais jogadores podem participar deste jogo. Uma única pilha é formada com todas as cartas. Cada participante inicia o jogo com sete cartas tiradas do topo da pilha. Cada participante inicia com 20 pontos de energia. Após um sorteio, define-se o jogador que começa e a ordem segue o sentido horário.

Em seu turno, o jogador pode realizar as seguintes ações:

1. Puxar uma carta da pilha (obrigatório);
2. Criar uma classe;
3. Adicionar variáveis e métodos as classes;
4. Editar classes (mover variáveis ou métodos de uma classe para outra);
5. Remover cartas da mesa para o lixo;
6. Criar objetos (Robôs);
7. Executar um método;
8. Encerrar o turno e passar a vez ao próximo jogador;

Restrições a serem observadas

- Ao executar um método, devem-se realizar as modificações nos valores das variáveis, conforme descrito na carta.
- Não é permitido criar objetos do tipo Robô sem que a classe tenha uma carta de *Energia*.
- Se o atributo *Energia* de um objeto chegar a ZERO, esse objeto é removido o jogo.
- Se um jogador não possui objetos para receber um ataque, o valor é removido de seus pontos de energia.

Duração da partida

A partida dura até que apenas um dos jogadores tenha ENERGIA em seu marcador;

Pontuação e vitória

Vence o jogo aquele jogador que possuir maior quantidade de pontos de energia ao

final da partida.

3.9.3 Jogo 3: Relações de herança

Objetivos Didáticos

- Fazer o aluno compreender que uma classe pode herdar características de outra classe;
- Explicar o conceito de sobreposição de métodos.
- Explicar sobre os modificadores de classe *abstract* e *final*.

Objetivos do Jogo

- Construir relações de herança entre as classes respeitando os conceitos ensinados.

Cartas utilizadas

- Cartas de Classe, incluindo as que possuem os modificadores *abstract* e *final*. Não usar classes com método estático;
- Cartas de Variáveis e constantes;
- Cartas de Métodos;

Mecânica do Jogo

Dois ou mais jogadores podem participar deste jogo. Uma única pilha é formada com todas as cartas. Cada participante inicia o jogo com sete cartas tiradas do topo da pilha. Após um sorteio, define-se o jogador que começa e a ordem segue o sentido horário.

Em seu turno, o jogador pode realizar as seguintes ações:

1. Puxar uma carta da pilha (obrigatório);
2. Criar uma classe;
3. Adicionar variáveis e métodos às classes;
4. Editar classes (mover variáveis ou métodos de uma classe para outra);
5. Criar relações de herança;
6. Remover cartas da mesa para o lixo;

Encerrar o turno e passar a vez ao próximo jogador;

Restrições a serem observadas

- Cartas com o modificador *abstract* só podem ser usadas como Superclasses.
- Cartas com o modificador *final* não podem ser usadas como superclasse.
- É permitido criar múltiplos níveis de herança.

Duração da partida

Os jogadores podem acordar uma quantidade de turnos antes do início da partida, por exemplo, 15 turnos.

Ou, os jogadores podem jogar até que não haja mais cartas na pilha.

Pontuação e vitória

Após encerrada a partida, para as cartas na mesa somam-se:

- 3 pontos para cada carta de classe;
- 2 pontos para cada carta de variável ou constante;
- 1 ponto para cada carta de método;
- A cada nível de herança dobram-se os pontos do nível anterior. Por exemplo, no segundo nível da herança as cartas valem: 6 pontos para classe, 4 pontos para variáveis e constantes, e 2 pontos para métodos. No terceiro nível de herança as cartas valem: 12 pontos para classe, 8 pontos para variáveis e constantes, 4 pontos para métodos. E assim por diante.

Vence o jogo aquele jogador que somar mais pontos ao final da partida.

3.9.4 Jogo 4: Todos os conceitos reunidos

Este jogo é uma expansão do jogo explicado no tópico 3.9.2, mas que inclui também a carta herança, classes com modificador *final*, classes com modificador *abstract*, classes com métodos estáticos, cartas de objeto arma, cartas de edição e qualquer outra carta disponível. Todas as restrições dos jogos 1, 2 e 3 devem ser observadas e a vitória é concedida ao jogador que possuir mais pontos de energia ao final da partida.

3.9.5 Jogo 5: Baseado em objetivos

Este jogo é uma variação dos jogos explicados nos tópicos 3.7.1 e 3.7.3. A diferença é que o objetivo do jogo muda a cada partida, mediante o sorteio de uma carta de objetivo.

A Figura 50 mostra exemplos de cartas Objetivo.

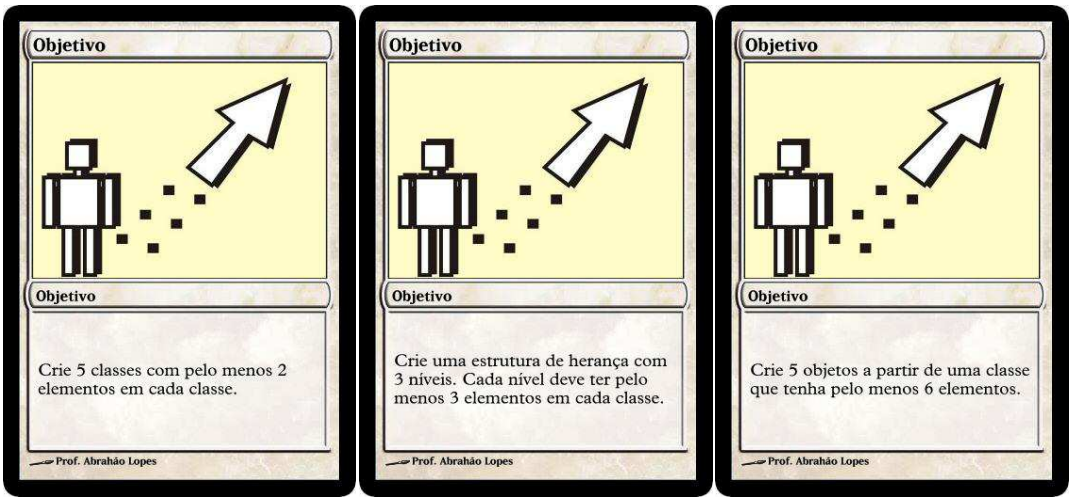


Figura 50: Cartas Objetivo

Vence a partida o jogador que conseguir atingir o objetivo primeiro.

3.9.6 Jogo 6: Reprodução de diagramas

Este jogo é uma variação do jogo baseado em objetivos (seção 3.7.6), porém ao invés de sortear uma carta, o objetivo é fixo: reproduzir através das cartas um diagrama em UML indicado pelo professor.

O professor deve conhecer as cartas disponíveis no baralho e desenhar um diagrama UML que use as variáveis, métodos e estruturas de herança existentes.

Vence o jogador que primeiro conseguir reproduzir o diagrama proposto.

3.9.7 Outros jogos

Além dos jogos apresentados nesta seção, os professores e alunos são incentivados a desenvolverem e compartilharem outros jogos utilizando as cartas de “P.O.O. – O Jogo”.

O Capítulo 4 apresenta os resultados de algumas experiências realizadas a partir da aplicação dos jogos de cartas de “P.O.O – O Jogo” entre alunos do nível médio do curso técnico em informática.

CAPÍTULO 4

4 APLICAÇÃO DO “P.O.O. – O JOGO”

Para validara metodologia apresentada neste trabalho foram realizadas várias experiências aplicando tal metodologia. Ao todo, 50 estudantes participaram das sessões de testes que reunia grupos de 6 a 10 participantes em média.

Os estudantes serão classificados em grupos de acordo com a série que cursam, devido às disciplinas que já estudaram. Foram convidados a participar os alunos do 2º, 3º e 4º anos do curso de Informática, pois todos já haviam cursado a disciplina de Programação Orientada a Objetos.

Cada sessão teve duração aproximada de 90 minutos, iniciando-se com a aplicação do Questionário 1 – Conhecimentos Prévios (ver Anexos).

A seguir era realizada uma breve explanação sobre a metodologia, apresentando a idéia do jogo, os tipos de cartas utilizadas e o objetivo de cada jogo.

Após essa introdução, os alunos formavam duplas e recebiam conjuntos de cartas a fim de participarem dos jogos propostos.

Entre as partidas eram realizadas atividades envolvendo a construção de diagramas e a implementação e explanações sobre o código fonte referente às ações realizadas pelas cartas, a fim de ilustrar melhor os conceitos.

Em cada sessão de teste eram apresentados 3 ou 4 dos jogos propostos, sendo que o Jogo 4, aquele que reúne todos os conceitos, foi utilizado em todas as sessões.

Após cada partida, também era solicitado que os alunos comentassem sobre suas impressões sobre o jogo que haviam participado, relatando os pontos positivos, pontos negativos e sugestões para melhorá-lo.

Ao final da sessão, foi solicitado aos alunos que respondessem o Questionário 2 – Avaliação da Experiência (ver Anexos), a fim de registrar formalmente sua opinião.

Em ambos os questionários o aluno não necessitava se identificar.

A Figura 51 apresenta fotos de uma das sessões da experiência realizada com os estudantes.

Figura 51: Alunos durante aplicação da experiência

Os tópicos 4.1, 4.2 e 4.3 apresentam detalhes sobre os dados coletados, agrupando-os por turma, apesar de, durante as experiências, alunos de diferentes turmas participaram conjuntamente.

4.1 ALUNOS DO 2º ANO

A experiência foi realizada com 19 alunos do 2º ano, que cursaram no semestre 2011.2 a disciplina Programação Orientada a Objetos. Anteriormente, a única disciplina cursada na área de programação foi Algoritmos, que envolve apenas programação estruturada.

A experiência foi aplicada no final do semestre, desta forma, subentende-se que os alunos estudaram todos os conceitos abordados pela metodologia e que ainda possuem recente lembrança de tais conceitos.

4.1.1 Resultados do questionário 1: conhecimentos prévios

Conforme explicado na introdução do Capítulo 4, este questionário foi aplicado antes de ser apresentada a metodologia descrita neste trabalho, portanto os alunos não sabiam de antemão que se tratava do uso de jogos, nem que seriam utilizadas cartas neste jogo.

A seguir são apresentados os resultados de cada uma das questões contidas no questionário, e traçados comentários sobre os pontos que se demonstram relevantes.

Questão 1: Em sua opinião, o uso de jogos educativos contribui para o aprendizado?

Esta pergunta visa saber se o aluno considera o uso de jogos traz alguma contribuição para o aprendizado.

Todos os 19 participantes responderam afirmativamente a esta pergunta.

Questão 2: Durante as disciplinas que você cursou, outro professor utilizou jogos como método de ensino?

Esta questão não se limita ao atual curso no ensino médio. Os alunos poderiam relatar experiências vividas em escolas anteriores, mesmo no curso do ensino fundamental.

Mesmo diante de tal abertura, apenas 2 alunos responderam que participaram alguma vez de aulas em que foram usados jogos didáticos para o ensino. As disciplinas em que os jogos foram aplicados foram: inglês e matemática.

O Gráfico 1 apresenta as demais respostas dessa pergunta.

Gráfico 1: Experiências anteriores com jogos didáticos

Questão 3: Você tem interesse em conhecer outros jogos educativos que auxiliem em seus estudos?

Todos os alunos responderam afirmativamente a esta pergunta e traçaram os seguintes comentários:

"Desenvolve habilidades mentais".

"É interessante principalmente quanto à fixação dos conceitos".

"Estimula o aluno".

"Ajuda a desenvolver o raciocínio lógico de forma mais rápida".

"É uma forma dinâmica e estimulante".

"A utilização de jogos educativos envolve os alunos no estudo sobre certo assunto, esse envolvimento é inquestionável para o aprendizado".

"Estimula a aprendizagem".

"Toda abordagem nova é interessante".

"Para uma maior interação entre os colegas e para uma maior assimilação do conteúdo".

"É uma melhor didática".

"Todos os métodos que auxiliem os estudos são bem vindos".

Questão 4: Em relação ao formato em que o jogo é apresentado, qual o seu tipo favorito?

Esta pergunta foi apresentada para verificar se o formato escolhido para apresentação dos jogos propostos neste trabalho (cartas) foi adequada.

O Gráfico 2 apresenta as respostas. Observa-se que a maioria dos alunos (10) prefere o uso de jogos eletrônicos, seguido de jogos de tabuleiro (6) e por fim jogos com cartas (3). Outros formatos de apresentação, como papel e caneta, não foram indicados.

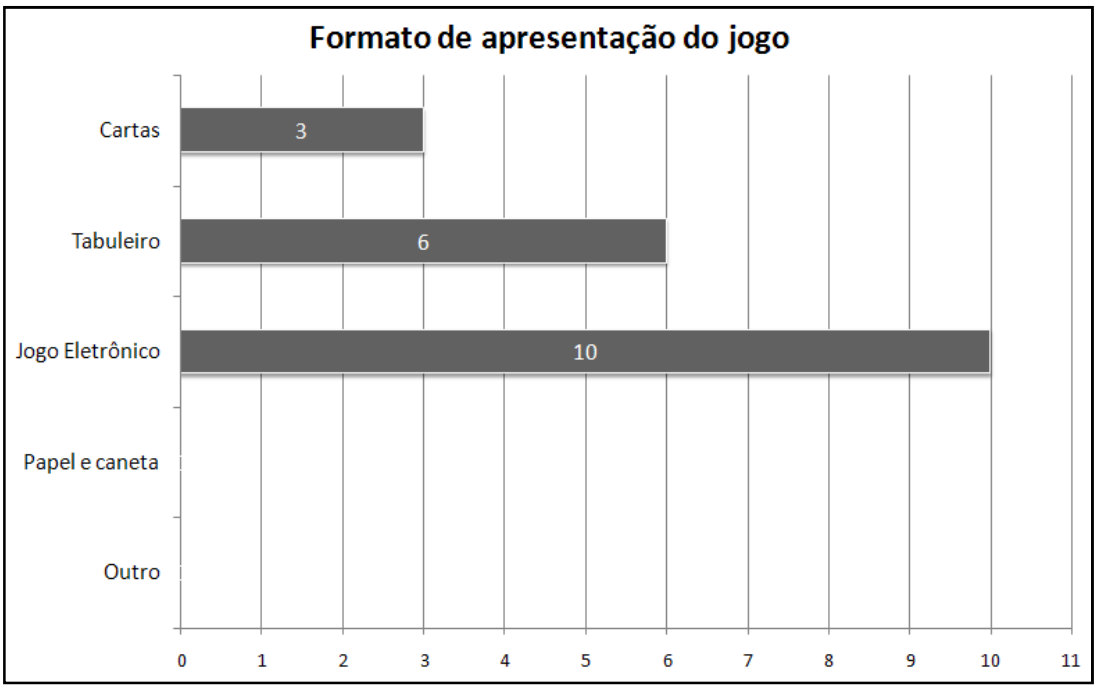


Gráfico 2: Formato de apresentação do jogo

Os seguintes comentários foram apresentados por alguns dos alunos que escolheram a opção Jogo Eletrônico:

"Tudo ao alcance de um click".

"Afinidade, costume, etc."

"Melhor visualização".

"Tem mais a ver com informática".

"Todos os formatos são bem-vindos se tiverem uma boa didática".

Questão 5: Qual o principal elemento que você busca em um jogo educativo?

Esta questão foi apresentada a fim de verificar que elemento os alunos buscam em um

jogo, o que os motiva a jogá-lo.

O Gráfico 3 mostra que a maioria dos alunos (11) deseja aprender a teoria enquanto joga, outros 6 alunos desejam aplicar os conhecimentos adquiridos para vencer o jogo.

Um único aluno apontou a competitividade como sendo o elemento indispensável no jogo e outro deseja que o jogo seja cooperativo. Curiosamente, nenhum dos alunos considera a diversão como elemento primordial em um jogo educativo.

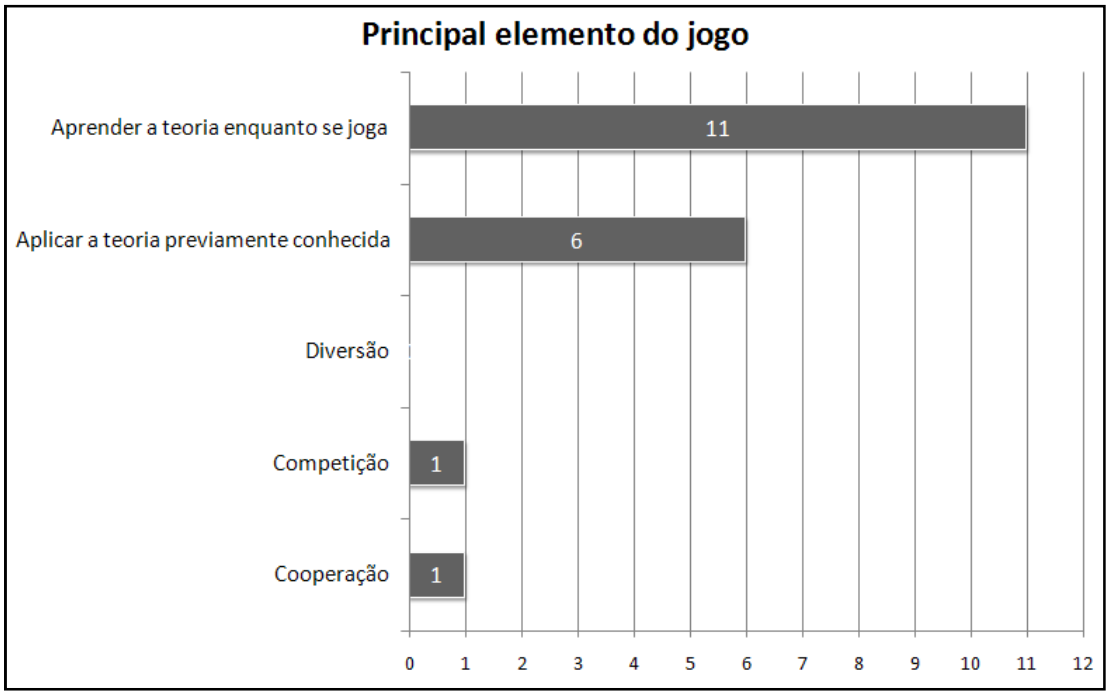


Gráfico 3: Principal elemento do jogo

Questão 6: Classifique seu conhecimento prévio em Programação Orientada a Objetos

Esta pergunta visa conhecer o nível de conhecimento do aluno. Como não houve possibilidade de aplicar provas para mensurar precisamente o conhecimento, optou-se por classificá-lo de acordo com a compreensão dos conceitos que o aluno possui.

No questionário 2: Avaliação da Experiência esta mesma pergunta é apresentada, a fim de verificar se houve acréscimo no conhecimento dos alunos.

O Gráfico 4 apresenta as respostas para esta pergunta.

Observa-se que são poucos (2) os alunos que obtiveram total compreensão dos conceitos de orientação a objetos, mesmo tendo cursado a disciplina no semestre 2011.2. A maioria dos alunos (9) compreende os conceitos, mas possuem dificuldade em programar

enquanto os demais afirmam ter dificuldades na maioria dos conceitos.

Um dos alunos comentou que mesmo tendo facilidade em compreender os conceitos, ainda assim, não consegue desenvolver os programas da maneira que gostaria.

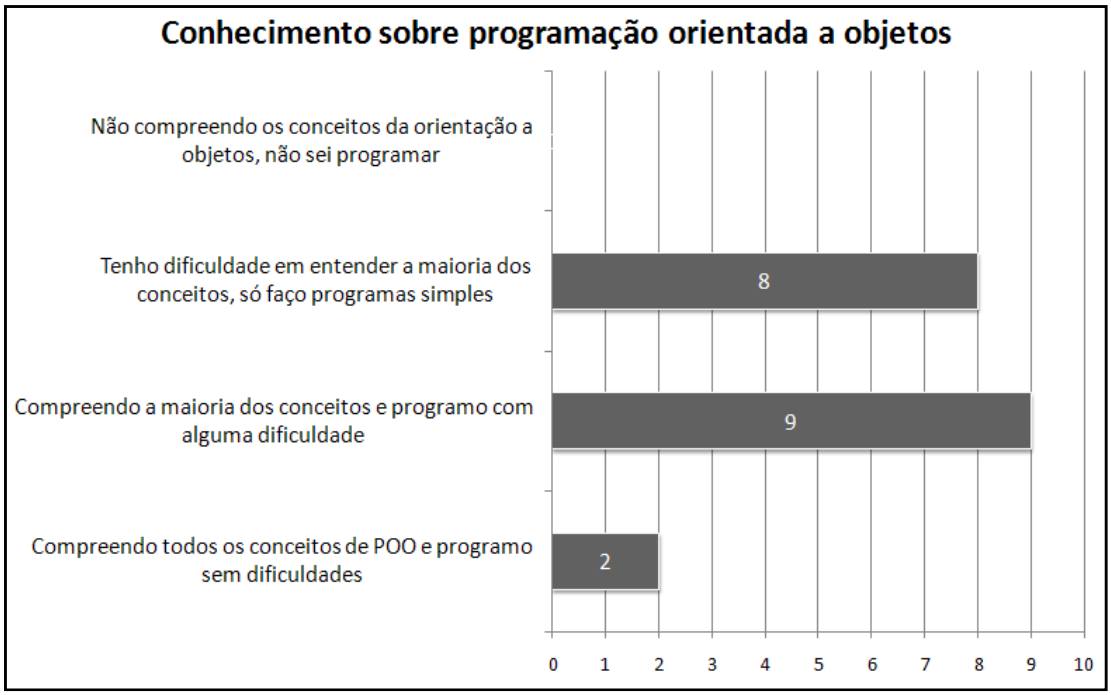


Gráfico 4: Nível de conhecimento sobre P.O.O.

Questões de 7 a 20: Conhecimentos sobre conceitos específicos

As questões 7 a 20 tem por objetivo investigar detalhadamente onde se encontram as dificuldades dos alunos.

Tais questões aparecem em ambos os questionários, a fim de permitir verificar se houve assimilação do conteúdo após a experiência.

As questões 7 (Classe), 10 (Método), 14 (Objeto), 17 (Variável) e 18 (Valor inteiro) foram respondidas afirmativamente por todos os participantes.

Na questão 15, sobre Herança, apenas 1 aluno respondeu não conhecer este conceito. De acordo com o comentário do aluno, este foi um dos últimos assuntos ministrado pelo professor e tal aluno afirma não ter estudado adequadamente.

A questão 16, sobre Constantes, apresentou 2 alunos com dúvida sobre este conceito. Algo curioso, uma vez que não se trata de um assunto específico da orientação a objetos e sim da construção de algoritmos. Aqui foi identificada uma pequena deficiência oriunda da disciplina de Algoritmos. Da mesma maneira, na questão 19, na qual apenas um aluno

apontou não compreender o que é um Valor lógico.

Na questão 20, sobre Diagramas de Classe, apenas 2 alunos afirmaram não saber ler ou desenhar esse tipo de diagrama.

O Gráfico 5 ilustra as respostas das demais questões.

Através das questões 8, 9 e 13, sobre Classe Abstrata, Classe Final e Sobreposição de Métodos, respectivamente, observa-se que apesar dos alunos compreenderem o conceito de herança, tópicos específicos sobre este assunto não foram assimilados adequadamente. Apresentam 78%, 73% e 78% de dúvidas, respectivamente.

Percebe-se também que há deficiência no assunto sobre métodos, pois mesmo o conceito de parâmetro, abordado na questão 11, algo indispensável para a troca de mensagens entre os objetos, não é compreendido por 47% dos 19 alunos entrevistados.

Da mesma maneira, o conceito perguntado na questão 12, Método Estático, teve uma deficiência ainda maior: 73% dos alunos não compreendem este conceito.

Acredita-se que os alunos apliquem tais conceitos nos programas que desenvolvem, mas não estejam associando a teoria com o código que constroem, pois é bastante comum o uso de métodos estáticos e parâmetros ao se trabalhar com a linguagem *Java*.

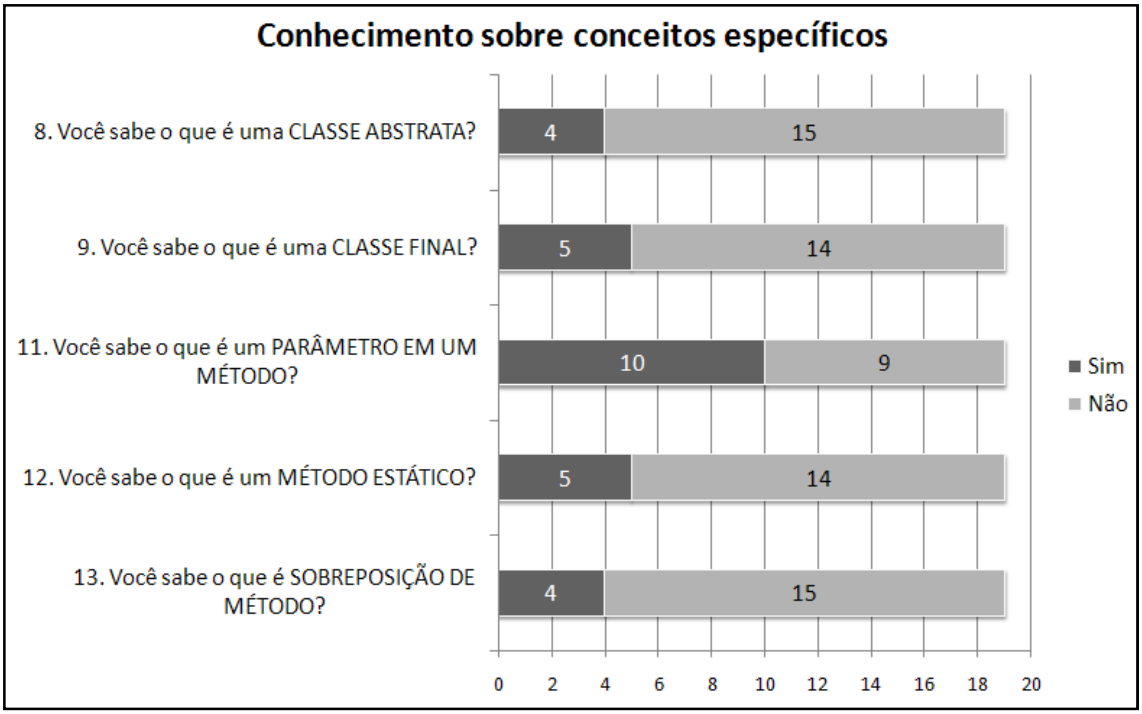


Gráfico 5: Conhecimento sobre conceitos específicos

4.1.2 Resultados do questionário 2: avaliação da experiência

Esta seção trata dos resultados obtidos após a aplicação da metodologia entre os alunos. O questionário 2: Avaliação da Experiência foi aplicado ao final de cada uma das sessões de teste a fim de obter a impressão imediata dos alunos, sem que eles tivessem oportunidade de consultar outras fontes, como livros, em busca de reforçar os conceitos apresentados.

As questões de 1 a 5 referem-se a metodologia em si, buscando a opinião a respeito do tema do jogo, facilidade no entendimento das regras, construção de código e diagramas.

As questões de 6 a 20 são idênticas às existentes no questionário 1: Conhecimentos Prévios, tendo por objetivo observar se houve progresso no conhecimento dos alunos.

Uma última questão solicita ao aluno que faça seus comentários e dê sugestões sobre como melhorar a metodologia.

A seguir são apresentados os resultados deste segundo questionário e a análise dos pontos relevantes.

Questão 1: Na sua opinião, a ambientação do jogo (Batalha de Robôs) é um tema atrativo e auxilia na compreensão dos conceitos?

Esta pergunta foi realizada a fim de verificar se o tema escolhido como plano de fundo para o jogo era adequado e atrativo para os participantes.

Nesta questão, 17 alunos responderam afirmativamente e os seguintes comentários foram recebidos:

"Eu gostei muito. Parece com o desenho animado Medabots que passava na televisão."

"Sim, pois tem a ver com tecnologia."

"Tudo a ver com programação."

Apenas 2 alunos responderam negativamente e apontaram as seguintes justificativas:

"Podia ter outros temas, não só robôs."

"Podia ser mais parecido com Magic, com feitiços e magias. Seria mais legal assim."

Questão 2: Com relação às diferentes possibilidades de jogos utilizando as cartas, você considera que:

Esta questão foi proposta a fim de verificar se a variedade de jogos propostos, que

utilizam o conjunto de cartas desenvolvido, ajuda ou atrapalha no momento de compreender os conceitos.

Nesta questão, 4 participantes responderam que um único modo de jogo seria suficiente, pois muitas regras diferentes acabam confundindo o jogador. Um dos alunos comentou: *"Muita coisa diferente confunde a cabeça de quem está jogando"*.

Os outros 15 alunos preferem que os conceitos sejam introduzidos aos pontos através dos diferentes jogos. O seguinte comentário foi recebido: *"Prefiro os conceitos pouco a pouco, assim fica mais fácil aprender"*.

Questão 3: Com relação à facilidade de uso:

A questão 3 foi proposta a fim de saber se os alunos consideraram o jogo fácil ou difícil de ser compreendido e jogado.

Nesta questão, 6 participantes responderam que é um jogo simples de ser jogado e é possível aprender sozinho, apenas lendo as regras, enquanto 13 disseram que apesar de ser fácil, é necessário alguém que possa ensinar como jogar.

Um dos alunos comentou o seguinte: *"É bom ter alguém que saiba sobre o jogo para ensinar, ou a pessoa fica perdida nos jogos que usam todas as regras"*.

Questão 4: Você considera que as implementações de código fonte utilizados para descrever o funcionamento das cartas auxiliam a entender os conceitos explicados?

A pergunta é clara e tem por objetivo verificar a eficácia do uso de códigos-fonte para reproduzir no computador o funcionamento das cartas.

Entre os participantes, 17 alunos responderam afirmativamente e apenas 2 responderam negativamente.

Entre os comentários recebidos foi sugerido por dois alunos ampliar o número de cartas para que houvesse mais exemplos e opções de implementação.

Questão 5: Você considera que a construção de diagramas a partir das cartas auxiliam a entender os conceitos explicados?

Esta questão também é clara e tem por objetivo saber a opinião a respeito da construção de diagramas de classe, tanto em classes simples, quanto nas relações de herança.

Todos os participantes responderam afirmativamente e demonstraram-se satisfeitos com as atividades propostas que envolvem a criação de diagramas.

Questão 6: Classifique o seu conhecimento prévio em programação orientada a objetos

As questões de 6 a 20 são idênticas às questões de mesma numeração apresentadas no questionário 1: conhecimentos prévios. A repetição das mesmas questões em ambos os questionários tem o objetivo de verificar se houve aumento do conhecimento após aplicação da metodologia.

Nesta questão, 50% dos alunos que afirmavam ter dificuldade em entender os conceitos, mudaram sua resposta, permanecendo 4 alunos com a mesma resposta. Um deles comentou: *"Aprendi várias coisas, mas é que programação é difícil pra mim"*.

A resposta "Compreendo todos os conceitos..." apresentou um crescimento de 100%, com 2 novos alunos mudando a resposta anterior para esta, agora 4 no total.

A resposta "Compreendo a maioria dos conceitos..." também ganhou 2 pontos, conforme visto no Gráfico 6. Para os dados anteriores, consultar o Gráfico 4, pág. 80.

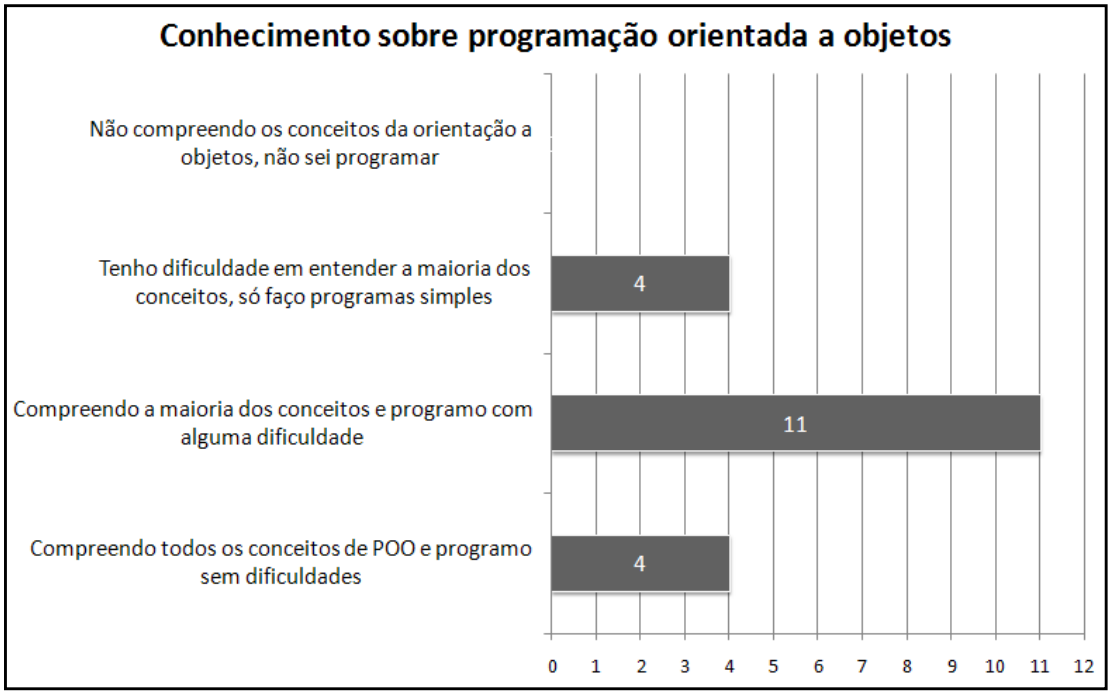


Gráfico 6: Conhecimento sobre P.O.O. após aplicação da metodologia

Dois alunos comentaram que apesar de agora compreenderem os conceitos, não mudaram sua resposta de "Compreendo a maioria dos conceitos..." para "Compreendo todos os conceitos..." devido ao complemento desta última alternativa que diz "programo sem dificuldades".

Questões de 7 a 20: Conhecimentos sobre conceitos específicos

As questões sobre Classe (7), Método (10), Objeto (14), Variável (17) e Valor inteiro (18), não sofreram alterações, continuando com respostas afirmativas para todos os participantes.

A questão 15, sobre Herança, e a questão 16, sobre Constante, neste segundo questionário receberam todas as respostas afirmativas, mostrando que o uso dos jogos foi suficiente para tirar a as dúvidas que os alunos tinham quanto a estes assuntos.

A questão 20, sobre Diagramas de Classe, ainda permaneceu com 1 aluno apresentando dúvida neste assunto. Tal aluno deixou o seguinte comentário registrado: *"Ainda tenho dificuldade de entender os diagramas, quando bota menos, quando bota mais, o que fica dentro dos parênteses, etc."*.

O comentário refere-se ao uso dos símbolos para representar os modificadores *private* e *public*, e sobre os parâmetros utilizados nos métodos.

O Gráfico 7 apresenta os dados das demais questões.

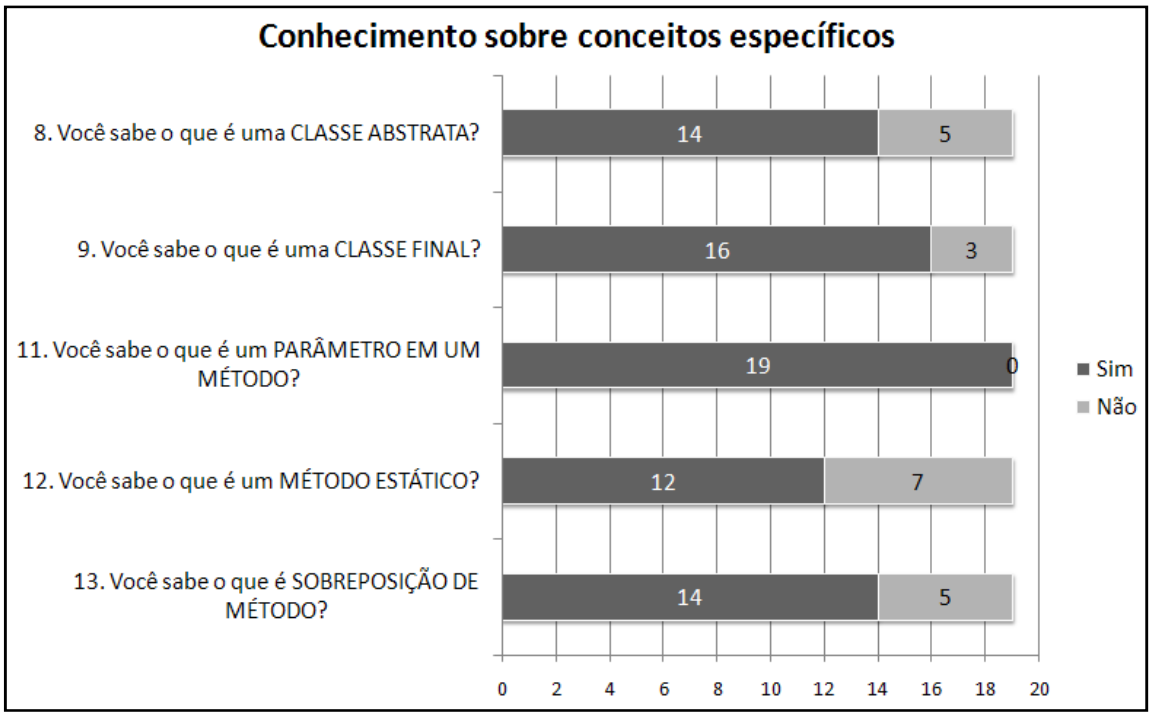


Gráfico 7: Conhecimentos sobre conceitos específicos após aplicação da metodologia

A questão 8, sobre Classe Abstrata, teve uma redução de alunos que não compreendiam o conceito, passando de 78% para 26%.

Na questão 9, sobre Classe Final, também houve redução nas dúvidas, caindo de 73% para 15%.

A questão 11, sobre Parâmetros de um método, obteve 100% de respostas afirmativas sobre a compreensão do conceito, um aumento de 47% em relação ao primeiro questionário.

A questão 12, sobre Método Estático foi, entre todas, a questão que manteve o maior número de dúvidas, mas, ainda assim, obteve uma redução considerável, em que metade dos alunos que tinham dúvidas, conseguiu esclarecê-las. Diminuiu de 73% para 36% os alunos com dúvidas nesse conceito.

Por fim, a questão 13, sobre Sobreposição de métodos também obteve uma redução satisfatória nas dúvidas, passando de 78% para 26%.

4.2 ALUNOS DO 3º ANO

Ao todo, 26 alunos do 3º ano participaram da aplicação da metodologia. Estes alunos cursaram nos semestres do ano de 2010 a disciplina Programação Orientada a Objetos, e no ano de 2011 a disciplina Técnicas de Programação, que utilizava a linguagem *Java* para construção de programas. Também estudaram Algoritmos anteriormente.

Alguns alunos comentaram que a turma não teve uma disciplina de POO proveitosa, apontando o excesso de teoria apresentada, dificuldades com a linguagem e falta de exercícios práticos motivadores como motivo do baixo rendimento.

4.2.1 Resultados do questionário 1: conhecimentos prévios

Uma vez que o objetivo de cada questão foi explicado na apresentação dos tópicos 4.1.1 e 4.1.2, esta seção focará especificamente nos resultados.

Questão 1: Em sua opinião, o uso de jogos educativos contribui para o aprendizado?

Todos responderam afirmativamente.

Os seguintes comentários foram apresentados:

“É Muito bom aprender com métodos diferentes e divertidos”.

“É preciso utilizar algo que chame atenção dos alunos”.

“É mais legal aprender brincando”.

“O uso de jogos permite o aprendizado de uma forma descontraída e talvez até de forma mais bem aproveitada”.

“É uma forma de chamar a atenção do aluno, podendo assim passar o assunto ao aluno”.

“Com uma metodologia diferente o aprendizado fica dinâmico”.

“Pois através de jogos podemos adquirir conhecimentos e fixar conhecimentos já adquiridos”.

“Jogos são interativos por si, eles focam nossa atenção no jogo e se as regras forem para o aprendizado, é mais fácil aprender”.

“Pois fazer com que o aprendizado seja um pouco ‘menos sério’ é muito mais interessante, fazendo com que o aluno tenha mais vontade de aprender”.

“Uma forma de aprendizagem onde há dinâmica e interação se torna mais fácil e divertido o processo de aprendizagem”.

Questão 2: Durante as disciplinas que você cursou, outro professor utilizou jogos como método de ensino?

Aqui, 8 alunos (30%) relataram ter participado de pelo menos uma atividade envolvendo jogos didáticos nas disciplinas Inglês e Química.

Questão 3: Você tem interesse em conhecer outros jogos educativos que auxiliem em seus estudos?

Todos responderam afirmativamente.

Um aluno comentou que deseja encontrar jogos para os assuntos que ele considera difícil, a fim de aprender com mais facilidade. Outro comentou que tem curiosidade, pois nunca participou de nenhuma experiência do tipo.

Questão 4: Em relação ao formato em que o jogo é apresentado, qual o seu tipo favorito?

A preferência dos alunos também é por jogos eletrônicos (50%), seguido de jogos de tabuleiro (44%). Jogos com cartas e usando papel-e-caneta obtiveram apenas 3% cada.

A seguir, são relacionados alguns dos comentários apresentados, todos relacionados ao

uso de jogos eletrônicos.

"Por oferecer mais comodidade".

"Permite uma maior dinamicidade entre os jogadores".

"Já que se trabalha na área da informação, é muito mais interessante aprender com jogos eletrônicos".

"É mais interativo".

Questão 5: Qual o principal elemento que você busca em um jogo educativo?

42% dos alunos desejam aprender a teoria enquanto jogam. 28% buscam diversão. 15% querem aplicar a teoria já conhecida e 15% buscam competição.

Um comentário recebido a respeito dessa questão:

"Para mim não seria necessário se jogar em sala de aula se não com o objetivo de aprender".

Questão 6: Classifique seu conhecimento prévio em programação orientada a objetos

58% dos alunos escolheram a alternativa "Tenho dificuldade em entender a maioria dos conceitos, só faço programa simples". Os outros 42% escolheram "Compreendo a maioria dos conceitos e programo com alguma dificuldade".

Questões de 7 a 20: Conhecimentos sobre conceitos específicos

Assim como aconteceu com os alunos do 2º ano, o Gráfico 8 mostra deficiência nos assuntos Classe Abstrata, Classe Final e Sobreposição de Método, relacionados ao assunto Herança, chegando a 88% no conceito de sobreposição.

Também se observa dificuldades nos conceitos Parâmetros de métodos e Métodos Estáticos.

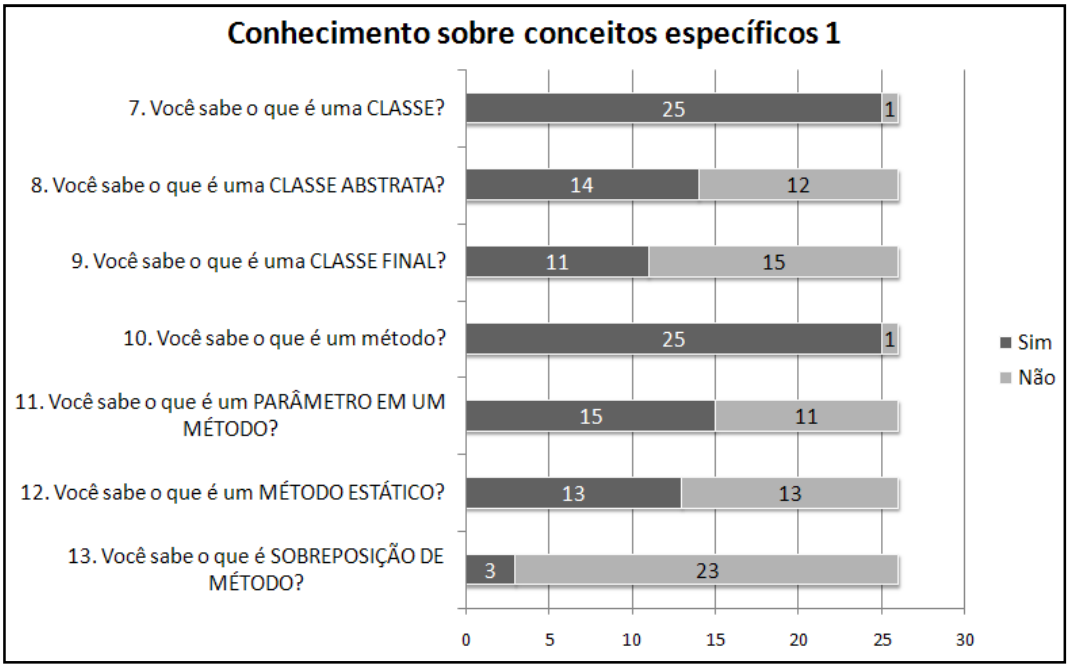


Gráfico 8: Conhecimentos sobre conceitos específicos 1

Com relação às demais respostas, apresentadas no Gráfico 9, destacam-se 23% dos alunos com dificuldades em ler ou escrever diagramas de classe.

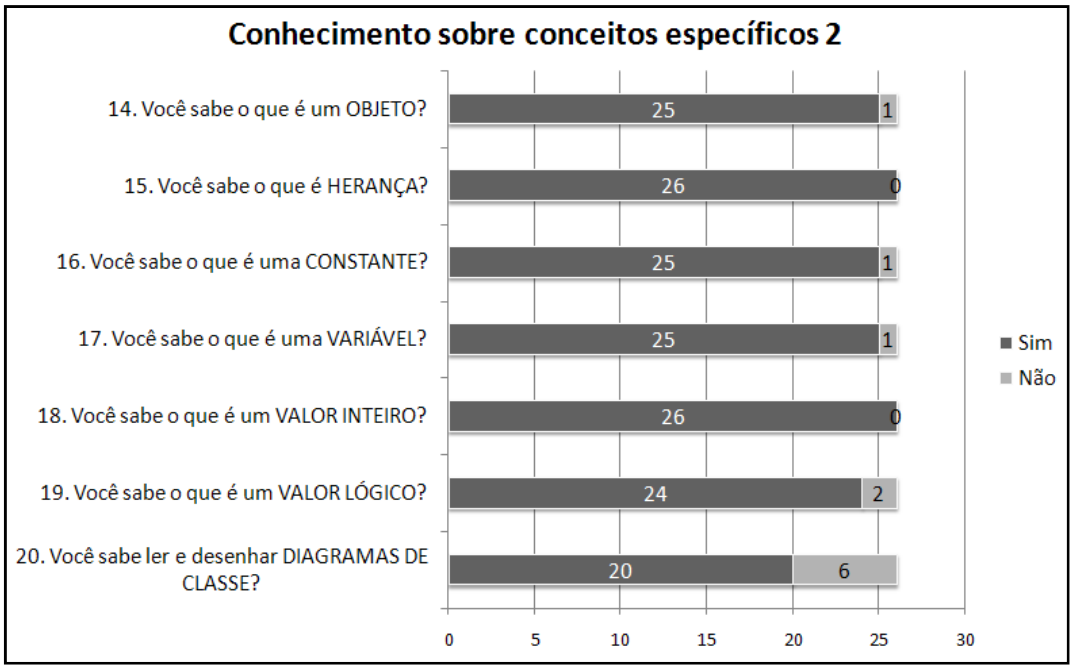


Gráfico 9: Conhecimentos sobre conceitos específicos 2

4.2.2 Resultados do questionário 2: avaliação da experiência

Questão 1: Na sua opinião, a ambientação do jogo (Batalha de Robôs) é um tema atrativo e auxilia na compreensão dos conceitos?

Todos os participantes responderam afirmativamente a esta questão.

Um aluno comentou: *"Pois faz com que haja incentivo para aprender mais e criar robôs mais fortes"*.

Questão 2: Com relação às diferentes possibilidades de jogos utilizando as cartas, você considera que:

88% preferem que os conceitos sejam ensinados pouco a pouco através de diferentes jogos, enquanto 12% desejam utilizar um único jogo envolvendo todos os conceitos.

Questão 3: Com relação à facilidade de uso:

54% responderam que é possível aprender a jogar sozinho, apenas lendo as regras. 46% disseram que é preciso alguém que ensine como jogar.

Os seguintes comentários foram apontados:

"O jogo faz menções a POO, e quem não tem uma noção básica, não consegue realizar a proposta do jogo".

"Para quem não fez uma boa disciplina de POO no período anterior, vai ficar voando em algumas coisas. É bom ter alguém pra ensinar".

"Até dá para aprender sozinho, mas tendo quem ensine é melhor, com certeza".

Questão 4: Você considera que as implementações de código fonte utilizados para descrever o funcionamento das cartas auxiliam a entender os conceitos explicados?

Todos os participantes responderam que o código fonte auxilia na compreensão dos conceitos.

Um aluno registrou o seguinte comentário: *"Até por que é pelo código fonte que saberemos como usar a carta"*.

Questão 5: Você considera que a construção de diagramas a partir das cartas auxiliam a entender os conceitos explicados?

Todos os participantes responderam afirmativamente a esta questão.

Os seguintes comentários foram apresentados:

“Ficou mais fácil entender os diagramas após ver o jogo”.

Questão 6: Classifique o seu conhecimento prévio em programação orientada a objetos

8% dos alunos responderam que após usar o jogo passaram a conhecer todos os conceitos de POO. Essa alternativa não foi escolhida no primeiro questionário.

73% dos participantes escolheram a alternativa “Compreendo a maioria dos conceitos...”, uma melhora de 31% em relação à primeira pesquisa.

19% mantiveram a alternativa “Tenho dificuldade em entender a maioria dos conceitos...”, entretanto observamos uma redução significativa, pois o questionário 1: Conhecimentos Prévios apontou 58% de respostas com esta opção.

O seguinte comentário foi registrado:

“*É tão mais fácil entender os assuntos usando esse jogo. Bem que eu queria que meu professor tivesse usando uma didática dessas quando deu a disciplina para nós. A maioria dos alunos teve muita dificuldade em POO”*.

Questões de 7 a 20: Conhecimentos sobre conceitos específicos

O Gráfico 10 mostra as respostas das questões de 7 a 13 e deve ser comparado com o Gráfico 8, pág. 89.

Observaram-se melhorias em todos os itens, com destaque para a Sobreposição de Método (item 13) que caiu de 88% de dúvidas para 27%.

O conceito de método (item 12) estático passou a ser compreendido por 88% dos alunos, quando anteriormente apenas 50% entendiam este assunto.

Também os conceitos de Classe Final (item 9) e Classe Abstrata (item 8) no segundo questionário figuram com 92% e 80% de compreensão, respectivamente.

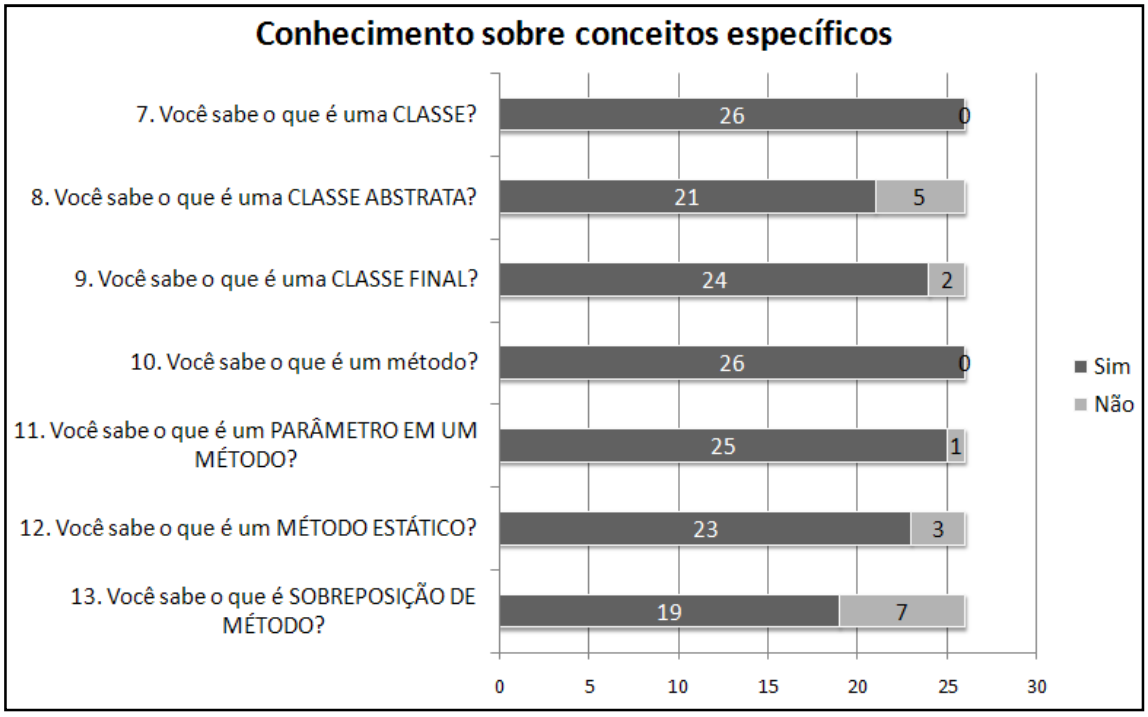


Gráfico 10: Conhecimentos sobre conceitos específicos após aplicação da metodologia

As questões 14 a 19, receberam respostas afirmativas de todos os alunos. Anteriormente de 1 a 2 alunos possuíam dúvidas nos conceitos de Constante, Objeto, Variável e Valor Lógico, que foram elucidadas após a aplicação da metodologia.

Na questão 20, sobre Diagramas de Classe, apresentou uma redução de 23% para 7% nos alunos com dúvidas.

4.3 ALUNOS DO 4º ANO

Cursaram as disciplinas de Algoritmos, Programação Orientada a Objetos (no ano 2009), Técnicas de Programação, Programação *Web* e Engenharia de *Software*. Com exceção de Algoritmos, as demais disciplinas abordam conceitos de Orientação a Objetos.

Apenas 5 alunos com este perfil participaram da experiência devido ao encerramento antecipado das aulas nesta turma em decorrência do início das aulas nas universidades ocorrer antes do término do semestre letivo de 2011.2.

4.3.1 Resultados do questionário 1: conhecimentos prévios

Questão 1: Em sua opinião, o uso de jogos educativos contribui para o aprendizado?

Todos os participantes afirmaram os jogos didáticos auxiliam o aprendizado.

Questão 2: Durante as disciplinas que você cursou, outro professor utilizou jogos como método de ensino?

Um único aluno afirmou ter participado de aulas em que o professor usou jogos didáticos. Os demais participantes não lembram ou afirmam não terem participado de qualquer aula envolvendo jogos didáticos, conforme o Gráfico 11.

Um dos alunos apresentou a seguinte justificativa a esta questão:

"O uso de uma metodologia "não-clássica", que torne a aprendizagem mais interessante e menos entediante sempre contribui para uma melhor aprendizagem, e os jogos educativos vão por esse caminho."

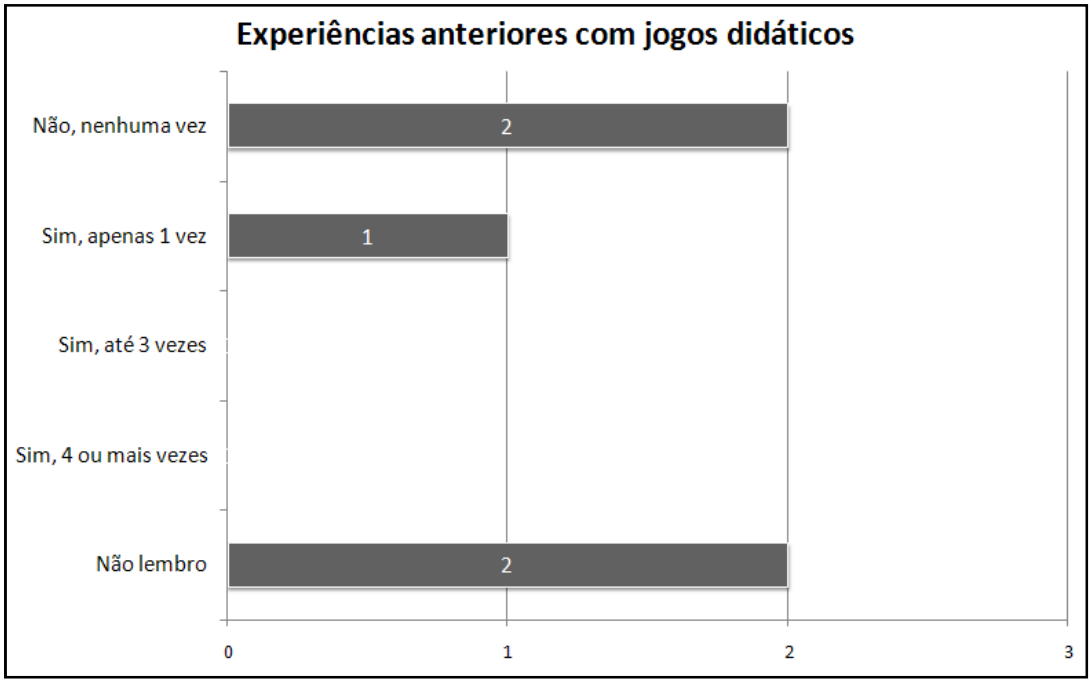


Gráfico 11: Alunos do 4º ano: Experiências anteriores com jogos didáticos

Questão 3: Você tem interesse em conhecer outros jogos educativos que auxiliem em seus estudos?

Todos afirmaram ter interesse em conhecer mais jogos didáticos.

Um aluno apresentou o seguinte comentário a esta questão:

"Gostaria que os professores se modernizassem e inserissem outros meios de ensino em suas aulas, como os jogos educativos, por exemplo."

Questão 4: Em relação ao formato em que o jogo é apresentado, qual o seu tipo favorito?

Nesta pergunta, 3 alunos optaram pelos jogos eletrônicos e 2 por jogos com cartas.

Uma justificativa apresentada foi: *"Apesar de ter marcado a opção Jogo Eletrônico, por proporcionar maiores possibilidades de ação, qualquer outro tipo também seria válido"*.

Questão 5: Qual o principal elemento que você busca em um jogo educativo?

Nesta pergunta, 3 alunos apontaram que desejam aprender a teoria enquanto jogam, enquanto 2 alunos afirmaram que buscam a diversão nos jogos.

Um aluno comentou: *"É muito melhor aprender a teoria já na prática, enquanto se joga. Torna o ensino mais dinâmico, menos metódico"*.

Questão 6: Classifique seu conhecimento prévio em programação orientada a objetos

Os valores para esta questão são apresentadas no Gráfico 12. Apenas 1 aluno diz compreender todos os conceitos de POO, enquanto 2 alunos possuem dificuldades em compreender conceitos e outros 2 alunos compreendem os conceitos mas possuem dificuldades em programar.

Sobre esta questão, um dos alunos que respondeu compreender os conceitos e programar com dificuldade, comentou:

"Até entendo os conceitos, só de vez em quando é que tenho que recorrer a um livro para lembrar ou aprender alguma coisa."

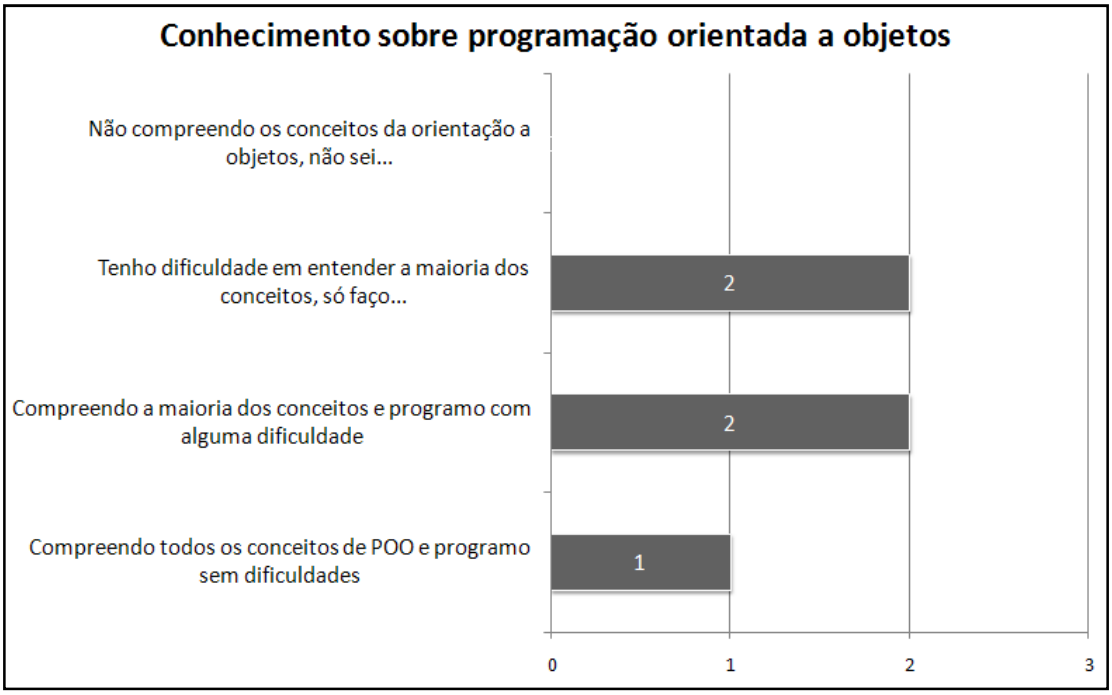


Gráfico 12: Conhecimento prévio sobre Orientação a Objetos

Questões 7 a 20: Conhecimento sobre conceitos específicos

O Gráfico 13 apresenta as respostas das questões 8, 9, 12 e 13. Observa-se que conceitos específicos do assunto Herança, como Classe final, Classe abstrata e Sobreposição não são compreendidos pela maioria dos alunos.

Com relação ao conceito de Método estático, 3 alunos compreendem e 2 não.

Para as demais questões, houve unanimidade nas respostas, sendo de conhecimento de todos os participantes os conceitos de Classe, Método, Parâmetro, Objeto, Herança, Constante, Variável, Valores inteiros, Valores lógicos e Diagrama de Classes.

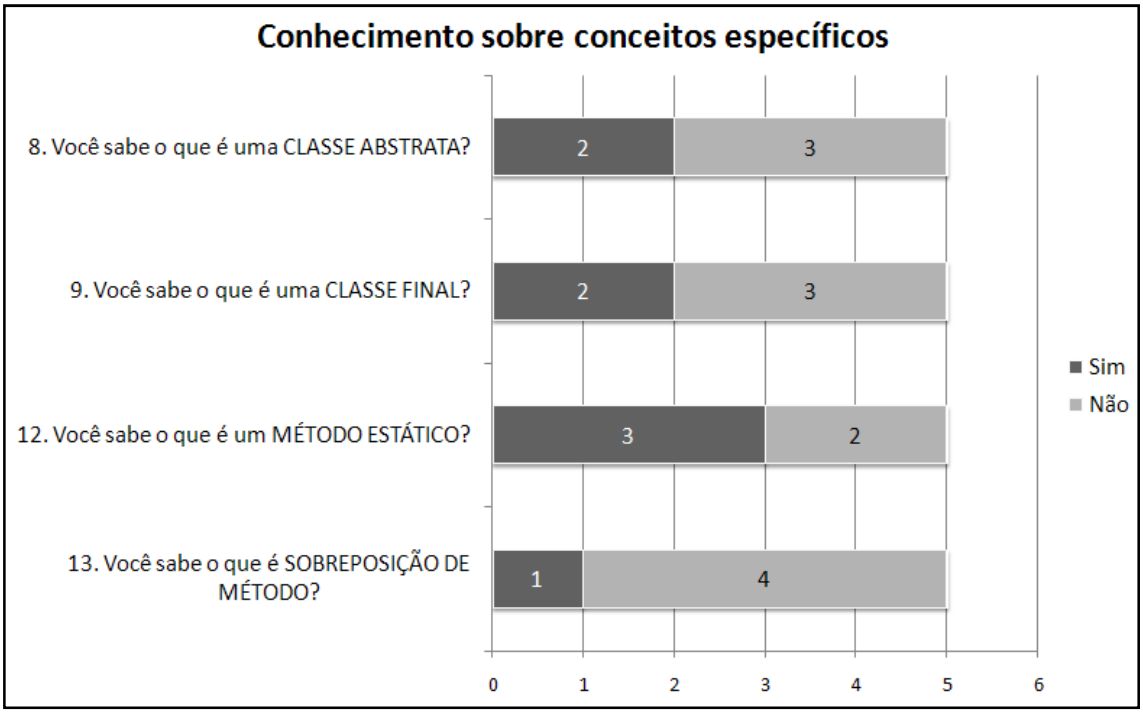


Gráfico 13: Conhecimentos sobre conceitos específicos

4.3.2 Resultados do questionário 2: avaliação da experiência.

Questão 1: Na sua opinião, a ambientação do jogo (Batalha de Robôs) é um tema atrativo e auxilia na compreensão dos conceitos?

Todos os participantes responderam afirmativamente a esta questão. Além disso, os seguintes comentários foram feitos pelos alunos:

"Gostei, pois junta conceitos de POO com a estratégia de jogos como Magic".

"Por que é um jogo divertido, parecido com outros jogos, porém com conceitos de POO."

"É interessante, mas podia pensar em algo que se aproximasse mais dos programas reais tipo cadastro, etc."

Questão 2: Com relação às diferentes possibilidades de jogos utilizando as cartas, você considera que:

Nesta questão, 3 alunos responderam que os diferentes jogos ajudam a compreender os

conceitos pouco a pouco, enquanto 2 pessoas responderam que preferem apenas um único modo de joga para evitar confusões quanto às diferentes regras.

Os seguintes comentários foram recebidos.

Sobre haver diversos modos de jogo: *"Por que fica mais fácil ir aprendendo os conceitos pouco a pouco."*

Sobre haver um único modo de jogo: *"Quando vai programar precisamos ter em mente todos os conceitos de POO e não apenas um em específico."*

Questão 3: Com relação à facilidade de uso:

Todos os participantes responderam que é um jogo simples de ser jogado e é possível aprender sozinho, apenas lendo as regras.

Os seguintes comentários foram apresentados:

"É simples, principalmente se você souber o básico de programação."

"Não são tantas regras e nem são complexas."

Questão 4: Você considera que as implementações de código fonte utilizados para descrever o funcionamento das cartas auxiliam a entender os conceitos explicados?

Todos os participantes responderam que escrever código fonte para refletir a ação das cartas auxilia na compreensão dos conceitos.

Os seguintes comentários foram apontados:

"É um pouco confuso a questão de retorno de valores nas funções."

"Por que já visualiza o modelo do código que vai ser utilizado."

"Ajudam, mas ainda acho meio complicado a questão de passar valor de um lado pra outro."

Questão 5: Você considera que a construção de diagramas a partir das cartas auxiliam a entender os conceitos explicados?

Todos os participantes responderam afirmativamente a esta questão.

Os seguintes comentários foram apresentados:

"Representa bem a ideia de herança, atributos e métodos presentes nas classes."

"Por que mostra a ideia de herança, método, classes, atributos."

Questão 6: Classifique o seu conhecimento prévio em programação orientada a objetos

Esta é a primeira que aparece no questionário 1: Conhecimentos Prévios e no questionário 2: Avaliação da Experiência.

No tópico 4.3.1, o Gráfico 12, pág. 95, mostrava que 2 alunos que declaravam ter dificuldades em compreender os conceitos. No Gráfico 14, observa-se que esses dois alunos elevaram sua compreensão dos conceitos de orientação a objetos, apesar de ainda terem dificuldades em programar.

O seguinte comentário foi realizado sobre essa questão: *"Entendi melhor alguns conceitos, como a herança, mas ainda preciso aprender mais."*

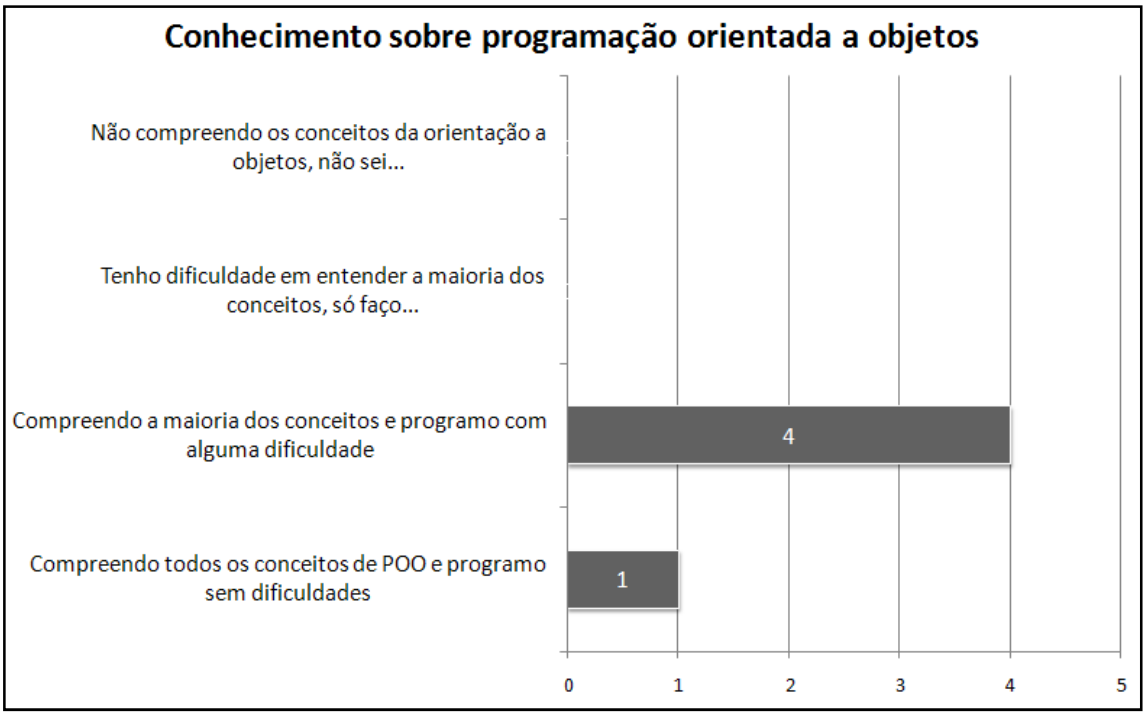


Gráfico 14: Conhecimento sobre P.O.O. após a experiência

Questões 7 a 20: Conhecimento sobre conceitos específicos

As questões 8 e 9 sobre os conceitos "Classe Abstrata" e "Classe Final", respectivamente, tiveram 100% de respostas afirmativas após a experiência, fazendo com que os 3 alunos que desconheciam estes assuntos pudessem compreende-los.

A questão 12, sobre "Método Estático" ganhou mais um ponto em relação aos dados apresentados no Gráfico 13, pág. 96, permanecendo apenas 1 aluno com dúvida sobre este assunto.

A questão 13, relacionada à "Sobreposição de métodos" obteve uma melhoria de 3

pontos, invertendo a situação anterior onde apenas 1 aluno compreendia este conceito. Após a experiência, apenas 1 de 5 alunos ainda tinha dúvidas sobre o assunto.

Os resultados destas quatro questões são representados no Gráfico 15.

Os demais itens não sofreram alteração, continuando com respostas afirmativas de todos os alunos em todos os assuntos.

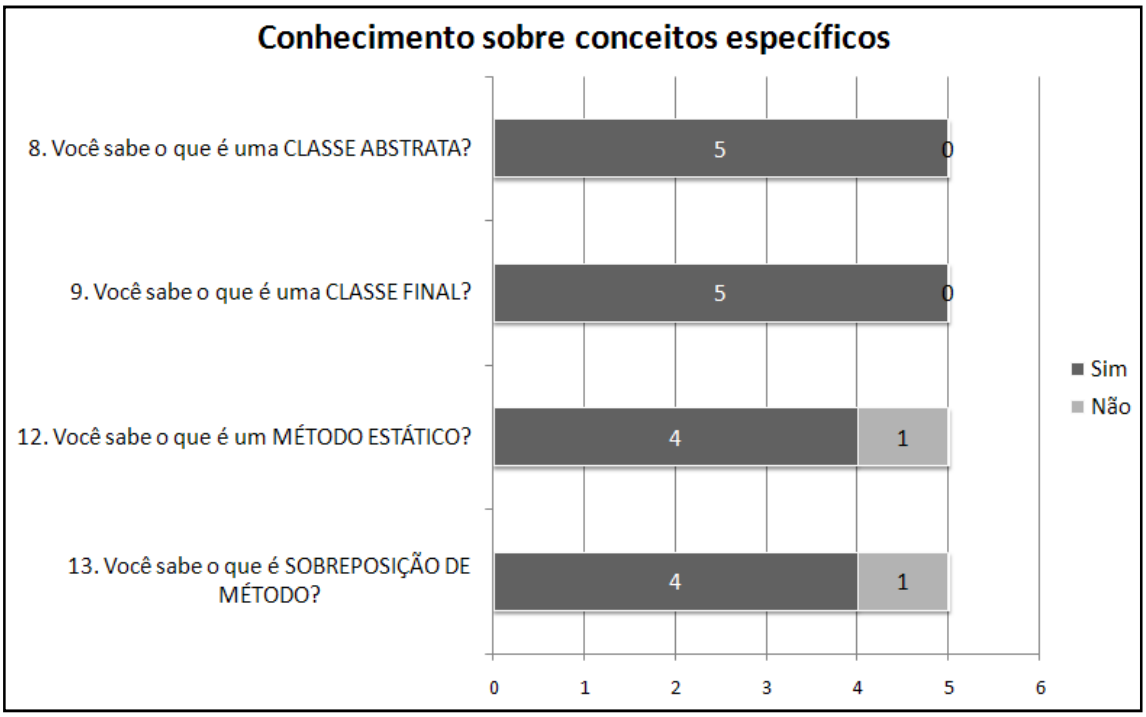


Gráfico 15: Conhecimentos específicos após a experiência

4.4 SUGESTÕES E COMENTÁRIOS DOS PARTICIPANTES

No questionário 2: Avaliação da Experiência, a questão número 21 solicita ao participante que aponte sugestões para a melhoria do método e dos jogos. A seguir são relacionadas algumas das sugestões mais relevantes.

"Um manual de instruções básico, só pra noção de POO, para quem nunca estudou a disciplina."

"Apenas ajustar no número de cartas do baralho, pois demora a apareceras cartas que a gente precisa."

“Ter um limite de ações por turno. Objetos só atacar no segundo turno. Adicionar cartas que permite realizar mais ações na rodada em que for usada essa carta.”

“Colocar mais cartas de classe.”

“Podia ter uma versão pra jogar no computador.”

“Tentar outras linguagens de programação. Acho Java muito difícil.”

“Algumas vezes quanto estamos jogando a gente fica um monte de tempo sem fazer nada, só puxando carta e passando a vez de um pra outro. É bom pensar num jeito de ter mais ação.”

“Mais opções de diagramas.”

Outros alunos não registraram sugestões para melhorar o jogo, mas deixaram sua opinião sobre a experiência e sobre o jogo em si. Abaixo estão tais relatos.

“Esse jogo é muito bom. Tem que ter cuidado para não viciar.”

“Foi bom ter usado esse jogo na aula, deu para aprender mais coisas que só com explicações teóricas.”

“Está ótimo assim.”

O Capítulo 5 trata das conclusões, contribuições e trabalhos futuros.

CAPÍTULO 5

5 CONCLUSÕES

Ao analisar separadamente os dados de cada turma, observa-se que há alunos com deficiências similares nos conceitos de orientação a objetos, principalmente com relação aos tópicos que envolvem Herança, como sobreposição de métodos, classes abstrata e finais.

Os conceitos de Classe, Objetos e Métodos são, no geral, bem compreendidos, assim como aqueles conceitos que não são exclusividade da orientação a objetos, como variáveis, constantes, valores inteiros e lógicos.

Alguns alunos que a princípio relataram terem dúvidas quanto a um destes assuntos, simplesmente não estavam associando a teoria à prática, pois, após explicar o conceito, era comum ouvir comentários como: *"ah, isso aí eu sei"*, *"eu já uso isso, só não sabia que se chamava assim"*, etc.

É notável a dificuldade que muitos possuem em programar. Mesmo compreendendo os conceitos não conseguem desenvolver código que solucione os problemas propostos.

Em conversas informais, alguns alunos afirmam que não imaginavam que teriam tamanha dificuldade nas disciplinas de programação e que esta deficiência faz com que não tenham o desejo de fazer um curso superior na área.

Observou-se que para motivar o aluno é importante que o professor se utilize de artifícios e metodologias diferentes. De acordo com as respostas dos questionários propostos, praticamente todos os alunos possuem o desejo de participar de aulas que utilize jogos como auxílio ao aprendizado, principalmente se o jogo facilitar a compreensão dos assuntos e não apenas aplique o conhecimento prévio, como acontece nos jogos de perguntas e respostas.

No geral, a metodologia e os jogos apresentados neste trabalho foram bem aceitos pelos alunos, que elogiaram, deram sugestões de melhorias e se dispuseram a divulgá-los entre os colegas estudantes.

A metodologia mostrou-se eficiente ao ensinar os assuntos sobre Herança, Classe Final e Classe Abstrata e apresentou bons resultados nos assuntos relacionados aos métodos, tendo reduções de dúvidas com índices superiores a 50%, em alguns casos.

Com relação ao uso de diagramas, mesmo a maioria dos alunos já tendo o

conhecimento de como desenhá-los e lê-los, as atividades propostas foram bem recebidas e conseguiram desenvolver o conhecimento dos poucos que ainda tinham dificuldades.

A construção de código fonte a partir das ações realizadas pelas cartas também foi bem recebida, apesar de ter contribuído para a compreensão dos conceitos apresentados, não foi possível verificar com maiores detalhes se o aluno conseguiria construir código em outras situações.

Durante a análise dos dados observou-se que algumas perguntas dos questionários poderiam ter sido escritas de outra forma e outras poderiam ter sido acrescentadas. Por exemplo: as alternativas do 6º item do questionário, sobre o conhecimento de orientação a objetos, deveriam omitir o nível de conhecimento de codificação em linguagem de programação, pois alguns alunos sentiram-se inibidos em mudar sua resposta para um nível de conhecimento maior após a experiência uma vez que a alternativa abordava também o desenvolvimento de programas.

Algumas perguntas poderiam ter sido acrescentadas aos questionários, por exemplo: “Você sabe o que é um método construtor?”, “Você sabe o que é sobrecarga de método?”, “Você sabe o que é um método com retorno?”, etc.

Verificou-se também que se faz necessária maior atenção quanto aos assuntos “Métodos Estáticos” e “Sobreposição de Métodos”, pois mesmo após a aplicação da metodologia, ainda obteve-se índices altos de alunos com dúvidas nestes conceitos.

Conclui-se, então, que a metodologia proposta neste trabalho facilita a compreensão dos conceitos de orientação a objetos, mas não garante que o aluno se tornará um bom programador, pois isso exige dedicação aos estudos e muita prática. E que os jogos, apesar de motivarem os estudos, não devem substituir os livros didáticos, pois eles ainda são a melhor fonte de pesquisa para o aluno.

5.1 CONTRIBUIÇÕES ACADÊMICAS

O presente trabalho oferece uma nova metodologia para o ensino dos conceitos de orientação a objetos a alunos que estão iniciando seus estudos. Também disponibiliza o conjunto de cartas do jogo “P.O.O. – O Jogo”, bem como as regras que refletem os conceitos de orientação a objetos e uma variedade de jogos para introduzir gradualmente tais conceitos. Espera-se que os jogos sejam utilizados por professores de cursos técnicos, de graduação e

pós-graduação como ferramenta didática.

5.2 RECOMENDAÇÕES PARA TRABALHOS FUTUROS

As cartas e jogos desenvolvidos durante este trabalho abordam conceitos iniciais da orientação a objetos, mas podem ser expandidas a fim de contemplar conceitos mais avançados ou aprofundar o nível de detalhes dos conceitos já apresentados.

Entre os assuntos a serem abordados em trabalhos futuros estão: polimorfismo, agregação, pacotes, modificadores de acesso *protected*, *private*, *public*, padrões de projeto, etc.

Pretende-se também realizar experiências com alunos de cursos de graduação e pós-graduação em computação, além de alunos de outras áreas que também estudam disciplinas de programação.

Pretende-se realizar a aplicação da metodologia durante as aulas de um semestre letivo, a fim de verificar com maior precisão a eficácia do método, não se limitando a breves sessões de teste.

Aplicar algumas das sugestões apontadas pelos alunos e desenvolver uma versão eletrônica do jogo a fim de facilitar a distribuição em formato digital e adicionar elementos interativos como animações, sons, etc.

Por fim, pretende-se desenvolver novos temas de ambientação além do atual “Batalha de Robôs”, diversificando os exemplos de código que podem ser criados a partir das cartas.

REFERÊNCIAS

ARNOLD, K.; GOSLING, J.; HOLMES, D. **A linguagem de prograjação Java**. 4ª. ed. Porto Alegre: Bookman, 2007.

BARNES, D. J.; KÖLLING, M. **Programação orientada a objetos com Java:** uma introdução prática usando o BlueJ. 4ª. ed. São Paulo: Pearson, 2008. ISBN 978-85-7605-187-9.

BAUM, L. F. **O Mágico de OZ**. 1900.

BELL, T.; WITTEN, I. H.; FELLOWS, M. **Computer Science Unplugged:** The Original Activities Book. 1ª Edição. ed. Computer Science Unplugged, 2005.

BÖRSTLER, J.; SCHULTE, C. **TEACHING OBJECT ORIENTED MODELLING WITH CRC-CARDS AND ROLEPLAYING GAMES**. WCCE. 2005.

BRATHWAITE, B.; SCHREIBER, I. **Challenges for game designers**. 1ª Edição. ed. Course Technology, 2009.

CANTO, A. R.; ZACARIAS, M. A. Utilização do jogo Super Trunfo Árvores Brasileiras como instrumento facilitador no ensino dos biomas brasileiros. **Ciências & Cognição**, Rio de Janeiro, 14, n. 1, Março 2009.

CARD GAME FORGE. Card Game Forge. **CardGameForge.com**, 2011. Disponível em: <http://www.cardgameforge.com>. Acesso em: 31 Dezembro 2011.

COMPLUBOT. Learning programming playing cards. **Arduino Blog**, 2011. Disponível em: <http://arduino.cc/blog/2011/02/14/learn-programming-playing-cards/>. Acesso em: 10 Setembro 2011.

COPAG. COPAG, 2011. Disponível em: <http://www.copag.com.br>. Acesso em: 29 Dezembro 2011.

CORREIA, C. H.; DOMINGUES, M. J. C. D. S. **Práticas inovadoras de ensino:** uso de brinquedos no ensino de algoritmos. Simpósio Brasileiro de Informática na Educação. 2010.

DAVID-MARSHALL, B.; VAN DREUNEN, J.; WANG, M. **Trading Card Game Industry From the T to the C to the G**. SuperData Research, Inc. New York. 2010.

DEVIR. Devir: Magic: The Gathering. **Devir.com.br**, 2011. Disponível em: <http://www.devir.com.br/magic/>. Acesso em: 29 Dezembro 2011.

ESTRELA, 2011. Disponível em: <www.estrela.com.br>. Acesso em: 29 Dezembro 2011.

GAMESPY. State of the Game Interview: Magic Online. **GameSpy.com**, 2007. Disponível em: <http://www.gamespy.com/articles/818/818114p1.html>. Acesso em: 30 Dezembro 2011.

GODOI, T. A. D. F.; OLIVEIRA, H. P. M.; CODOGNOTO, L. Tabela Periódica - Um super trunfo para alunos do ensino fundamental e médio. **Química Nova Escola**, 32, n. 1, Fevereiro 2009. 22-25.

GROW. http: //www.grow.com.br, 2011. Acesso em: 29 Dezembro 2011.

HANSEN, S. **The Game of SetR – An Ideal Example for Introducing Polymorphism and Design Patterns**. SIGCSE. Norfolk: 2004. p. 110-114.

HASBRO, 2011. Disponivel em: <http://www.hasbro.com/pt_BR>. Acesso em: 29 Dezembro 2011.

ICV2. 'Magic' Doubled Since 2008. **ICV2.com**, 2011. Disponível em: <http://www.icv2.com/articles/news/21471.html>. Acesso em: 30 Dezembro 2011.

IGN. Magic: The Gathering - Duels of the Planeswalkers 2012. **IGN.com**, 2011. Disponível em: <http://media.xbox360.ign.com/media/102/102879/imgs_1.html>. Acesso em: 30 Dezembro 2011.

ISTOÉ DINHEIRO. IstoÉ Dinheiro. **Rei do Baralho**, 2011. Disponível em: <http://www.istoedinheiro.com.br/noticias/58458_REI+DO+BARALHO>. Acesso em: 29 Dezembro 2011.

JANDL JUNIOR, P. **Java:** guia do programador. São Paulo: Novatec, 2007. ISBN 978-85-7522-109-9.

JANN, P. N.; LEITE, M. D. F. Jogo do DNA: um instrumento pedagógico para o ensino de ciência e biologia. **Ciências & Cognição**, 15, n. 1, 20 Abril 2010. Acesso em: 05 Janeiro 2012.

JOYSTIQ. Magic: The Gathering taps XBLA for 500K sales. **Joystiq.com**, 2011. Disponível em: <http://www.joystiq.com/2011/01/28/magic-the-gathering-taps-xbla-for-500k-sales/>. Acesso em: 30 Dezembro 2011.

KIM, S. et al. **Smalltalk Card Game for Learning Object-Oriented Thinking in an Evolutionary Way**. OOPSLA. Portland: 2006. p. 683-684.

KÖLLING, M. BlueJ, 2011. Disponível em: <www.bluej.org>. Acesso em: 2011 Dezembro 15.

KONAMI. Yu-Gi-Oh! Card, 2011. Disponível em: <http://www.yugioh-card.com>. Acesso em: 05 Agosto 2011.

LIM, D. **Taking Students Out for a Ride:** Using a Board Game to Teach Graph Theory. SIGCSE. Covington: 2007.

MATTEL, 2011. Disponível em: <http://mattelbrasil.com.br>. Acesso em: 29 Dezembro 2011.

MORENO, C. E. L. **Trunfo Químico**. Rio de Janeiro: FAPERJ, 2010.

ORION'S BELL. OZ The Card Game of OZ. **OZ The Card Game of OZ**, 2011. Disponível em: <http://orionsbell.com/>. Acesso em: 31 Dezembro 2011.

PEREIRA, R. F.; FUSINATO, P. A.; NEVES, M. C. D. **DESENVOLVENDO JOGOS EDUCATIVOS PARA O ENSINO DE FÍSICA:** UM MATERIAL DIDÁTICO ALTERNATIVO DE APOIO AO BINÔMIO ENSINO-APRENDIZAGEM. XIX Simpósio Nacional de Ensino de Física. Manaus: 2011.

PÍCCOLO, H. L. et al. **Ambiente Interativo e Adaptável para ensino de Programação**. 2010.

REBOUÇAS, A. D. D. S. et al. **Aprendendo a Ensinar Programação Combinando Jogos e Python**. 2010.

RESNICK, M. et al. Scratch, 2009. Disponível em: <http://scratch.mit.edu>.

ROCHA, I. F.; FIGUEIREDO, A. M. T. A. Trunfo Químico: Uma Atividade Lúdica na Aprendizagem da Tabela Periódica. **49º Congresso Brasileiro de Química**, Porto Alegre, 04 Outubro 2009.

SALES, C. G.; DANTAS, V. F. **ProGame:** um jogo para o ensino de algoritmos e programação. 2010.

SANTOS, R. **Introdução à programação orientada a objetos usando Java**. Rio de Janeiro: Elsevier, 2003. ISBN 85-352-1206-X.

SIERRA, K.; BATES, B. **Use a Cabeça! Java**. Rio de Janeiro: Alta Books, 2007. ISBN 0-596-00920-8.

SPIEGEL, C. et al. Célula Adentro, 2001. Disponível em: <http://celulaadentro.ioc.fiocruz.br>. Acesso em: 05 Janeiro 2012.

THE POKÉMON COMPANY. Pokémon Trading Card Game, 2011. Disponível em: <www.pokemontcg.com>. Acesso em: 2011 Agosto 05.

TOYSTER, 2011. Disponível em: <http://www.toyster.com>. Acesso em: 29 Dezembro 2011.

VAN LAARHOVEN, T. Magic Set Editor, 2011. Disponível em: <http://magicseteditor.sourceforge.net>. Acesso em: 2011 Setembro 01.

VENTURA, B. Trunfo nas escolas. **Ciência Hoje**, 2010. Disponível em: <http://cienciahoje.uol.com.br/alo-professor/intervalo/trunfo-nas-escolas>. Acesso em: 05 Janeiro 2012.

WIZARDS OF THE COAST. Magic the Gathering, 2011. Disponível em: <http://www.wizards.com/magic/>. Acesso em: 05 Agosto 2011.

WTACTICS. WTactics.org. **WTactics.org**, 2011. Disponível em: <http://wtactics.org/>. Acesso em: 31 Dezembro 2011.

WULVEN GAME STUDIOS. Shadow Era. **Shadow Era**, 2011. Disponível em: <http://www.shadowera.com>. Acesso em: 29 Dezembro 2011.

ZANON, D. A. V.; GUERREIRO, M. A. D. S.; OLIVEIRA, R. C. D. Jogo didático Ludo Químico para o ensino de nomenclatura dos compostos orgânicos. **Ciências & Cognição**, v. 13, n. 1, Março 2008. ISSN 1806-5821.

ANEXOS

QUESTIONÁRIO 1 - CONHECIMENTOS PRÉVIOS

Instruções:

- Marque apenas UMA alternativa em cada questão.
- Escreva justificativas às suas respostas, se considerar necessário.

Qual o ano/período que você cursa?

( ) 1º ( ) 2º ( ) 3º ( ) 4º

Sobre jogos educativos em geral

1. Em sua opinião, o uso de jogos educativos contribui para o aprendizado?

( ) Sim ( ) Não Justificativa: _____

2. Durante as disciplinas que você cursou, outro professor utilizou jogos como método de ensino?

( ) Não, nenhuma vez. ( ) Sim, apenas uma vez. ( ) Sim, até 3 vezes.

( ) Sim, 4 ou mais vezes. ( ) Não lembro

Se sim, qual o assunto/disciplina: _____

3. Você tem interesse em conhecer outros jogos educativos que auxiliem em seus estudos?

( ) Sim ( ) Não Justificativa: _____

4. Em relação ao formato em que jogo é apresentado, qual o seu tipo favorito?

( ) Tabuleiro ( ) Cartas ( ) Jogo eletrônico ( ) Papel e caneta ( ) Outro

Justificativa: _____

5. Qual o principal elemento que você busca em um jogo educativo?

( ) Diversão ( ) Competição ( ) Cooperação

( ) Aplicar a teoria previamente conhecida ( ) Aprender a teoria enquanto se joga

Justificativa: _____

Sobre o conhecimento em Programação Orientada a Objetos

6. Classifique o seu conhecimento prévio em programação orientada a objetos:

( ) Não compreendo os conceitos da orientação a objetos, não sei programar.

( ) Tenho dificuldade em entender a maioria dos conceitos, só faço programas simples.

( ) Compreendo a maioria dos conceitos e programo com alguma dificuldade.

( ) Compreendo todos os conceitos de POO e programo sem dificuldades.

( ) Outro: _____

7. Você sabe o que é uma classe?	( ) Sim	( ) Não
8. Você sabe o que é uma classe abstrata?	( ) Sim	( ) Não
9. Você sabe o que é uma classe final?	( ) Sim	( ) Não
10. Você sabe o que é um método?	( ) Sim	( ) Não
11. Você sabe o que é um parâmetro em um método?	( ) Sim	( ) Não
12. Você sabe o que é um método estático?	( ) Sim	( ) Não
13. Você sabe o que é sobreposição de método?	( ) Sim	( ) Não
14. Você sabe o que é um objeto?	( ) Sim	( ) Não
15. Você sabe o que é herança?	( ) Sim	( ) Não
16. Você sabe o que é uma constante?	( ) Sim	( ) Não
17. Você sabe o que é uma variável?	( ) Sim	( ) Não
18. Você sabe o que é um valor inteiro?	( ) Sim	( ) Não
19. Você sabe o que é um valor lógico?	( ) Sim	( ) Não
20. Você sabe ler e desenhar diagramas de classes?	( ) Sim	( ) Não

QUESTIONÁRIO 2 – AVALIAÇÃO DA EXPERIÊNCIA

Instruções:

- Marque apenas UMA alternativa em cada questão.
- Escreva justificativas às suas respostas, se considerar necessário.

Qual o ano/período que você cursa?

( ) 1º ( ) 2º ( ) 3º ( ) 4º

Sobre o jogo proposto neste trabalho

1. Na sua opinião, a ambientação do jogo (Batalha de Robôs) é um tema atrativo e auxilia na compreensão dos conceitos?

( ) Sim ( ) Não Justificativa: _____

2. Com relação às diferentes possibilidades de jogos utilizando as cartas, você considera que:

( ) Os diferentes jogos ajudam a compreender os conceitos pouco a pouco.

( ) Deveria haver um único modo de jogo, pois é confuso usar tantas regras diferentes.

( ) Outro Justificativa: _____

3. Com relação à facilidade de uso:

( ) É um jogo simples de ser jogado, e é possível aprender sozinho, apenas lendo as regras.

( ) É um jogo simples de ser jogado, mas só é possível aprender com ajuda de alguém que já saiba jogar.

( ) É um jogo difícil de ser jogado, são muitas regras a serem aprendidas.

Justificativa: _____

4. Você considera que as implementações de código fonte utilizados para descrever o funcionamento das cartas auxiliam a entender os conceitos explicados?

( ) Sim ( ) Não Justificativa: _____

5. Você considera que a construção de diagramas a partir das cartas auxiliam a entender os conceitos explicados?

( ) Sim ( ) Não Justificativa: _____

Sobre o conhecimento em Programação Orientada a Objetos após o uso do jogo

6. Classifique o seu conhecimento em programação orientada a objetos:

( ) Não compreendo os conceitos da orientação a objetos, não sei programar.

( ) Tenho dificuldade em entender a maioria dos conceitos, só faço programas simples.

( ) Compreendo a maioria dos conceitos e programo com alguma dificuldade.

( ) Compreendo todos os conceitos de POO e programo sem dificuldades.

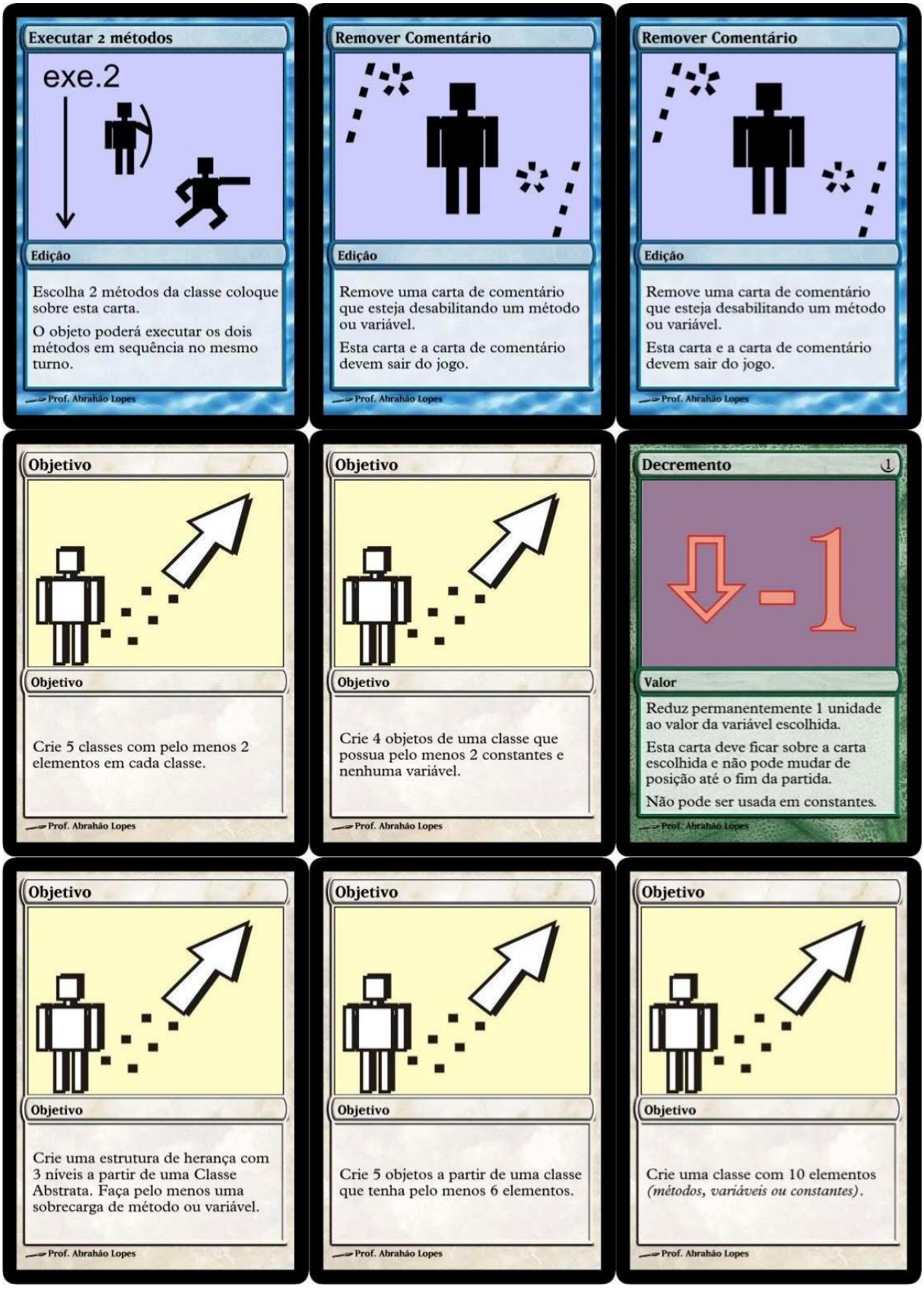
( ) Outro: _____

Após a utilização do jogo proposto neste trabalho,

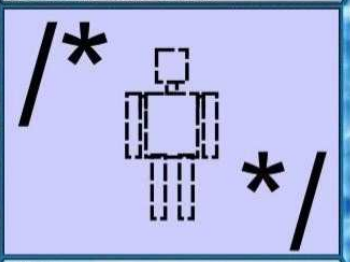
7. Você sabe o que é uma classe?	( ) Sim	( ) Não
8. Você sabe o que é uma classe abstrata?	( ) Sim	( ) Não
9. Você sabe o que é uma classe final?	( ) Sim	( ) Não
10. Você sabe o que é um método?	( ) Sim	( ) Não
11. Você sabe o que é um parâmetro em um método?	( ) Sim	( ) Não
12. Você sabe o que é um método estático?	( ) Sim	( ) Não
13. Você sabe o que é sobreposição de método?	( ) Sim	( ) Não
14. Você sabe o que é um objeto?	( ) Sim	( ) Não
15. Você sabe o que é herança?	( ) Sim	( ) Não
16. Você sabe o que é uma constante?	( ) Sim	( ) Não
17. Você sabe o que é uma variável?	( ) Sim	( ) Não
18. Você sabe o que é um valor inteiro?	( ) Sim	( ) Não
19. Você sabe o que é um valor lógico?	( ) Sim	( ) Não
20. Você sabe ler e desenhar diagramas de classes?	( ) Sim	( ) Não

21. Que modificações você sugere para melhorar o jogo?

CARTAS DE “P.O.O – O JOGO” PARA IMPRIMIR E JOGAR



Adicionar Comentário



Edição

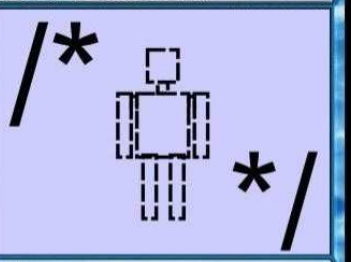
Desabilita um trecho de código escolhido pelo jogador.

Neste jogo, esta carta pode ser usada apenas em métodos e variáveis.

Ponha esta carta sobre a carta que será desabilitada.

Prof. Abrahão Lopes

Adicionar Comentário



Edição

Desabilita um trecho de código escolhido pelo jogador.

Neste jogo, esta carta pode ser usada apenas em métodos e variáveis.

Ponha esta carta sobre a carta que será desabilitada.

Prof. Abrahão Lopes

Executar 2 métodos

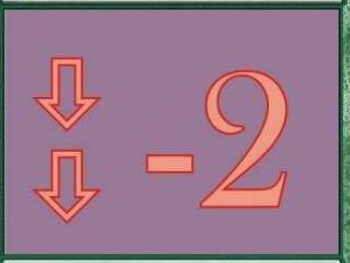
Edição

Escolha 2 métodos da classe coloque sobre esta carta.

O objeto poderá executar os dois métodos em sequência no mesmo turno.

Prof. Abrahão Lopes

Decremento 2



Valor

Reduz permanentemente 1 unidade ao valor da variável escolhida.

Esta carta deve ficar sobre a carta escolhida e não pode mudar de posição até o fim da partida.

Não pode ser usada em constantes.

Prof. Abrahão Lopes

Incremento 2

Valor

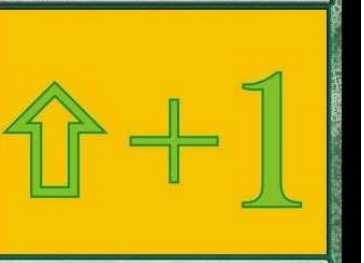
Aumenta permanentemente 2 unidade ao valor da variável escolhida.

Esta carta deve ficar sobre a carta escolhida e não pode mudar de posição até o fim da partida.

Não pode ser usada em constantes.

Prof. Abrahão Lopes

Incremento 1



Valor

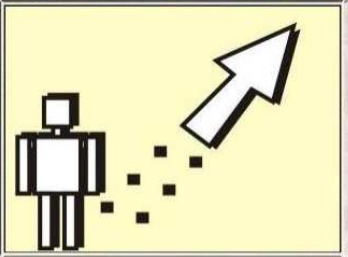
Aumenta permanentemente 1 unidade ao valor da variável escolhida.

Esta carta deve ficar sobre a carta escolhida e não pode mudar de posição até o fim da partida.

Não pode ser usada em constantes.

Prof. Abrahão Lopes

Objetivo

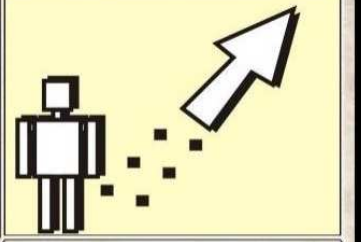


Objetivo

Crie uma estrutura de herança com 2 níveis. Cada nível deve ter pelo menos 2 elementos em cada classe.

Prof. Abrahão Lopes

Objetivo

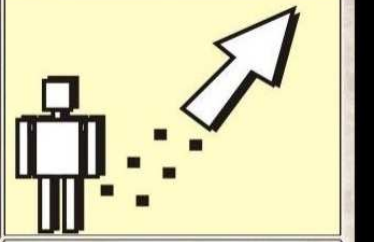


Objetivo

Crie uma estrutura de herança com 3 níveis. Cada nível deve ter pelo menos 3 elementos em cada classe.

Prof. Abrahão Lopes

Objetivo

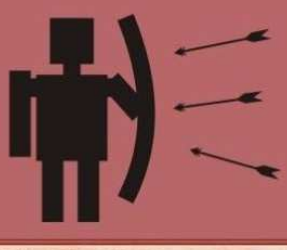


Objetivo

Crie 5 objetos do tipo robô, e 5 objetos do tipo arma.

Prof. Abrahão Lopes

Defender (Ataque)

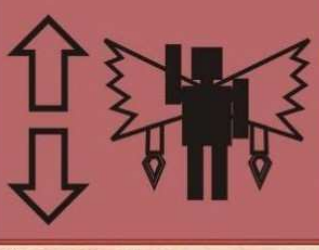


Método — sem retorno

Bloqueia ESCUDO de dano no Ataque recebido.

Prof. Abrahão Lopes

Voar ( )



Método — sem retorno

Esta carta só pode ser usada se a classe possuir a variável Vôo.

Gire a carta de objeto para a posição horizontal para ativar o vôo.

Retorne a carta de objeto à posição vertical para desativar o vôo.

Prof. Abrahão Lopes

Vôo

Variável — lógico

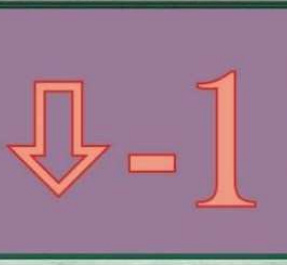
Concede a habilidade de voar.

O vôo deve ser ativado através de uma carta de método.

Gire a carta de objeto quando estiver voando, retorne à posição normal quando estiver em terra.

Prof. Abrahão Lopes

Decremento 1



Valor

Reduz permanentemente 1 unidade ao valor da variável escolhida.

Esta carta deve ficar sobre a carta escolhida e não pode mudar de posição até o fim da partida.

Não pode ser usada em constantes.

Prof. Abrahão Lopes

Decremento 2

Valor

Reduz permanentemente 1 unidade ao valor da variável escolhida.

Esta carta deve ficar sobre a carta escolhida e não pode mudar de posição até o fim da partida.

Não pode ser usada em constantes.

Prof. Abrahão Lopes

Incremento 2

Valor

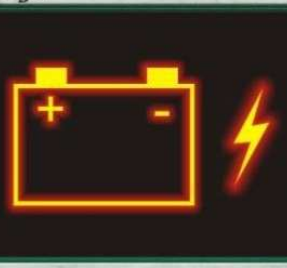
Aumenta permanentemente 2 unidade ao valor da variável escolhida.

Esta carta deve ficar sobre a carta escolhida e não pode mudar de posição até o fim da partida.

Não pode ser usada em constantes.

Prof. Abrahão Lopes

Energia 2

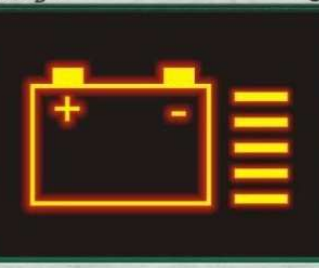


Variável — inteiro

Resiste a 2 pontos de dano.

Prof. Abrahão Lopes

Energia 1



Variável — inteiro

Resiste a 1 ponto de dano.

Prof. Abrahão Lopes

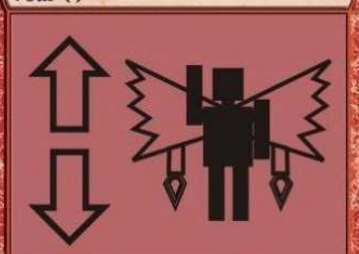
Escudo 2

Constante — inteiro

Resiste a 2 pontos de dano.

Prof. Abrahão Lopes

Voar ( )



Método — sem retorno

Esta carta só pode ser usada se a classe possuir a variável Vôo.

Gire a carta de objeto para a posição horizontal para ativar o vôo.

Retorne a carta de objeto à posição vertical para desativar o vôo.

Prof. Abrahão Lopes

Vôo

Variável — lógico

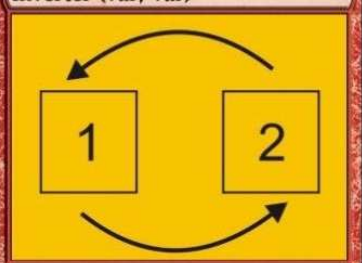
Concede a habilidade de voar.

O vôo deve ser ativado através de uma carta de método.

Gire a carta de objeto quando estiver voando, retorne à posição normal quando estiver em terra.

Prof. Abrahão Lopes

Inverter (Var, Var)



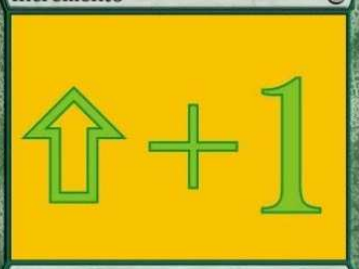
Método — sem retorno

Permite tocar o valor de duas variáveis do objeto atual até o final do turno.

Ex. Força 1, Energia 2, após usar esta carta: Força 2, Energia 1,

Prof. Abrahão Lopes

Incremento



Valor

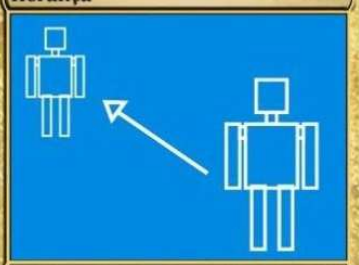
Aumenta permanentemente 1 unidade ao valor da variável escolhida.

Esta carta deve ficar sobre a carta escolhida e não pode mudar de posição até o fim da partida.

Não pode ser usada em constantes.

Prof. Abrahão Lopes

Herança



Classe — extensão

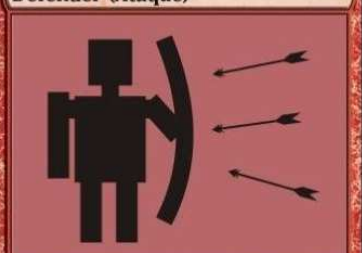
Copia variáveis e métodos de uma superclasse para uma subclasse.

A subclasse passa a ter todas as características da superclasse além das que forem definidas nela.

Métodos e variáveis com mesmo nome e parâmetros na subclasse sobrepõem os da superclasse.

Prof. Abrahão Lopes

Defender (Ataque)



Método — sem retorno

Bloqueia ESCUDO de dano no Ataque recebido.

Prof. Abrahão Lopes

Escudo

Variável — inteiro

Resiste a 1 ponto de dano.

Prof. Abrahão Lopes

Escudo

Variável — inteiro

Resiste a 2 pontos de dano.

Prof. Abrahão Lopes

Escudo

Constante — inteiro

Resiste a 1 ponto de dano.

Prof. Abrahão Lopes

Refletir (Ataque)
Método — retorno inteiro
Se durante a defesa o objeto receber um Ataque maior que ENERGIA + ESCUDO, a diferença é devolvida ao atacante.
Prof. Abrahão Lopes
UsarArma ( )
Método — sem retorno
Permite chamar o método de disparo de uma arma escolhida pelo jogador.
Prof. Abrahão Lopes
Voar ( )
Método — sem retorno
Esta carta só pode ser usada se a classe possuir a variável Vôo.
Gire a carta de objeto para a posição horizontal para ativar o vôo.
Retorne a carta de objeto à posição vertical para desativar o vôo.
Prof. Abrahão Lopes
Energia 2
Variável — inteiro
Resiste a 2 pontos de dano.
Prof. Abrahão Lopes
UsarArma ( )
Método — sem retorno
Permite chamar o método de disparo de uma arma escolhida pelo jogador.
Prof. Abrahão Lopes
Refletir (Ataque)
Método — retorno inteiro
Se durante a defesa o objeto receber um Ataque maior que ENERGIA + ESCUDO, a diferença é devolvida ao atacante.
Prof. Abrahão Lopes
Força 1
Variável — inteiro
Causa 1 ponto de dano no objeto ou fábrica alvo.
Prof. Abrahão Lopes
Força 1
Variável — inteiro
Causa 1 ponto de dano no objeto ou fábrica alvo.
Prof. Abrahão Lopes
Energia 1
Variável — inteiro
Resiste a 1 ponto de dano.
Prof. Abrahão Lopes

Vôo
Variável — lógico
Concede a habilidade de voar.
O vôo deve ser ativado através de uma carta de método.
Gire a carta de objeto quando estiver voando, retorne à posição normal quando estiver em terra.
Prof. Abrahão Lopes
Voar ( )
Método — sem retorno
Esta carta só pode ser usada se a classe possuir a variável Vôo.
Gire a carta de objeto para a posição horizontal para ativar o vôo.
Retorne a carta de objeto à posição vertical para desativar o vôo.
Prof. Abrahão Lopes
Vôo
Variável — lógico
Concede a habilidade de voar.
O vôo deve ser ativado através de uma carta de método.
Gire a carta de objeto quando estiver voando, retorne à posição normal quando estiver em terra.
Prof. Abrahão Lopes
Força 3
Constante — inteiro
Causa 3 pontos de dano no objeto ou fábrica alvo.
Prof. Abrahão Lopes
Força 4
4t
4t
Constante — inteiro
Causa 4 pontos de dano no objeto ou fábrica alvo.
Prof. Abrahão Lopes
Energia 2
Variável — inteiro
Resiste a 2 pontos de dano.
Prof. Abrahão Lopes
Força 2
Variável — inteiro
Causa 2 pontos de dano no objeto ou fábrica alvo.
Prof. Abrahão Lopes
Energia 1
Variável — inteiro
Resiste a 1 ponto de dano.
Prof. Abrahão Lopes
Atacar (Objeto)
Método — retorno inteiro
Causa FORÇA de dano no objeto alvo.
Prof. Abrahão Lopes

Energia
1
Variável — inteiro
Resiste a 1 ponto de dano.
Prof. Abrahão Lopes
Escudo
2
Variável — inteiro
Resiste a 2 pontos de dano.
Prof. Abrahão Lopes
Escudo
1
Constante — inteiro
Resiste a 1 ponto de dano.
Prof. Abrahão Lopes
Energia
2
Variável — inteiro
Resiste a 2 pontos de dano.
Prof. Abrahão Lopes
Escudo
2
Constante — inteiro
Resiste a 2 pontos de dano.
Prof. Abrahão Lopes
Escudo
1
Variável — inteiro
Resiste a 1 ponto de dano.
Prof. Abrahão Lopes
Atacar (Objeto, Objeto)
Método — retorno inteiro
Permite atacar até dois objetos numa única jogada.
Causa FORÇA de dano distribuído entre os objetos-alvo, conforme escolha do jogador atacante.
Prof. Abrahão Lopes
Atacar (Objeto, Objeto)
Método — retorno inteiro
Permite atacar até dois objetos numa única jogada.
Causa FORÇA de dano distribuído entre os objetos-alvo, conforme escolha do jogador atacante.
Prof. Abrahão Lopes
AgruparEAtacar (Aliado, Inimigo)
Método — retorno inteiro
Agrupa dois robôs do jogador atacante para atingir um único alvo adversário.
O dano causado é a soma da FORÇA dos dois atacantes.
O robô agrupado não joga neste turno.
Prof. Abrahão Lopes

Força 2
Variável — inteiro
Causa 2 pontos de dano no objeto ou fábrica alvo.
Prof. Abrahão Lopes
Força 3
Constante — inteiro
Causa 3 pontos de dano no objeto ou fábrica alvo.
Prof. Abrahão Lopes
Força 4
4t
4t
Constante — inteiro
Causa 4 pontos de dano no objeto ou fábrica alvo.
Prof. Abrahão Lopes
Herança
Classe — extensão
Copia variáveis e métodos de uma superclasse para uma subclasse.
A subclasse passa a ter todas as características da superclasse além das que forem definidas nela.
Métodos e variáveis com mesmo nome e parâmetros na subclasse sobrepõem os da superclasse.
Prof. Abrahão Lopes
Herança
Classe — extensão
Copia variáveis e métodos de uma superclasse para uma subclasse.
A subclasse passa a ter todas as características da superclasse além das que forem definidas nela.
Métodos e variáveis com mesmo nome e parâmetros na subclasse sobrepõem os da superclasse.
Prof. Abrahão Lopes
Inverter (Var, Var)
1
2
Método — sem retorno
Permite tocar o valor de duas variáveis do objeto atual até o final do turno.
Ex. Força 1, Energia 2, após usar esta carta: Força 2, Energia 1,
Prof. Abrahão Lopes
Explodir (Objeto)
Método
Remove do jogo o objeto atacante e o objeto alvo.
Prof. Abrahão Lopes
Explodir (Objeto)
Método
Remove do jogo o objeto atacante e o objeto alvo.
Prof. Abrahão Lopes
Inverter (Var, Var)
1
2
Método — sem retorno
Permite tocar o valor de duas variáveis do objeto atual até o final do turno.
Ex. Força 1, Energia 2, após usar esta carta: Força 2, Energia 1,
Prof. Abrahão Lopes

Atacar (Objeto)

Método — retorno inteiro

Causa FORÇA de dano no objeto alvo.

Prof. Abrahão Lopes

Atacar (Objeto)

Método — retorno inteiro

Gaste 2 pontos de energia para comprar 1 ponto extra de força.

Causa FORÇA + 1 de dano no objeto alvo.

Prof. Abrahão Lopes

Atacar (Objeto)

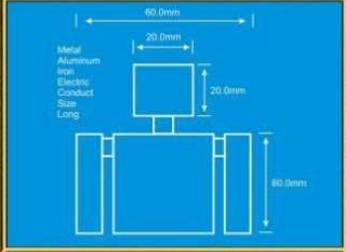
Método — retorno inteiro

Gaste 2 pontos de energia para comprar 1 ponto extra de força.

Causa FORÇA + 1 de dano no objeto alvo.

Prof. Abrahão Lopes

SuperClasse



Classe — abstrata

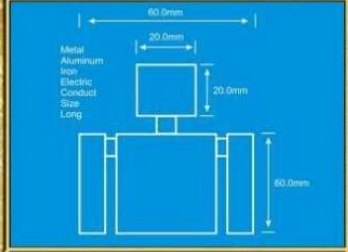
Não se pode criar objetos a partir desta classe. Deve ser usada apenas como superclasse em relações de herança.

Escudo = 1; Energia = 1;

Defender *(Ataque)*: Bloqueia Escudo de dano no Ataque recebido

Prof. Abrahão Lopes

SuperClasse



Classe — abstrata

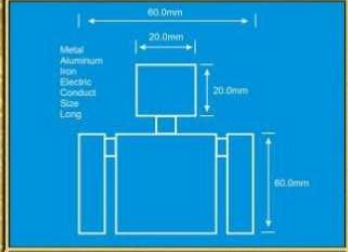
Não se pode criar objetos a partir desta classe. Deve ser usada apenas como superclasse em relações de herança.

Força = 1; Energia = 2;

Atacar *(Robô)*: Causa força de dano no objeto alvo

Prof. Abrahão Lopes

SuperClasse



Classe — abstrata

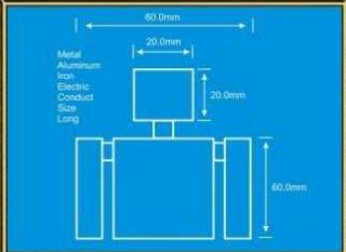
Não se pode criar objetos a partir desta classe. Deve ser usada apenas como superclasse em relações de herança.

Escudo = 2; Energia = 1;

Defender *(Ataque)*: Bloqueia Escudo de dano no Ataque recebido

Prof. Abrahão Lopes

Projeto



Classe — final

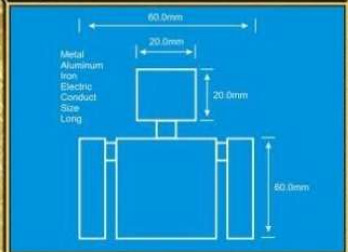
Descreve o funcionamento do robô.

Agrupa variáveis e métodos.

Não pode ser superclasse numa relação de herança.

Prof. Abrahão Lopes

Projeto



Classe — final

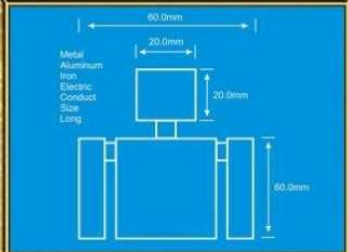
Descreve o funcionamento do robô.

Agrupa variáveis e métodos.

Não pode ser superclasse numa relação de herança.

Prof. Abrahão Lopes

Projeto



Classe — final

Descreve o funcionamento do robô.

Agrupa variáveis e métodos.

Não pode ser superclasse numa relação de herança.

Prof. Abrahão Lopes

DestruirArma (Arma)

Método — sem retorno

Destrói um OBJETO de arma alvo.

A arma só pode ser destruída se estiver em período de recarga.

Prof. Abrahão Lopes

DestruirArma (Arma)

Método — sem retorno

Destrói um OBJETO de arma alvo.

A arma só pode ser destruída se estiver em período de recarga.

Prof. Abrahão Lopes

AgruparEAtacar (Aliado, Inimigo)

Método — retorno inteiro

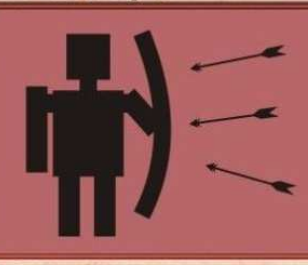
Agrupa dois robôs do jogador atacante para atingir um único alvo adversário.

O dano causado é a soma da FORÇA dos dois atacantes.

O robô agrupado não joga neste turno.

Prof. Abrahão Lopes

Defender (Ataque)

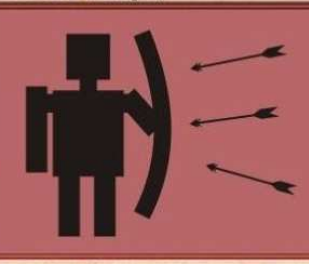


Método — sem retorno

Bloqueia ESCUDO de dano no Ataque recebido.

Prof. Abrahão Lopes

Defender (Ataque)

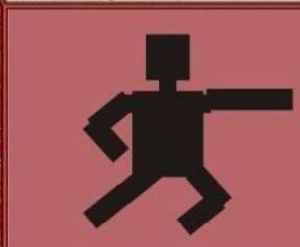


Método — sem retorno

Bloqueia ESCUDO de dano no Ataque recebido.

Prof. Abrahão Lopes

Atacar (Objeto)

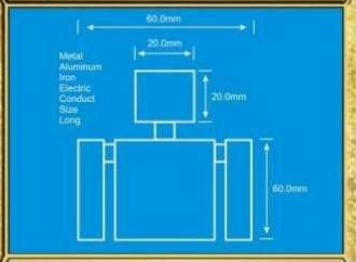


Método — retorno inteiro

Causa FORÇA de dano no objeto alvo.

Prof. Abrahão Lopes

Projeto



Classe — final

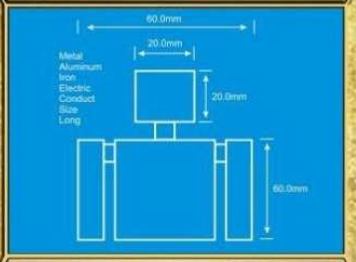
Descreve o funcionamento do robô.

Agrupa variáveis e métodos.

Não pode ser superclasse numa relação de herança.

Prof. Abrahão Lopes

SuperClasse



Classe — abstrata

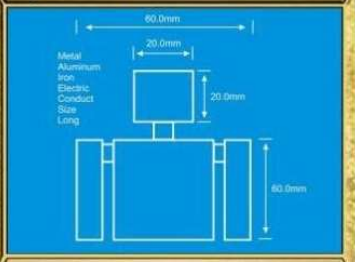
Não se pode criar objetos a partir desta classe. Deve ser usada apenas como superclasse em relações de herança.

Força = 2; Energia = 1;

Atacar *(Robô)*: Causa força de dano no objeto alvo

Prof. Abrahão Lopes

SuperClasse



Classe — abstrata

Não se pode criar objetos a partir desta classe. Deve ser usada apenas como superclasse em relações de herança.

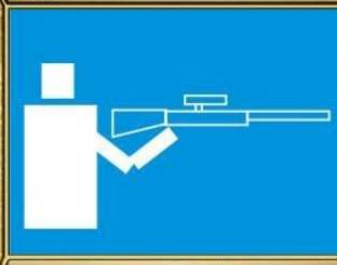
Vôo = falso; Força = 1;

Voar *( )*: Gire a carta de objeto para a posição horizontal para ativar o vôo.

Prof. Abrahão Lopes

Herança
Classe — extensão
Copia variáveis e métodos de uma superclasse para uma subclasse.
A subclasse passa a ter todas as características da superclasse além das que forem definidas nela.
Métodos e variáveis com mesmo nome e parâmetros na subclasse sobrepõem os da superclasse.
Prof. Abrahão Lopes
Herança
Classe — extensão
Copia variáveis e métodos de uma superclasse para uma subclasse.
A subclasse passa a ter todas as características da superclasse além das que forem definidas nela.
Métodos e variáveis com mesmo nome e parâmetros na subclasse sobrepõem os da superclasse.
Prof. Abrahão Lopes
Herança
Classe — extensão
Copia variáveis e métodos de uma superclasse para uma subclasse.
A subclasse passa a ter todas as características da superclasse além das que forem definidas nela.
Métodos e variáveis com mesmo nome e parâmetros na subclasse sobrepõem os da superclasse.
Prof. Abrahão Lopes
Projeto
60.0mm
20.0mm
Metal
Aluminum
Iron
Electric
Conduct
Size
Long
20.0mm
60.0mm
Classe
Descreve o funcionamento do robô.
Agrupa variáveis e métodos.
Prof. Abrahão Lopes
Projeto
60.0mm
20.0mm
Metal
Aluminum
Iron
Electric
Conduct
Size
Long
20.0mm
60.0mm
Classe
Descreve o funcionamento do robô.
Agrupa variáveis e métodos.
Prof. Abrahão Lopes
Projeto
60.0mm
20.0mm
Metal
Aluminum
Iron
Electric
Conduct
Size
Long
20.0mm
60.0mm
Classe
Descreve o funcionamento do robô.
Agrupa variáveis e métodos.
Prof. Abrahão Lopes
Arma
Objeto — arma
Possui as características definidas pela classe de arma atual.
Prof. Abrahão Lopes
Arma
Objeto — arma
Possui as características definidas pela classe de arma atual.
Prof. Abrahão Lopes
Arma
Objeto — arma
Possui as características definidas pela classe de arma atual.
Prof. Abrahão Lopes

Fuzil



Classe — com método estático

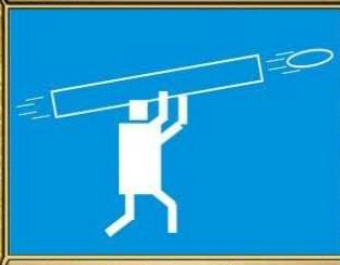
Não é necessário criar um objeto de Arma para usar o método desta classe.

O robô precisa ter a habilidade de usar armas para poder usar esta carta.

AtirarEm *(Objeto)* - Causa 4 pontos de dano no alvo. Demora 2 turnos para recarregar.

Prof. Abrahão Lopes — 4/2

Bazuca



Classe — com método estático

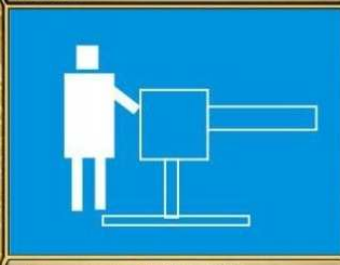
Não é necessário criar um objeto de Arma para usar o método desta classe.

O robô precisa ter a habilidade de usar armas para poder usar esta carta.

AtirarEm *(Objeto)* - Causa 5 pontos de dano no alvo. Demora 3 turnos para recarregar.

Prof. Abrahão Lopes — 5/3

Canhão



Classe — com método estático

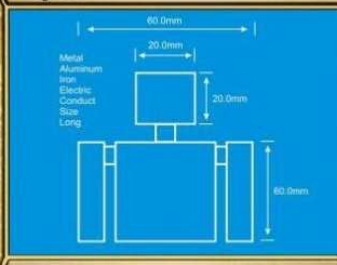
Não é necessário criar um objeto de Arma para usar o método desta classe.

O robô precisa ter a habilidade de usar armas para poder usar esta carta.

AtirarEm *(Objeto)* - Causa 8 pontos de dano no alvo. Demora 5 turnos para recarregar.

Prof. Abrahão Lopes — 8/5

Projeto



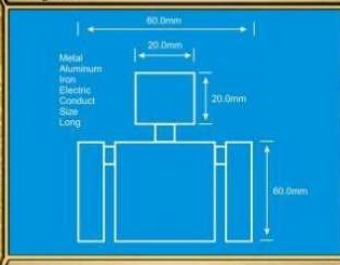
Classe

Descreve o funcionamento do robô.

Agrupa variáveis e métodos.

Prof. Abrahão Lopes

Projeto



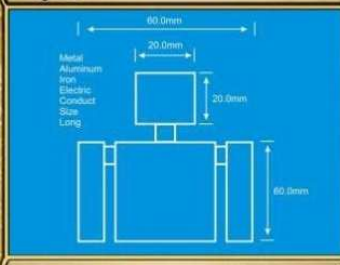
Classe

Descreve o funcionamento do robô.

Agrupa variáveis e métodos.

Prof. Abrahão Lopes

Projeto



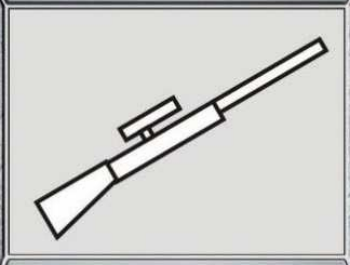
Classe

Descreve o funcionamento do robô.

Agrupa variáveis e métodos.

Prof. Abrahão Lopes

Arma

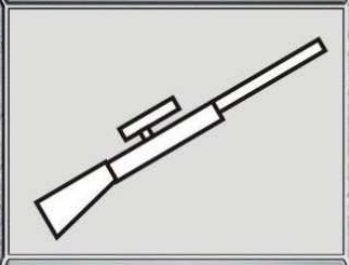


Objeto — arma

Possui as características definidas pela classe de arma atual.

Prof. Abrahão Lopes

Arma

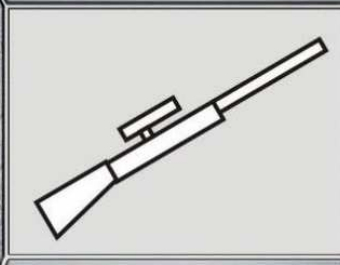


Objeto — arma

Possui as características definidas pela classe de arma atual.

Prof. Abrahão Lopes

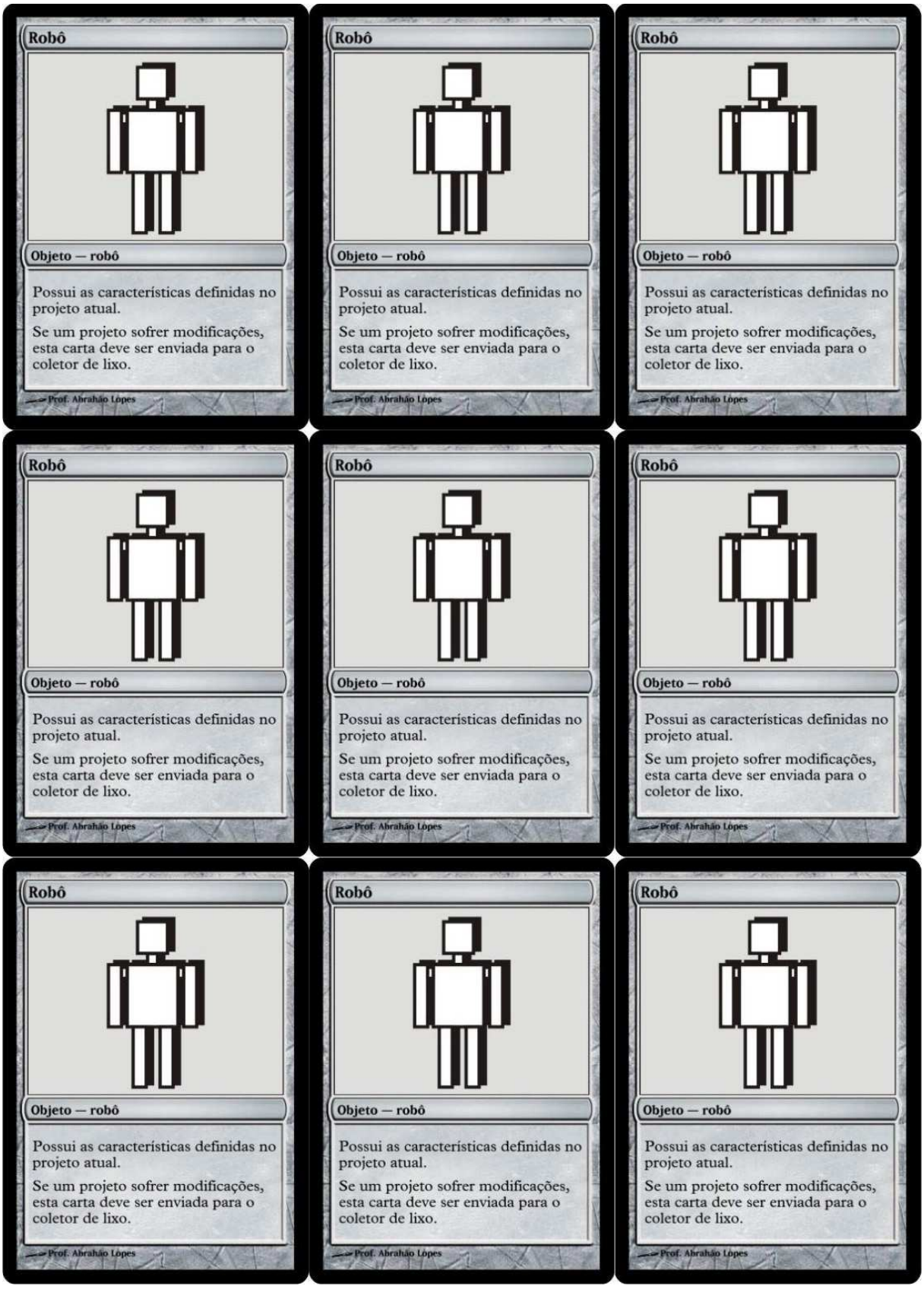
Arma



Objeto — arma

Possui as características definidas pela classe de arma atual.

Prof. Abrahão Lopes



(imprima duas vezes esta página)