LATIN.SCIENCE XVI Congresso Latino-americano de Software Livre e Tecnologias Abertas LATINOWARE 2019

Utilização de Pygame para Ensino e Aprendizado de Orientação a Objetos

Ramon G. Camargo *, Carlos E. Ribeiro†, Fabio Sordi Junior ‡, Paulo R. Anastácio§ e José R. Merlin¶
Universidade Estadual do Norte do Paraná
Rodovia BR 369 - KM 54 - Bandeirantes - PR
*ramon.g.camargo42@gmail.com
†biluka@uenp.edu.br
‡fabiodsj@uenp.edu.br
§paulinho.r.a@gmail.com
¶merlin@uenp.edu.br

***Resumo*—A programação de computadores se apresenta como complexa para muitos estudantes da área. Uma das causas desta complexidade é a dificuldade em lidar com o nível de abstração exigido para construção de programas. Por isto, diversos autores buscam métodos alternativos de ensino. Neste artigo, é abordada a programação de jogos com a utilização de Pygame, uma biblioteca da linguagem Python. Pretende-se, com isto, que a aprendizagem de programação orientada a objetos possa ser tratada como uma experiência visual, diminuindo a abstração. Os temas normalmente vistos nas disciplinas são analisados no contexto de Pygame. Os resultados preliminares mostram que é possível o ensino dos conceitos de orientação a objetos por meio do desenvolvimento de jogos simples.**

Palavras-chave - Pygame; programação orientada a objetos; jogos.

I. Introdução

Uma das causas da alta evasão nos cursos ligados à computação é a dificuldade em entender a lógica necessária a criação de programas, devido ao nível de abstração necessário [1] [2]. Para Viegas, o estudante precisa visualizar o que deve ser feito antes e durante a construção do código fonte [7].

Leite Junior, Murakami e Almeida [3] afirmam que, no contexto do ensino, os "nativos digitais" estão fortemente envolvidos em atividades virtuais, ao mesmo tempo em que as atividades geralmente desenvolvidas em sala de aula ainda são tradicionais, gerando desmotivação nos alunos. Rizzo [4] afirma que "os jogos constituem um poderoso recurso de estimulação do desenvolvimento integral do educando. Eles desenvolvem a atenção, disciplina, autocontrole, respeito às regras e habilidades perceptivas e motoras relativas a cada tipo de jogo oferecido".

Do ponto de vista do estudante de programação, mais do que utilizar jogos para o aprendizado, é interessante que ele aprenda a desenvolver jogos. Com isto, ele passa da situação de consumidor passivo de um *software* à condição de desenvolvedor ativo. Para a criação de jogos, existem diversas bibliotecas, comumente chamadas de *frameworks*. Entretanto, estes *frameworks* trazem muitas funcionalidades prontas, não necessitando muita escrita de código. Embora possam ser adequados ao desenvolvimento de jogos, se o objetivo é aprender programação, talvez eles não contribuam muito significativamente.

Uma alternativa aos *frameworks* é a biblioteca Pygame [1]. Esta biblioteca, escrita em linguagem Python, utiliza o paradigma de orientação a objetos e oferece um conjunto de classes necessárias ao desenvolvimento de jogos. A vantagem é que o código é totalmente em Python, por isso o aluno com algum conhecimento nesta linguagem pode facilmente começar a programar. Pygame é distribuído sob LGPL (*GNU Lesser General Public Licence*), o que permite que seja usado tanto em *softwares* comerciais quanto livres.

Resta estabelecer uma relação entre os conceitos estudados nas disciplinas sobre programação orientada a objetos (POO) e as funcionalidades de Pygame, sendo este o objeto do presente projeto. Espera-se, com isto, propor diretrizes para a utilização de Pygame no ensino de OO.

Para Medeiros, Brasil e Aranha [5], aprender a codificar não é uma tarefa fácil. Para contornar esta dificuldade, os jogos digitais podem ser importantes aliados, por introduzir o aspecto lúdico, tornando-se mais atrativos para os estudantes. No entanto, não foram encontrados na literatura estudos relacionando o desenvolvimento de jogos com os conceitos tradicionalmente vistos nas ementas das disciplinas, exceto em [6]. Assim, muitas vezes, aprender a fazer um jogo não contribuiu, necessariamente, com o aprendizado de programação, pois a montagem de blocos não é associada à escrita de código. Por isso, este estudo visa suprir esta lacuna, investigando como desenvolver um jogo de forma "manual", em outras palavras, escrevendo o código.

O objetivo geral deste trabalho é realizar um mapeamento entre os conceitos de orientação a objetos e as funcionalidades da biblioteca Pygame.

O restante deste texto está organizado como segue. Após esta introdução, é mostrado como a orientação a objetos pode ser trabalhada durante a criação de jogos com Pygame. Em seguida, são mostrados os resultados preliminares obti-

[1] www.pygame.org

dos e, por último as considerações finais.

II. Orientação a objetos e Pygame

Em [6] é mostrado um estudo conduzido para verificar quais são os conteúdos ministrados nas disciplinas sobre orientação a objetos nos cursos de computação. Os 14 principais temas que aparecem em 10 universidades de renome foram:

1) Classes e objetos;
2) Atributos e métodos;
3) Agregação e composição;
4) Herança e generalização;
5) Polimorfismo;
6) Encapsulamento e modificadores;
7) Tratamento de exceções;
8) Programação genérica;
9) Interfaces;
10) Pacotes e classes utilitárias;
11) Streams;
12) Serialização;
13) Sobrecarga; e
14) Sobrescrita.

A partir destes temas, o presente trabalho procura analisar como podem ser abordados em Pygame.

A. Pygame

Na maioria das linguagens, o desenvolvimento de jogos não é algo trivial e recomendável para iniciantes em programação. No entanto, com a linguagem Python e o módulo Pygame, é possível ter uma experiência visual com a programação. Por meio do desenvolvimento de jogos, o aprendiz pode "ver" a lógica funcionando [9].

No sentido de diminuir o grau de abstração envolvido, o que é investigado no presente trabalho é como o desenvolvimento de um jogo ajuda na visualização concreta dos conceitos de POO. Em outras palavras, seria mais fácil visualizar um personagem que anda em um cenário do que uma pilha de inteiros em que se adicionam elementos.

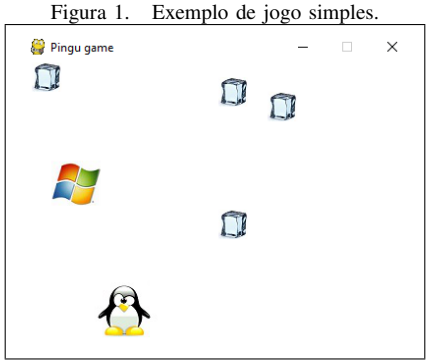
Na Figura 1 é mostrado um exemplo de jogo simples em que o pinguim deve pegar os blocos de gelo (ganha pontos) e evitar o logo do Windows (morre). Este é um exemplo de programa que mesmo estudantes das fases iniciais de programação podem desenvolver, o que confirma a simplicidade de Pygame.

As subseções seguintes analisam alguns dos conceitos de POO com Pygame.

B. Classes e objetos

Os conceitos de classes e objetos costumam ser os primeiros abordados, pois são a base para o que vem depois. Encontram-se na literatura, com frequência, exemplos que se referem a entidades como plantas e prédios. Isto pode não ser muito fácil de entender para um estudante de programação. Apesar de correto, o exemplo não pode ser relacionado

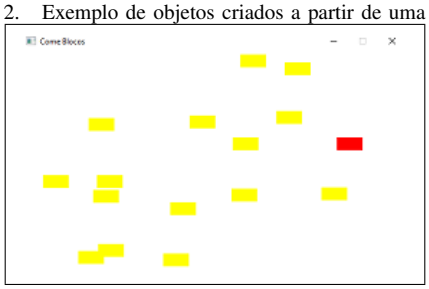
Figura 1. Exemplo de jogo simples.



Fonte: os autores.

com código em linguagem de programação. Para contornar esta abstração, com a utilização de Pygame, o professor poderia utilizar elementos "visíveis". Um jogo com blocos poderia ser criado. A partir da classe `Bloco`, poderiam ser instanciados diversos objetos com as mesmas características, como mostrado na Figura 2.

Figura 2. Exemplo de objetos criados a partir de uma classe.



Fonte: os autores.

C. Atributos e métodos

Atributos são características comuns a objetos de uma mesma classe. Métodos são ações que os objetos executam. Assim, o mesmo exemplo da Figura 2 pode ser utilizado para demonstrar que todos os blocos possuem os atributos `cor`, `tamanho` e `posição`. Neste contexto, pode–se aproveitar para inserir o conceito de *estado do objeto*. Embora os objetos tenham atributos comuns, em dado momento cada um tem um valor para estes elementos. No exemplo, os blocos têm cores e posições diferentes.

D. Herança e generalização

Herança é a possibilidade, oferecida por linguagens orientadas a objetos, de se criar classes derivadas de outras. Criam-se classes mais genéricas (superclasses ou classes base) e, a partir delas, derivam-se classes específicas. No que se refere a Pygame, herança está bem presente. Para se criar uma classe que represente um elemento do jogo (por exemplo, `Herói`, `Inimigo`), é preciso herdar da superclasse `Sprite`. Esta é a classe base para todos os

objetos visíveis em Pygame. Para se criar a classe `Bloco` como subclasse de `Sprite`, o código é

```
class Bloco(pygame.sprite.Sprite):
```

em que a classe qualificada entre parênteses constitui a superclasse de `Bloco`.

O exemplo pode ser usado para explicar que, ao herdar de uma superclasse, a classe pode aproveitar todas as funcionalidades já implementadas, sem ao menos saber como foram implementadas.

O tema Herança pode ainda ser explorado com Pygame de maneira poderosa. A classe `Bloco`, anteriormente citada, tem o método `cair()` que faz o bloco se movimentar na tela de cima para baixo. Caso se queira um outro tipo de bloco que também se movimente na horizontal, não é necessário criar outra classe do zero. Pode-se simplesmente criar a classe `BlocoEspecial` que tivesse o método `mover()`. Com isto, o aluno poderia ver claramente o reuso de código proporcionado pela herança.

E. Agregação e composição

Agregação e composição são tipos de associações entre classes. Uma associação ocorre quanto uma classe possui atributos do tipo de outra classe.

A composição pressupõe uma relação de dependência entre objetos. A diferença fundamental é que o objeto que contém os outros é responsável por controlar, criar e destruir seus agregados. Em Pygame, composição está presente, por exemplo, quando se cria uma classe principal e nela se instanciam outros objetos. Suponha que a classe principal seja `Jogo` e nela sejam instanciados o herói, o inimigo, as armas, objetos do cenário... Então, pode–se dizer que Jogo é uma composição de diversos objetos.

Agregação também é uma associação do tipo todo/parte. Ocorre quando um objeto de determinada classe faz parte (está contido) em outro objeto. No contexto de um jogo, um exemplo seria uma espaçonave que tivesse objetos *bamba* que fossem da classe `Bomba`. A classe `Espaçonave` (todo) precisa da classe `Bomba` (parte) para executar suas ações.

F. Encapsulamento e modificadores

Encapsulamento, em POO, fornece dois benefícios: vincula os dados ao código que os trata e controla o acesso aos membros [8], impedindo que o objeto entre em um estado inválido.

Em Python e, por consequência, em Pygame, uma das formas de promover o encapsulamento é com a utilização de um *decorator* chamado *property*. Um breve exemplo ajuda a explicar o conceito. Em um jogo, o objeto `herói` (classe `Heroi`) possui o atributo `vidas`, que não pode ser inferior a zero. A classe `Herói` seria:

```
class Heroi:
   def __init__(self, vidas=1):
      self.__vidas = vidas

   @property
   def vidas(self):
      return self.__vidas

   @vidas.setter
   def vidas(self, vidas):
      if (vidas < 0):
         return
      else:
         self.__vidas = vidas
```

O sublinhado duplo que precede o atributo `vidas` indica que ele é privado. Para acessar este atributo privado, deve-se criar um método com a anotação `@property` (acesso) e outro com a anotação `@vidas.setter` (modificação). Isto pode ser utilizado para demonstrar a proteção contra modificação externa. Infelizmente, em Python não há os modificadores explícitos como em outras linguagens, o que pode dificultar o entendimento destes.

III. Resultados

Neste trabalho foram mostrados apenas alguns conceitos de POO, tais como classes e objetos, atributos e métodos, herança, associação e encapsulamento. Mesmo assim, os resultados iniciais mostram que é possível a utilização de Pygame na construção se sequências didáticas para ensino de orientação a objetos.

Outros temas sobre programação também estão envolvidos na construção de jogos, por mais simples que sejam. Por exemplo, ao somar pontos, pode ser trabalhado o conceito de acumuladores. Listas são necessárias para manter todos os objetos, além de também conterem os eventos a serem processados. A tela precisa ser atualizada uma quantidade de vezes por segundo, do início ao fim do jogo, o que é uma oportunidade para se trabalhar o conceito de *loop*. Enfim, a construção de um jogo envolve muitos conceitos de programação de forma integrada.

Além das questões de linguagem de programação, é preciso destacar a motivação presente durante o desenvolvimento de um jogo. De modo geral, os estudantes gostam de criar jogos pelo conteúdo lúdico envolvido. Isto leva a maior envolvimento e persistência com a tarefa. Conforme relatam Abrantes et al. [10], a inovação e diversificação de práticas pedagógicas são essenciais para o sucesso no ensinar e aprender.

Como continuidade do estudo, pretende-se aplicar o desenvolvimento de jogos com Pygame em turmas da disciplina de POO, para mensurar qualitativamente os resultados, uma vez que no presente trabalho apenas se verificou a viabilidade.

IV. Considerações finais

O aprendizado de orientação a objetos pode se apresentar abstrato demais para alguns estudantes, o que pode ocasionar um alto índice de reprovação e evasão nos cursos ligados à computação.

Para diminuir as dificuldades, algumas abordagens são mostradas na literatura. Neste artigo foi apresentada a utilização da biblioteca Pygame para desenvolvimento de jogos na linguagem Python. O estudo em andamento vem mostrando que os conceitos de orientação a objetos podem ser abordados de maneira mais "concreta" com esta biblioteca. Elementos como classes e objetos, herança, encapsulamento, entre outros, foram implementados em um jogo. Com isto, podendo "ver" o comportamento dos objetos, pretende–se que diminuir o grau de abstração envolvido na programação.

O trabalho continua com a análise de outros elementos da POO ainda não abordados.

Agradecimentos

Os autores agradecem à Fundação Araucária pelo apoio financeiro ao projeto por meio do PIBIC - Programa Institucional de Bolsas de Iniciação Científica.

Referências

[1] Fernandes, V. S.; Freitas Junior, V. Evasão e Reprovação: uma análise das metodologias de ensino para a disciplina de lógica de programação. In.: Mostra Nacional de Iniciação Científica e Tecnológica Interdisciplinar. 2014

[2] Raabe, A. L. A.; Silva, J. M. C. Um ambiente para atendimento as dificuldades de aprendizagem de algoritmos. In: XIII Workshop de Educação em Computação – WEI, p.2326-2335. 2005

[3] Leite Junior, A. J. M; Murakami, L. C.; Almeida, R. B. Criação de Jogos no curso de Administração (Relato de Práticas). In.: 3º Congresso Brasileiro de Informática na Educação (CBIE 2014). 2014, Dourados. Anais... Dourados:

[4] Rizzo, G. Jogos inteligentes: a construção do raciocínio na escola natural. Rio de Janeiro: Bertrand Brasil, 1996.

[5] Medeiros, T.; Brasil, P.; Aranha, E. Um framework para criação de jogos voltados para o ensino de lógica de programação. In.: 3º Congresso Brasileiro de Informática na Educação (CBIE 2014). 2014, Dourados. Anais... Dourados:

[6] Merlin, J. R.; Saques, V. O. ; Braz, R. S. ; Anastacio, P. R. . Análise da Ferramenta Robocode Para Aprendizado de Orientação a Objetos. In: CONIEN - II Congresso Internacional de Ensino, 2019, Cornélio Procópio. Anais do CONIEN, 2019. p. 2036-2045.

[7] Viegas, T. R.; Okuyama, F. Y.; Paravisi, M.; Bertagnolli, S. D. (2015). Uso das tics no processo de ensino-aprendizagem de programação. In Nuevas Ideas en Informática Educativa – TISE, pages 780–785.

[8] H. Schildt. Java para iniciantes. 6. ed. Porto Alegre: Bookman, 2015.

[9] H. Kinsley; W. McGugan. Python Games Development with Pygame. New York: Apress, 2015.

[10] Abrantes, P., Mauritti, R., Roldão, C., Alves, L., Amaral, P., Baptista, I., Teixeira, A. (2011). Efeitos TEIP: Avaliação de impactos escolares e sociais em sete territórios educativos de intervenção prioritária. Centro de Investigação e Estudos de Sociologia do Instituto Universitário de Lisboa, 12-27.