Trabalho de INF1301

15/04/2021

Prof. Ivan Mathias Filho

Objetivo da 1ª Iteração

O objetivo da 1ª iteração é implementar todas as regras do jogo Mastermind. Para tal será necessário definir as estruturas de dados que serão usadas para representar

- os pinos que compõem a senha;
- os pinos usados em cada tentativa de descoberta da senha;
- as marcações usadas para informar o progresso do jogador tendo em vista a descoberta da senha.

Além disso, terão de ser elaboradas funções para

- a escolha do nível de dificuldade de uma partida;
- a definição, de forma aleatória, da senha;
- a comparação das combinações de cores resultantes das jogadas com a combinação de cores da senha.

As regras que serão implementadas neste trabalho estão publicadas em http://www.gomes-mota.nome.pt/joao/www/jgcores/regras_mm.html

Em https://www.jogosjogos.com/jogar-jogo/mastermind!.html existe uma implementação do Mastermind que pode ser usada para que as regras possam ser bem compreendidas.

Arquitetura da Implementação

O software para jogar Mastermind será organizado segundo a arquitetura MVC. Na 1ª iteração será iniciada a especificação e codificação do componente Model dessa arquitetura. Esse componente será formado por um conjunto de funções que fornecerá uma API (Application Programming Interface) para que os demais componentes (View e Controller) solicitem, quando necessário, serviços ao componente Model.

Implementação do Componente Model

O componente Model **tem de ser** um pacote Python contendo pelo menos um módulo (arquivo .py). Todas as variáveis, funções e estruturas de dados usadas para

implementar as regras do Mastermind, e apenas as regras, farão parte do componente Model.

Segundo o princípio da ocultação da informação, apenas as funções de um módulo que serão diretamente chamadas pelos demais módulos de uma aplicação – tendo em vista obter os serviços necessários à execução das tarefas desses módulos – deverão ser públicas. Isto é, as variáveis usadas por um módulo, além das funções que realizam tarefas internas desse módulo, não devem ser visíveis aos demais módulos da aplicação.

O princípio da ocultação da informação deverá ser implementado por meio da variável **__all__** , que irá definir todas as funções públicas (API) de um módulo (**X**). Além disso, os demais módulos importarão a API do módulo **X** por meio do comando **from X import *** .

Não será obrigatório implementar todos os detalhes das regras do jogo nesta 1ª iteração, mesmo porque quando os componentes View e Controller começarem a ser implementados surgirão, certamente, demandas por parte desses componentes que não foram inicialmente previstas. Sendo assim, o componente Model deverá permitir futuras alterações para atender a essas demandas.

Para tal, deve-se observar o princípio aberto/fechado, cuja formulação é atribuída a Bertrand Meyer, autor do livro **Object Oriented Software Construction**. Esse princípio, que lhes foi apresentado no capítulo 1, diz o seguinte:

- Um módulo será dito aberto se ele ainda está disponível para extensão. Por exemplo, se for possível adicionar campos às estruturas de dados desse módulo, ou novos elementos ao conjunto de funções que executa.
- Um módulo será dito ser fechado se estiver disponível para uso por outros módulos. Isso pressupõe que o módulo tenha sido bem-definido. Isto é, que tenha uma descrição estável e que sua API abstraia suas características específicas.

Em termos de processo de desenvolvimento de software, é importante que o relatório da iteração diga o que foi implementado e testado; o que foi implementado, mas não foi totalmente testado; e o que não foi implementado no curso de uma iteração. Ele deve, também, relatar os problemas encontrados e os motivos pelos quais uma ou outra funcionalidade não foi implementada ou testada. Por último, esse relatório deve apontar as funcionalidades que serão implementadas na próxima iteração e os responsáveis por cada uma delas.

O relatório da 1ª iteração deve usar o modelo de relatório que está disponível na seção **Trabalho** da página de INF1301 no EAD.

Testes Unitários

Testes unitários são realizados tendo em vista testar unidades individuais de código fonte. Tais unidades podem ser funções, métodos, classes, módulos e etc. Eles têm por objetivo verificar se cada unidade atende corretamente à sua especificação.

Todas as funções que compõem um módulo têm de ser testadas individualmente. Para tal, crie um caso de teste para cada função a ser testada. Utilize o modelo de caso de teste que foi publicado na página de INF1301 no EAD.

Caso um função **f1** chame uma função **f2**, deve-se, primeiro, testar a função **f2** para garantir que ela está funcionado segundo a sua especificação. Em seguida testa-se a função **f1**, pois mesmo que ela não esteja funcionando corretamente, não será necessário retestar a função **f2**, pois já se sabe que ela está funcionado de acordo com o especificado.