

U.C. 21025

Desenvolvimento de Software

12 de julho de 2017

INSTRUÇÕES

PARA A RESOLUÇÃO DA PROVA, ACONSELHA-SE QUE LEIA ATENTAMENTE O SEGUINTE:

1. Verifique o exemplar que lhe foi entregue e, no caso de estar incompleto ou com qualquer deficiência, dirija-se ao professor vigilante. A prova termina com a palavra **FIM**.
2. A prova é efetuada sem consulta de outros materiais. No final do enunciado tem material de referência para consulta durante a prova.
3. A prova está dimensionada para um tempo de resolução de uma hora e trinta minutos.
4. A prova é constituída por 2 partes. A cotação de cada uma das partes está indicada.
5. A resolução deve ser escrita a esferográfica. Utilize, sempre, uma letra legível e não use uma caneta de outra cor que não seja o preto ou o azul - as respostas a lápis não serão consideradas.
6. A interpretação dos enunciados das questões também faz parte da sua resolução, pelo que, se existir alguma ambiguidade, deve indicar claramente como foi resolvida.

Bom Trabalho!

Nome:

N.º de Estudante: B. I. n.º

Turma: Assinatura do Vigilante:

1.ª Parte (4 Valores)

Código de referência

Adaptado de: <https://www.codeproject.com/Articles/42236/Game-of-Life-Simulator-as-Learning-Tool>

```
01 static void Main(string[] args)
02 { //Valores iniciais
03     LifeGame lifeGame = new LifeGame(60, 40);
04     double lifeProbability = 0.25;
05     lifeGame.Randomize(lifeProbability);
06     bool exit = false;
07     bool first = true;
08     int genCount = 0;
09     while (!exit) //Ciclo global do jogo
10     {
11         Console.Clear(); //Limpar o ecrã
12         if (!first) //Avançar uma geração no jogo da vida
13         {
14             lifeGame.Step();
15             genCount += 1;
16         }
17         first = false;
18         Console.Write(lifeGame.ToString()); //Atualizar ecrã
19         Console.WriteLine("Any key to continue, 'r' to randomize, 'q' to quit.
20                               Gen: " + genCount);
21         char key = Console.ReadKey(false).KeyChar; //Comandar o jogo
22         if (key == 'q')
23         {
24             exit = true; //Permite sair do jogo
25         }
26         else if (key == 'r') //Reiniciar o jogo da vida
27         {
28             lifeGame.Randomize(lifeProbability);
29             first = true;
30             genCount = 0;
31         }
32     }
```

As alíneas seguintes são todas de resposta curta.

1.

a) Indique as linhas do código de referência onde há operações de *input*:

b) Indique as linhas do código de referência onde há operações de *output*:

2. Suponha que queria reescrever o código de referência segundo o estilo arquitetónico MVC. Indique como distribuiria pelos componentes as tarefas descritas nos comentários desse código.

Por ex., se considerar que a tarefa “Valores iniciais” (linha 02) corresponde ao Controller, escreva “C: 02”. Se considerar que corresponde ao Model, escreva “M: 02”. Se considerar que corresponde à View, escreva “V: 02”.

a) Segundo a abordagem de Krasner & Pope (1988):

b) Segundo a abordagem de Curry & Grace (2008):

Nome:

N.º de Estudante: B. I. n.º

Turma: Assinatura do Vigilante:

c) Explique as dúvidas ou dilemas com que se debateu para responder às alíneas a) e b) e justifique as principais opções que tomou ao dar as suas respostas: _____

3. O código anterior depende de vários componentes não explicitados: (1) o sistema operativo, que fornece argumentos de linha de comandos, processa os pedidos de escrita no ecrã e leitura do teclado; (2) a classe LifeGame, que fornece a lógica do jogo da vida; (3) o gerador de número aleatórios do sistema, a que apoia a classe LifeGame.

a) Desenhe um diagrama onde o código anterior e estes três componentes surjam num padrão de desenho do tipo “Arquitetura estratificada”.

b) Um dos objetivos da etapa de desenvolvimento dedicada à produção de código-fonte é *desenvolver os aspetos que não tenham sido cobertos pela fase de análise e conceção do sistema de software* (Guerreiro, 2015, p. 155). Considere que a fase de análise e conceção produziu, como únicos resultados, a lista de *inputs* e *outputs* da pergunta 1 e a arquitetura estratificada da alínea 3 a). Que aspetos do código-fonte fornecido não foram cobertos?

Nome:

N.º de Estudante: B. I. n.º

Turma: Assinatura do Vigilante:

4. Considere o código de referência sob a perspectiva da verificação e validação do software. Dê alguns exemplos de falhas, erros e faltas que possam ocorrer nesse código.

2.ª Parte (8 Valores)

Em duas páginas, efetue uma descrição do trabalho desenvolvido ao longo dos e-fólios, refletindo sobre os temas que se seguem.

- a) A separação em módulos (Model-View-Controller ou o estilo arquitetónico efetivamente seguido);
 - i) qual o principal problema que encontrou;
 - ii) qual a abordagem seguida e como resultou;
 - iii) em retrospectiva, que mudaria na abordagem a esta separação.
- b) O recurso a eventos e interfaces para separação de interesses e independência de dados;
 - i) qual o principal problema que encontrou;
 - ii) qual a abordagem seguida e como resultou;
 - iii) em retrospectiva, que mudaria na abordagem a esta técnica.
- c) O recurso a exceções personalizadas para gestão de erros.
 - i) qual o principal problema que encontrou;
 - ii) qual a abordagem seguida e como resultou;
 - iii) em retrospectiva, que mudaria na abordagem a esta técnica.

FIM

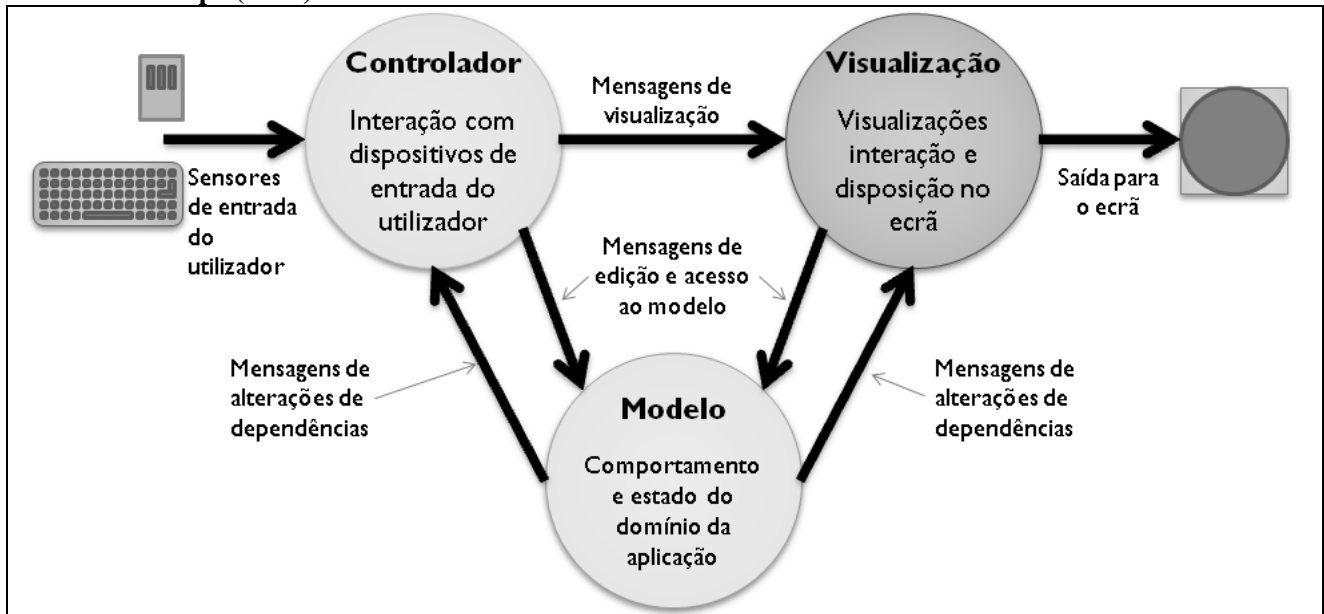
Nome:

N.º de Estudante: B. I. n.º

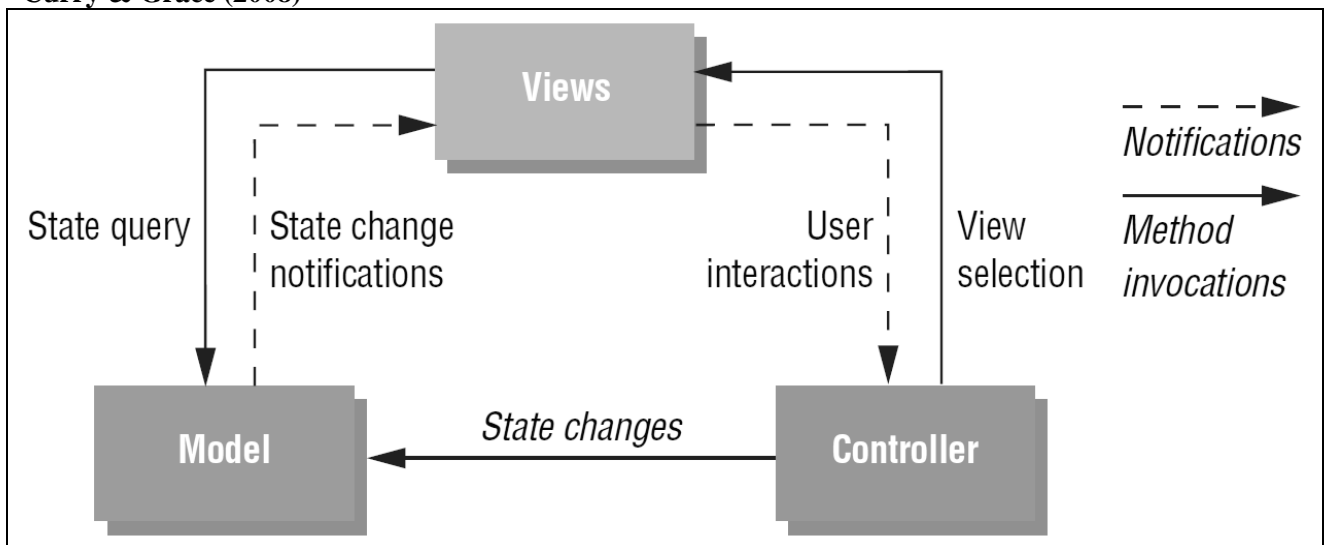
Turma: Assinatura do Vigilante:

Material de referência

Krasner & Pope (1988)



Curry & Grace (2008)



Apoios de sintaxe

```
throw new ClasseDaExcecao(ParametroDaExcecao);  
catch (ClasseDaExcecao VariavelDaExcecao) { }  
public delegate TipoDevolvido NomeDoTipoDeDelegado(Parametro1, Parametro2);  
NomeDoTipoDeDelegado delegado = new NomeDoTipoDeDelegado(MetodoAtribuido);  
public event NomeDoTipoDeDelegado NomeDoEvento;  
NomeDoEvento += delegado;  
NomeDoEvento += new NomeDoTipoDeDelegado(MetodoAtribuido);
```