Sobre acoplamento complexo

5/7 points (71%)

Teste para praticar, 7 questions

✖ **Tente novamente quando estiver preparado.**

Pontuação necessária para ser aprovado: 75% ou mais

Você pode refazê-lo quantas vezes quiser.

De volta à semana 6

Refazer

Sobre acoplamento complexo

1/1 pontos

5/7 points (71%)

Teste para praticar, 7 questions

1.

Diferença entre delegação e redirecionamento:

- [ ] Delegação é quando um método m apenas repassa sua responsabilidade para outro método de outra classe, cujo nome desse outro método é igual a m!

Não selecionado está correto

- [ ] Redirecionamento é quando um método m apenas repassa sua responsabilidade para outro método de outra classe, cujo nome desse outro método é igual a m!

Correto

Alguns autores consideram essa definição como estando incluída na definição de delegação, não reconhecendo o termo redirecionamento exclusivo para esse caso!

- [ ] Redirecionamento é quando um método m apenas repassa sua responsabilidade para outro método de outra classe, cujo nome desse outro método é diferente de m!

Não selecionado está correto

- [ ] Delegação é quando um método m apenas repassa sua responsabilidade para outro método de outra classe, cujo nome desse outro método é diferente de m!

Correto

Alguns autores consideram que a definição de delegação também inclui o caso do nome do outro método ser igual a m! Ou seja, inclui o que estamos chamando de redirecionamento!

Sobre acoplamento complexo

1/1 pontos

5/7 points (71%)

Teste para praticar, 7 questions

2.

Reconheça o conceito de delegação no código Java abaixo, seguindo o seguinte formato: [classe delegadora–método delegador/objeto delegado–classe delegada–método delegado]

```
1  public class GerenteDeVendas{
2     . . .
3     public void aumenteVendasTrimestralmente( ){. . .}
4     . . .
5  }
6  public class VicePresidenteDeVendas{
7     . . .
8     public void aumenteVendasTrimestralmente( ){
9        gerDeVendas.aumenteVendasTrimestralmente( );
10    }
11    . . .
12    GerenteDeVendas gerDeVendas;
13 }
14 public class Presidente{
15    . . .
16    public void aumenteLucros(Financeiro financeiro){
17       vpDeVendas.aumenteVendasTrimestralmente( );
18    }
19    . . .
20    VicePresidenteDeVendas vpDeVendas;
21 }
```

- VicePresidenteDeVendas–aumenteVendasTrimestralmente( )/gerDeVendas–GerenteDeVendas–aumenteVendasTrimestralmente( )
- Presidente–aumenteLucros( )/vpDeVendas–VicePresidenteDeVendas–aumenteVendasTrimestralmente( )

Correto

Sobre acoplamento complexo

1/1 pontos

5/7 points (71%)

Teste para praticar, 7 questions

3

O trecho em negrito "**vpDeVendas.getGerDeVendas( )**.monitoraMarketing( )" corresponde a objeto anônimo de que classe?

```
1  public class GerenteDeVendas{
2     . . .
3     public void aumenteVendasTrimestralmente( ){. . .}
4     public void monitoraMarketing( ){. . .}
5     . . .
6  }
7  public class VicePresidenteDeVendas{
8     public void aumenteVendasTrimestralmente( ){
9        gerDeVendas.aumenteVendasTrimestralmente( )
10    }   public GerenteDeVendas getGerDeVendas( ){return gerDeVendas;}
11    . . .
12    GerenteDeVendas gerDeVendas;
13 }
14 public class Presidente{
15    . . .
16    public void aumenteLucros(Financeiro financeiro){
17       vpDeVendas.aumenteVendasTrimestralmente( );
18       // analisa estratégia de marketing
19       vpDeVendas.getGerDeVendas( ).monitoraMarketing( );
20       . . .
21    }
22    . . .
23    VicePresidenteDeVendas vpDeVendas;
24 }
```

gerDeVendas.aumenteVendasTrimestralmente( )

- GerenteDeVendas

Correto

- Presidente
- VicePresidenteDeVendas

Sobre acoplamento complexo

0/1 pontos

5/7 points (71%)

Teste para praticar, 7 questions

4.

Dentre as mensagens ou invocações de métodos no método aumenteLucros(Financeiro financeiro) da classe Presidente abaixo, aponte qual não satisfaz o princípio "Law of Demeter". Escolha dentre os números 1 a 4!

```
1   public class Presidente{
2      . . .
3      public void aumenteLucros(Financeiro financeiro){
4   1     vpDeVendas.aumenteVendasTrimestralmente( );
5   2     this.verificaDadosContábeis();
6         // analisa estratégia de marketing
7   3     vpDeVendas.getGerDeVendas( ).monitoraMarketing( );
8         Estoque estoque = new Estoque();
9   4     estoque.getSaldoGeral();
10  5     financeiro.consisteDados();
11     }
12     . . .
13     VicePresidenteDeVendas vpDeVendas;
14  }
```

. . .

- 4

Isso não deve ser selecionado

Mensagem oportunista a objeto da classe Estoque

- 5
- 3
- 1
- 2

Sobre acoplamento complexo

5/7 points (71%)

Teste para praticar, 7 questions

0/1 pontos

5.

Reconheça o conceito de redirecionamento no código Java da questão 3, seguindo o seguinte formato: [classe delegadora–método delegador/objeto delegado–classe delegada–método delegado]

- Presidente–aumenteLucros( )/vpDeVendas–VicePresidenteDeVendas–aumenteVendasTrimestralmente( )

Isso não deve ser selecionado

- VicePresidenteDeVendas–aumenteVendasTrimestralmente( )/gerDeVendas–GerenteDeVendas–aumenteVendasTrimestralmente( )

Sobre acoplamento complexo

1/1 pontos

5/7 points (71%)

Teste para praticar, 7 questions

6.

Ao usar os princípios "Law of Demeter", "Tell, Don't Ask!" e delegação, chegamos ao código Java abaixo. Está tudo correto?

```
1  public class GerenteDeVendas{
2     . . .
3     public void aumenteVendasTrimestralmente( ){. . .}
4  1  public void monitoraMarketing( ){. . .}
5     . . .
6  }
7  public class VicePresidenteDeVendas{
8     public void aumenteVendasTrimestralmente( ){. . .}
9  2  public void analisaEstrategiaMarketing( ){
10 3     gerDeVendas.monitoraMarketing( );
11 4  }
12    . . .
13    GerenteDeVendas gerDeVendas;
14 }
15 public class Presidente{
16    . . .
17    public void aumenteLucros(Financeiro financeiro){
18       . . .
19 5     vpDeVendas.analisaEstrategiaMarketing( );
20    }
21    . . .
22    VicePresidenteDeVendas vpDeVendas;
23 }
```

- Sim

Correto

Verifique o código e constate que houve uma delegação da linha de nr. 5 para o método da linha nr. 2 e outra delegação da linha de nr. 3 para o método da linha nr. 1.

- Não

Sobre acoplamento complexo

1/1 pontos

5/7 points (71%)

Teste para praticar, 7 questions

7.

Considerando a "Law of Demeter", o que eu não posso fazer:

- [x] invocar método de amigo de amigo

Correto

- [ ] Usar variáveis estáticas!

Não selecionado está correto

- [x] invocar método de amigo de amigo de amigo

Correto

- [ ] invocar método de amigo

Não selecionado está correto