IFBA – Instituto Federal de Educação, Ciência e Tecnologia
Análise e Desenvolvimento de Sistemas
INF011 – Padrões de Projeto
Prof.: Sandro Santos Andrade

3ª Avaliação Individual – 2016.2

Instruções:
- Todos os códigos-fonte produzidos, exceto arquivos produtos da compilação (ex: .class), devem ser enviados em um único arquivo .zip para sandro.andrade@gmail.com ao final da avaliação.
- O e-mail deve obrigatoriamente ter o subject: INF011-20162-P3.

Questão 1) (5,0) Uma das decisões de implementação do *Observer* é a definição de quem irá disparar a notificação dos *Observers*: se o próprio *Subject* ou se o cliente. Deixar a cargo do cliente é interessante para evitar a notificação de múltiplas pequenas mudanças, deixando para notificar somente quando a última de um conjunto de mudanças for realizada. Entretanto, corre-se o risco de um cliente esquecer de chamar o método de notificação. Suponha o código-fonte abaixo, onde os dados de um cliente são atualizados e deseja-se notificar os *Observers* somente ao final do processo:

```
ICliente cliente1 = new ClientePessoaFisica();
cliente1.setEndereco("Rua tal");
cliente1.setTelefone("1111-2222");
cliente1.setCNPJ("123.123.123/0001-11");
cliente1.save();
cliente1.notifyObservers();
```

No futuro, novos tipos de cliente poderão aparecer (ex: ClientePessoaJuridica) e a operação *save()* deve ser implementada de modo específico para cada tipo de cliente. Implemente uma solução onde o uso combinado do *Observer* e do *Template Method* faz com que os clientes não precisem se lembrar de executar o método *notifyObservers()*. Tal notificação deverá acontecer como parte da execução do método *save()*.

(1,0) – implementação correta das interfaces
(2,0) – implementação correta do *Subject*
(2,0) – implementação correta do *Template Method*

Questão 2) (5,0) Sabe-se que um dos problemas do Visitor é a impossibilidade de adição de novos tipos concretos de elementos do agregado sem requerer modificações na API de todos os Visitors já implementados. Implemente uma solução para este problema, de modo que novos tipos concretos de elementos do agregado possam ser dinâmicamente adicionados, sem requerer **nenhuma** modificação nos Visitors já implementados. OBS: lembrem-se que um Visitor realiza diferentes operações para diferentes tipos concretos de elementos. Implemente sua solução com base no código *Visitor.tar.gz* anexo a esta prova.

(1,0) – implementação correta da nova interface dos *Visitors*
(4,0) – implementação correta do método de visitação

Boa sorte !