Atividade Supervisionada AS10(a)

1. Em programação orientada a objetos, o padrão de projeto denominado *Iterator* define uma forma de acesso sequencial aos elementos de um objeto agregado, sem expor sua representação interna.

C. Certo

E. Errado

2. O padrão *Command* encapsula uma requisição em um objeto, permitindo a parametrização de clientes com diferentes requisições, filas ou requisições de log.

C. Certo

E. Errado

3. Para definir uma classe que possui apenas uma instância e provê um ponto de acesso global a ela, é correto o uso do padrão *Singleton*.

C. Certo

E. Errado

4. O padrão de projeto que tem como finalidade separar a construção de um objeto complexo de sua representação, de forma que um mesmo processo de construção possa criar diferentes representações é conhecido como

A. *Abstract Factory*

B. *Builder*

C. *Composite*

D. *Factory Method*

E. *Prototype*

5. A definição de que um sistema deve ser desenvolvido em três níveis é feita pelo padrão de projeto

A. *MVC (Model View Controller).*

B. *MVC-Dev (Model Value Constructive Development).*

C. *TMS (Time Milestones Setting).*

D. *PMC (Project Main Controller).*

E. *MCA (Model Classes Assignment).*

6. Um programador deve criar um projeto que envolva vários tipos de produtos com as mesmas funções, mas com peculiaridades diferentes. Por exemplo, o produto do tipo *gold* realiza as mesmas funções que o produto *standard*, mas, a cada uma delas, armazena a última configuração para fornecer uma memória para o usuário. O padrão apropriado para representar as classes deste projeto é o

A. *Facade*, pois ele pode juntar vários comportamentos em um só, criando uma visão simplificada do sistema.

B. *Strategy*, pois ele permite criar uma única interface com várias implementações que diferem apenas em seu comportamento.

C. *Proxy*, pois este permite criar uma representação menos custosa de cada um dos objetos do sistema.

D. *Iterator*, pois este permite visitar todos os objetos do sistema sem se preocupar com a classe real de cada um deles.

E. *Singleton*, pois este garante a existência de uma única instância de produto e evita a confusão entre as classes.

7. Programadores se deparam muitas vezes com a situação onde é preciso acrescentar responsabilidades a objetos e não a classe. Uma alternativa é atribuir dinamicamente a um dado objeto. Este padrão é chamado de:

A. *Singleton*.
B. *Instance*.
C. *Decorator*.
D. *Prototype*.
E. *Bridge*.

8. O padrão *Adapter* é bastante utilizado para compatibilizar classes implementas por programadores diferentes, ou desenvolvidas em momentos diferentes, ou ainda para unir classes com interfaces diferentes em uma estrutura hierárquica única, sem precisar implementar novamente todas as funcionalidades e interfaces da classe já existente, considere as afirmativas:

I – A ideia é criar uma classe (*Adapter*) com a interface padrão que se deseja para fazer a conexão com a classe já existente (*Adaptee*) com interface diferente da estrutura de classes que se pretende utilizar no sistema.

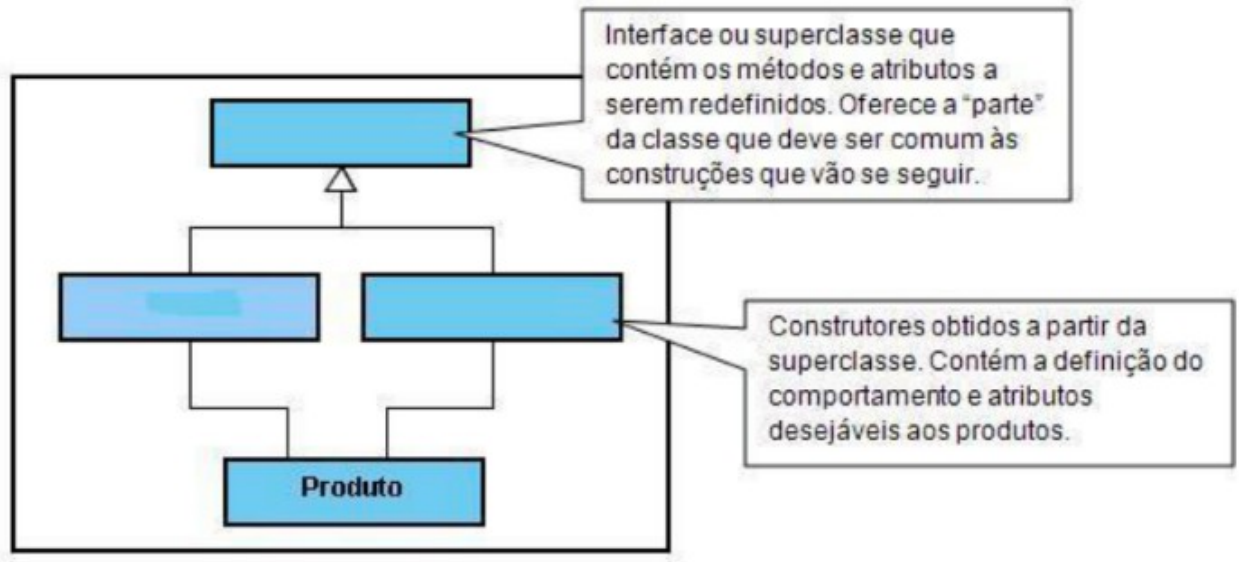
III – Além de reaproveitar totalmente a classe antiga sem precisar alterar o código e entender a complexa implementação realizada, mantém a uniformidade do seu projeto original.

III – cria uma hierarquia de classes diferentes categorias de objetos sem relação de herança.

A. As afirmativas I, II, estão corretas e a afirmativa III está errada.
B. As afirmativas I, III, estão corretas e II errada.
C. As afirmativas II, III estão corretas e a afirmativa I está errada.
D. As afirmativas I, II, III estão incorretas.
E. Todas as afirmativas estão corretas.

9. Conhecido como padrões da gangue dos quatro, por terem sido desenvolvidos por quatro autores, os Padrões GoF (Group of Four) estão divididos pelas seguintes famílias de padrões: Padrões de Criação ou de Construção, Padrões Estruturais e Padrões Comportamentais.

Considerando os padrões de Criação ou de Construção, analise o modelo abaixo e em seguida marque a alternativa que define a representação.



A. *MEDIATOR.*
B. *BUILDER.*
C. *FACADE.*
D. *SINGLETON.*
E. *FACTORY METHOD.*

10 – Sobre o padrão *Composite* podemos afirmar:

I – É utilizado quando se pretende representar hierarquias partes-todo (ou todo-parte) de objetos, ou ainda, quando se pretende modelar relacionamento de agregação.

II – o cliente poderá acessar objetos compostos ou não de maneira uniforme, pois irá se relacionar com a classe abstrata.

III – Um processamento pode ser realizado diretamente por uma superclasse Componente, quando se trata de uma composição, e o processamento parcial é feito pela classe filha, montando todas as partes que compõem o objeto.

A. As afirmativas II, III estão corretas e a afirmativa I está errada.
B. As afirmativas I, III, estão corretas e II errada.
C. As afirmativas I, II, estão corretas e a afirmativa III está errada.
D. As afirmativas I, II, III estão incorretas.
E. As afirmativas estão todas corretas.