Nome: Rafael dos Santos Parisi
RA: 148418

1. Descreva três benefícios da propriedade de projeto chamada ocultamento de informação (information hiding)?

R: Desenvolvimento em paralelo, trazendo a possibilidade de classes serem desenvolvidas enquanto outras também estão sendo. Reduzindo o tempo de implementação do sistema.
Flexibilidade a mudanças, trazendo um menor acoplamento entre as classes, sendo assim, a troca de classes fica mais prática.
Facilidade de entendimento, pois o escopo da classe estará bem definido e facilita o entendimento de novos integrantes.

2. Suponha que um programador adote a seguinte estratégia: ao implementar qualquer nova funcionalidade ou corrigir um bug que implique na modificação de duas classes A e B localizadas em arquivos diferentes, ele conclui a tarefa movendo as classes para o mesmo arquivo. Explicando melhor: após terminar a tarefa de programação que ficou sob sua responsabilidade, ele escolhe uma das classes, digamos a classe B, e a move para o mesmo arquivo da classe A. Agindo dessa maneira, ele estará melhorando qual propriedade de projeto? Por outro lado, qual propriedade de projeto estará sendo afetada de modo negativo? Justifique.

R: Aumentando a coesão do projeto. Mas, ao mover a classe B para o escopo da classe A, o programador estará aumentando o acoplamento do projeto, deixando os escopos misturados ao invés de deixar cada arquivo para uma classe específica.

3. Classitis é o nome dado por John Ousterhout à proliferação de pequenas classes em um sistema. Segundo ele, classitis pode resultar em classes que individualmente são simples, mas que aumentam a complexidade total de um sistema. Usando os conceitos de acoplamento e coesão, como podemos explicar o problema causado por essa doença?

R: A doença seria uma baixa coesão com grande acoplamento, isso significa que as classes são interdependentes e possuem escopos não coesos.

4. Defina: (a) acoplamento aceitável; (b) acoplamento ruim; (c) acoplamento estrutural; (d) acoplamento evolutivo (ou lógico).

R: a) Aceitável é quando uma classe consome métodos públicos de outra classe, a qual possui interface estável que não muda frequentemente.
b) Ruim é quando temos injeção de dependência de uma classe para a outra, na qual as mudanças da classe acoplada implicam em mudanças da classe que recebe o acoplamento.
c) Estrutural é quando uma classe possui a declaração explícita de outra classe dentro de sua estrutura.
d) Evolutivo (ou lógico): É quando mudanças de uma classe se propagam para mudanças de outra classe.

5. Dê um exemplo de: (1) acoplamento estrutural e aceitável; (2) acoplamento estrutural e ruim.

R: (1) Estrutural e aceitável é de acordo com o exemplo onde a classe Estacionamento, utiliza a classe Hashtable, isso é aceitável pois a interface dessa classe é estável.

(2) Estrutural e ruim é quando tenho uma classe Singleton por exemplo, e esta é consumida por outras que podem ser impactadas diretamente pelas mudanças de acoplamento que essa mesma referência global possui com outras classes do sistema que também podem ser alteradas.

6. É possível que uma classe A esteja acoplada a uma classe B sem ter uma referência para B em seu código? Se sim, esse acoplamento será aceitável ou será um acoplamento ruim?

R: Sim, é possível ter um acoplamento indireto e esse acoplamento não é tão preocupante quanto o direto, podendo ser caracterizado como aceitável.

7. Suponha um programa em que todo o código está implementado no método main. Ele tem um problema de coesão ou acoplamento? Justifique.

R: Um problema de acoplamento, pois está coeso com o escopo todo em um local. Alguns problemas gerados por isso são: a manutenção dificultada e a falta de separação de obrigações desse grande código monólito.

8. Qual princípio de projeto é violado pelo seguinte código?

R: Primeiro que o fluxo de pagamento está todo definido em uma função click, ou seja, não respeito o princípio de responsabilidade única.

9. Costuma-se afirmar que existem três conceitos chaves em orientação a objetos: encapsulamento, polimorfismo e herança. Suponha que você tenha sido encarregado de projetar uma nova linguagem de programação. Suponha ainda que você poderá escolher apenas dois dos três conceitos que mencionamos. Qual dos conceitos eliminaria então da sua nova linguagem? Justifique sua resposta.

R: Herança, pois com o tempo tende a introduzir no sistema um forte acoplamento, diminuindo a manutenibilidade, além de violar o encapsulamento das classes pai.

10. Qual princípio de projeto é violado pelo seguinte código? Como você poderia alterar o código do método para atender a esse princípio?

R: Viola o princípio da responsabilidade única.

11. Qual princípio de projeto é violado pelo seguinte código? Como você poderia alterar o código do método para atender a esse princípio?

R: Princípios: responsabilidade única, segregação de interfaces, inversão de dependências. Criaria uma classe responsável por obter as informações de contratação de usuário, outra responsável pelo log do sistema que irá formatar o output das informações obtidas pela classe responsável pelos dados do funcionário.

12. As pré-condições de um método são expressões booleanas envolvendo seus parâmetros (e, possivelmente, o estado de sua classe) que devem ser verdadeiras antes da sua execução. De forma semelhante, as pós-condições são expressões booleanas envolvendo o resultado do método. Considerando essas definições, qual princípio de projeto é violado pelo código abaixo?

R: Não está de acordo com o princípio da substituição de Liskov, pois o funcionamento parece ser afetado ao utilizar a subclasse B, que possui requisitos diferentes da classe base, principalmente no retorno do método.

13. Calcule o CBO e LCOM da seguinte classe:

R: LCOM = 1

CBO = 5

Pares de métodos	Conjunto A	Intersecção dos Conj A
(m1, m2)	A(m1) = {f1, f2} A(m2) = {f2, f3}	{f2}
(m1,m3)	A(m1) = {f1, f2} A(m3) = {f3}	vazio
(m2, m3)	A(m2) = {f2, f3} A(m3) = {f3}	{f3}

14. Qual das seguintes classes é mais coesa? Justifique computando os valores de LCOM de cada uma delas.

R: LCOM = 3 (Maior falta de coesão) na Classe A, segundo a tabela.

Por extensão, percebe-se que na Classe B, temos maior coesão.

Pares de métodos	Conjunto A	Intersecção dos Conj A
(f, g)	A(f) = {x} A(g) = {x}	{x}
(f,h)	A(f) = {x} A(h) = {x}	{x}
(g,h)	A(g) = {x} A(h) = {x}	{x}

15. Por que a métrica LCOM mede a ausência e não a presença de coesão? Justifique.

R: Pois mostra o uso das variáveis dentro de uma classe, ou seja, se acessada por vários métodos menor coesão terá.

16. Todos os métodos de uma classe devem ser considerados no cálculo de LCOM? Sim ou não? Justifique.

R: Não são utilizados os métodos getters e setters, nem os construtores.

17. A definição de complexidade ciclomática é independente de linguagem de programação. Sim ou não? Justifique.

R: Depende das palavras reservadas da linguagem que irão definir comandos de decisão.

18. Dê um exemplo de código com complexidade ciclomática mínima. Qual é essa complexidade?

R:
```
void main(){
    int x = 0;
    return;
}
```

19. A versão utilizando OO fica muito mais legível, facilitando a compreensão do sistema. Além disso, temos que perceber a divisão de tarefas que a abstração OO possui. Sendo assim, todo o sistema se torna mais facilmente mantido, além de trazer certa facilidade para novas implementações.