1) Escreva um pequeno resumo estendido (aproximadamente 5000 caracteres)

que descreva e comente sobre cada um dos itens abaixo. O texto deverá ser

feito com suas palavras sem cópias da internet. Ou seja, estude o conteúdo e

crie seu resumo.

a) Classe: A classe é um ambiente onde serão elaborados os codigos de uma meneira geral,

para enfim podermos classificá-las para melhor entendimento e funcionamento do codigo.

b) Objeto: O objeto é de certa forma realmente um objeto na qual ele terá suas caracteristicas

(atributos), e suas funcionalidades ou objetivos(métodos), que serão organizados seus atributos

e metodos de forma a condizer com a realidade para otimizar a escrita repetitiva, entendimento

e otimização de código.

c) Método e atributo: Os métodos serão as funcionalidades/funções/tarefas que o objeto poderá

executar durante sua funcionalidade; Os atributos por sua vez serão as caracteristicas/adjetivos/

qualidades do objeto onde mesmo objetos de mesma classe poderão ter diferente caracteristicas, ex:

um objeto carro, pode ter diferentes cores, motor, placa, etc.

d) Encapsulamento: O encapsulamento é uma maneira de proteção de codigo, pois vários objetos

têm seus dados públicos ou pessoais/reservados, nas quais não é desejável que o usuário possa

alterar livremente, pois assim usuários poderiam "bagunçar" o programa.

e) Visibilidade de atributos e métodos (public , private e protected): A visibilidade é justamente

a aplicação do conceito de encapsulamento, onde há dados de "segurança baixa" onde usa-se a palavra

"public" onde qualquer parte do projeto poderá utilizar de todas as funcionlidades da classe,

"segurança média" onde utilizamos "protected" onde as funcionalidades das classes podem ser utilizadas

quando houver um codigo no mesmo pacote e "segurança alta" onde seus atributos e metodos serão restritos

apenas a modificação na própria classe.

f) Herança: A herança por sua vez é utilizada para otimização de escrita de código, onde a partir

de classes amplas nas quais não possuem uma especificidade, podemos criar classes com os mesmos

atributos e metodos iguais ao da classe "pai" e criar novos atributos e metodos da classe "filho"

na qual será uma classe mais especifica, ex: "classe pai veiculo" e "classe filho carro/moto/onibus".

g) Polimorfismo: O polimorfismo é a capacidade de um objeto se comportar de várias formas dependendo

de como será utilizado no contexto em específico, ex: eu posso ter na classe pai veiculo o metodo

ligar, mas nas classes filhas carro e moto esse metodo pode se comportar diferente, pois no carro

basta girar a chave, enquanto que na moto deve-se girar a chave e apertar o botão de partida.

h) Classes abstratas: As classes abstratas são classe com apenas o objetivo de gerar filhos,

onde o objeto com o nome da classe na realidade não existe, e sim suas classes mais especificas,

ex: a classe animal sendo abstrata e a classe cachorro sendo o filho que terá comportamentos específicos

onde até mesmo a classe cachorro pode ser tratada como abstrata para classes filhas sendo as raças

específicas de cada cachorro.

i) Construtores: Os construtores serão responsáveis por especificar dados que serão atribuídos a

cada objeto diferente no momento de sua criação, para que não seja possível alterá-los durante o código,

gerando assim uma falha de segurança.

j) Get e Set: Get e Set são métodos presentes em objetos, nas quais garantem a segurança de atributos

protegidos ou privados, na qual podemos atribuir um valor com o set e ler um valor com o método get.

k) Sobrecarga de métodos: A sobrecarga é uma maneira de utilizar métodos semelhantes, porém de maneira

diferente na mesma classe, podendo ter mais parametros, tipos diferentes de parametros e até mesmo

ordem diferente, ex: public int soma(int a, int b) , public float soma(float a, float b).

l) Sobrescrita de métodos: A sobrescrita é como o nome sugere uma escrita por cima, que significa

que um classe filho herdara um metodo do pai e com mesmo nome e parametros, porém ela poderá ter

mais funcionalidades ou uma funcionalidade diferente/mais especifica da classe filho, ex: a classe

animal pode conter um metodo fazer barulho, na qual gera um barulho aleatório, e na classe filho

cachorro o metodo fazer barulho pode gerar um latido "au au".

m) Palavras reservadas (super, this e final): A utilização do super é para podermos herdar um

construtor do pai, pois o construtor deve ter o nome da sua classe, porém o filho deve ter um

nome diferente de uma classe já existente, logo para o construtor ter o mesmo uso que o construtor

de seu pai utilizamos o super();, ja a palavra this serve para que a variavel estabelecida localmente,

seja no construtor ou no metodo não seja atribuida localmente, e sim ao atributo presente na classe,

já a palavra final tem a utilização oposta da classe abstract, já que ela estabelece que uma

classe não possa ter mais filhos, logo ninguém pode herdar desta classe.

n) Relação de objetos: “ter”, “usar” e “ser”: A relação dos objetos de "ter" é a parte onde uma

classe de objeto tem nele presente outra classe de objetos, a parte de "usar" é quando dentro de

um objeto usamos outro objeto para realizar funções ou metodos, a parte "ser" é quando o objeto esta

pronto para ser utilizado.