Exercícios – Orientação a objetos em C++:

1- Verdadeiro ou falso

a- ( ) Classes são tipos de dados definidos pelo usuário. Elas podem conter itens de dados e funções para operar esses itens.

b- ( ) Um objeto é uma variável de um tipo **classe**.

c- ( ) As palavras-chave **public** e **private:** especificam a visibilidade dos membros de uma classe.

d- ( ) Todo construtor pode ter um valor de retorno, e pode receber qualquer número de argumentos.

e- ( ) Os membros **static** são criados uma única vez para a classe como um todo, não importando o número de objetos declarados. Esses membros são visíveis a todos os objetos de sua classe.

f- ( ) Membros privados definem a interface com a classe e membros públicos, a atividade interna da classe.

g- ( ) Os membros públicos de um objeto são acessados por meio do operador ponto. Este operador liga um objeto a um membro dele.

h- ( ) Numa classe pode haver mais de um construtor de mesmo nome.

i- ( ) Numa classe pode haver mais de um destrutor de mesmo nome.

j- ( ) Um construtor não pode receber argumentos

k- ( ) Um destrutor não pode receber argumentos.

l- ( ) O processo de herança permite a uma classe herdar todas as características de outra classe

m- ( ) A classe **derivada** pode incluir características próprias, além das adquiridas por herança. Por meio de heranças, é possível ampliar as propriedades de classes existentes e adaptá-las a novas situações.

n- ( ) Dados ou funções de uma classe-base precedidos da palavra **protected** podem ser acessados por uma classe derivada, e também por objetos da classe derivada.

o- ( ) Uma classe pode ter derivação pública ou privada de uma classe-base. Os objetos de uma classe de derivação pública podem acessar os membros públicos da classe-base, enquanto os objetos de uma classe de derivação privada não podem.

p- ( ) Uma classe derivada não pode servir de base para outra classe.

2- A relação entre classes e objetos é a mesma existente entre:

a) Tipos básicos e variáveis desses tipos
b) Variáveis e funções
c) Uniões e estruturas
d) Estruturas e funções

3- Na definição de uma classe, os membros designados como privados:

a) São acessados por qualquer função do programa
b) São protegidos de pessoas não autorizadas
c) São acessados por qualquer função-membro da classe
d) São acessados pelos membros públicos da classe

4- Métodos são:

a) Classes
b) Dados-membro
c) Funções-membro
d) Chamadas a funções-membro

5- Mensagens são:

a) Classes
b) Dados-membro
c) Funções-membro

d) Chamadas à funções-membro

6- O nome de um construtor é sempre o mesmo _____.

7- O nome de um destrutor é sempre o mesmo _____ precedido por _____.

8- Assuma a seguinte definição:

```
class facil {
        private:
                int x, y, z;
        public:
                static int p;
                facil() {}
                ~facil() {}
};
```

Como podemos atribuir 5 ao membro p?

a) p = 5;
b) facil.p=5;
c) facil a; a.p=5; (dentro do main()). Antes do main() escrever: int facil::p;)
d) facil::p=5; (dentro do main()). Antes do main() escrever int facil::p;)
e) int facil::p=5; (após definição da classe e antes do main())

9- Defina uma classe de nome ALUNO com dados privados para armazenar o primeiro nome do aluno, a série e o grau. Inclua duas funções públicas: uma para solicitar os dados para o usuário e outra para imprimir os dados.

a) Escreva uma instrução que declare um objeto chamado alu1 da classe ALUNO.
b) Escreva uma instrução para executar a função que solicite os dados de entrada para o usuário.
c) Escreva uma instrução para executar a função que imprima os dados digitados.
d) Inclua um membro *static* privado para contar o número de alunos cadastrados.
e) Inclua um construtor que incremente o contador de alunos cadastrados.
f) Inclua um destrutor que decremente o contador de alunos cadastrados.
g) Inclua uma função pública que imprima o número de alunos cadastrados.
h) Escreva a instrução necessária para declarar uma matriz de 10 objetos da classe ALUNO.
i) Preencha os dados dessa matriz e exiba-os.

10- Exercício maior idade: Implemente um programa no qual o usuário deverá informar o nome e a idade de três pessoas. O programa deverá informar o nome da pessoa que possuir a maior idade.

Regras que deverão ser seguidas para a implementação do algoritmo:

- É obrigatório o uso de classe para representar uma pessoa e a mesma deverá possuir como propriedades (características) um nome e uma idade.
- A classe também deverá possuir um método chamado ExibirDados. Esse método deverá exibir o nome e a idade da pessoa em questão.
- Ao implementar a classe é obrigatório implementar dois construtores: (Sobrecarga), um que não recebe parâmetro algum e outro que irá receber o nome e a idade de uma pessoa.

11- Área de um triângulo:

Crie um programa que exiba para o usuário qual é a área de um triângulo. Para implementar esse programa, você deverá seguir as seguintes regras:

- O usuário deverá informar para o programa o valor da base e altura do triângulo.
- É obrigatório criar/utilizar uma classe para representar o triângulo.
- A base e a altura informada pelo usuário deveram ser representadas na classe como propriedades.
- O objeto deverá possuir uma propriedade para exibir o valor da área do triângulo
- O objeto deverá possuir um método que exiba os dados de todas as suas propriedades.

12- Com relação aos conceitos de orientação objeto, existe uma característica que faz com que detalhes internos do funcionamento dos métodos de uma classe permaneçam ocultos para os objetos e que por conta dessa técnica, o conhecimento a respeito da implementação interna da classe é desnecessário do ponto de vista do objeto, uma vez que isso passa a ser responsabilidade dos métodos internos da classe. A característica apresentada se refere a:

a) Encapsulamento
b) Polimorfismo
c) Abstração
d) Herança

13- Criar uma calculadora usando conceitos de orientação a objetos. Apresentar um método para cada operação matemática e perguntar ao usuário qual operação ele deseja. Terminar o programa quando o usuário digitar -1 na operação desejada. Usar o comando switch() para selecionar a operação desejada.

14- Herança é um processo que permite:

a) A inclusão de um objeto dentro de outro;
b) Transformar classes genéricas em classes mais específicas
c) Adicionar propriedades a uma classe existente sem reescrevê-la
d) Relacionar objetos por meio de seus argumentos

15- Para derivar uma classe de outra já existente, deve-se:

a) Alterar a classe existente
b) Usar o código-objeto da classe existente
c) Reescrever a classe existente
d) Nenhuma das anteriores

16- Para que membros de uma classe-base possam ser acessados por membros da classe derivada, eles devem ser:

a) public
b) protected
c) private
d) todas as anteriores.

17- Qual das seguintes instruções executa a função-membro base(f), da classe-base BAS, a partir de uma função membro da classe derivada DERIV?

a) basef();
b) BAS::basef();
c) DERIV::basef();
d) BAS:basef();

18- Programa de herança

Parte 1: Imagine que você deva escrever um programa para armazenar veículos. Primeiro, crie a classe **Motor** que contém **NumCilindro** (int) e **Potencia** (int). Inclua um construtor sem argumentos que inicialize os dados com zeros e um construtor que inicialize os dados com os valores recebidos como argumento. Acrescente uma função para a entrada de dados, **getdata()**, e uma função que imprima os dados, **putdata().**

Parte 2: Escreva a classe **Veiculo** contendo **Peso** em quilos (int), **VelocMax** em km/h (int) e **Preco** em US$ (float). Inclua um construtor sem argumentos que inicialize os dados com zeros e um construtor que inicialize os dados com os valores recebidos como argumento. Acrescente uma função para a entrada de dados, **getdata()**, e uma função que imprima os dados, **putdata().**

Parte 3: Crie a classe **CarroPasseio** usando as classes **Motor** e **Veiculo** como base. Inclua **Cor** (string) e **Modelo (**string). Inclua um construtor sem argumentos que inicialize os dados com zeros e um construtor que inicialize os dados com os valores recebidos como argumento. Acrescente uma função para a entrada de dados, **getdata()**, e uma função que imprima os dados, **putdata().**

Crie um programa para testar as classes.