Lista 2 - Classes e Objetos

Técnicas de Programação 2019.2

Programação Estruturada 2019.2

linha curta

# Resumo

Programação Orientada a Objetos é um estilo de programação que tem o objetivo de tornar o pensamento sobre programação mais próximo do mundo real. Na Programação Orientada a Objetos, os objetos são unidades independentes e cada objeto tem sua própria identidade, assim como os objetos no mundo real. Uma maçã é um objeto; assim como uma caneca. Cada um tem sua identidade única. É possível ter duas canecas idênticas, mas elas ainda são objetos únicos e separados.

A lista está dividida em duas partes. Na primeira parte você vai responder questões de múltipla escolha que irão ajudar a revisar e entender melhor alguns conceitos de classes e objetos. Na segunda parte é hora de colocar a mão na massa e praticar como criar classes e objetos em C++.

Parte 1: <https://forms.gle/AKJx8yDNeriuzz5Q9>

Parte 2: <https://forms.gle/Vz9iJeewFmXYyNRU7>

Os formulários ficarão abertos para entrega até o dia 23/09/2019.

# Questões da Parte 2

Q1. Implemente uma classe Aluno com os atributos nome, notas, media, disciplinas. Faça um programa para testar a classe.

Q2. Crie uma classe chamada Data que inclua três atributos - um mês (tipo int), um dia (tipo int) e um ano (tipo int). Sua classe deve ter um construtor com três parâmetros para inicializar os três atributos. Para os efeitos deste exercício, presuma que os valores fornecidos para o ano e o dia estão corretos, mas assegure-se de que o valor do mês esteja no intervalo de 1-12; se não estiver, defina o mês como 1. Fornecer um método *set* e um método *get* para cada atributo. Forneça um método mostrarData que exibe o dia, o mês e o ano separados por barras oblíquas ( dd / mm / aaaa ). Escreva um programa de teste que demonstra as capacidades da classe Data.

Q3. Implemente uma classe chamada Conta que um banco poderia utilizar para representar contas bancárias dos clientes. Sua classe deve incluir um atributo de tipo float para representar o saldo da conta. A classe deve fornecer um construtor que recebe um saldo inicial e o utiliza para inicializar o atributo.

O construtor deve validar o saldo inicial para assegurar que ele seja maior ou igual a zero. Se não, o saldo deve ser configurado como 0 e o construtor deve exibir uma mensagem de erro, indicando que o saldo inicial era inválido.

A classe deve fornecer três métodos:

* O método *fazerDeposito* deve adicionar uma quantia ao saldo atual.
* O método *fazerSaque* deve retirar o dinheiro da conta e assegurar que a quantia de débito não exceda o saldo da conta. Se exceder, o saldo deve permanecer inalterado e a função deve imprimir uma mensagem que indica que não há saldo suficiente.
* O método *mostrarSaldo* deve mostrar o saldo atual do cliente.

Crie um programa que cria dois objetos Conta e testa os métodos da classe Conta.

Q4. Faça uma classe ContaCorrente que herda as funcionalidades da classe Conta. A nova classe deve possuir um atributo chamado *limite* que pode ser inicializado através do construtor da classe ou do método *setLimite*. Quando o método *mostrarSaldo*  for utilizado ele deve mostrar o saldo do cliente adicionado do limite concedido.

Q5. Crie uma classe chamada Fatura que uma loja de hardware pode usar para representar uma fatura para um item vendido na loja. Uma Fatura deve incluir quatro atributos - número de série (tipo string), uma descrição parcial (tipo string), uma quantidade do item que está sendo comprado (tipo int) e um preço por item (tipo float).

Sua classe deve ter um construtor que inicialize os quatro atributos.

Fornecer um método *set* e um método *get* para cada atributo.

Além disso, fornecer um método *faturaTotal* que calcula o montante da fatura (ou seja, multiplica a quantidade pelo preço do item) e, em seguida, retorna o valor da fatura.

Se a quantidade não for positiva, deve ser definida como 0.

Se o preço por item for não positivo, ele deve ser definido como 0. Escreva um programa de teste que demonstra as capacidades classe Fatura.

Q6. Crie uma classe chamada Empregado que inclua três atributos - um primeiro nome (tipo string), um sobrenome (tipo string) e um salário mensal (tipo float).

* Sua classe deve ter um construtor que inicialize os atributos de dados.
* Fornecer um método *set* e um método *get* para cada atributo.
* Se o salário mensal não for positivo, ajuste-o para 0.

Escreva um programa de teste que demonstra as capacidades classe Empregado. Crie dois objetos Empregado e exiba o salário anual de cada objeto. Em seguida, dê a cada Empregado um aumento de 10% e mostre novamente o salário anual de cada Empregado.

Q7. Escreva uma classe chamada Point2d. O Point2d deve conter dois atributos do tipo float: m\_x e m\_y, ambos inicializados com 0.0 caso o usuário não forneça os valores iniciais. Forneça os construtores e uma função de impressão.

O programa a seguir deve ser executado:

int main(){

Point2d inicio;

Point2d final(3.0, 4.0);

inicio.print();

final.print();

return 0;

}

Isso deve imprimir:

Point2d(0, 0);  
Point2d(3, 4);

Q8. Agora adicione um método chamado *distancia*, que usa outro Point2d como parâmetro, e calcula a distância entre eles. Dados dois pontos (x1, y1) e (x2, y2), a distância entre eles pode ser calculada como *sqrt ((x1 - x2) \* (x1 - x2) + (y1 - y2) \* (y1 - y2))*. O programa a seguir deve ser executado:

int main(){

Point2d inicio;

Point2d fim(3.0, 4.0);

inicio.print();

fim.print();

cout << "A distância entre os pontos é : "<<inicio.distancia(fim)<<endl;

return 0;

}

Isso deve imprimir:

Point2d(0, 0);  
Point2d(3, 4);

A distância entre os pontos é : 5

Q9. Escreva uma classe Maca e uma classe Banana derivadas de uma classe Fruta. A classe Fruta deve ter dois atributos: um nome e uma cor.

O programa a seguir deve ser executado:

int main(){

Maca a("vermelha");

Banana b;

cout << "A " << a.getNome() << " é " << a.getCor() << ".\n";

cout << "A " << b.getNome() << " é " << b.getCor() << ".\n";

return 0;

}

Isso deve imprimir:

A maçã é vermelha.

A banana é amarela.

Q10. Adicione uma nova classe ao programa anterior chamado MacaVerde, que herda da da classe Maca.

O programa a seguir deve ser executado:

int main(){

Maca a("vermelha");

Banana b;

MacaVerde c;

cout << "A " << a.getNome() << " é " << a.getCor() << ".\n";

cout << "A " << b.getNome() << " é " << b.getCor() << ".\n";

cout << "A " << c.getNome() << " é " << c.getCor() << ".\n";

return 0;

}

Isso deve imprimir:

A Maçã é vermelha.

A Banana é amarela.

A Maçã-Verde é verde.