Faculdade IMPACTA	**Programação Orientada a Objetos** **Atividade Contínua 02** **Grupos: Máximo 6 pessoas** **Data de Entrega: 21/09 (23:59)**

OBJETIVO:

Utilizar boas práticas na resolução de problemas que envolvem conceitos de classes, atributos, métodos e associação entre classes.

DESCRIÇÃO:

João confecciona placas por encomenda. Como o volume dos pedidos tem aumentado, ele precisa de uma aplicação que controle o cadastro de seus clientes e os pedidos realizados.

Quando ele recebe uma encomenda, João anota o nome do cliente, seu endereço completo e seu telefone.

Para a encomenda, ele registra: a altura e largura da placa, a frase a ser escrita, a cor da placa ("branca" ou "cinza"), a cor da frase ("azul", "vermelha", "amarela", "preta" ou "verde").

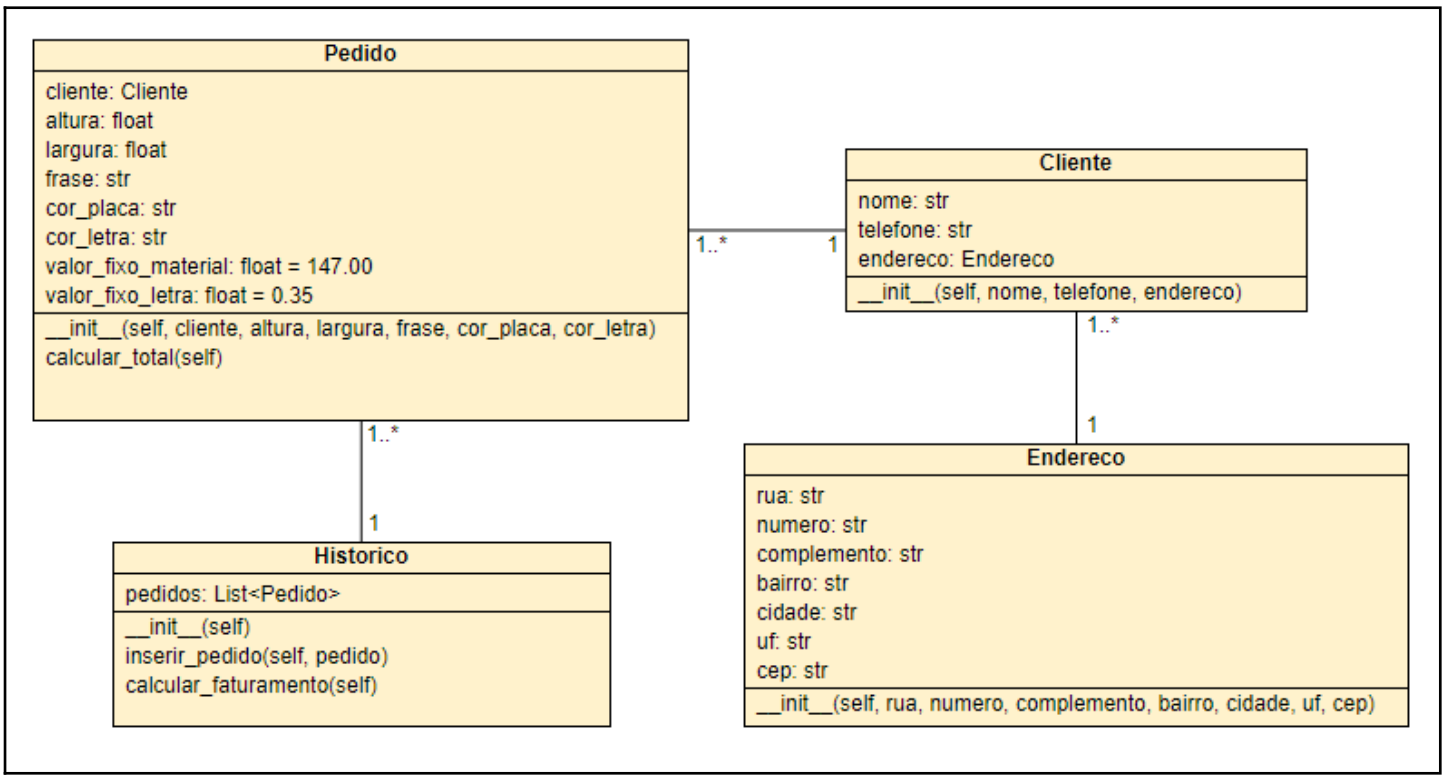
Com base nessas informações ele calcula manualmente o valor da placa, utilizando as seguintes fórmulas:

área = altura x largura
custo do material = área x R$ 147,00
custo do desenho = quantidade de letras x R$ 0,35 *(espaços devem ser desconsiderados)*
valor final da placa = custo do material + custo do desenho

João deseja que o sistema controle os pedidos e calcule automaticamente o valor final das placas. João deseja ainda manter um histórico de todos os pedidos realizados, a fim de verificar o seu faturamento.

A partir do cenário descrito e do diagrama de classes apresentado, implemente em Python as classes solicitadas, aplicando os conceitos de programação orientada a objetos.

DIAGRAMA DE CLASSES:



Classe Endereço:

Atributos

- **rua**
- **numero**
- **complemento**
- **bairro**
- **cidade**
- **uf**
- **cep**

Métodos

Não possui

Classe Cliente:

Atributos

- **nome**
- **telefone**
- **endereco**: objeto da classe Endereco

Métodos

Não possui

Classe Historico:

Atributos:

- **pedidos:** lista de objetos da classe Pedido. Definido no construtor como uma lista vazia.

Métodos:

- **inserir_pedido**: recebe como parâmetro um objeto Pedido e inclui na lista de pedidos
- **calcular_faturamento**: retorna o faturamento total dos pedidos (somatório do valor total de todos os pedidos)

Classe Pedido:

Atributos:

- **cliente:** objeto da classe Cliente
- **altura:** altura da placa
- **largura:** largura da placa
- **frase:** frase a ser escrita na placa
- **cor_placa:** cor da placa
- **cor_letra:** cor da letra
- **valor_fixo_material:** valor fixo do material. Definido no construtor (R$ 147,00).
- **valor_fixo_letra:** valor fixo de cada letra. Definido no construtor (R$ 0,35).

Métodos:

- **calcular_total:** retorna o valor total da placa.
  Para calcular o valor da placa, as seguintes fórmulas são usadas:
  área = altura x largura
  custo do material = área x R$ 147,00
  custo do desenho = quantidade de letras x R$ 0,35 *(espaços devem ser desconsiderados)*
  valor final da placa = custo do material + custo do desenho

CRITÉRIOS DE AVALIAÇÃO:

- 8.0 pontos: Implementação e execução correta do programa conforme as especificações acima e conforme ilustrado no diagrama de classes.
- 2.0 pontos: Formatação e organização do código de acordo com o padrão PEP8.

FORMA DE ENTREGA:

- Anexar na atividade do Classroom o arquivo ac02.py com a implementação das classes solicitadas.
- Pode ser entregue por apenas um aluno do grupo, mas não esqueça de colocar o nome de todos os integrantes no código do programa.

ATENÇÃO:

- Não serão aceitos trabalhos entregues em atraso.
- Não serão aceitos trabalhos com mais de 6 alunos no grupo.
- Não deve ser utilizado input dentro das classes.
- Entregar apenas o arquivo ac02.py.
- Arquivos em outros formados que não sejam .py, não serão aceitos (doc, pdf, txt, link para git, etc)
- O arquivo ac02_teste.py contém um conjunto de testes que pode ser utilizado para testar o funcionamento das classes implementadas.
    - Outros testes poderão ser realizados na correção da atividade, para verificar se a implementação está de acordo com o solicitado.
- Se for identificada cópia de trabalhos entre os grupos, a nota será zerada.