**Pontifícia Universidade Católica de Minas Gerais**
**Bacharelado em Engenharia de Software - Campus Lourdes**
**Programação Modular**
**Prof.** João Caram - **Semestre** 1/2025
**Exercício - Fundamentos de POO - Valor:** 1 ponto

Esta é uma atividade de revisão e nivelamento da disciplina ***Programação Modular***, com valor de 1 ponto. Seu objetivo é praticar os conceitos básicos da POO em Java, tendo sempre em vista os princípios SOLID e os aspectos de modularidade.

**ATENÇÃO: SOMENTE A QUESTÃO 3 É PONTUADA E TEM ENTREGA OBRIGATÓRIA.**

1) Faça o projeto de uma classe "Retangulo" para representar um retângulo que será desenhado na tela do Console. Este retângulo terá uma altura, uma largura, um caractere de borda e um deslocamento na tela definidos pelo usuário. O deslocamento indica quantas colunas de tela devem ser saltadas desde a borda esquerda da tela até a borda esquerda do retângulo. Somente o caractere de borda e o deslocamento podem ser modificados após a criação de um retângulo. Implemente sua classe e a utilize num código principal.

2) Use um diagrama de classes UML para modelar uma calculadora que consiga fazer as quatro operações matemáticas básicas. Implemente sua classe.

3) Projete e implemente uma classe "Hora" para ser utilizada em sistemas diversos. A hora será representada até o nível de segundos e deve obedecer às regras básicas:

- Um objeto "Hora" só pode armazenar estados válidos.
- Um objeto "Hora" pode ser incrementado em minutos, segundos ou horas. Esta operação não modifica o valor da "Hora" original, sendo seu objetivo retornar uma nova "Hora" válida.
- Um objeto "Hora" pode ser comparado com outro para verificação de qual horário está mais adiante.

4) Em um curso técnico, que tem uma carga total de 40 aulas, cada aluno é avaliado pelos seguintes critérios:
a) Quatro notas de exercícios de 0 a 100, com peso de 20% na nota final.
b) Duas notas de prova de 0 a 100, com peso de 60% na nota final.
c) Um trabalho prático de 20 pontos, a ser somada com as anteriores para calcular a nota final.
Para ser aprovado, o aluno precisa atingir pelo menos 60 pontos na nota final, bem como cumprir no mínimo 75% de frequência às aulas. Projete e implemente uma classe "Aluno" que atenda aos requisitos descritos.

5) Uma partida de basquete é disputada por duas equipes. A partida é dividida em 4 quartos de 10 minutos e será considerada vencedora a equipe que somar mais pontos na soma das pontuações dos quartos. Em caso de empate, serão disputadas quantas *prorrogações* de 5 minutos forem necessárias, até que uma equipe supere a outra em pontuação. Modele e implemente uma classe "PartidaBasquete" que registre os placares das equipes e, indique quem venceu a partira e quando solicitada, forneça o placar no formato abaixo:

```
           Q1    Q2    Q3    Q4    FINAL
EQUIPE1    25    19    26    21    91
EQUIPE2    22    25    28    25    100
```

No caso de prorrogações, devem aparecer no formato acima como "P1", "P2" etc., após o Q4.

6) Uma *mensagem criptografada* é um tipo de mensagem que cria uma *cifra* a partir de uma mensagem de texto. A *cifra* é um texto que esconde a mensagem original e que pode, posteriormente, ser *decifrada* para mostrá-la. Utilizando seus conhecimentos de POO, implemente uma classe "Cifra" que usa o seguinte algoritmo: a mensagem original é dividida em blocos de N caracteres. A cifra será obtida lendo-se a posição correspondente de cada bloco, sendo inserido ainda um "`*`" a cada mudança de posição. Por exemplo, se a mensagem original for "`mensagem secreta`" e o bloco tiver tamanho 5:

```
mensa
gem s
ecret
a
```

A cifra seria "`mgea*eec *nmr *s e *ast *`" (atenção para os espaços em branco). A classe também deve fornecer a mensagem original, mediante um pedido do usuário informando o tamanho do bloco utilizado.