3ª VERIFICAÇÃO DE APRENDIZAGEM

Instruções gerais:
Salve as implementações a cada modificação, caso aconteça alguma falha de energia o trabalho será preservado. Lembre-se que uma vez removido o arquivo do eclipse, seu contendo será perdido.

A prova é prática e deverão ser entregues no SIGAA: [3ª V.A. - Prova Prática] em Semana 15. Deverá ser entregue todo o projeto Eclipse contendo os códigos-fonte implementados em Java. *A Nota máxima desta prova é de 10,0 pontos.* A Pontuação da Prova está distribuída entre os conceitos vistos na disciplina MPOO CCMP5012.

A composição da nota da 3ª V.A. obedecerá a seguinte pontuação: 3ª V.A. = 100% * Prova Prática

Criação de Projeto e sua organização (0,5 ponto)

1) No Eclipse limpe todos os projetos existentes.
   - Crie um novo **projeto chamado**
     br.3vampoo.edu.NomeSobrenome
     Este deverá ter uma **pasta de pacotes** chamada **sistemaBudega** contendo todos os arquivos necessários para as respectivas questões.
   - Ao finalizar a prova **compacte o projeto** contendo toda a codificação do projeto (arquivos texto, *bytecodes* e imagens) e envio-o no SIGAA:
     [3ª V.A. - Prova Prática] em Semana 15
     O SIGAA aceitará submissões até às 22h

O Problema:

A Empresa "MPOO Development Edu" recebeu do proprietário da "Budega MPOO" a demanda de inclusão de novas funcionalidades no seu sistema. Dessa vez, ele quer um sistema que trate de descontos e controle dos produtos que estão a vencer. Sendo assim, você assumirá o papel de programador e apresentará um protótipo que atenda a essa demanda, mostre que você está preparado no desenvolvimento de sistemas Orientado a Objetos em Java. Para isso, tem-se o diagrama de classes com a modelagem da solução (Apêndice A) e a descrição de GUI's e funcionalidades do sistema a serem apresentadas.

Descrição:

2) É descrição do sistema:

Design Pattern: MVC (0,5 ponto)

a) Utilize os padrões de projeto: Model-View-Controller (MVC)

Classes, atributos e métodos construtores (1,0 ponto)
Encapsulamento e métodos de acesso (0,5 ponto)
static e no-static (0,5 ponto)
abstract (0,5 ponto)
Herança (0,5 ponto)
Agregação e Cardinalidade por ArrayList (0,75 pontos)
Composição (0,5 ponto)
Definição de métodos e suas implementações (1,0 ponto)

b) Implemente as definições do modelo do diagrama de classes do Apêndice A. É descrição:
   - BaseDados é uma classe que contém uma estrutura de dados para Produtos e métodos de manipulação dessa estrutura de dados;
   - codBarras é chave primária de produto, mas todo produto tem um id que é do tipo auto incrementável;
   - Produto contém Estoque, sendo esta uma relação de composição;
   - Produto contém um Fornecedor que é uma Pessoa;

Interface (1,0 ponto)

c) O sistema é composto por 3 interfaces. Mostre que você tem potencial de ser um programador de MPOO Development Edu apresentando suas utilizações e distinção quando estas possuem definições abstratas e estáticas:
   - VencidoInterface possui método que possibilita verificar se um produto está vencido. É sua codificação:

```
public boolean isVencido() {
   Calendar c = Calendar.getInstance();
   c.set(Calendar.DATE, this.validade.getDate());
   c.set(Calendar.MONTH, this.validade.getMonth());
   c.set(Calendar.YEAR, this.validade.getYear());
   Date validadeProduto = c.getTime();
   if (validadeProduto.before(new Date()))
       return true;
   return false;
}
```

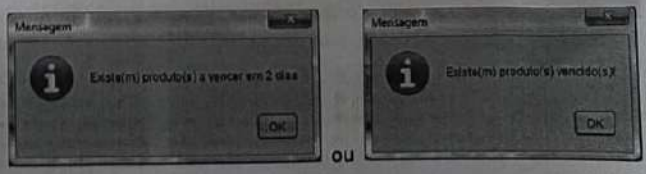
   - AvencerInterface possui método que informa quantos dias um produto estar de vencer. É sua codificação:

```
public static long diasAVencer(Produto produto){
   LocalDateTime atual = LocalDateTime.now();
   LocalDateTime produtoDate =
         LocalDateTime.of(produto.getValidade().getYear(),
         produto.getValidade().getMonth(),
         produto.getValidade().getDate(),0,0);
   long diasaVencer =
          ChronoUnit.DAYS.between(atual.toLocalDate(),
                                  produtoDate.toLocalDate());
   return (diasaVencer);
}
```

   - CPFValidadorInterface possui método para validar um CPF. No sistema este não deverá ter corpo codificado, apenas criada a sua definição com retorno booleano verdadeiro.

Thread: (1,0 ponto)

d) O sistema possui um robô (Thread) que a cada 24 horas (VERIFICACAO_DIAS = 86400000) executam-se as verificações para saber se há produtos a vencer ou vencidos. Por estratégia de Budega MPOO as verificações automáticas avaliam se um produto está a 2 dias de vencer (PRAZO_DIAS = 2). Quando encontrado produtos deve-se exibir:

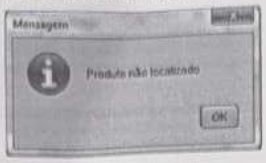


Mas atenção: não se deve confundir uma verificação automática realizada por um robô *started* em uma aplicação com a possibilidade de verificar a existência de produtos a vencer pela utilização de uma GUI! Quando encontrado um produto:

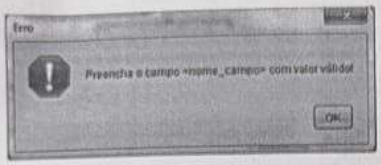
Métodos e Manipulação de Exceção (0,75 ponto)

e) Faça a devida manipulação de Exception quando:

- O método buscarProduto(String codBarras) de BaseDados não encontrar um produto na base. Deve-se tratar por throws/throw a exceção ProdutoNaoExisteException.
  Como tratamento deve-se exibir as seguintes mensagens de alerta ao usuário:



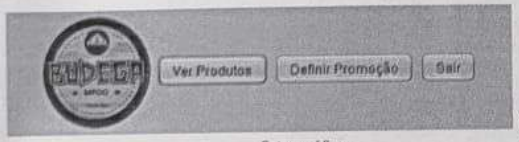
- Em chamadas de métodos exigir try/catch.
- Os campos numéricos devem ser numéricos. Se um usuário confirmar uma ação que faça uso de um campo numérico e este tenha digitado um texto, então se deve tratar a exceção NumberFormatException e em seu tratamento exibir mensagem alertando o usuário.



Componentes Gráficos (1,5 pontos)

É descrição das GUI's do sistema da Budega:

f) Possui a tela de opções:

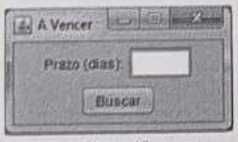


(450x110)

EstoqueView

Utilize:
`UIManager.setLookAndFeel("javax.swing.plaf.nimbus.NimbusLookAndFeel")`

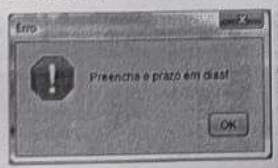
g) A opção "Ver Produtos" permite verificar os produtos a vencer em um prazo a ser informado:



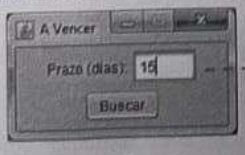
(190x100)

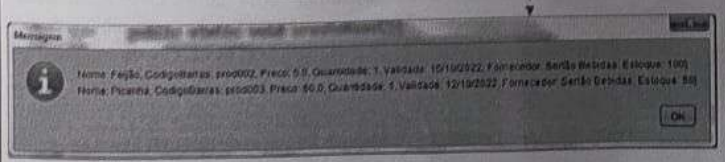
AVencerView

Caso seja informado um prazo inválido (valor negativo ou não numérico) para os produtos a vencer por este prazo, deve-se exibir a mensagem de alerta ao usuário:



Caso informado o valor de dias válido, exibe-se os produtos que irão vencer dentro desse prazo, mas utilize toString() para recuperá-los:

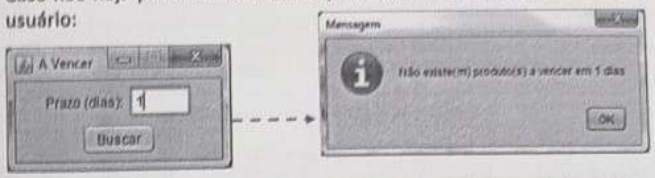




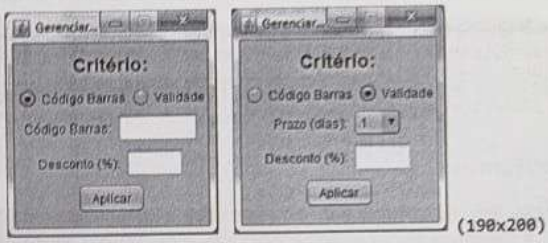
É codificação de toString() de Produto::

```
public String toString() {
    return "Nome: " + nome + ", CodigoBarras: " + codigoBarras
        + ", Preco: " + preco + ", Quantidade: " + quantidade
        + ", Validade: " + validade.getDate() + "/" +
                           validade.getMonth()+ "/" +
                           validade.getYear()
        + ", Fornecedor: " + fornecedor.getNomeFantasia()
        + ", Estoque: " + estoque.getQuantidade() + "]";
}
```

Caso não haja produto a vencer no prazo informado, deve-se exibir ao usuário:



h) A opção "Definir Promoção" permite ao gestor da Budega MPOO habilitar promoções aos produtos por código de Barras ou que estão a vencer. Para isso, têm-se as telas, respectivamente:



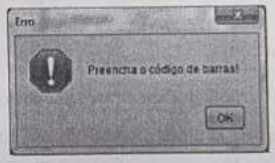
(190x200)

DescontoView

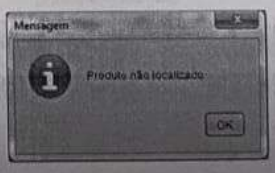
i) Observe que os campos para código de barras e prazo (dias) são exibidos a partir da seleção das opções dispostas nos JRadioButtons Código de Barras e Validade, respectivamente.

*Quando a opção **Código de Barras** está selecionada:*

- Se ao tentar aplicar um desconto em um produto indicado por Código de Barras vazio, exibe-se a mensagem de alertar ao usuário:

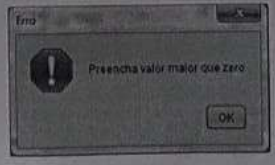


- E se um produto não for localizado, deve-se exibir a mensagem de alerta ao usuário. Utilize buscarProduto(String codBarras) de BaseDados e ProdutoNaoExisteException.

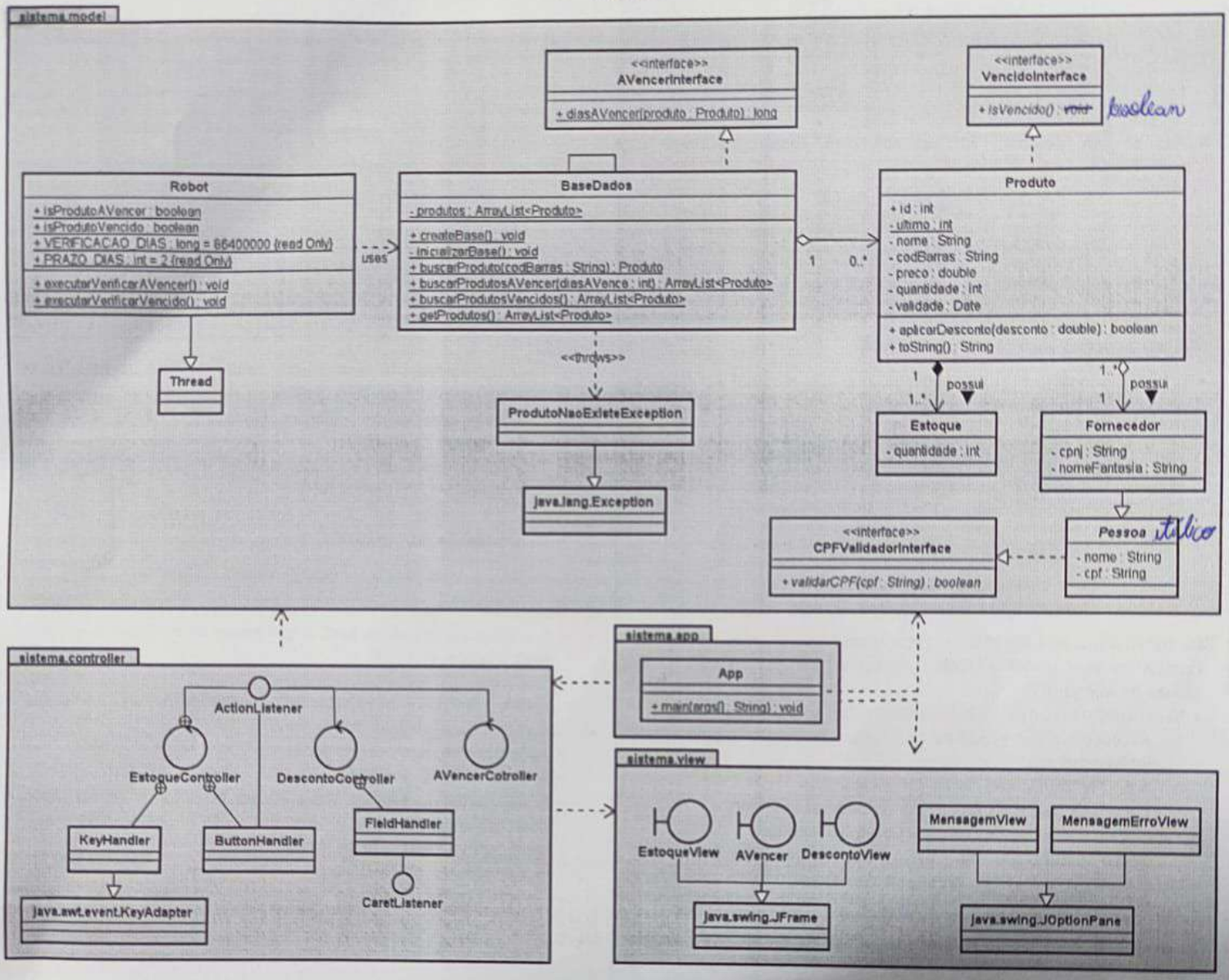


*Quando **qualquer uma das opções** (Código Barras ou Validade) está selecionada:*

- Caso seja informado um valor negativo para um desconto a ser aplicado, deve-se exibir a mensagem de alerta ao usuário:

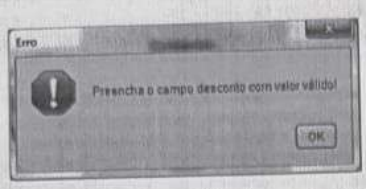


APÊNDICE A

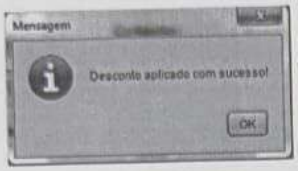



```java
public class Window{
    JButton button = new JButton();
    public static void main(String[] args) {
        Window window = new Window();
        window.button.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                JOptionPane.showMessageDialog(null, "Boa Prova!");
            }
        });
        window.button.doClick();
    }
}
```

- Ou se preencher em desconto valor não numérico, deve-se exibir a mensagem:

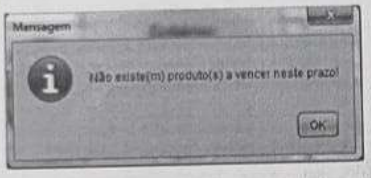


- Mas se um desconto for aplicado com sucesso, exibe-se a confirmação:



*Quando a opção **Validade** está **selecionada**:*

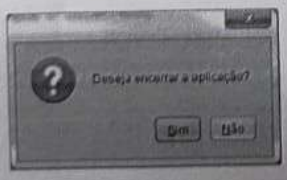
- Se um desconto tentar ser aplicado, mas não existem produtos a vencer no prazo informado, então:



Tratamento de Eventos (1,5 pontos)
Design Pattern: Bônus: Adapter (0,25 ponto)

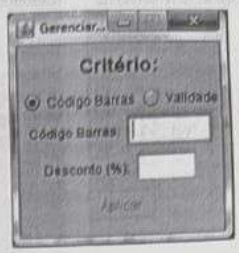
j) Os tratamentos dos eventos das telas (sistema.view) devem estar nos respectivos controladores (sistema.controller) conforme diagrama de classes do Apêndice A:
   - Para EstoqueView tem-se EstoqueController, sendo:
     - As opções "Ver Produtos" e "Definir Promoção" como classe interna privada.
     - A opção "Sair" como classe interna anônima
     - Encerrada pela tecla Escape por classe interna privada.
   - Para AVencerView tem-se AVencerController, sendo:
     - Botão Buscar tratado na própria classe (classe realiza interface)
   - DescontoView tem-se DescontoController, sendo:
   - Botão Aplicar tratado na própria classe (classe realiza interface).

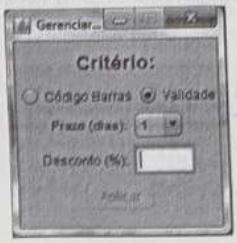
k) As opções de encerrar do sistema (opção Sair de EstoqueView e tecla Esc) devem ter confirmação do usuário:



Bônus: CaretListener (0,5 ponto)

l) O botão Aplicar só é liberado quando todos os campos estão preenchidos:





Instâncias (0,5 pontos)

m) Na Base de Dados:
   - Criação dos produtos do Fornecedor de nome Zé Santos e nome fantasia Sertão Bebidas, cpf 111.111.111-11 e cnpj 01.000.000/0001-11:
     - 100 unidades de Pinga Sertão, codBarras prod001, R$ 1.000,00, 800 ml, validade: 25/10/2025;
     - 100 unidades de Feijão, codBarras prod002, R$ 5,00, 1 kg, validade: 15/10/2022;
     - 50 unidades de Picanha, codBarras prod003, R$ 60,00, 1 kg, validade: 12/10/2022; e
     - 10 unidades de Danone, codBarras prod004, R$ 12,00, 1 L, validade: 01/09/2022.

Instâncias e chamadas de métodos (0,25 pontos)

n) Deve-se ter em App:
   - A criação de instâncias MVC e chamadas de métodos (base, telas, controladores e thread).

Obs.: Realize a devida verificação das funcionalidades (depuração) das GUI's visando validar suas funcionalidades, para:
- Ações de componentes e teclado;
- Entrada de dados inválidos;
- Manipulação de produto (consulta de produtos a vencer, aplicar desconto em produto); e
- Verificação de execução de Thread.