**Universidade Federal de Campina Grande – UFCG**

**Centro de Engenharia Elétrica e Informática – CEEI**

**Unidade Acadêmica de Sistemas e Computação – UASC**

Disciplina: Laboratório de Programação 2

# Laboratório 03

O objetivo do Lab 3 é:

1) Praticar a criação de classes e objetos;
2) Praticar o uso de arrays para armazenamento de vários objetos;

A submissão e as respectivas instruções estão disponíveis no Canvas.

## Descrição da Implementação:

Com o sucesso do LojaoP2, seu chefe decidiu criar um supermercado EconomizaP2. Esse supermercado vende produtos de diversos tipos. São eles: alimentos, limpeza, ferramentas e eletrodomesticos. Devido ao sucesso do seu código na loja de R$1.99 você foi contratado para implementar o código para realizar o cadastro, a venda e o balanço do supermercado. Siga as instruções abaixo para implementar as entidades responsáveis pelo funcionamento do supermercado.

## Passo 1: Implementação do Menu de Entrada

Neste passo serão praticadas operações de entrada e saída em java.
Seguindo o padrão dos **laboratórios anteriores**, implemente um menu no qual o usuário digita a **opção da operação** desejada. A opção será lida do teclado usando a classe java.util.Scanner. Lembre-se de deixar o main modularizado por meio de métodos. Utilize o modelo abaixo.

```
= = = = Bem-vindo(a) ao EconomizaP2 = = = =
Digite a opção desejada:
1 - Cadastrar um Produto
2 - Vender um Produto
3 - Imprimir Balanço
4 - Sair

Opção:
```

## Passo 2: Implementação do Cadastro do Produto

Neste passo serão praticadas a criação de classes e de objetos. Além disso também vamos praticar a inicialização e manutenção de tamanhos e conteúdo de array (ou seja, criar arrays dinâmicos).
Ao selecionar o **cadastro do produto**, o gerente irá criar produtos no seu supermercado. Lembre que esses produtos devem ser **armazenados** para que depois você possa acessá-los. Utilize o seguinte menu para a criação de produtos:

```
= = = = Cadastro de Produtos = = = =
Digite o nome do produto: Sabonete
Digite o preço unitário do produto: 5,98
Digite o tipo do produto: Limpeza

Sabonete cadastrado com sucesso.

Deseja cadastrar outro produto? Sim

Digite o nome do produto: Lata de Leite Ninho
Digite o preço unitário do produto: 11,20
Digite o tipo do produto: Alimento
Lata de Leite Ninho cadastrado com sucesso.

Deseja cadastrar outro produto? Nao
```

## Passo 3: Implementação da Venda de Produtos

Neste passo será praticado o uso de objetos e também a implementação de equals de objetos.
Ao selecionar a Venda de produtos, o usuário **digita o nome do produto** que deseja vender. Se o produto estiver cadastrado, ele então informa quantas **unidades** daquele produto deseja vender. Só então **é informado o total que deve ser pago pelo cliente**. Lembre de armazenar o valor pago para que depois seja impresso no balanço. Por exemplo:

```
= = = = Venda de Produtos = = = =
Digite o nome do produto: Panela
==> Panela nao cadastrada no sistema.
Deseja vender outro produto? Sim

Digite o nome do produto: Sabonete
==> Sabonete (Limpeza). R$5.98

Digite a quantidade que deseja vender: 10
==> Total arrecadado: R$ 59.80
Deseja vender outro produto? Nao
```

## Passo 4: Imprimir o Balanço

O objetivo desse passo é praticar a criação e uso de método toString.
Nessa funcionalidade você deve imprimir duas informações: todos os **produtos cadastrados**, e o **total de dinheiro** arrecadado com vendas até o momento. Então, pelo nosso exemplo será impresso o seguinte:

```
= = = = Impressao de Balanco = = = =
Produtos cadastrados:
    1) Sabonete (Limpeza). R$5.98
    2) Lata de Leite Ninho (Alimento). R$11.20

Total arrecadado em vendas: R$ 59.80
```

## Passo 5: Modificação

Neste passo o aluno irá modificar classes e produzir alterações em código OO.
Com o sucesso do seu sistema, o seu chefe pede que você permita o controle dos produtos de acordo com uma **quantidade especificada durante o cadastro de produtos**. Por exemplo, ao receber os produtos dos fornecedores, são entregues 50 sabonetes, e 100 latas de leite ninho. Portanto o cadastro deve alterado para que o cadastro prossiga da seguinte forma:

```
= = = = Cadastro de Produtos = = = =
Digite o nome do produto: Sabonete
Digite o preço unitário do produto: 5,98
Digite o tipo do produto: Limpeza
Digite a quantidade no estoque: 50
50 “Sabonete” cadastrado(s) com sucesso.

Deseja cadastrar outro produto? Sim

Digite o nome do produto: Lata de Leite Ninho
Digite o preço unitário do produto: 11,20
Digite o tipo do produto: Alimento
Digite a quantidade no estoque: 100
100 “Lata de Leite Ninho” cadastrado(s) com sucesso.

Deseja cadastrar outro produto? Nao
```

Observe que pode ocorrer agora uma situação onde
Isso também implica numa **modificação no Imprime Balanço**, pois agora **dois novos** elementos devem ser impressos: A quantidade de produtos restante no estoque, e o total que ainda pode ser faturado pelo supermercado. Considerando que foram vendidos 10 sabonetes, ao imprimir o balanço, temos, por exemplo:

= = = = Impressao de Balanco = = = =
Produtos cadastrados:

1) Sabonete (Limpeza). R$5.98 Restante: 40
2) Lata de Leite Ninho (Alimento). R$11.20 Restante: 100

Total arrecadado em vendas: R$ 59.80
Total que pode ser arrecadado: R$ 1359.20

Note também que não podemos vender uma quantidade maior do que a restante. Por exemplo, se ao realizar a venda, o usuário deseja vender 50 sabonetes (tendo apenas 40 restantes), seu sistema deve informar que **não há Sabonete suficiente**, como ilustrado na execução abaixo:

= = = = Venda de Produtos = = = =
Digite o nome do produto: Panela
==> Panela nao cadastrada no sistema.
Deseja vender outro produto? Sim

Digite o nome do produto: Sabonete
==> Sabonete (Limpeza). R$5.98

Digite a quantidade que deseja vender: 50
Não é possível vender pois não há Sabonete suficiente.

Deseja vender outro produto? Nao

## Dicas para ajudar na implementação:

Arrays dinâmicos:

Note que em nenhum momento o usuário define quantos produtos serão cadastrados. Portanto, implemente seu array de forma que ele possa crescer dinamicamente à medida que a quantidade de produtos aumenta. Comece com 5 produtos, e ao ultrapassar o limite (ao adicionar o sexto produto):

1) Crie um novo array com o dobro do tamanho atual;
2) Copie as informações atuais no novo array;
3) Atualize seu contador para que os novos produtos sejam inseridos no final do array.

Um exemplo:

```
if (quantidadeDeProdutos == produtos.length) {
     Produto[] novoArray = new Produto[quantidadeDeProdutos * 2];
     copiaProdutosParaNovoArray(novoArray);
     produtos = novoArray;
}
produtos[quantidadeDeProdutos++] = new Produto(nome, preco, tipo);
```

Modularização:

Cada Menu abre para uma funcionalidade diferente. Lembre dos quatro prinípios de boa Programação Orientada a Objetos para manter um bom design de sua classe: Creator, Expert, Coesão, Acoplamento. Nem sempre cada funcionalidade deve ser expressa como uma classe diferente. Saiba agrupar suas funcionalidades pensando:

1) Coesão: Faz sentido elas estarem nesse local (classe, método, etc.)?
2) Expert: Que informações a funcionalidade precisa como entrada para funcionar corretamente?
3) Creator: Estou criando meus objetos da forma correta? Será se estou garantindo a integridade do Objeto?
4) Acoplamento: Minhas entidades (variáveis, classes, métodos) estão muito dependentes umas das outras? Meu código está "amarrado"?

Além das funcionalidades diferentes, alguns trechos de código se tornam reusáveis. Por exemplo, algumas operações realizadas com o array podem ser encapsuladas em métodos.Se pergunte: que operações eu posso fazer nesse Array? Adicionar, buscar, aumentar o array, imprimir os elements, etc.

Quantidades de Produtos:

Note que criar um objeto de Produto para cada produto recebido do fornecedor diminui a eficência do seu programa pois muita memória é necessária. Imagine um supermercado que recebe milhares de exemplares de cada um dos milhares de tipos de produtos. Antes que você perceba, você acaba criando um milhão de objetos. Essa quantidade pode ser expressa de outra forma. Note que é necessário que essa quantidade reflita precisamente a quantidade de produtos antes e depois de realizar uma compra. Como seria essa variável de controle da quantidade de produtos restante?

Preços de Produtos:

Se queremos gerenciar os exemplares do mesmo tipo de produto em uma variável de controle da quantidade de produtos restante, guardaremos também um único preço para esses exemplares. Considere o último preço cadastrado como sendo esse valor.

## Como o seu lab será avaliado?

Serão considerados os seguintes critérios com suas respectivas notas:

- (2.0) Funcionalidade: observar se foram implementadas as funcionalidades pedidas

- (2.0) Legibilidade: observar, por exemplo, nomeação de métodos e variáveis, além da organização do código (endentação), lógica clara e ainda o bom uso de modularização.
- (4.0) Escolha e Criação das classes: lembrar que classes são entidades do sistema que encapsulam dados+comportamentos; devemos obedecer ao princípio da ocultação da informação. Que classes devem ser criadas?
- (1.0) Uso de objetos: atentar para instanciação e uso dos objetos a partir da chamada correta aos seus métodos
- (1.0) Uso de array

** atenção para o uso de “static” nesse lab, o que deve ser muito cuidadoso. Queremos que os alunos explorem uso de objetos, acima de tudo.

**Boa sorte e boa implementação!**