MÓDULO 05 – PRÁTICA DE LABORATÓRIO I

1 INTRODUÇÃO

Nos tópicos a seguir **estaremos praticando os conceitos visto no módulo 3 e 4 de Programação I**. Revise estes conceitos nos seguintes capítulos do livro de apoio de Programação I:

- Capítulo 04 – CLASSES E OBJETOS;
- Capítulo 05 – MÉTODOS DE CONFIGURAÇÃO E ACESSO AOS ATRIBUTOS.

Na sequencia, leia atentamente o resumo dos conceitos e os exemplos apresentados neste documento e tire suas dúvidas com o tutor no fórum de dúvidas.

Atenção: Para que você possa comparar as suas classes, com as classes desenvolvidas neste material, faça o download deste projeto que está disponível para download no módulo 05.

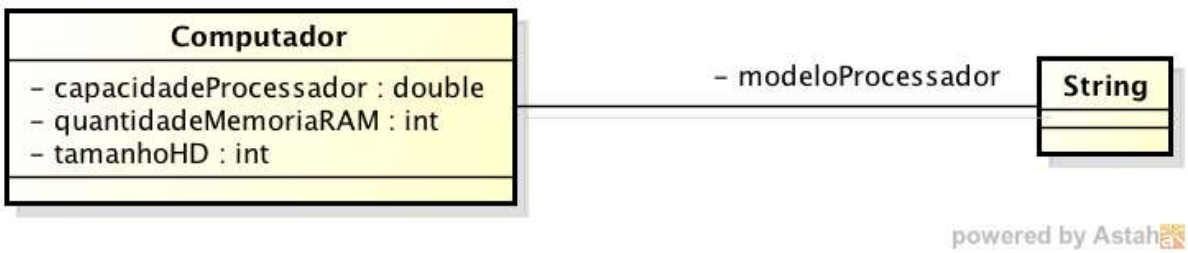
2 ASSOCIAÇÃO SIMPLES ENTRE CLASSES

No módulos anteriores de Programação I e Laboratório I, **ao utilizarmos o tipo de dados objeto: String, trabalhamos com um conceito de associação simples entre classes**. Neste módulo estaremos praticando a associação simples também, entre classes que foram criadas por nós. Porém, antes de praticarmos, vamos verificar o conceito de associação simples entre classes.

A associação simples entre as classes pode ser considerado como um tipo de ligação entre classes, cujo objetivo é apresentar o relacionamento ou dependência de uma classe com a outra.

Considere os seguintes exemplos abaixo:

1. **Associação simples** entre a classe **Computador** e a classe **String** da biblioteca Java:



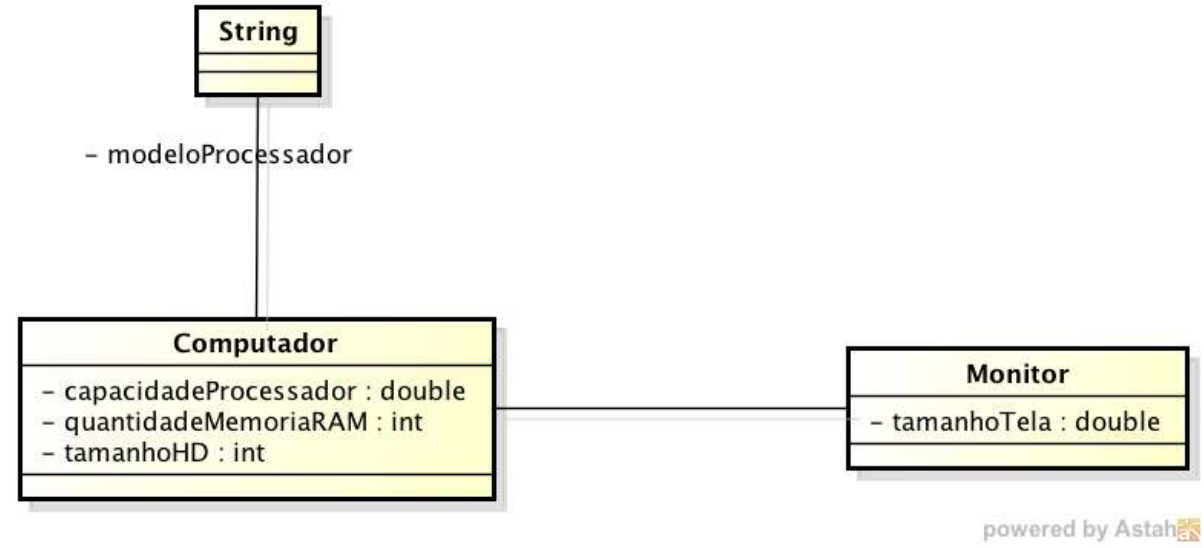
O diagrama de classes UML acima apresenta a associação simples entre a classe Computador e a classe String da biblioteca Java. Este exemplo de associação está baseado na classe: Computador, utilizada nos exemplos das práticas de Laboratório I dos módulos 3 e 4.

Se você abrir a classe Computador e visualizar os atributos desta classe, vai perceber que o atributo: modeloProcessador é do tipo de dados objeto String. Como o tipo de dados objeto String é criado (instanciado) por meio de uma classe da biblioteca Java chamada: String, podemos dizer que há uma associação simples entre estas classes neste exemplo.

```
public class Computador
{// o simbolo "{" simboliza o inicio do corpo da classe

    //atributos da classe:
    private String modeloProcessador;
    private double capacidadeProcessador;
    private int quantidadeMemoriaRAM;
    private int tamanhoHD;
```

2. **Associação simples** entre a classe **Computador** e uma nova classe: **Monitor**:

Vamos agora apresentar um outro exemplo de associações simples entre classes, utilizando duas classes criadas por nós, as classes: Computador e Monitor.



Importante: Neste exemplo também foi mantida a associação entre computador e String, pois na nossa classe: Computador estamos fazendo uso do tipo de dados objeto: String para representarmos o tipo de dados do atributo: modeloProcessador. Esta associação com a classe String está sendo representada apenas para fins didáticos, pois na prática, não é necessário detalhar este tipo de associação em um diagrama de classes UML.

Analisando a associação entre Computador e Monitor, podemos ter a seguinte compreensão: um Computador possui um Monitor. Sendo assim, é possível dizer que a classe: Computador depende da classe: Monitor. Este tipo de dependência, para as associações simples, é chamado de: **dependência por atributo**, sendo que o Monitor será um atributo da classe Computador.

A imagem abaixo apresenta a classe: Computador com o novo atributo: monitorComputador. Veja que o tipo de dados utilizado para este atributo é a classe: Monitor, apresentada na segunda imagem.

```java
public class Computador
{// o simbolo "{" simboliza o inicio do corpo da classe

    //atributos da classe:
    private String modeloProcessador;
    private double capacidadeProcessador;
    private int quantidadeMemoriaRAM;
    private int tamanhoHD;
    private Monitor monitorComputador;
```

```java
public class Monitor
{
    // atributos da classe
    private double tamanhoTela;

}
```

2.1 EXEMPLO ASSOCIAÇÃO ENTRE CLASSES

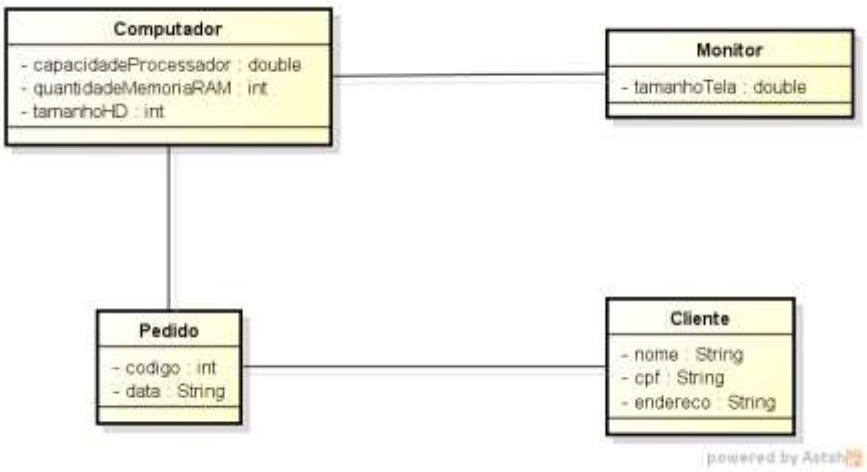
Vamos então praticar a criação de associações simples entre classes, utilizando duas novas classes. Utilizando o cenário de uma loja virtual que vende computadores, estaremos criando duas novas classes no nosso projeto: Cliente e Pedido. A classe Cliente possuirá os seguintes atributos:

- CPF;
- Nome;
- Endereço;

A classe: Pedido por sua vez, possuirá relacionamento com as classes: Computador e Cliente, pois um cliente da loja virtual poderá comprar um computador para cada pedido emitido. Desta forma, inicialmente a classe Pedido possuirá os seguintes atributos:

- Código;
- Data;
- Cliente => atributo que será relacionado com a classe Cliente;
- Computador => atributo que será relacionado com a classe Computador.

O diagrama de classes abaixo apresenta a associação entre estas 3 classes:



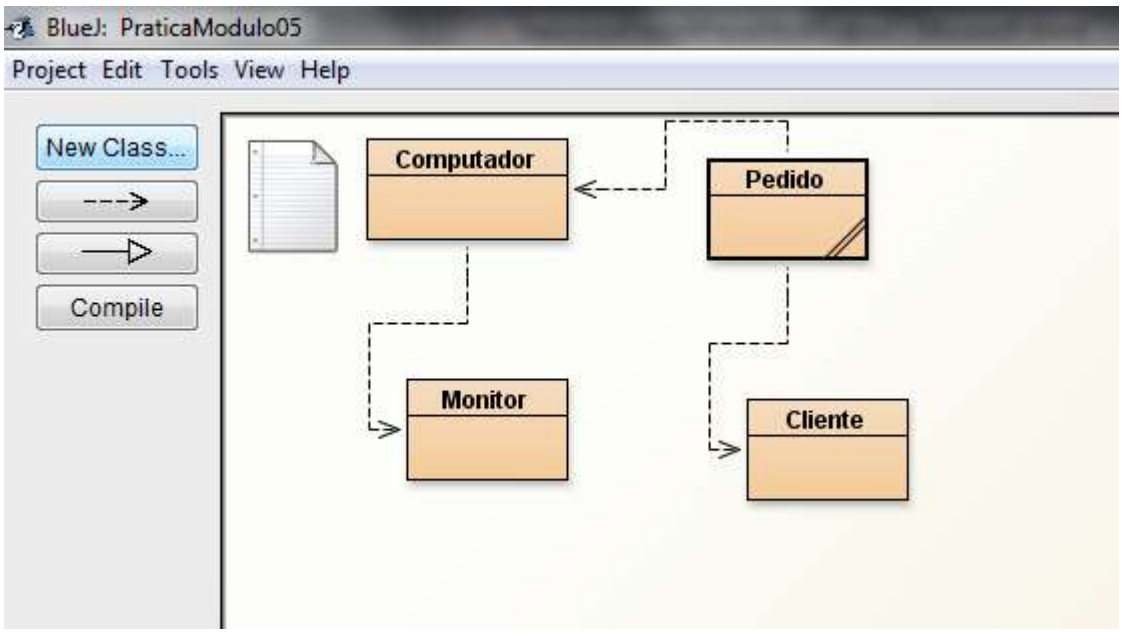
Vamos então criar a classe: Cliente. A classe: Cliente deverá ser criada conforme a imagem abaixo.

```
public class Cliente
{
    // atributos da classe
    private String cpf;
    private String nome;
    private String endereco;

    /**
     * Construtor
     */
    public Cliente(String cp, String no, String en)
    {
        cpf = cp;
        nome = no;
        endereco = en;
    }
}
```

Agora, vamos criar a classe: Pedido. A classe: Pedido deverá ser criada conforme a imagem abaixo.

```
public class Pedido
{
    // atributos da classe
    private int codigo;
    private String data;
    private Cliente cliente;
    private Computador computador;

    /**
     * Construtor
     */
    public Pedido(int cd, String da, Cliente cl, Computador co)
    {
        codigo = cd;
        data = da;
        cliente = cl;
        computador = co;
    }
}
```

Após a criação e compilação das classes, o BlueJ apresentará a associação das classes na tela principal, conforme a imagem abaixo:



Pronto! Agora podemos utilizar os comandos do BlueJ para testar a criação dos objetos e/ou criar os métodos de atribuição e acesso para as classes criadas. Para praticar, revise as práticas de laboratório dos módulos 3 e 4.