# A5LP1 – Exercícios

## Aula 6 - Programação Orientada a Objetos com C#

**Instruções para entrega das listas de exercícios:**

**Meio de Entrega:** As resoluções das listas de exercícios devem ser entregues exclusivamente por meio do ambiente Moodle (http://eadcampus.spo.ifsp.edu.br). Usar o mesmo usuário e senha do Sistema Aurora.

**Forma de Entrega:** Para exercícios com implementação de programas console, devem ser entregues os arquivos das classes (extensão CS). Para exercícios com implementação de programas Windows Forms, devem ser entregues as pastas dos projetos que contêm as aplicações desenvolvidas. Para exercícios com banco de dados, também devem ser entregues as instruções SQL usadas (extensão SQL ou TXT). Todos os arquivos da lista devem ser compactados em um único arquivo (extensão RAR ou ZIP), cujo nome deverá conter a aula, o nome e um sobrenome do aluno. Por exemplo: Aula2_JoaoSilva.zip.

**Prazo de Entrega:** O prazo de entrega está definido na própria página de exercícios do Moodle, lembrando que o sistema bloqueia o envio de arquivos após a data e horário indicados.

**Obs.:** O material da disciplina e as listas de exercícios estão disponíveis no ambiente Moodle.

1. Faça um programa em C# que apresente ao usuário as opções a seguir, enquanto ele não digitar a opção 0 (zero). De acordo com o número da opção informado, o programa deverá calcular a quantidade de tempo, em dias (opção 1) ou em horas (opção 2), entre uma data/hora inicial e uma data/hora final. Após realizar o cálculo, o programa deverá apresentar o resultado ao usuário.
   1) **Calcular dias** – Solicite a data/hora inicial e a data/hora final no formato "dd/mm/yy hh:mm". Os dados recebidos devem ser passados como argumento pelo construtor da classe **Dia**. A partir deste construtor, deve ser chamado o método abstrato **CalcularTempo()** implementado nesta classe.
   2) **Calcular horas** – Solicite a data/hora inicial e a data/hora final no formato "dd/mm/yy hh:mm". Os dados recebidos devem ser passados como argumento pelo construtor da classe **Hora**. A partir deste construtor, deve ser chamado o método abstrato **CalcularTempo()** implementado nesta classe.

   **Obs.:** Deverá haver uma classe abstrata **Tempo** que possuirá o método abstrato **CalcularTempo()**, o qual é implementado nas classes **Dia** e **Hora**. **Dica:** Para fazer os cálculos, crie atributos do tipo da classe **TimeSpan**. Use o método **Subtract** desta classe, para obter a diferença de dias e horas entre as datas, e as propriedades **TotalDays** e **TotalHours**, para extrair os dias e as horas.

2. Faça um programa em C# que apresente ao usuário as opções a seguir, enquanto ele não digitar a opção 0 (zero). De acordo com o número da opção informado, o programa deverá calcular a área da respectiva forma, solicitando as informações necessárias ao usuário. Após realizar o cálculo, o programa deverá informar a área calculada e a cor da forma ao usuário.
   1) **Retângulo** – Solicite a base, a altura e uma cor para o retângulo. Os dados recebidos devem ser passados como argumento pelo construtor da classe **Retangulo**. A partir deste construtor, devem ser chamados o método abstrato **CalcularArea()** da classe **Retangulo** e o método concreto **ConfigurarCor(string cor)** da classe abstrata **Forma**.

2) **Círculo** – Solicite o raio e uma cor para o círculo. Os dados recebidos devem ser passados como argumento pelo construtor da classe **Circulo**. A partir deste construtor, devem ser chamados o método abstrato **CalcularArea()** da classe **Circulo** e o método concreto **ConfigurarCor(string cor)** da classe abstrata **Forma**.
3) **Triângulo Equilátero** – Solicite a medida dos lados e uma cor para o triângulo equilátero. Os dados recebidos devem ser passados como argumento pelo construtor da classe **Triangulo**. A partir deste construtor, devem ser chamados o método abstrato **CalcularArea()** da classe **Triangulo** e o método concreto **ConfigurarCor(string cor)** da classe abstrata **Forma**.

**Obs.:** A classe abstrata **Forma** deve possuir o método abstrato **CalcularArea()**, além do método **ConfigurarCor(string cor)**, que deve atribuir o valor recebido ao atributo **cor** da classe **Forma**.

3. Faça um programa em C# que apresente ao usuário as opções a seguir, enquanto ele não digitar a opção 0 (zero). De acordo com o número da opção informada, o programa deverá efetuar a operação, solicitando as informações necessárias ao usuário.
   1) **Informar texto** – Deverá chamar o método público **SetTexto(string texto)** da classe abstrata **Texto**. Para isso, crie um objeto **Texto** instanciando uma classe filha **Extracao** ou **Substituicao**.
   2) **Extrair string** – Solicite a posição inicial e o número de caracteres da string a ser extraída. Os dados recebidos, assim como o texto informado na opção 1, devem ser passados como argumento pelo construtor da classe **Extracao**. A partir deste construtor, deve ser chamado o método abstrato **ManipularString(string texto, string posInicio, string numCaracteres)** implementado nesta classe.
   3) **Substituir string** – Solicite a string a ser substituída e a string substituta. Os dados recebidos, assim como o texto informado na opção 1, devem ser passados como argumento pelo construtor da classe **Substituicao**. A partir deste construtor, deve ser chamado o método abstrato **ManipularString(string texto, string strSubstituida, string strSubstituta)** implementado nesta classe.

   **Obs.:** A classe abstrata **Texto** possuirá o atributo estático **string texto**, os métodos concretos **GetTexto()** e **SetTexto(string txt)**, e o método abstrato **ManipularString(string texto, string argumento1, string argumento2)**, o qual é implementado nas classes **Extracao** e **Substituicao**. **Dica:** Para manipular as strings, use os métodos **Substring** e **Replace** da classe String.

4. Faça uma cópia do programa do exercício 1. Transforme a classe abstrata **Tempo** em uma interface e faça as alterações necessárias para que o programa funcione.

5. Faça uma cópia do programa do exercício 2. Transforme a classe abstrata **Forma** em uma interface e faça as alterações necessárias para que o programa funcione. **Obs.:** Lembre-se que uma interface não permite a implementação de métodos dentro dela. Logo o método **ConfigurarCor(string cor)** deverá ser transformado em método abstrato (sem implementação), devendo ser implementado nas classes **Retangulo**, **Circulo** e **Triangulo**.