Universidade Federal de Uberlândia
Faculdade de Computação – Gestão da Informação – Prof. Daniel A. Furtado
7º Trabalho de Programação Orientada a Objetos
Namespaces e Biblioteca de Classes .NET

Instruções Gerais

- Esta atividade deve ser realizada **individualmente**;
- Esteja atento às **observações sobre plágio** apresentadas no final desde documento;
- Trabalhos com implementações utilizando trechos de códigos retirados de sites da Internet, gerados por ferramentas de IA (como ChatGPT, Gemini e similares), copiados de trabalhos de semestres anteriores, serão anulados;
- O trabalho deve ser entregue até a data/hora definida pelo professor. Não deixe para enviar o trabalho nos últimos instantes, pois eventuais problemas relacionados à eventos adversos como instabilidade de conexão, congestionamento de rede etc., não serão aceitos como motivos para entrega da atividade por outras formas ou em outras datas;
- Trabalhos enviados por e-mail ou pelo MS Teams **não serão considerados**.

OBS: Este trabalho deve ser feito utilizando o **Visual Studio Community** ou o **VS Code** conforme instruções do slide 97. Ambientes executados no navegador (como o Programiz) não devem ser utilizados, pois algumas funcionalidades necessárias não estão disponíveis pelo navegador.

Material de Apoio

https://furtado.prof.ufu.br/site/teaching/POO/POO-Modulo2-Fundamentos.pdf (slides 97-111)

Exercício 1

a) Defina uma classe para modelar **pontos luminosos** no contexto de uma aplicação gráfica. Não é necessário criar um construtor explícito. Deve haver:
   - Atributos **privados** para armazenar as coordenadas de exibição x e y na tela (inteiro);
   - Propriedades **públicas** para expor as coordenadas para leitura e atualização;
   - Um método de nome **Brilhar**, sem parâmetros. Quando for chamado, o método deve alterar a posição do cursor da janela de console para as coordenadas **x** e **y** e então apresentar o caractere '+'. Utilize o método Console.SetCursorPosition.

b) Crie uma classe para modelar um **painel luminoso** contendo vários **pontos luminosos** em posições aleatórias:
   - Deve haver atributos privados para armazenar os pontos luminosos do painel (**array de pontos luminosos**), a **largura** e **altura** do painel, um objeto para geração de números aleatórios e um objeto contendo a cor dos pontos de luz no painel (do tipo ConsoleColor);
   - O construtor deve ter os seguintes parâmetros: número de pontos luminosos, largura e altura do painel, cor dos pontos luminosos. O construtor deve instanciar o array de pontos luminosos conforme tamanho recebido por parâmetro, assim como cada ponto luminoso do array. Deve instanciar também o objeto de geração de números aleatórios.
   - Deve haver um método para distribuir aleatoriamente os pontos luminosos dentro do painel. O método deve percorrer o array de pontos luminosos e atribuir valores aleatórios para as posições x e y de cada ponto luminoso, sem ultrapassar os limites de tamanho do painel.

- Deve haver um método para exibir todos os pontos luminosos do painel. O método deve percorrer o array de pontos e chamar o método **Brilhar** de cada ponto luminoso. Antes de executar essa ação, o método deve alterar a cor dos pontos luminosos utilizando a propriedade Console.ForegroundColor e a cor dos pontos armazenada no atributo.

c) Crie um programa principal para utilizar as classes definidas. O programa deve criar quatro painéis luminosos com 25 pontos de luz cada um, nas cores branco, verde, azul e vermelho, respectivamente (ConsoleColor.White, ConsoleColor.Green etc.). Os painéis devem ter tamanhos iguais à janela de console (Console.WindowWidth, Console.WindowHeight). Utilize um laço **while** para criar uma animação exibindo os pontos luminosos dos quatro painéis de maneira conjunta. Dentro do **while**, chame primeiramente o método para distribuir os pontos luminosos de cada painel. Em seguida, limpe o console e chame o método de exibição dos pontos luminosos de cada painel. Utilize Thread.Sleep para pausar temporariamente a execução dentro do **while** e criar um efeito de animação (veja exemplo do slide 110).

Exercício 2

Crie um projeto no Visual Studio Community e defina uma **classe estática** contendo um **método estático** para verificar, utilizando uma **pilha** da biblioteca de classes do framework .NET, se uma string passada por parâmetro corresponde a um **palíndromo** (o método deve retornar **true** ou **false**). Para simplificar, considere que a string terá sempre um número par de caracteres. A string deve ser percorrida apenas uma vez.

Crie um programa principal para testar os métodos da classe estática. O programa deve solicitar ao usuário a string a ser verificada.

Entrega

Compacte os arquivos com as respostas dos exercícios em um único arquivo no formato **zip** e envie pelo Sistema **SAAT** até a data limite indicada pelo professor em sala de aula. Caso não tenha o **Visual Studio Community**, coloque o código criado para cada exercício em um arquivo .cs (exercicio1.cs, exercicio2.cs etc.), compacte todos os arquivos e envie o arquivo compactado.

Sobre Eventuais Plágios

Este é um trabalho individual. Os alunos envolvidos em qualquer tipo de plágio, total ou parcial, seja entre equipes ou de trabalhos de semestres anteriores ou de materiais disponíveis na Internet (exceto os materiais de aula disponibilizados pelo professor), serão duramente penalizados (art. 196 do Regimento Geral da UFU). Todos os alunos envolvidos terão seus **trabalhos anulados** e receberão **nota zero**.