Universidade Federal de Uberlândia
Faculdade de Computação – Gestão da Informação – Prof. Daniel A. Furtado
8º Trabalho de Programação Orientada a Objetos
Tratamento de Exceções e Manipulação de Arquivos

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 96-118)

Exercício 1

a) Defina uma classe para modelar **clientes** com atributos **nome**, **email** e **telefone**. Crie **propriedades** para expor os dados para leitura e atualização. Crie um **construtor** para permitir que novos objetos sejam criados mediante o fornecimento desses dados. A classe deve ter um método (Mostrar) para apresentação dos dados do objeto utilizando Console.WriteLine.

b) Crie uma classe com o nome **RepositorioClientes** para modelar um repositório de dados de clientes:

   i. Deve haver um atributo do tipo lista para armazenar objetos do tipo **Cliente** (List<Cliente>, veja slide 105) e um atributo do tipo **string** para armazenar o caminho do arquivo onde os dados serão salvos.

   ii. O construtor deve ter apenas um parâmetro: o caminho do arquivo.

   iii. Deve haver um método de nome **Adicionar** com um parâmetro do tipo **Cliente** para permitir que novos objetos sejam adicionados à lista de clientes do repositório.

   iv. Deve haver um método para exibição de todos os dados de todos os clientes correntemente no repositório.

   v. Deve haver um método de nome Salvar para gravar todos os dados de todos os clientes em arquivo no formato **JSON**. O método deve abrir o arquivo utilizando a classe **StreamWriter**. A lista de clientes deve ser convertida em uma **string** JSON utilizando "**string** json = JsonSerializer.Serialize(listaClientes);". A string json deve ser gravada no arquivo utilizando o método **WriteLine** do objeto criado utilizando a classe **StreamWriter**. Exceções devem ser tratadas adequadamente. O arquivo deve ser fechado assim que os dados forem salvos.

c) Crie um programa principal para ilustrar o uso das classes. O programa deve criar um objeto utilizando a classe **RepositorioClientes** e apresentar um menu (com um laço **do**-**while**) com as seguintes opções:

1 – Cadastrar novo cliente
2 – Ver clientes cadastrados
3 – Salvar clientes em arquivo
4 – Sair

A opção 1 deve coletar os dados do novo cliente do usuário, criar um novo objeto do tipo Cliente, e chamar o método **Adicionar** do repositório para inclusão do objeto na lista de clientes do repositório. A opção 2 deve apresentar os dados de todos os clientes do repositório. A opção 3 deve chamar o método de salvamento em arquivo do repositório e a opção 4 deve finalizar o programa.

Depois de executar o programa, cadastrar vários clientes e salvar, abra o arquivo de texto utilizando um editor qualquer (Bloco de Notas) e verifique os dados.

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**.