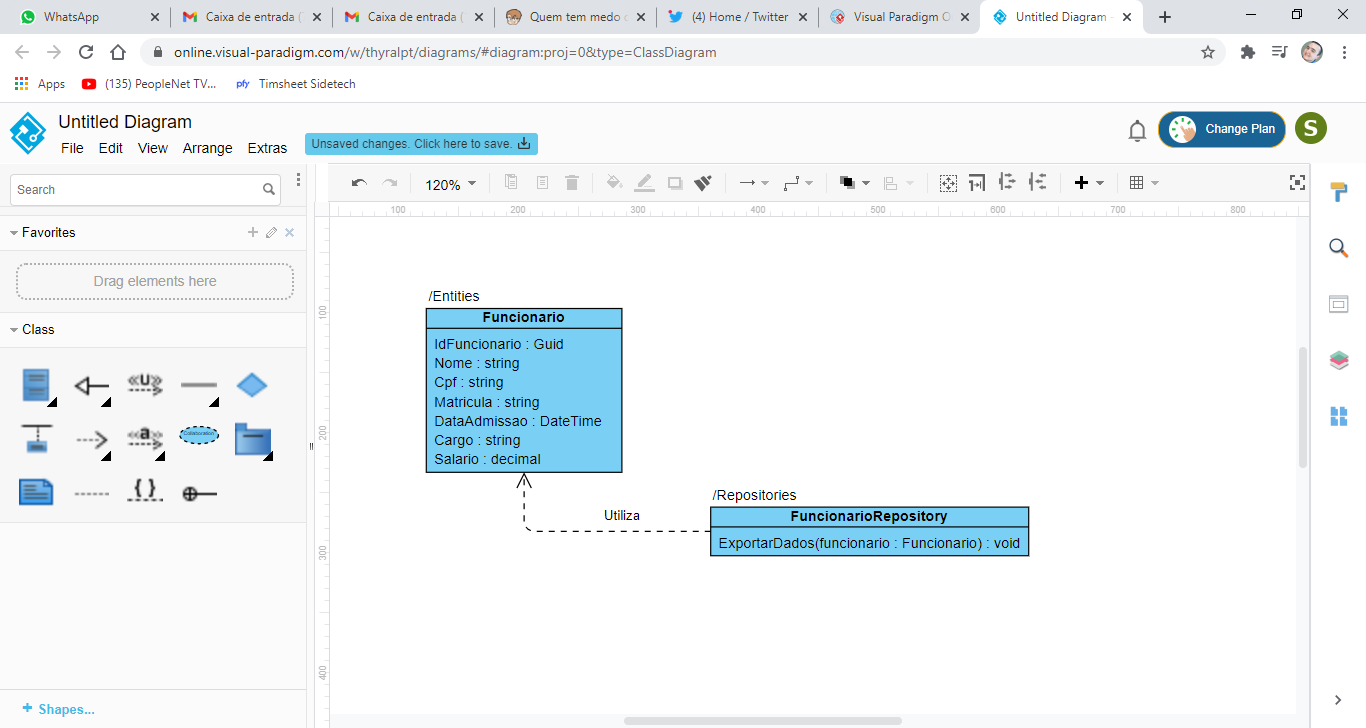
Teoria:

1. Defina classes, atributos e métodos.  
   R: Classes: Estrutura de dados que combina ações, fornece definições para as instancias e dão suporte a herança e polimorfismo.
2. O que são propriedades set e get? Qual a função de cada uma delas?  
   R: São métodos públicos que permitem acessar indiretamente os atributos de uma classe.   
   Set – serve para fazer a entrada de dados   
   Get – serve para fazer a saída dos dados.
3. Explique para que servem as seguintes palavras reservadas da linguagem:
   1. class – Serve para indicar a declaração de uma classe
   2. namespace – usada para declara um escopo que contém um conjunto de objetos relacionados
   3. void – Tipo de retorno de um método ou função local que especifica que tal método ou função não retorna nenhum valor.
   4. using – Cria um Alias para um namespace ou importa tipos definidos em outros namespace
   5. public – Definir acesso total para uma classe ou método
   6. new – Instanciar um objeto
   7. var – Declarar variável local
4. Coloque o nome de cada princípio SOLID a abaixo. Explique o que significa o princípio da responsabilidade única.
   1. SRP – Single Responsability Principle
   2. OCP – Open Closed Simple
   3. LSP - Liskov Substution Principle
   4. ISP - Interface Segregation Principle
   5. DIP – Dependency Inversion Principle  
        
      SRP - Uma classe deve ter um e somente um motivo para mudar.
5. Defina entidades.  
   Objetos, estrutura de dados ou classes que representam objetos do mundo real.
6. O que são variáveis de instância?  
   São atributos de uma classe.

Prática:

1. Crie um projeto do tipo “Aplicativo do Console”  
   Neste projeto, desenvolva uma aplicação que leia os dados de um Funcionário (informado pelo usuário) e grave as informações em um arquivo do tipo TXT na máquina local (em c:\\temp\\)  
   Desenvolva a aplicação conforme o diagrama abaixo: https://github.com/CabralJayme/ExercicioCotiAula1



Qual dos seguintes princípios é fundamental para escrever código limpo em C#?

a) Uso excessivo de comentários.

b) Manter classes pequenas e métodos concisos.

c) Adotar o padrão de codificação baseado em espaços em branco.

d) Utilizar nomes de variáveis e métodos longos para facilitar a compreensão.

Em relação aos comentários em C#, qual das seguintes afirmações é verdadeira?

a) Comentários extensivos são essenciais para um código limpo.

b) Comentários devem explicar cada linha de código em detalhes.

c) O código deve ser autoexplicativo, minimizando a necessidade de comentários.

d) Comentários devem ser escritos apenas no idioma inglês.

Qual é a prática recomendada para nomear variáveis e métodos em C#?

a) Usar abreviações sempre que possível para reduzir o tamanho do código.

b) Utilizar nomes descritivos que indiquem claramente a finalidade da variável ou método.

c) Nomear variáveis e métodos com uma única letra para economizar espaço.

d) Adotar nomes genéricos como "var1", "var2", etc., para facilitar a leitura.

Qual dos seguintes conceitos contribui para tornar o código mais limpo e legível em C#?

a) Aninhar vários loops e condicionais em um único método.

b) Ignorar o uso de constantes e valores literais.

c) Evitar a duplicação de código através da criação de métodos e classes reutilizáveis.

d) Escrever classes com muitas responsabilidades para aumentar a coesão.

O que é acoplamento em C# e por que é importante considerá-lo ao escrever código limpo?

a) Acoplamento refere-se à medida em que uma classe depende de outra. É importante reduzir o acoplamento para tornar o código mais flexível e fácil de manter.

b) Acoplamento é a quantidade de comentários presentes no código. Quanto mais comentários, mais acoplamento.

c) Acoplamento é o processo de criar relacionamentos entre classes, o que é fundamental para garantir a coesão do código.

d) Acoplamento é a técnica de mesclar várias classes em uma única, facilitando a comunicação entre elas.

Qual é a importância de escrever testes unitários para garantir código limpo em C#?

a) Testes unitários são desnecessários quando se escreve código limpo.

b) Testes unitários ajudam a verificar se o código está funcionando corretamente e fornecem documentação viva.

c) Testes unitários aumentam a complexidade do código e devem ser evitados.

d) Testes unitários são úteis apenas para desenvolvedores experientes.

Quais são algumas das características de uma classe bem projetada em C#?

a) Classes com muitas responsabilidades e métodos longos.

b) Classes que dependem fortemente de outras classes e não possuem coesão.

c) Classes que aderem ao princípio de responsabilidade única, têm baixo acoplamento e alta coesão.

d) Classes que não utilizam encapsulamento.

O que é "Refatoração" em C# e como ela contribui para o código limpo?

a) Refatoração é o processo de introduzir mais bugs no código para melhorar sua qualidade.

b) Refatoração é o ato de reescrever completamente o código de um projeto.

c) Refatoração é o processo de fazer pequenas melhorias incrementais no código, mantendo seu comportamento externo inalterado.

d) Refatoração é uma técnica usada apenas por programadores novatos.