# **Relatório de trabalho prático**

# **Gestão de Auditorias**

**Yuri Serediuk Lemos**

**Nº 19256**

**João Pedro Marques Figueiredo**

**Nº 19337**

Trabalho realizado sob a orientação de:

Luís Ferreira

**Linguagens de Programação II**

**Licenciatura em Engenharia de Sistemas Informáticos**

Barcelos, 25 de Abril de 2020

Índice

[**Relatório de trabalho prático** 1](#_Toc38829933)

[**Gestão de Auditorias** 1](#_Toc38829934)

[1. **Introdução** 5](#_Toc38829935)

[1.1. Contextualização 5](#_Toc38829936)

[1.2. Motivações e objetivos 6](#_Toc38829937)

[2. **Estado da arte** 7](#_Toc38829938)

[3. **Implementação** 8](#_Toc38829939)

[3.1. Descrição 8](#_Toc38829940)

[3.2. Solução 9](#_Toc38829941)

[3.2.1. Diagrama 9](#_Toc38829942)

[3.2.2. Criação de Classes 10](#_Toc38829943)

[3.2.2.1. Classe Pessoa 10](#_Toc38829944)

[3.2.2.2. Classe Funcionário 11](#_Toc38829945)

[3.2.2.3. Classe Funcionários 12](#_Toc38829946)

[3.2.2.4. Classe Equipamento 13](#_Toc38829947)

[3.2.2.5. Classe Equipamentos 14](#_Toc38829948)

[3.2.2.6. Classe Vulnerabilidade 15](#_Toc38829949)

[3.2.2.7. Classe Vulnerabilidades 16](#_Toc38829950)

[3.2.2.8. Classe Ocorrência 17](#_Toc38829951)

[3.2.2.9. Classe Interface IMetodosGenericos 18](#_Toc38829952)

[3.2.2.10. Classe Program 19](#_Toc38829953)

[4. **Conclusão** 20](#_Toc38829954)

[4.1. Lições aprendidas 20](#_Toc38829955)

[4.2. Apreciação Final 20](#_Toc38829956)

Índice de Imagens

[Figura 1 Diagrama de classes 9](file:///C:\Users\Yuri%20Lemos\Desktop\Relatório%20de%20trabalho%20prático%20LP2.docx#_Toc38830033)

[Figura 2 Classe Pessoa 10](#_Toc38830034)

[Figura 3 Classe Funcionário 11](#_Toc38830035)

[Figura 4 Classe Funcionários 12](#_Toc38830036)

[Figura 5 Classe Equipamento 13](#_Toc38830037)

[Figura 6 Classe Equipamentos 14](#_Toc38830038)

[Figura 7 Classe Vulnerabilidade 15](#_Toc38830039)

[Figura 8 Classe Vulnerabilidades 16](#_Toc38830040)

[Figura 9 Classe Ocorrência 17](#_Toc38830041)

[Figura 10 Classe Interface IMetodosGenericos 18](#_Toc38830042)

[Figura 11 Classe Program 19](#_Toc38830043)

# Introdução

Este projeto foi realizado tendo como objetivo a avaliação da compreensão dos alunos para a linguagem de programação orientada a objetos (POO). Nesta primeira parte do projeto desenvolvido mostra a capacidade de compreensão dos alunos dos termos básicos do paradigma desta linguagem.

# Contextualização

O advento da tecnologia fez com que fossem necessários novos arranjos para a sociedade e para o poder público. Não raro, a lei não consegue acompanhar as mudanças velozes do mundo conectado, mas há anos que a demanda por leis mais protetivas aos dados pessoais do cidadão ganha espaço entre especialistas e entusiastas da tecnologia. Isto porque, o cidadão comum, por vezes, não é capaz de reconhecer os riscos e oportunidades criadas para a exploração de seus dados pessoais. Assim, muitas vezes abrem mão de informações pessoais importantes em troca de simples acesso a sites e aplicativos de telemóveis. Não apenas isto, mas toda a troca de dados realizada em estabelecimentos comerciais físicos e em registos quaisquer transações do tipo também deveriam constar no escopo de uma lei geral de proteção de dados. A União Europeia há alguns anos já gozava de tal proteção com a Diretiva 95/46 CE, a qual buscava regular a coleta, uso e tratamento de dados no território europeu. Porém, acompanhando as tendências de uso global, transferência e necessidade de apagamento ou prestação de contas sobre dados pessoais, um novo Regulamento foi elaborado a fim de expandir a proteção ao indivíduo. Sendo assim, após um período de dois anos, entrou em vigor o Regulamento Geral de Proteção de Dados europeu — GDPR. Com texto extenso e uma preocupação notável em abranger as mais diversas possibilidades de transação envolvendo dados, o GDPR sagrou-se um marco na proteção de dados e da proteção à privacidade do utilizador. Instituiu princípios sólidos e claros, a fim de não abrir margem para interpretações diversas. Assim, o documento cria no indivíduo a possibilidade de domínio sobre os próprios dados, reclamando a propriedade destes como algo pessoal e não comercial, pertencente às empresas ou amplamente explorado pelo Poder Público.

# Motivações e objetivos

O propósito do trabalho foi montar de forma criativa, consoante ao RGPD um programa de fácil acessibilidade utilizando a linguagem C#, com isso pode-se então ter uma melhor perceção das falhas ao RGPD numa determinada empresa.

# Estado da arte

O Regulamento Geral sobre a Proteção de Dados (RGPD), aprovado no Parlamento Europeu em abril de 2016, visa regular a proteção das pessoas singulares no que diz respeito ao tratamento de dados pessoais e à livre circulação desses dados, revogando assim a Diretiva 95/46/CE. Este Regulamento introduziu alterações significativas às regras mantidas em matérias de proteção de dados, impondo novas e rigorosas obrigações e severas coimas como punição ao seu no seu não cumprimento.

O RGPD proporcionou um período de transição de 2 anos para que as organizações implementassem um plano de transformação que proporcione o cumprimento de todos os requisitos necessários à aplicabilidade do RGPD, em 25 de maio de 2018. Em Portugal, a 8 de agosto de 2019, foi publicada a nova Lei nº 58/2019 que assegura a execução, na ordem jurídica nacional, do RGPD, e que revoga a anterior Lei nº 67/98.

O mundo virtual está em crescimento todos os dias, e a proteção dos dados é essencial para que a confidencialidade de uma companhia não seja comprometida. E para que os dados sejam resguardados de forma segura judicialmente, surgiu o *Data Protection Officer* (DPO) ou, em português, diretor de Proteção de Dados.

De acordo com a lei LGPD (Lei Geral de Proteção de Dados – 13.709/2018), a profissão entrará em vigor em agosto de 2020. O especialista será responsável pelo aconselhamento sobre as normas vigentes relativas à proteção de dados pessoais, o monitoramento do cumprimento do GDPR (*General Data Protection Regulation*) pela entidade a que está vinculado e a cooperação com as autoridades públicas supervisoras da norma.

# Implementação

# Descrição

As auditorias internas podem ser efetuadas por vários colaboradores. Cada auditoria deve ter em conta um código (ano/numero), quem foi o colaborador responsável, a data, quantidade de vulnerabilidade identificadas. Depois de inseridos os dados acerca das auditorias, quando terminar, a aplicação deverá mostrar:

* Uma relação das auditorias efetuadas, indicando o código e o colaborador responsável.
* Qual a auditoria onde foram detetadas mais e menos vulnerabilidades, indicando o seu código, data e quantidade de vulnerabilidades.

A aplicação deverá ter também uma estrutura de colaboradores, equipamentos, vulnerabilidades e um registo de ocorrência

# Solução

# Diagrama

Uma imagem com captura de ecrã

Descrição gerada automaticamente

Figura 1 Diagrama de classes

# Criação de Classes

# Classe Pessoa

Esta classe representa uma Pessoa, nesta classe contém:

* Nome
* Bilhete de identidade
* Contacto
* Morada
* Número Contribuinte

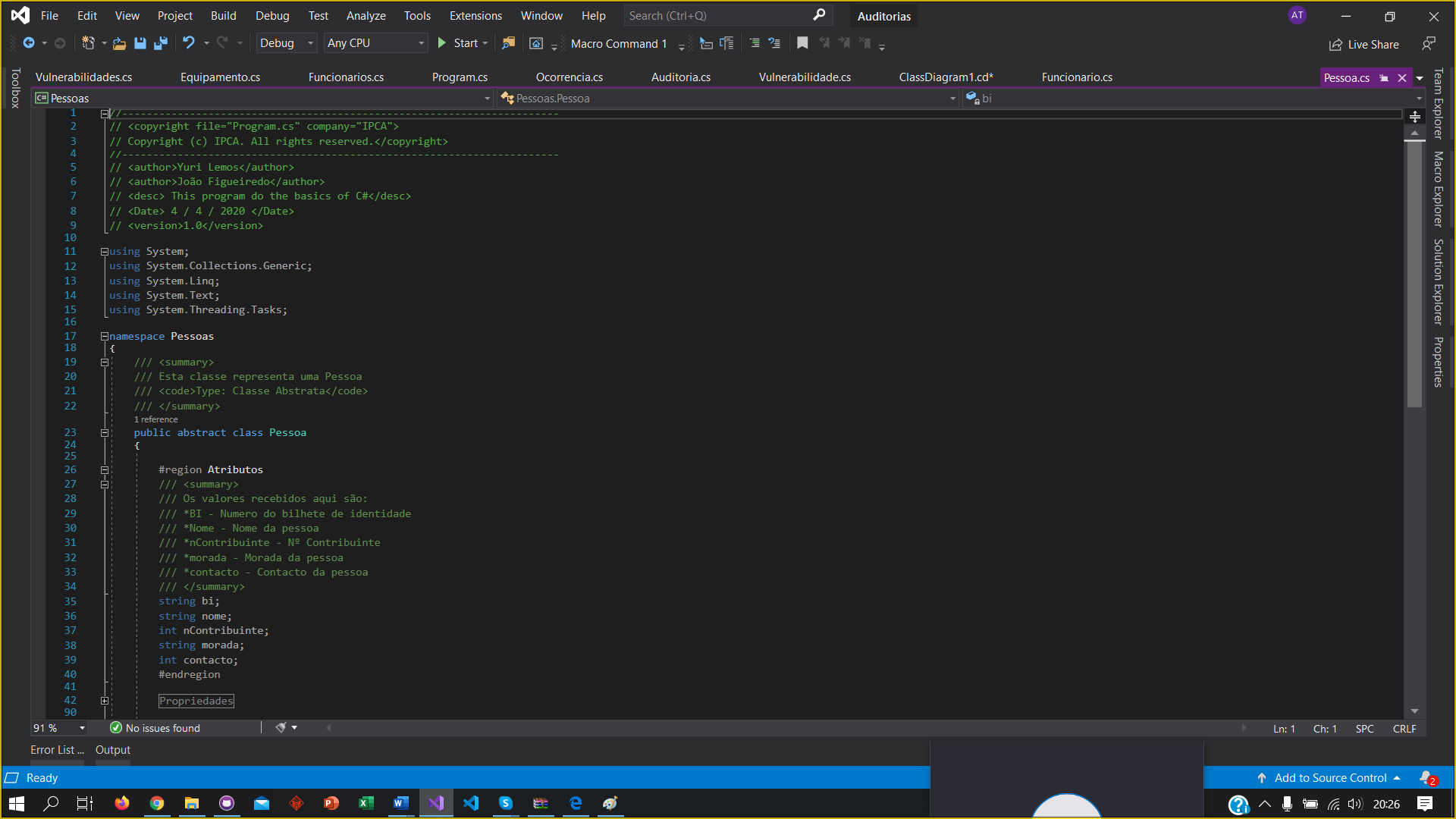


Figura 2 Classe Pessoa

# Classe Funcionário

Esta classe representa um Funcionário, ela é uma classe abstrata e por isso não poderá ser criada instâncias a partir desta classe. Nesta classe contém os seguintes atributos:

* Cargo
* Data Admissão
* Estado
* Identificação do funcionário

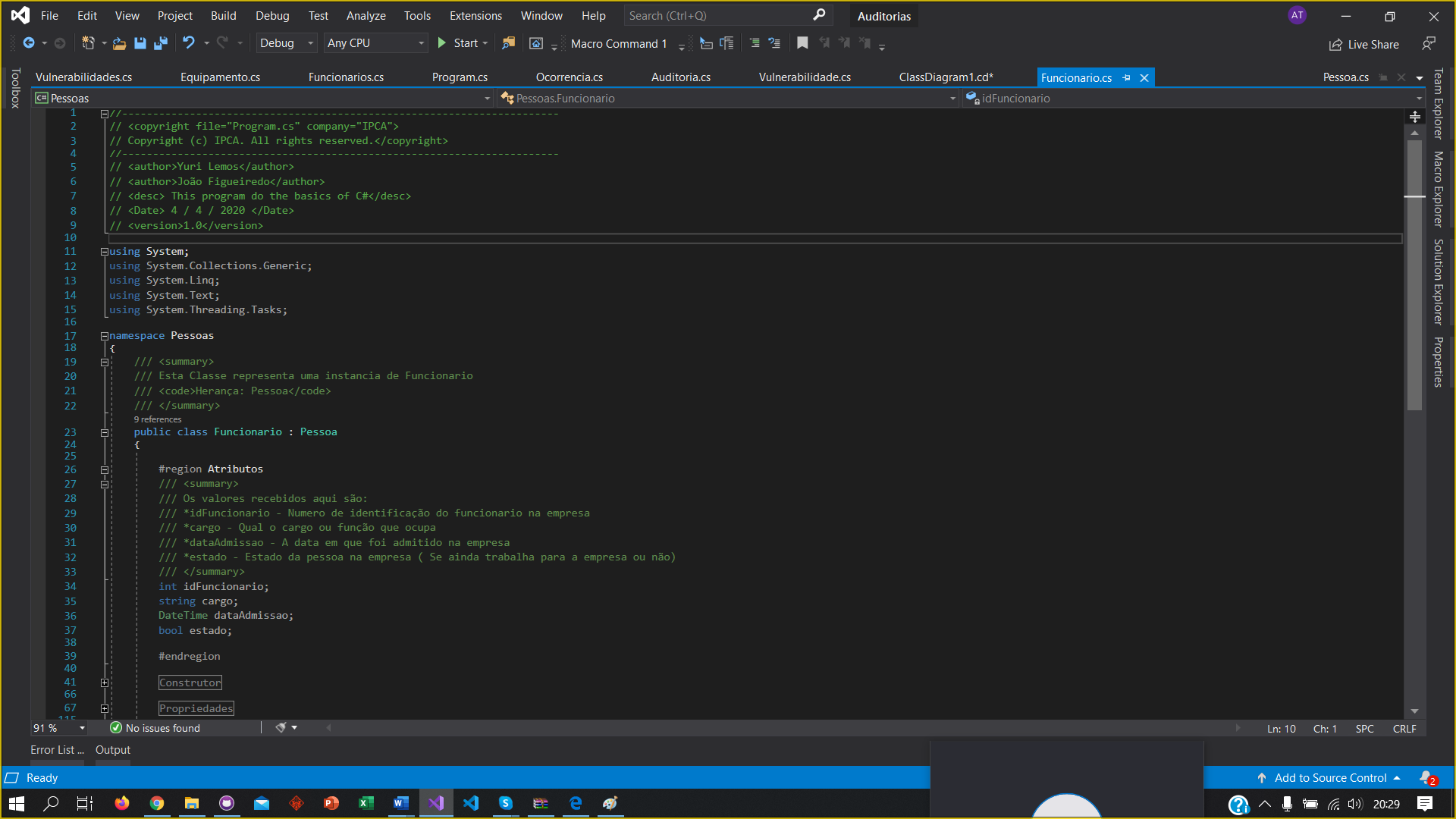


Figura 3 Classe Funcionário

# Classe Funcionários

Esta classe representa vários funcionários, nesta classe contém:

* Lista de funcionários
* Quantidade de funcionários

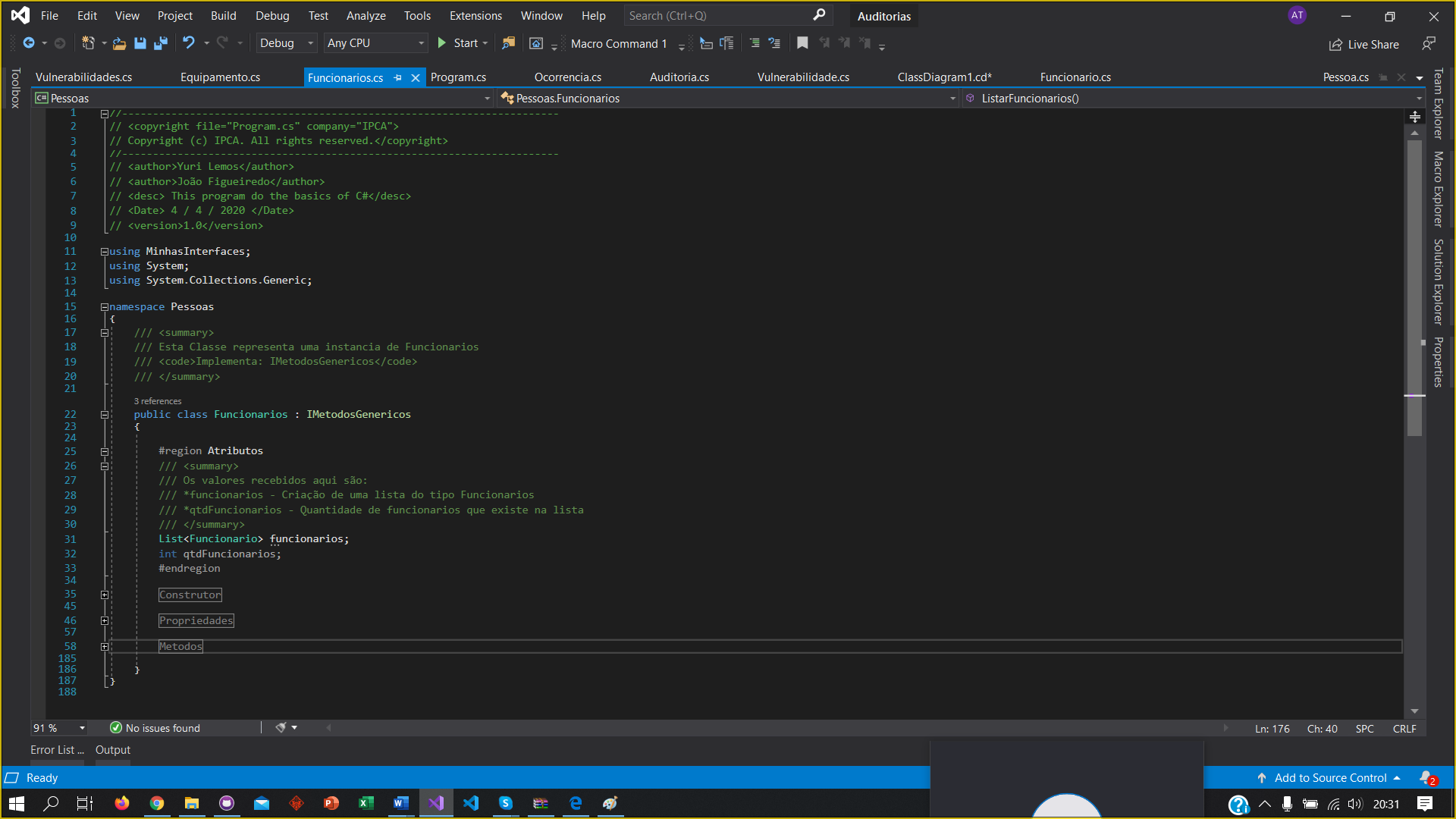


Figura 4 Classe Funcionários

# Classe Equipamento

Esta classe representa um equipamento, nesta classe contém:

* Código
* Marca
* Modelo
* Data aquisição
* Estado
* Tipo de equipamento

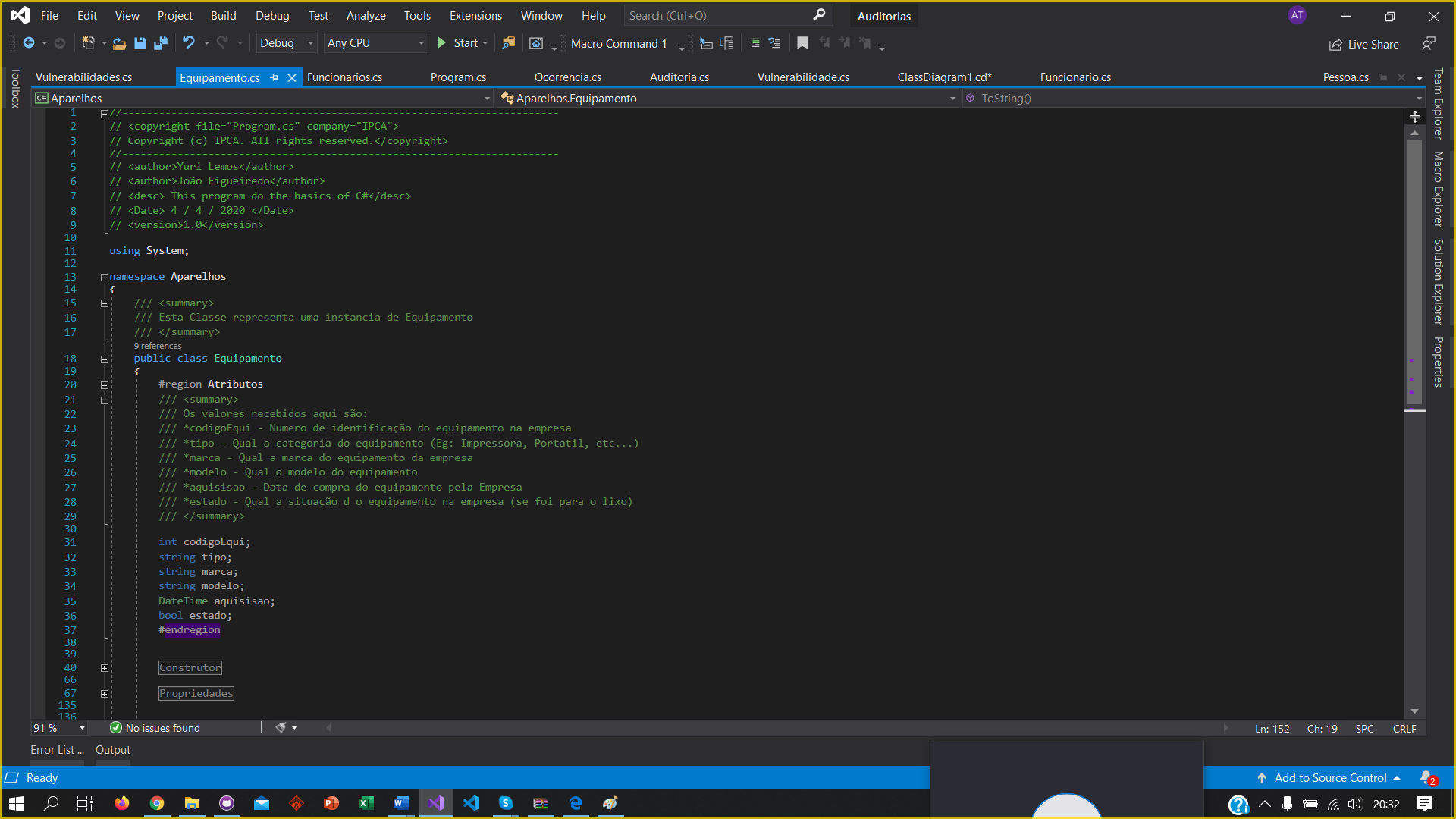


Figura 5 Classe Equipamento

# Classe Equipamentos

Esta classe representa vários equipamentos, nesta classe contém:

* Lista de equipamentos
* Quantidade de Equipamentos

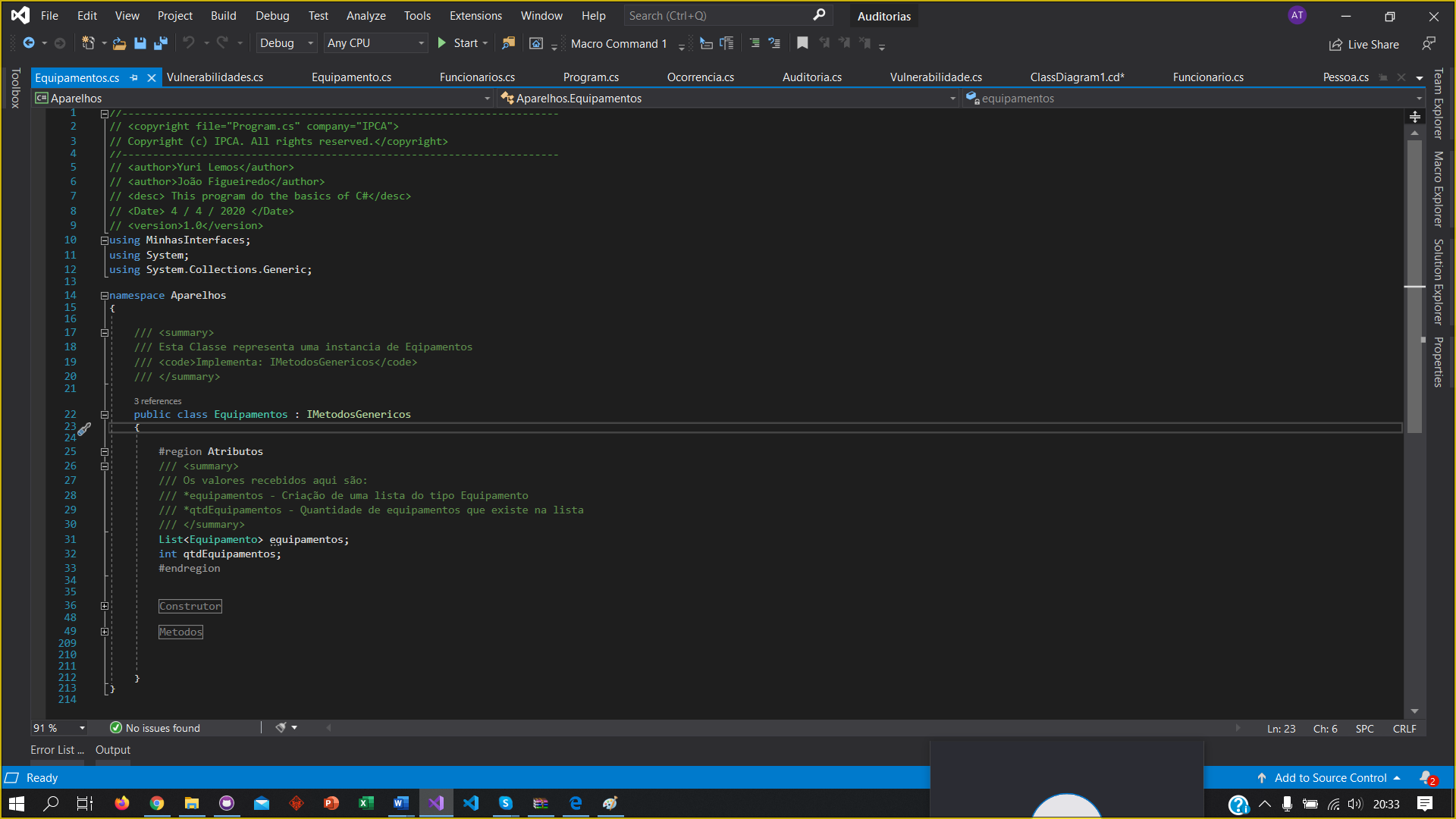


Figura 6 Classe Equipamentos

# Classe Vulnerabilidade

Esta classe representa uma vulnerabilidade, nesta classe contém:

* Código
* Descrição
* Nível de impacto

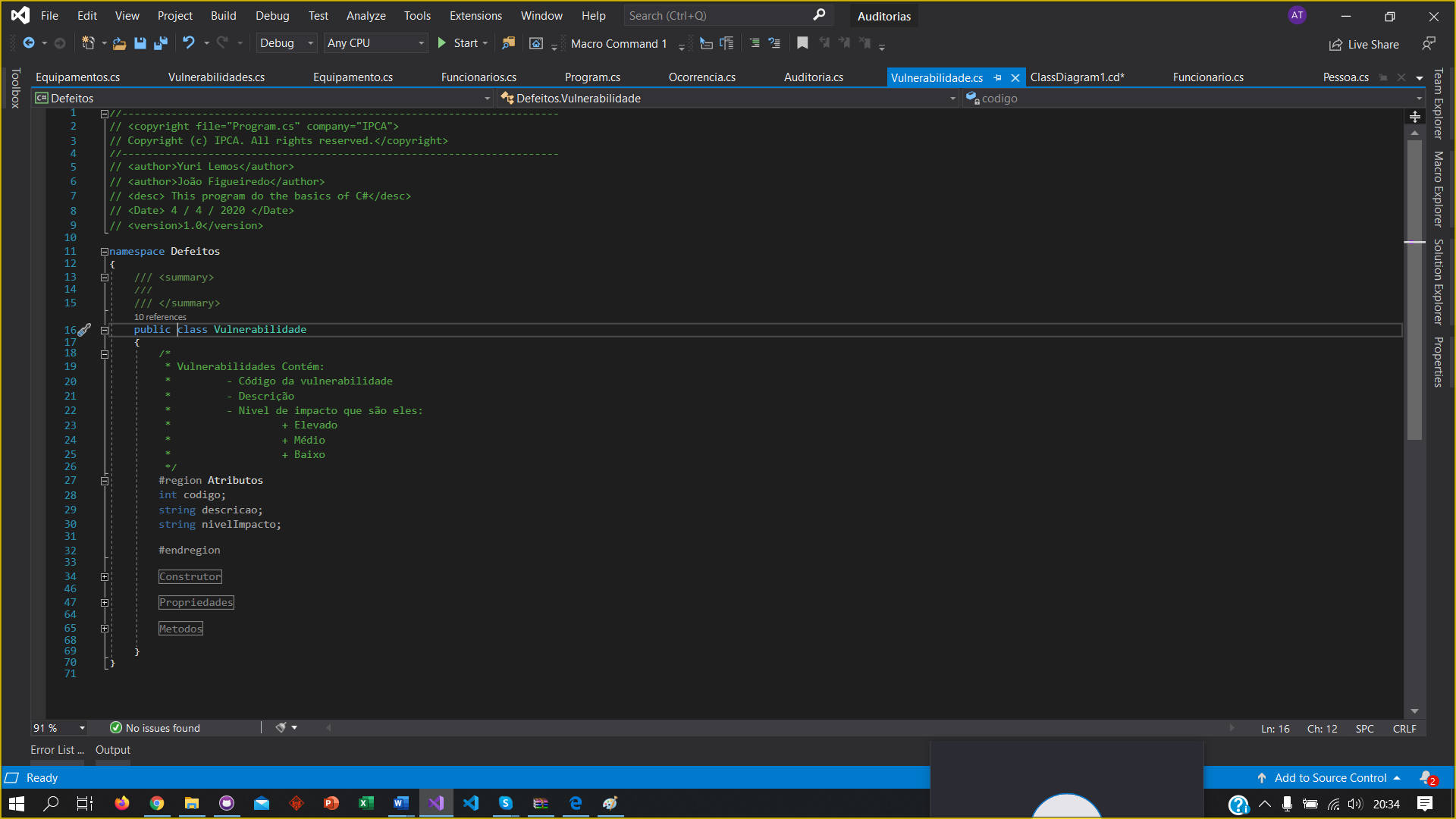


Figura 7 Classe Vulnerabilidade

# Classe Vulnerabilidades

Esta classe representa uma lista de vulnerabilidades, nesta classe contém:

* Lista de vulnerabilidades
* Quantidade de vulnerabilidades

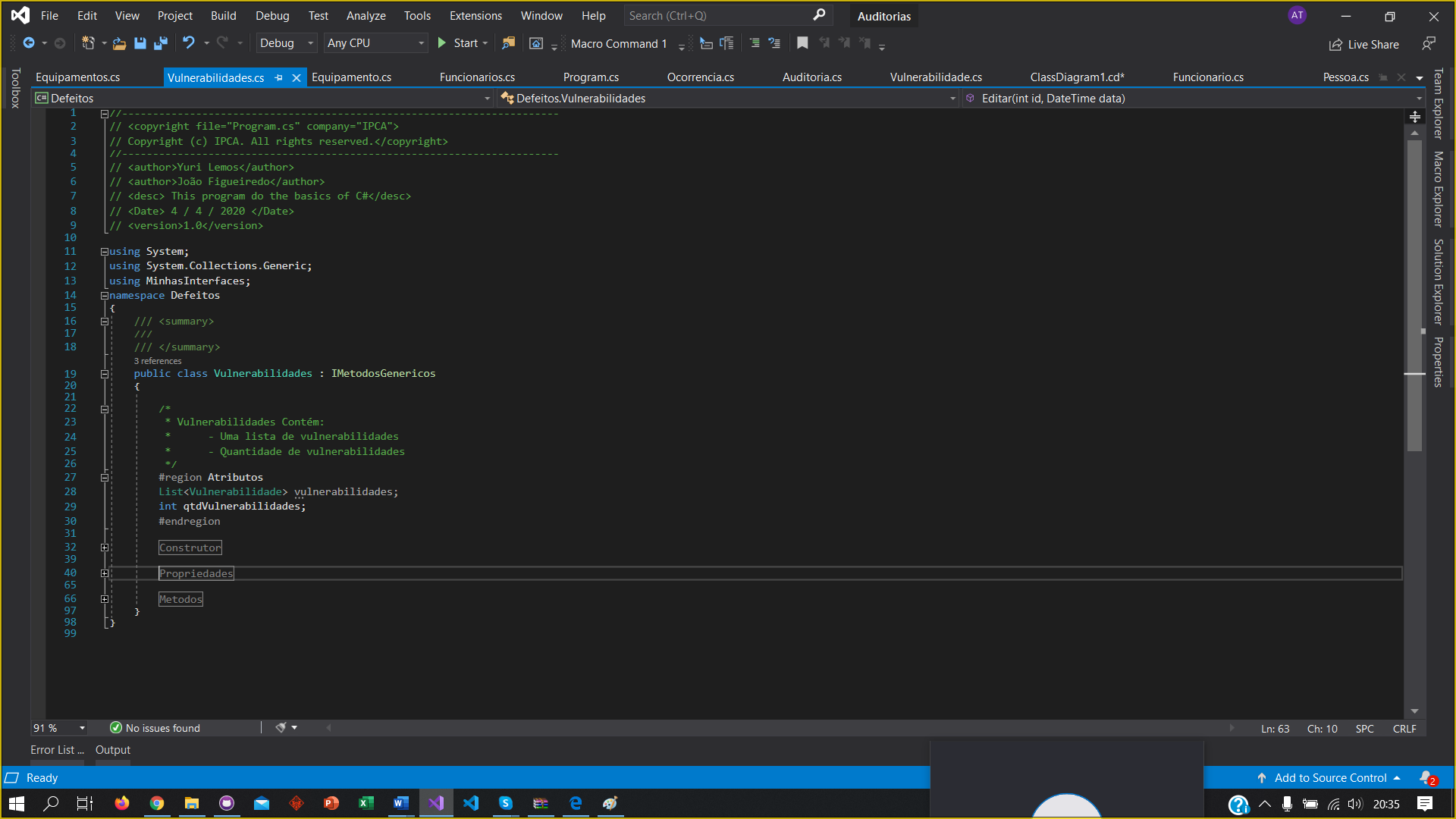


Figura 8 Classe Vulnerabilidades

# Classe Ocorrência

Esta classe representa uma ocorrência, nesta classe contém:

* Código da auditoria
* Código do equipamento
* Código da vulnerabilidade
* Estado

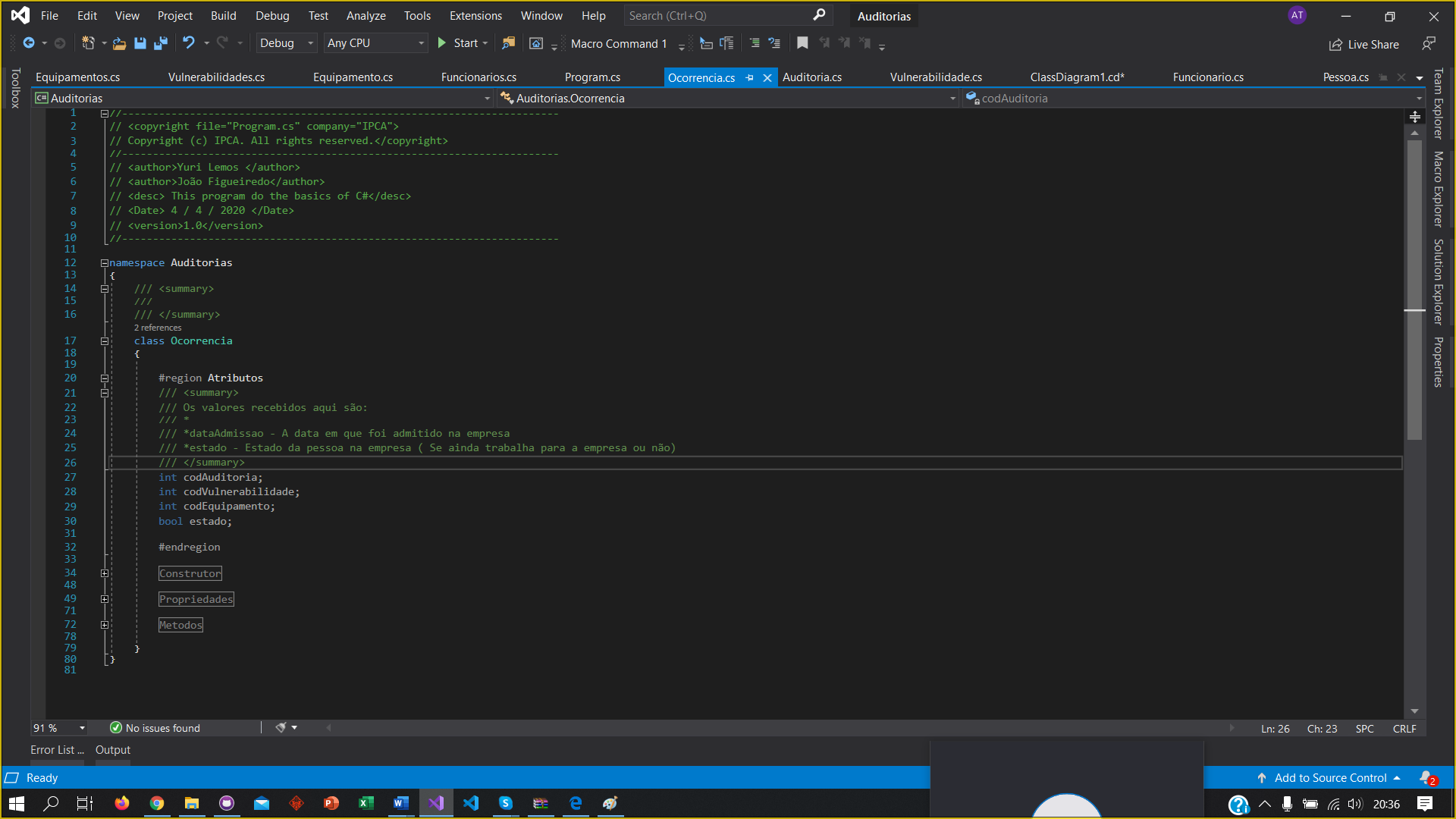


Figura 9 Classe Ocorrência

# Classe Interface IMetodosGenericos

Esta classe é uma interface contendo vários métodos abstratos, estes métodos são:

* Editar
* Procura
* Remove

Estes métodos podem ser utilizados por qualquer classe desde que esta interface seja chamada e que todos seus métodos inseridos na classe que está a implementá-lo.

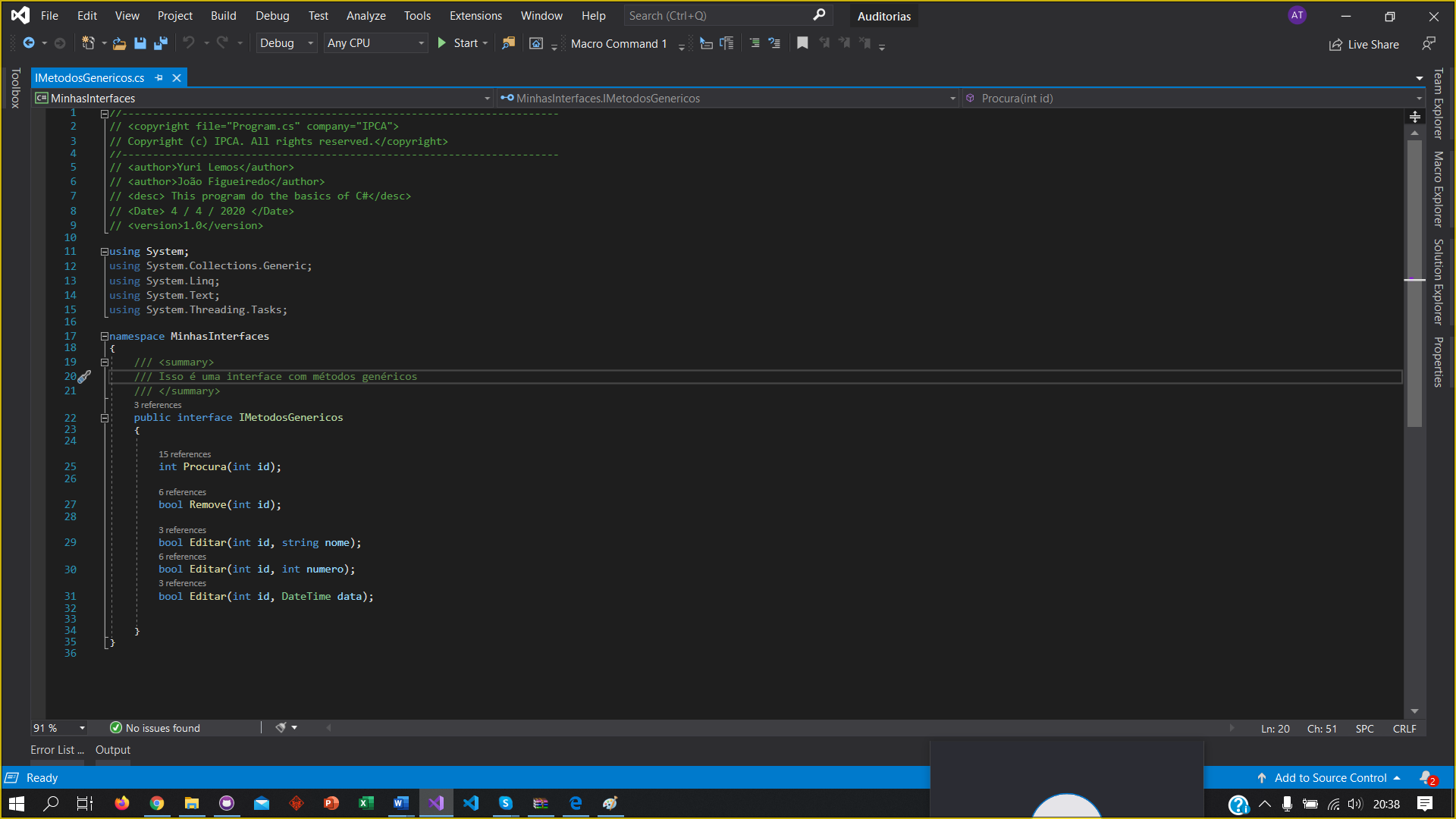


Figura 10 Classe Interface IMetodosGenericos

# Classe Program

Esta classe é onde executa o método principal para inicialização do programa, aqui podem ser criados as instâncias das classes e executar os métodos das mesmas.

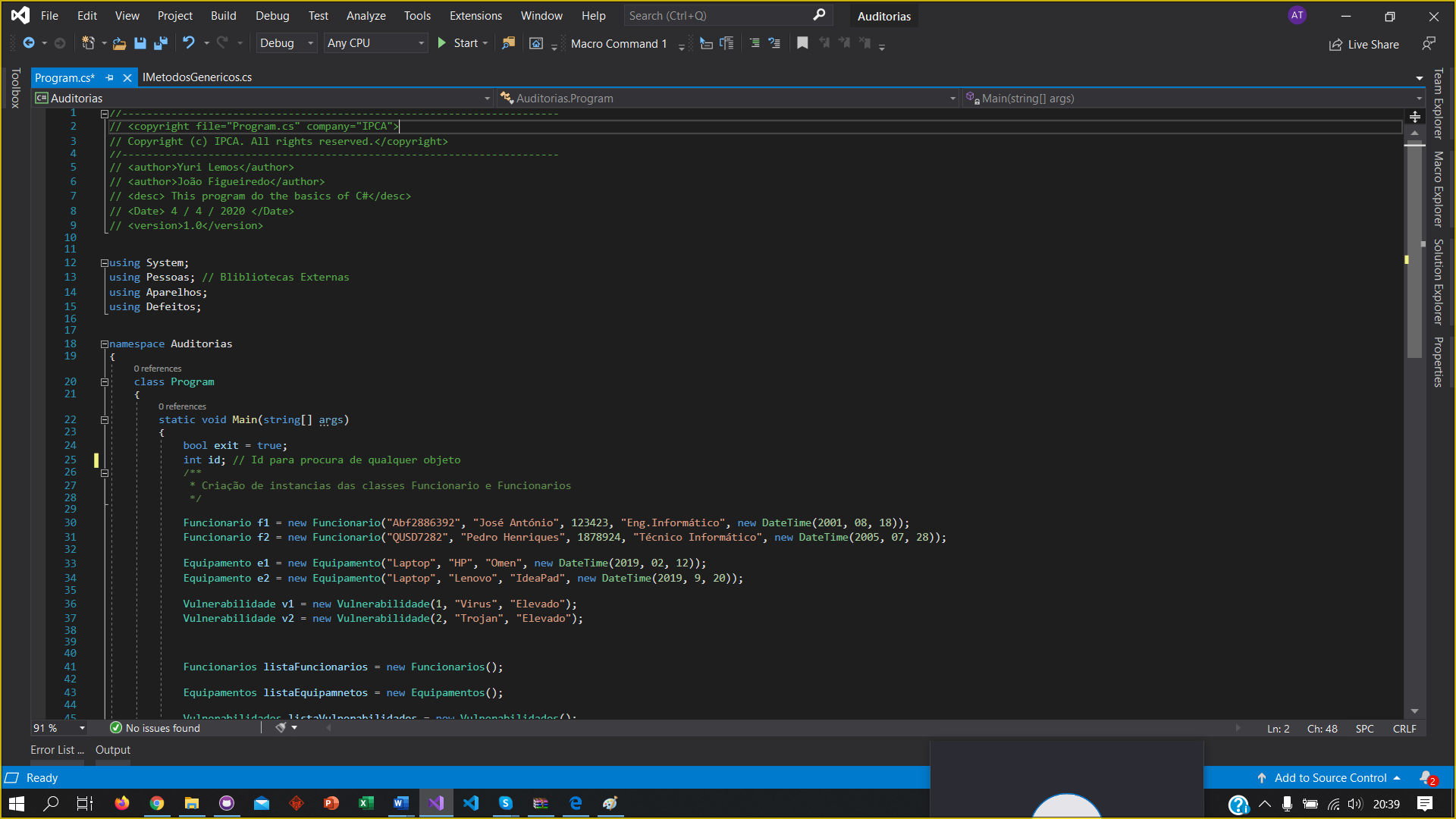


Figura 11 Classe Program

# Conclusão

# Lições aprendidas

Os conhecimentos mais técnicos e teóricos sobre RGPD como algumas normas, legislações e como deve ser implementado ajudou muito a compreender e adquirir mais conhecimento sobre esta área que é a privacidade e proteção de dados pessoais. Isso ajudou muito a ter um olhar mais voltado ao mercado e á área de tecnologia com suas revoluções.

Na esfera da prática, as habilidades e conhecimentos ao desenvolver o

programa, revisando e implementando novos cálculos foram aperfeiçoadas.

# Apreciação Final

A experiência propiciou a real prática das teorias vivenciadas em sala de aula, atribuindo mais gosto pelas disciplinas além de evoluir o raciocínio para adaptar propostas ao desenvolvimento.