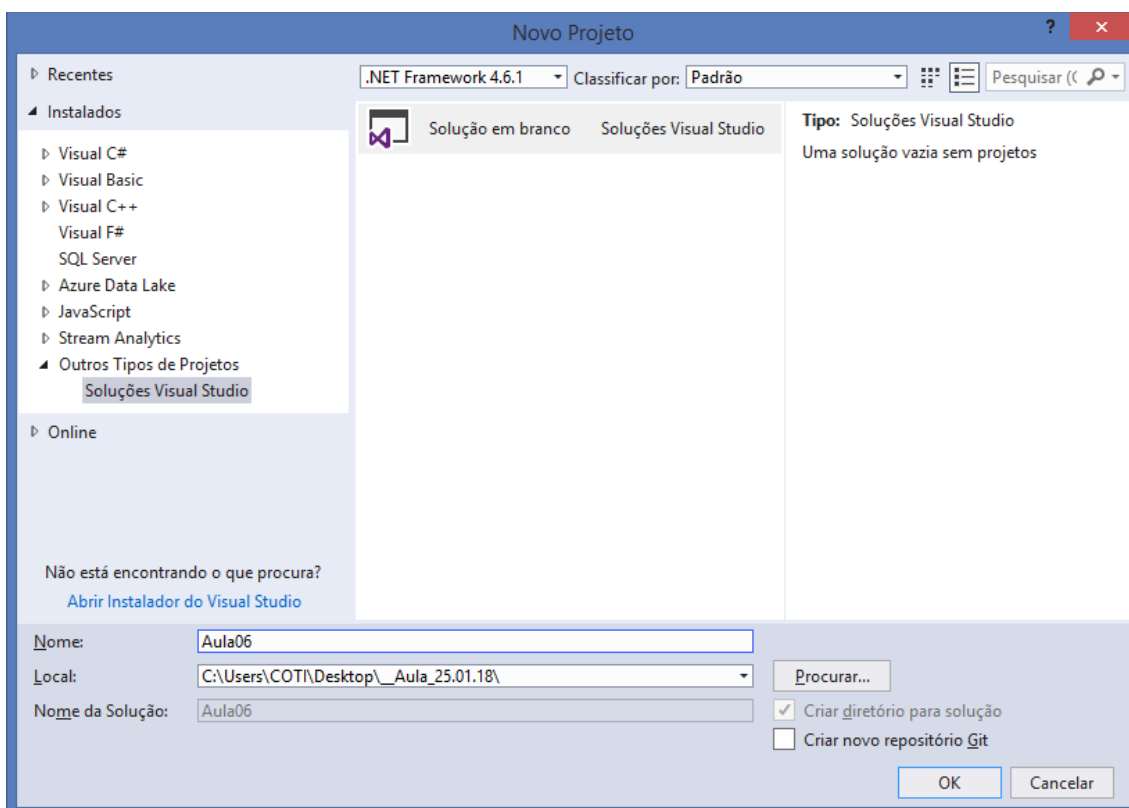
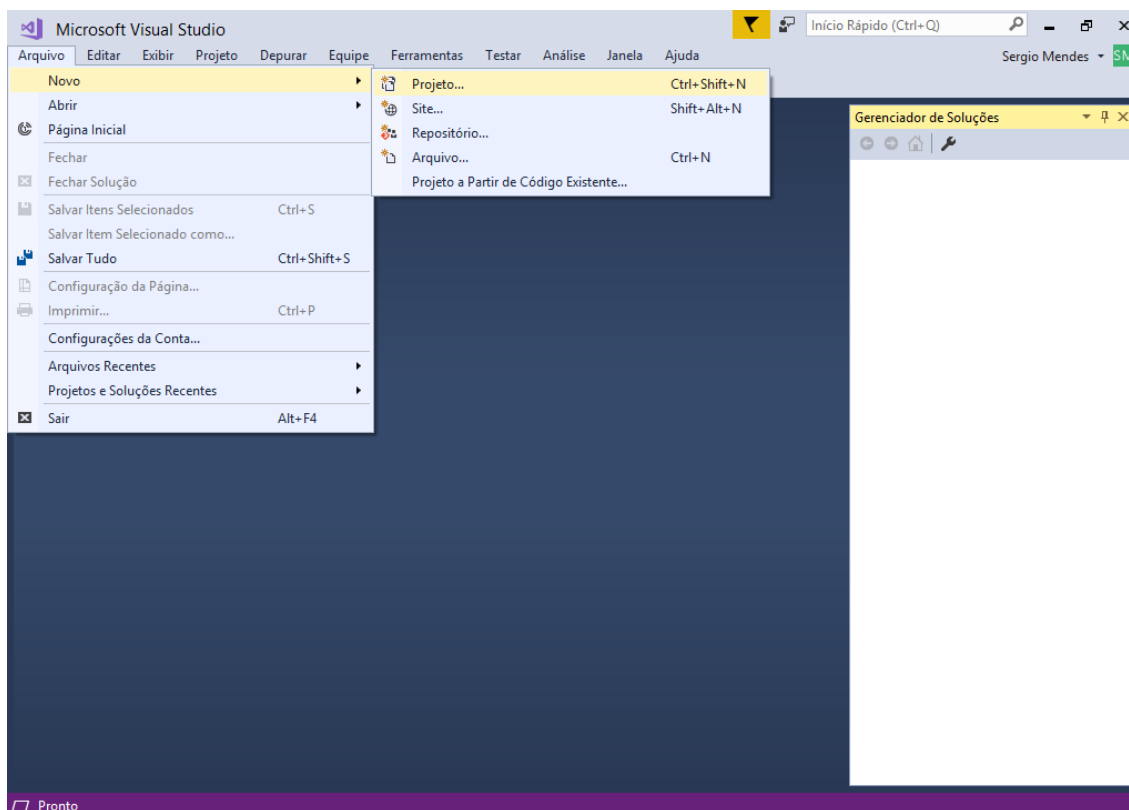
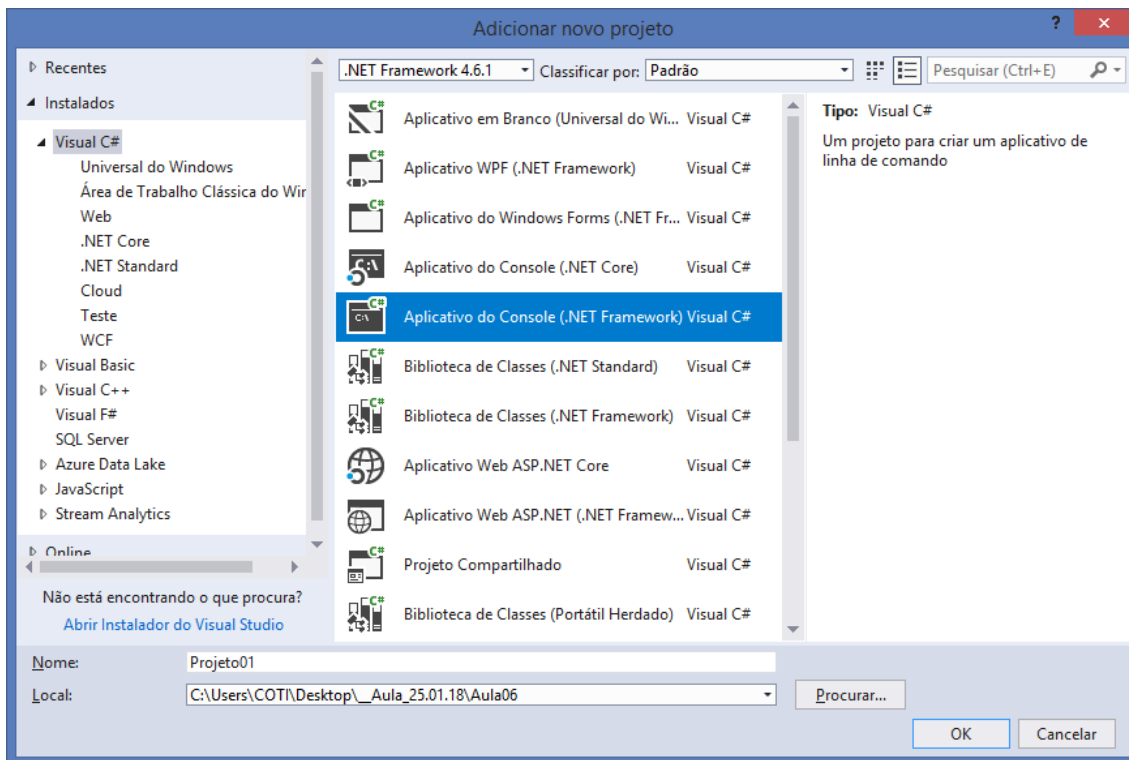


## Criando uma nova solution:

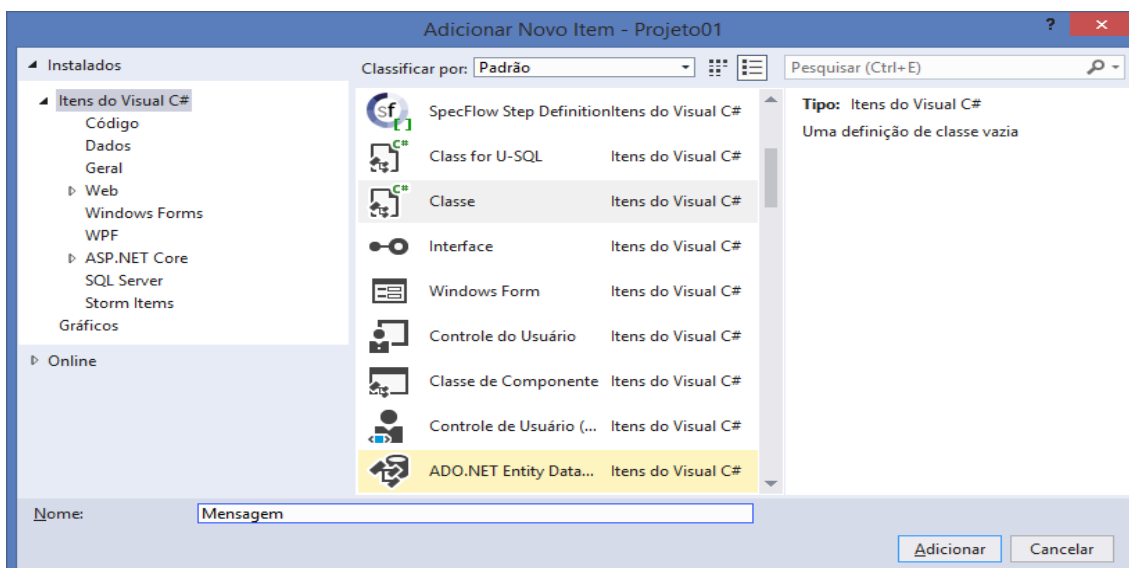


## Criando o projeto **Console Application** Prompt de comando (MS-DOS)



## Classe de entidade

Modelagem para os dados de envio de mensagem



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

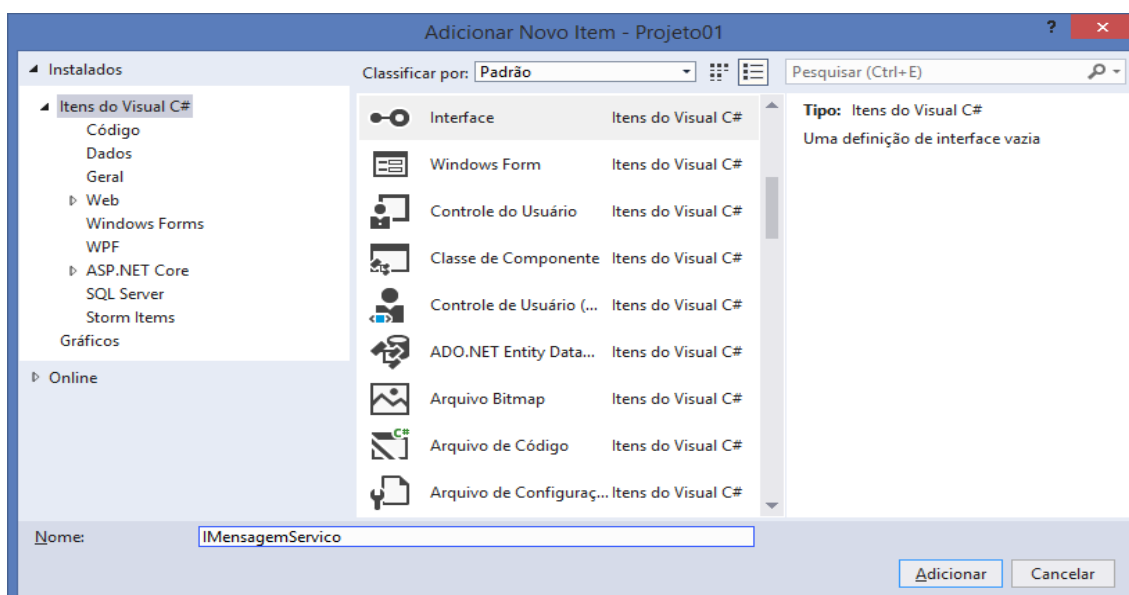
```
namespace Projeto01.Entidades
{
    public class Mensagem
    {
        //propriedades..
        //[prop] + 2x[tab]
        public Guid IdMensagem { get; set; }
        public string EmailDestino { get; set; }
        public string Assunto { get; set; }
        public string Conteudo { get; set; }

        //construtor default
        public Mensagem()
        {
            //vazio..
        }

        //sobrecarga de construtores (overloading)
        public Mensagem(Guid idMensagem, string emailDestino,
            string assunto, string conteudo)
        {
            IdMensagem = idMensagem;
            EmailDestino = emailDestino;
            Assunto = assunto;
            Conteudo = conteudo;
        }

        //sobrescrita do método ToString()..
        public override string ToString()
        {
            return $"{IdMensagem.ToString()}, {EmailDestino},
                {Assunto}, {Conteudo}";
        }
    }
}
```

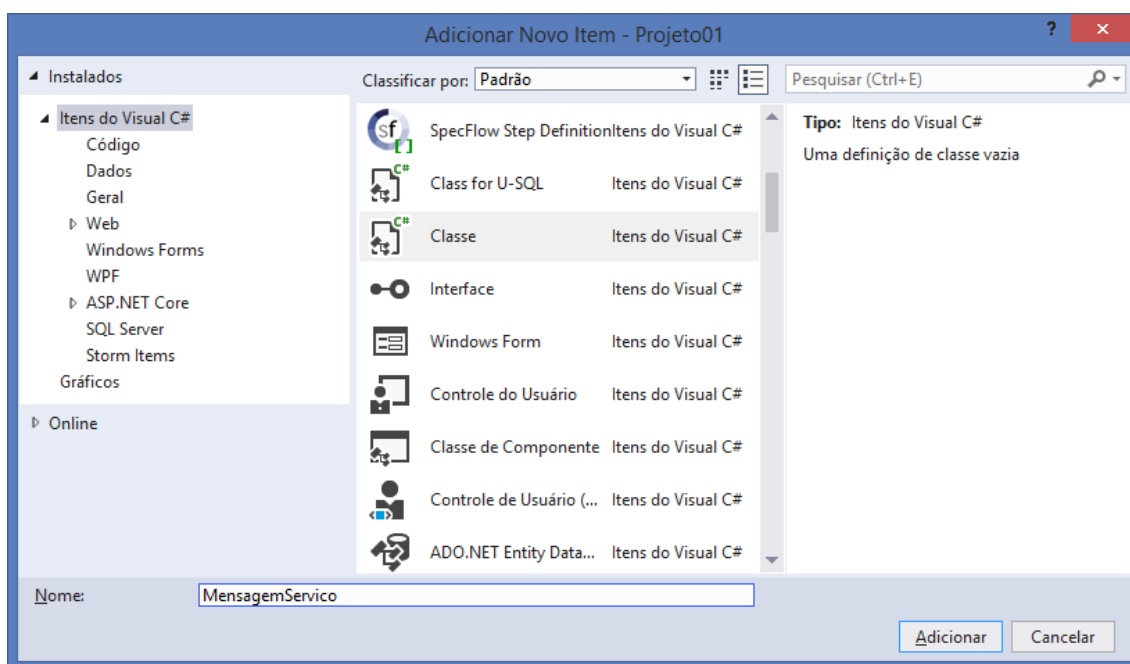
Criando uma interface para definir os métodos abstratos que serão implementados para realizar o envio de email.



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Projeto01.Entidades;

namespace Projeto01.Contratos
{
    public interface IMensagemService
    {
        //método abstrato..
        void EnviarMensagem(Mensagem msg);
    }
}
```

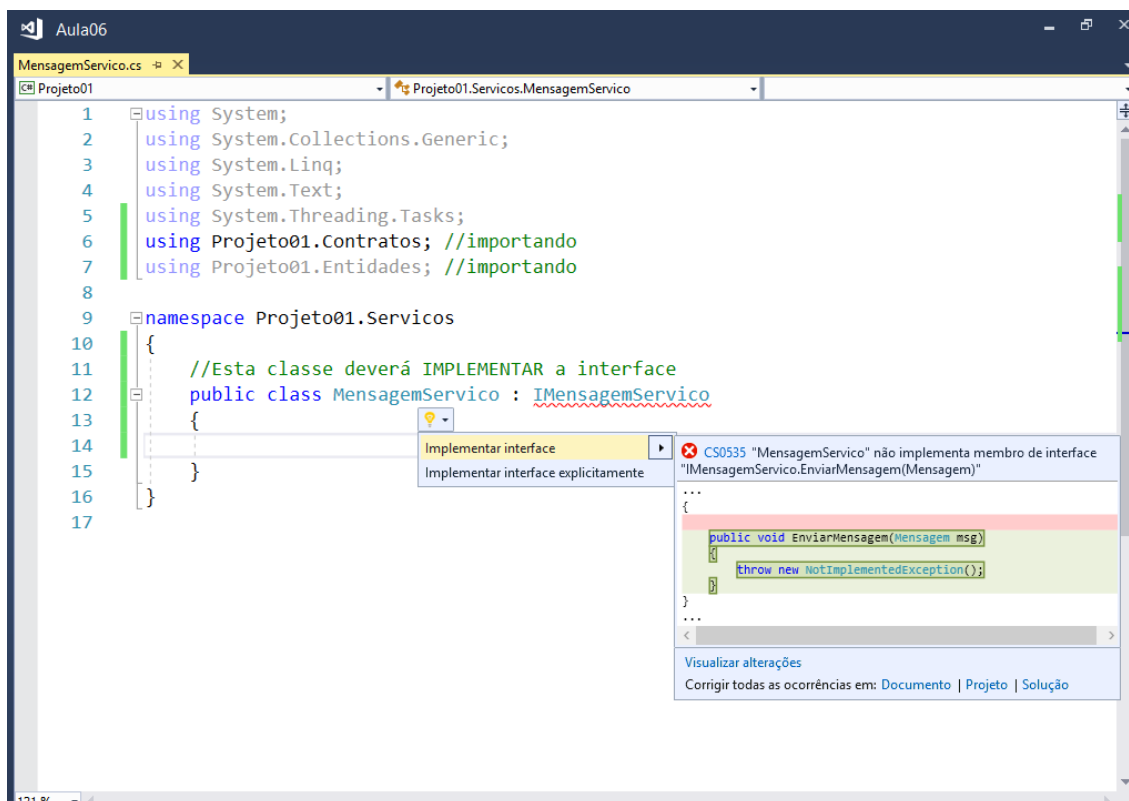
Implementando a interface:



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Projeto01.Contratos; //importando
using Projeto01.Entidades; //importando

namespace Projeto01.Servicos
{
    //Esta classe deverá IMPLEMENTAR a interface
    public class MensagemService : IMensagemService
    {
    }
}
```

## Implementando a interface:



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Projeto01.Contratos; //importando
using Projeto01.Entidades; //importando
using System.Net;
using System.Net.Mail;

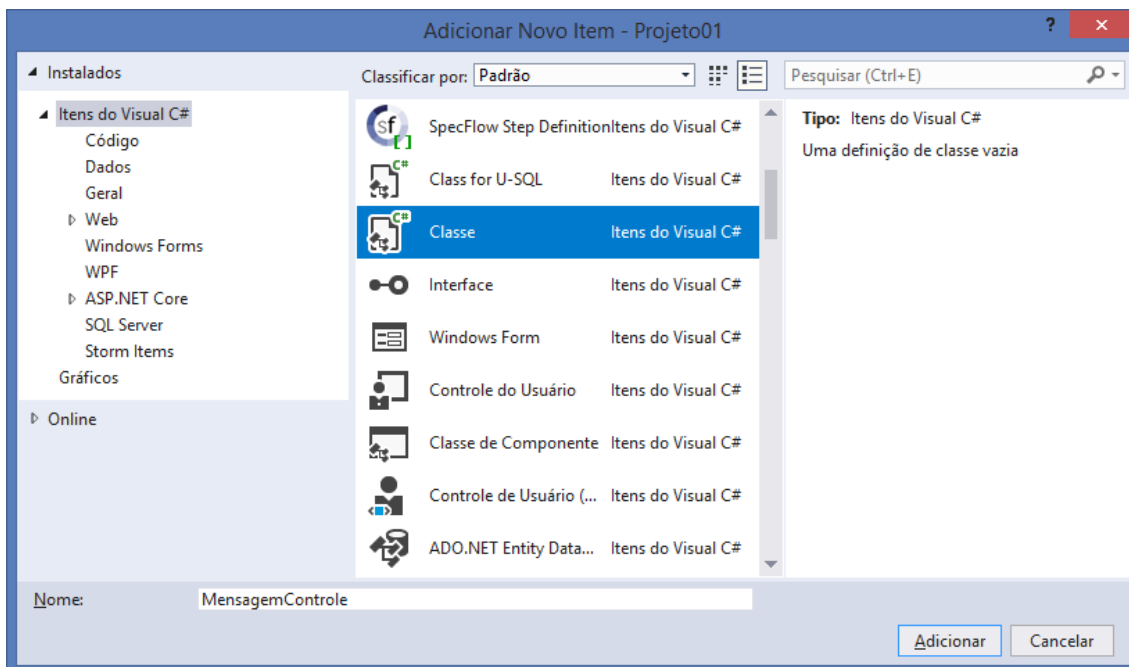
namespace Projeto01.Servicos
{
    //Esta classe deverá IMPLEMENTAR a interface
    public class MensagemServico : IMessageService
    {
        public void EnviarMensagem(Mensagem msg)
        {
            string emailOrigem = "testeaulacoti@gmail.com";
            string senhaOrigem = "coti123456";
        }
    }
}
```

```
//criando a mensagem..
MailMessage mail = new MailMessage(emailOrigem, msg.EmailDestino);
mail.Subject = msg.Assunto;
mail.Body = $"{msg.Conteudo}\n\nID da Mensagem:
                {msg.IdMensagem.ToString()}"

//autenticar no gmail e enviar a mensagem..
SmtpClient smtp = new SmtpClient("smtp.gmail.com", 587);
smtp.EnableSsl = true; //gmail só envia mensagens criptografadas
smtp.Credentials = new NetworkCredential(emailOrigem, senhaOrigem);
smtp.Send(mail); //enviando..
    }
}
}
```

-----

Criando uma classe de controle para realizar a interação com usuário do prompt de comando (DOS)



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Projeto01.Entidades; //importando..
using Projeto01.Servicos; //importando..
```

```
namespace Projeto01.Controles
{
    public class MensagemControle
    {
        //método para realizar o envio de email..
        public void ExecutarEnvioDeMensagem()
        {
            try
            {
                Console.WriteLine("\n - ENVIO DE EMAIL - \n");

                Mensagem msg = new Mensagem(); //instanciando..

                //gerando o id da mensagem..
                msg.IdMensagem = Guid.NewGuid();

                Console.WriteLine("Informe o email de destino..: ");
                msg.EmailDestino = Console.ReadLine();

                Console.WriteLine("Informe o assunto.....: ");
                msg.Assunto = Console.ReadLine();

                Console.WriteLine("Informe o corpo da mensagem.: ");
                msg.Conteudo = Console.ReadLine();

                MensagemServico svc = new MensagemServico();
                svc.EnviaMensagem(msg);

                Console.WriteLine("\nMensagem enviada com sucesso");
                Console.WriteLine(msg.ToString());
            }
            catch (Exception e)
            {
                Console.WriteLine("Erro ao enviar email: " + e.Message);
            }
        }
    }
}
```

## Executando a classe de controle:

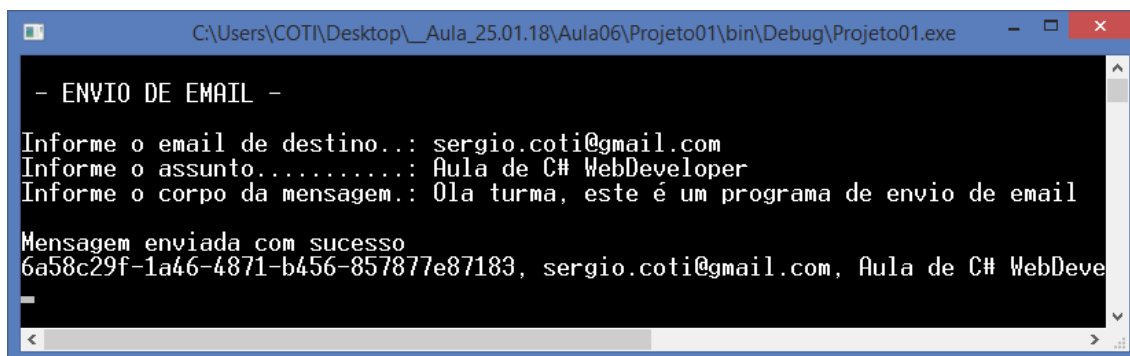
### Classe Program, Método Main()

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Projeto01.Controles; //importando..

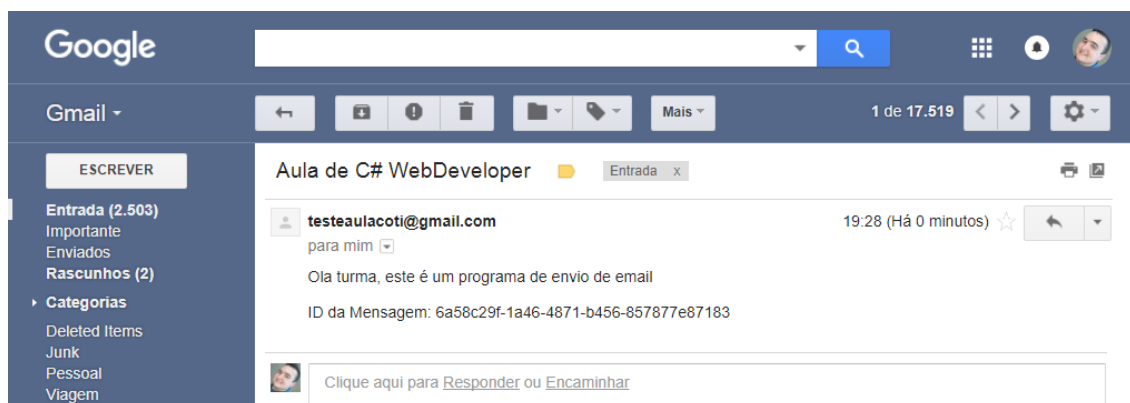
namespace Projeto01
{
    class Program
    {
        static void Main(string[] args)
        {
            MensagemControle mc = new MensagemControle();
            mc.ExecutarEnvioDeMensagem();

            Console.ReadKey(); //pausar..
        }
    }
}
```

### Saida do programa:

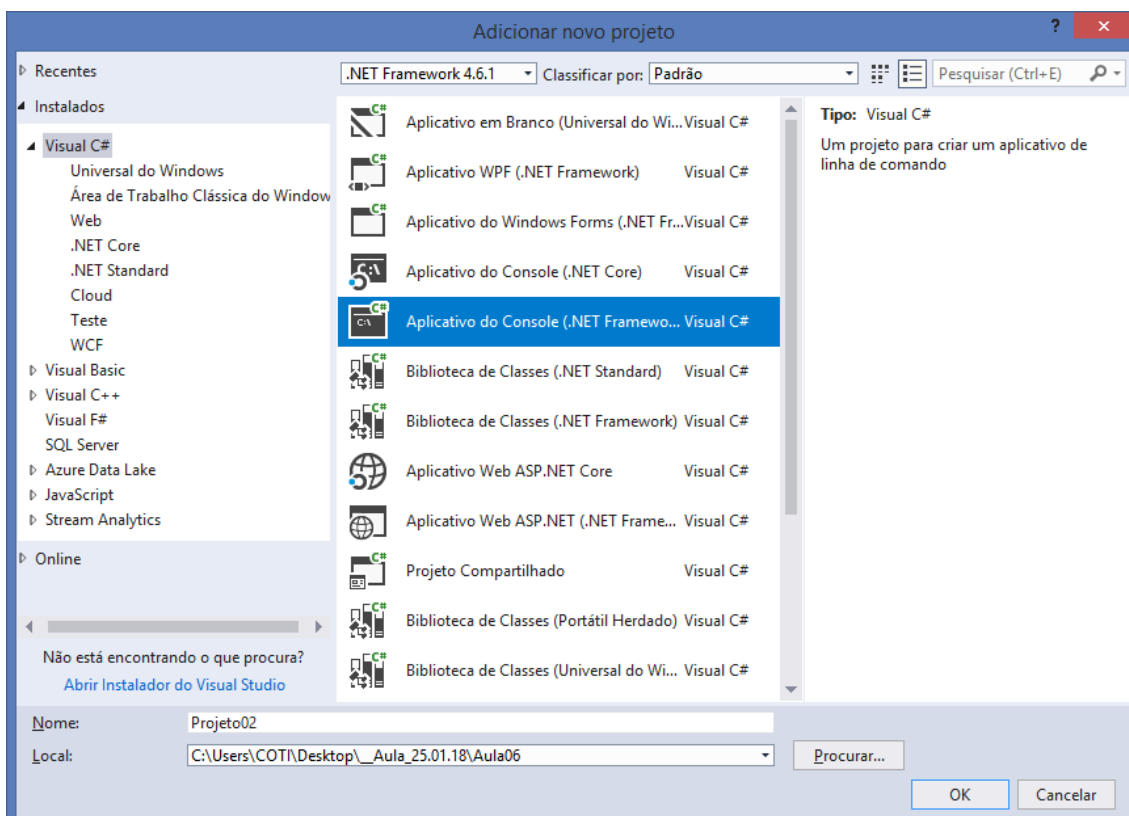
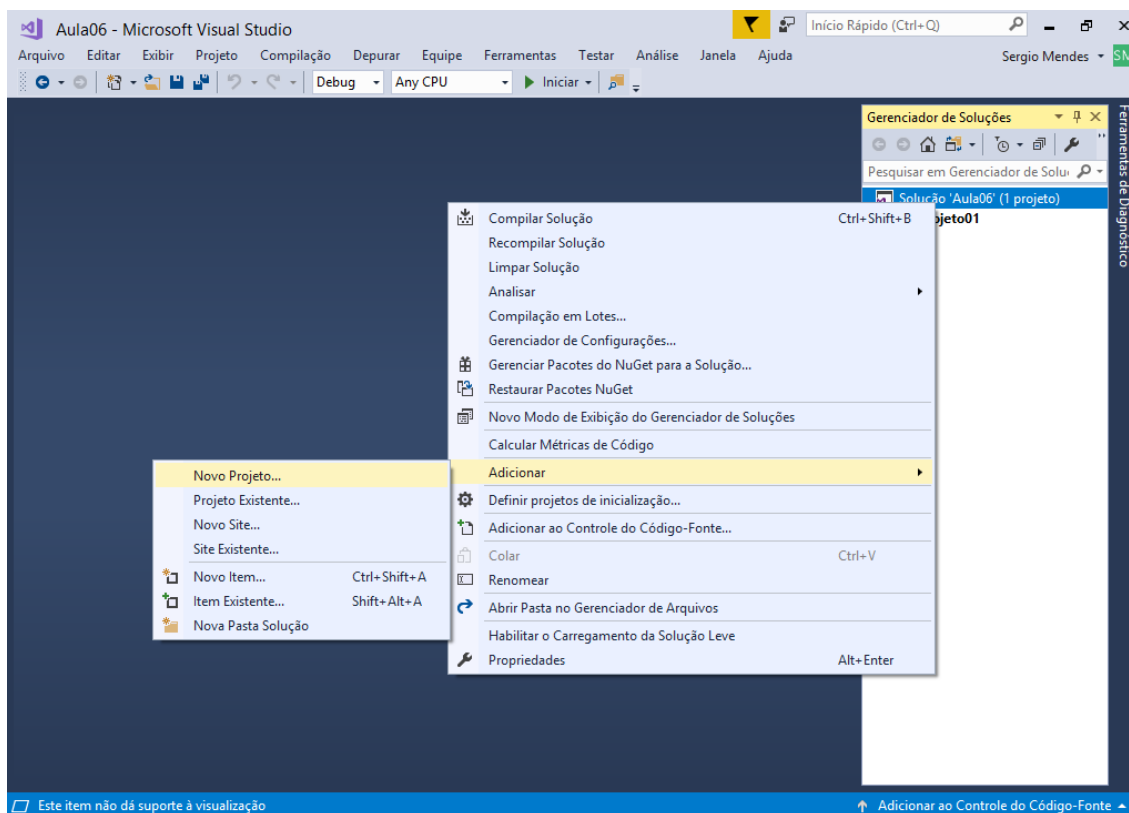


### Email recebido:

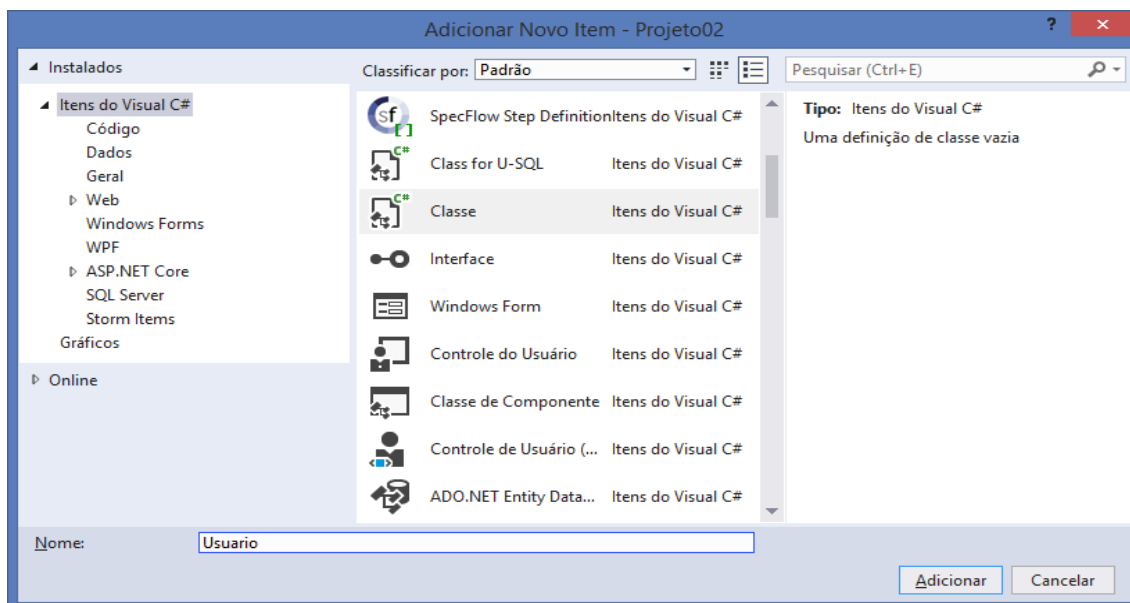




## Abrindo um novo projeto:



## Criando uma classe de entidade:



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Projeto02.Entidades
{
    public class Usuario
    {
        //propriedades..
        public Guid IdUsuario { get; set; }
        public string Login { get; set; }
        public string Senha { get; set; }

        //construtor default..
        public Usuario()
        {
            //vazio..
        }

        //sobrecarga de construtores (entrada de argumentos)
        public Usuario(Guid idUsuario, string login, string senha)
        {
            IdUsuario = idUsuario;
            Login = login;
            Senha = senha;
        }

        //sobrescrita do método ToString()..
        public override string ToString()
        {
            return $"Id: {IdUsuario.ToString()}, Login: {Login}, Senha: {Senha}";
        }
    }
}
```

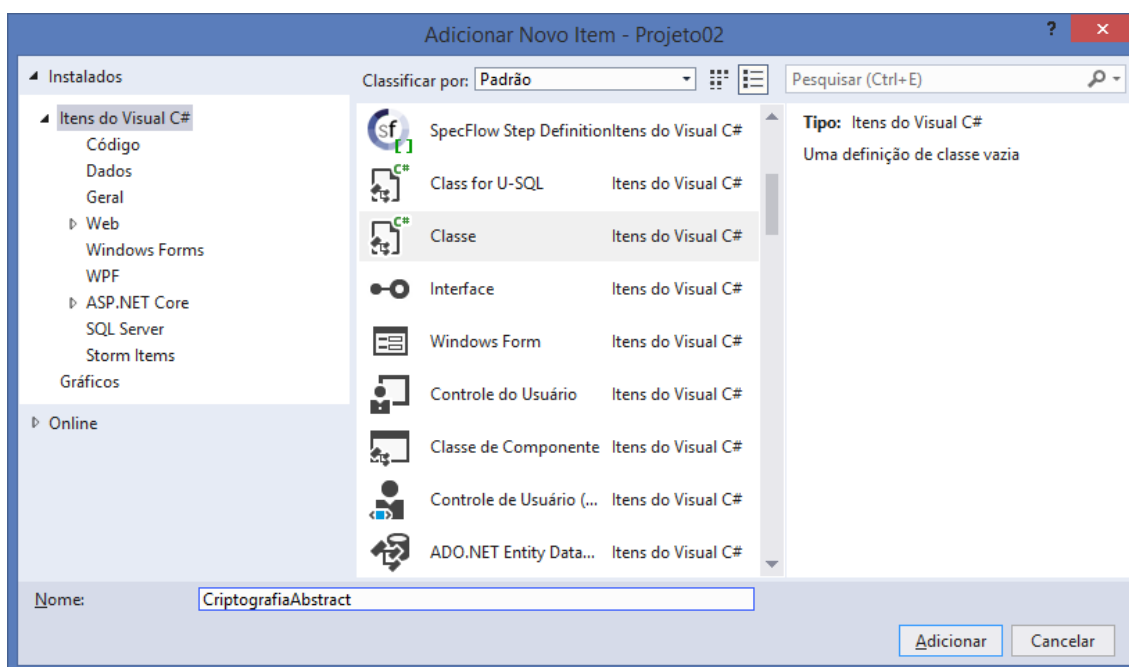
## Classe Abstrata

É um tipo de classe que pode conter atributos, construtores, métodos etc tal qual uma classe "comum". A diferença principal é que uma classe abstrata pode conter métodos abstratos, ou seja, métodos que não possuem corpo, apenas assinatura.

### Exemplo:

Criando uma classe abstrata para criptografia de dados.

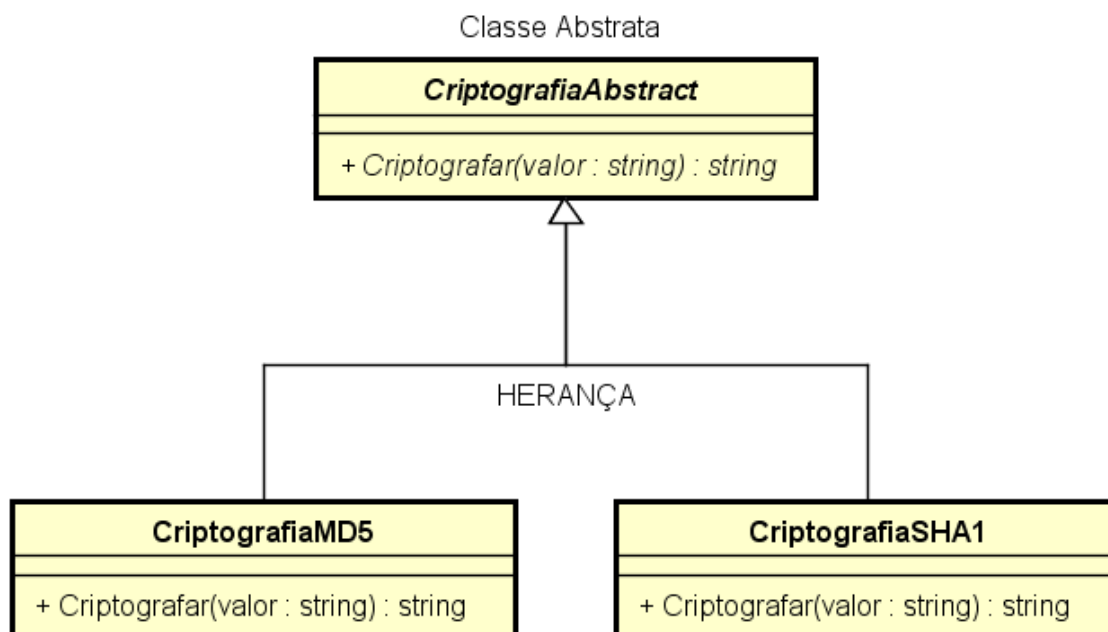
\*\* É uma boa pratica o nome de uma classe abstrata terminar com a palavra Abstract



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
namespace Projeto02.Seguranca
```

```
{
    public abstract class CriptografiaAbstract
    {
    }
}
```

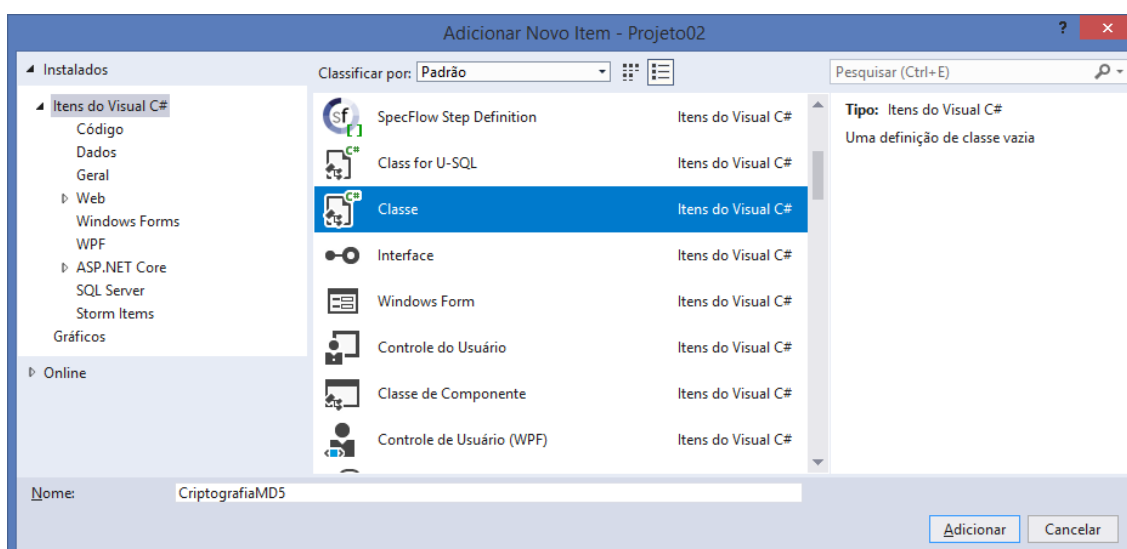


```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Projeto02.Seguranca
{
    public abstract class CriptografiaAbstract
    {
        //método abstrato..
        public abstract string Criptografar(string valor);
    }
}
  
```

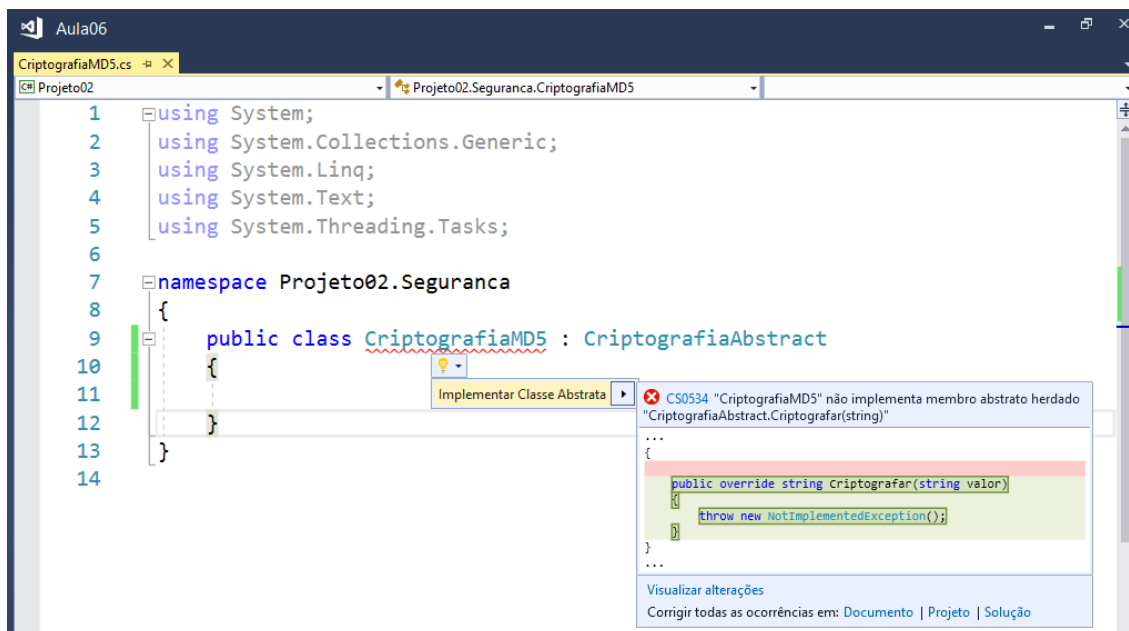
## Criando as classes concretas



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Projeto02.Seguranca
{
    public class CriptografiaMD5 : CriptografiaAbstract
    {
    }
}
```

## Implementando a classe abstrata:



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Projeto02.Seguranca
{
    public class CriptografiaMD5 : CriptografiaAbstract
    {
        public override string Criptografar(string valor)
        {
            throw new NotImplementedException();
        }
    }
}
```

## MD5 (Message Digest 5)

Algoritmo de criptografia baseado em HASH, ou seja, retorna o valor encriptado como uma string de 32 dígitos hexadecimal.  
O MD5 está na categoria de algoritmos de criptografia que não pode ser descriptografado.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Security.Cryptography; //criptografia..

namespace Projeto02.Seguranca
{
    public class CriptografiaMD5 : CriptografiaAbstract
    {
        public override string Criptografar(string valor)
        {
            //instanciando a classe de criptografia..
            MD5 md5 = new MD5CryptoServiceProvider();

            //converter o valor que será encriptado para bytes..
            byte[] valorEmBytes = Encoding.UTF8.GetBytes(valor);

            //executo a criptografia..
            byte[] hash = md5.ComputeHash(valorEmBytes);

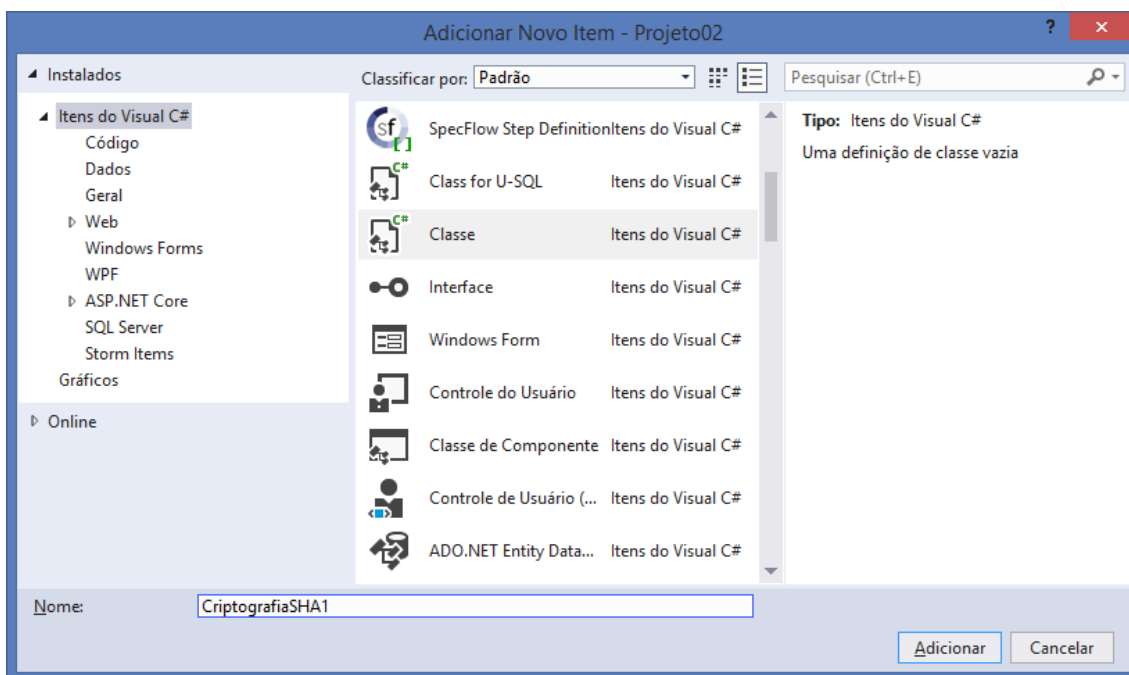
            //converter cada elemento de byte para string..
            string resultado = string.Empty;

            foreach(byte b in hash) //varrendo..
            {
                resultado += b.ToString("X"); //X -> Hexadecimal..
            }

            return resultado; //retornando..
        }
    }
}
```

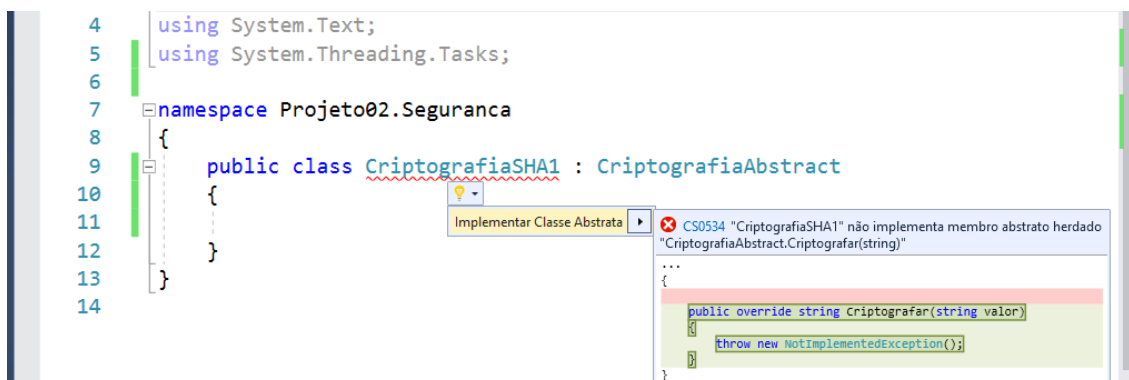
## SHA1

Algoritmo de criptografia baseado em HASH, ou seja, retorna o valor encriptado como uma string de 40 dígitos hexadecimal.  
O SHA1 também está na categoria de algoritmos de criptografia que não pode ser descryptografado.



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Projeto02.Seguranca
{
    public class CriptografiaSHA1 : CriptografiaAbstract
    {
    }
}
```



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Security.Cryptography;

namespace Projeto02.Seguranca
{
    public class CriptografiaSHA1 : CriptografiaAbstract
    {
        public override string Criptografar(string valor)
        {
            //instanciando a classe de criptografia..
            SHA1 sha1 = new SHA1CryptoServiceProvider();

            //converter o valor que será encriptado para bytes..
            byte[] valorEmBytes = Encoding.UTF8.GetBytes(valor);

            //executo a criptografia..
            byte[] hash = sha1.ComputeHash(valorEmBytes);

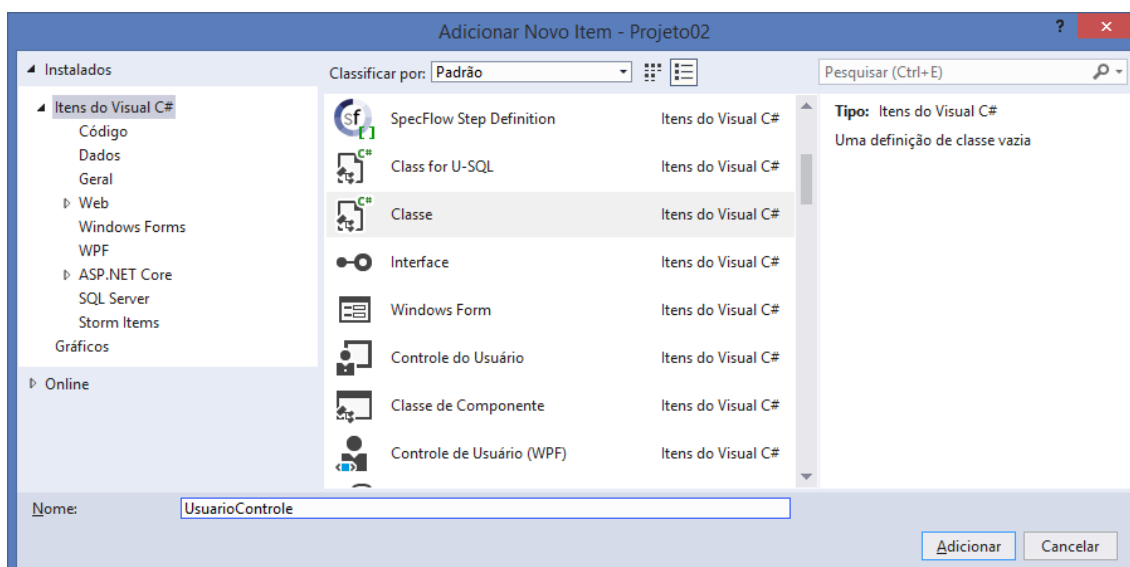
            //converter cada elemento de byte para string..
            string resultado = string.Empty;

            foreach (byte b in hash) //varrendo..
            {
                resultado += b.ToString("X"); //X -> Hexadecimal..
            }

            return resultado; //retornando..
        }
    }
}
```

-----

Classe de controle:





```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Projeto02.Entidades; //importando..
using Projeto02.Seguranca; //importando..

namespace Projeto02.Controles
{
    public class UsuarioControle
    {
        //método para ler e imprimir os dados do usuario..
        public void ObterDadosUsuario()
        {
            try
            {
                Console.WriteLine("\n - CONTROLE DE USUARIOS - \n");

                //instanciando um objeto da classe usuario..
                Usuario u = new Usuario();

                //definindo um ID para o usuario utilizando o GUID..
                u.IdUsuario = Guid.NewGuid();

                Console.WriteLine("Informe o Login do Usuário....: ");
                u.Login = Console.ReadLine();

                Console.WriteLine("Informe (1)MD5 ou (2)SHA1.....: ");
                int opcao = int.Parse(Console.ReadLine());

                //declarando um objeto da classe abstrata..
                CriptografiaAbstract c = null; //sem espaço de memória..

                switch(opcao)
                {
                    case 1:
                        c = new CriptografiaMD5(); //POLIMORFISMO!
                        break;
                }
            }
        }
    }
}
```

```

        case 2:
            c = new CriptografiaSHA1(); //POLIMORFISMO!
            break;
    }

    Console.WriteLine("Informe a Senha do Usuário....: ");
    u.Senha = c.Criptografar(Console.ReadLine());

    //imprimir os dados..
    Console.WriteLine("\nDados do Usuário:");
    Console.WriteLine("ID.....: " + u.IdUsuario.ToString());
    Console.WriteLine("Login....: " + u.Login);
    Console.WriteLine("Senha....: " + u.Senha);
}
catch(Exception e)
{
    //imprimir mensagem de erro..
    Console.WriteLine("Erro: " + e.Message);
}
}
}
}

```

## Polimorfismo

Ocorre quando um objeto executa um método que pode em tempo de execução apresentar resultados diferentes baseado na instancia da classe.

Exemplo:

Abaixo, declaramos um objeto 'c' para a classe abstrata mas não instanciamos este objeto, ou seja, seu valor é 'null'

**CriptografiaAbstract c = null;**

[Classe Abstrata]

[Objeto]

[Vazio]

Abaixo, instanciamos o objeto 'c' baseado nas classes que herdaram e implementaram a classe abstrata:

**c = new CriptografiaMD5();**

Ou

**c = new CriptografiaSHA1();**

Desta forma, quando o objeto 'c' executar o método Criptografar, o seu resultado irá depender da forma como o objeto foi instanciado (MD5 ou SHA1)

```
CriptografiaAbstract c = null;
//sem espaço de memória..

switch(opcao)
{
    case 1:
        c = new CriptografiaMD5(); //POLIMORFISMO!
        break;

    case 2:
        c = new CriptografiaSHA1(); //POLIMORFISMO!
        break;
}
```

### Executando no método Main()

Classe: Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Projeto02.Controles; //importando..

namespace Projeto02
{
    class Program
    {
        static void Main(string[] args)
        {
            UsuarioControle uc = new UsuarioControle();
            uc.ObterDadosUsuario(); //executando..

            Console.ReadKey(); //pausando..
        }
    }
}
```

## Executando:

```

C:\Users\COTI\Desktop\_Aula_25.01.18\Aula06\Projeto02\bin\Debug\Projeto02.exe

- CONTROLE DE USUARIOS -

Informe o Login do Usuário....: smendes
Informe (1)MD5 ou (2)SHA1.....: 1
Informe a Senha do Usuário....: adminadmin

Dados do Usuário:
ID.....: 00f6ad5d-8e91-4cae-8572-ceb01d8e49bf
Login...: smendes
Senha...: F6FDFFE48C908DEBF4C3BD36C32E72
  
```

```

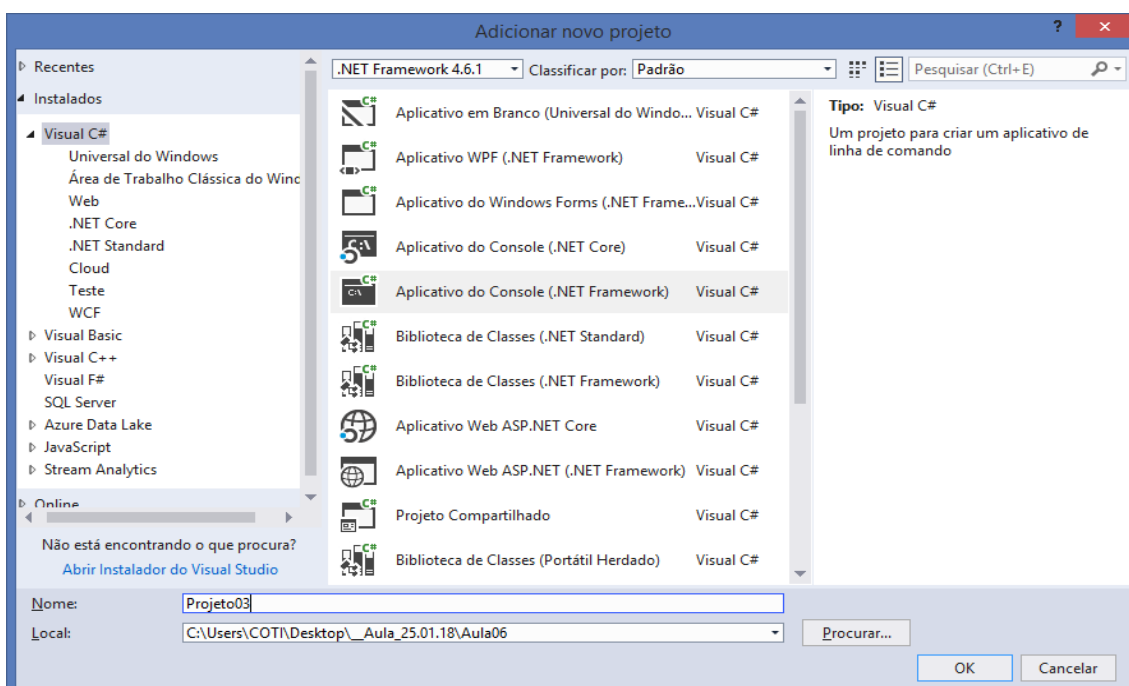
C:\Users\COTI\Desktop\_Aula_25.01.18\Aula06\Projeto02\bin\Debug\Projeto02.exe

- CONTROLE DE USUARIOS -

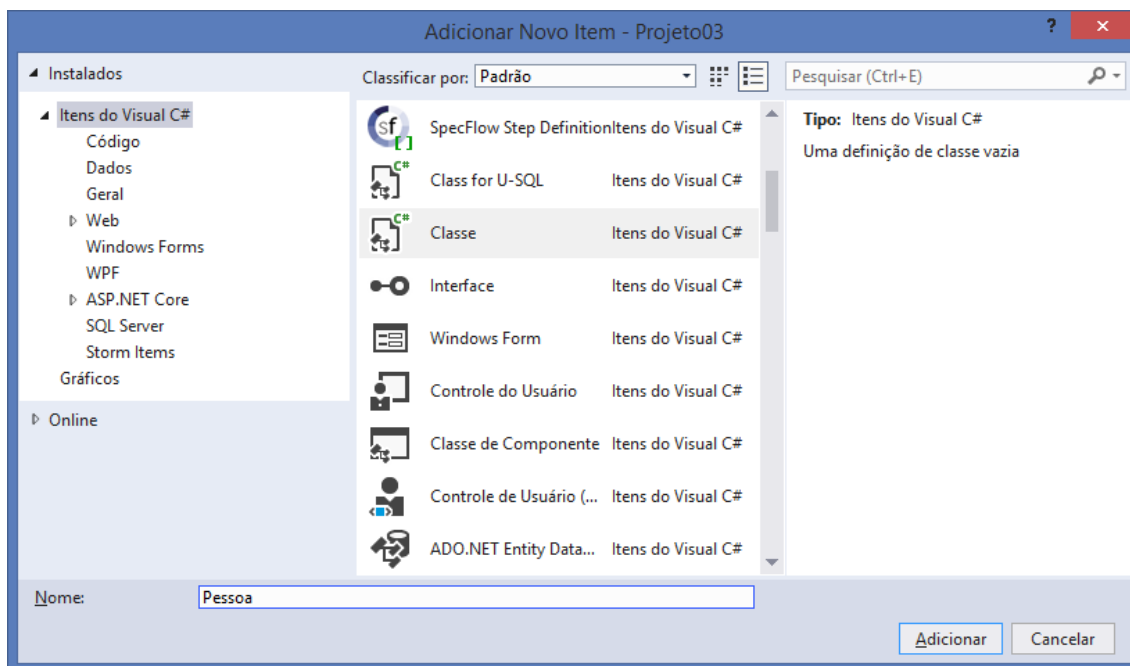
Informe o Login do Usuário....: smendes
Informe (1)MD5 ou (2)SHA1.....: 2
Informe a Senha do Usuário....: adminadmin

Dados do Usuário:
ID.....: bd2e470e-6ad0-4cc5-b9e0-2f845c3c9c28
Login...: smendes
Senha...: DD94709528BB1C83D08F3088D443F4742891F4F
  
```

## Novo projeto:



Classe de entidade: **Pessoa**



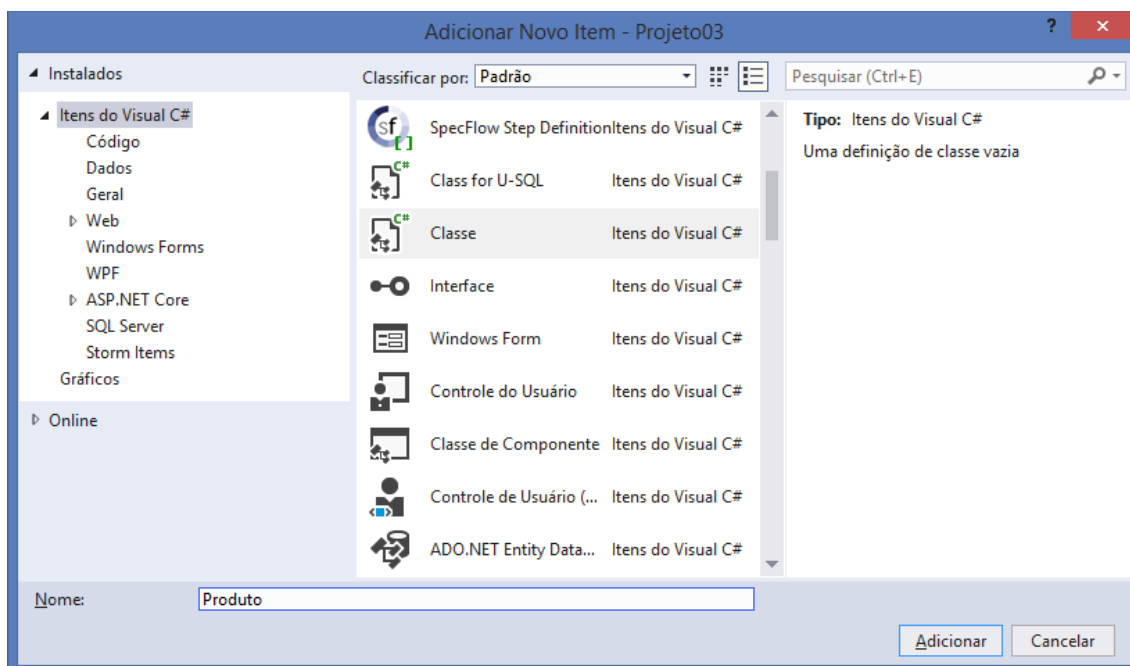
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Projeto03.Entidades
{
    public class Pessoa
    {
        public int IdPessoa { get; set; }
        public string Nome { get; set; }

        public Pessoa()
        {
        }

        public Pessoa(int idPessoa, string nome)
        {
            IdPessoa = idPessoa;
            Nome = nome;
        }

        public override string ToString()
        {
            return $"Id Pessoa: {IdPessoa}, Nome: {Nome}";
        }
    }
}
```



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Projeto03.Entidades
{
    public class Produto
    {
        public int IdProduto { get; set; }
        public string Nome { get; set; }
        public decimal Preco { get; set; }

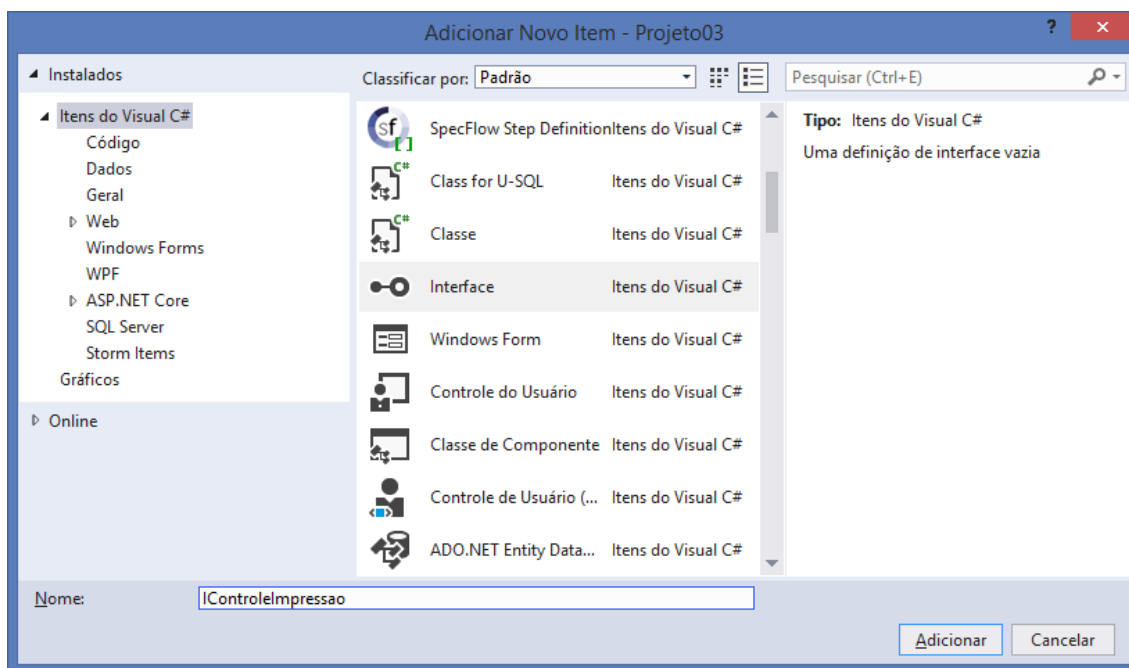
        public Produto()
        {
        }

        public Produto(int idProduto, string nome, decimal preco)
        {
            IdProduto = idProduto;
            Nome = nome;
            Preco = preco;
        }

        public override string ToString()
        {
            return $"Id Produto: {IdProduto}, Nome: {Nome}, Preço: {Preco}";
        }
    }
}
```

## Tipos Genéricos <T>

São tipos de dados que podemos declarar em classes e interfaces para representar qualquer tipo de classe, estrutura etc em C#



```
using System;
```

```
using System.Collections.Generic;
```

```
using System.Linq;
```

```
using System.Text;
```

```
using System.Threading.Tasks;
```

```
namespace Projeto03.Contratos
```

```
{
```

```
    // <T> Tipo Genérico..
```

```
    public interface IControleImpressao<T>
```

```
    {
```

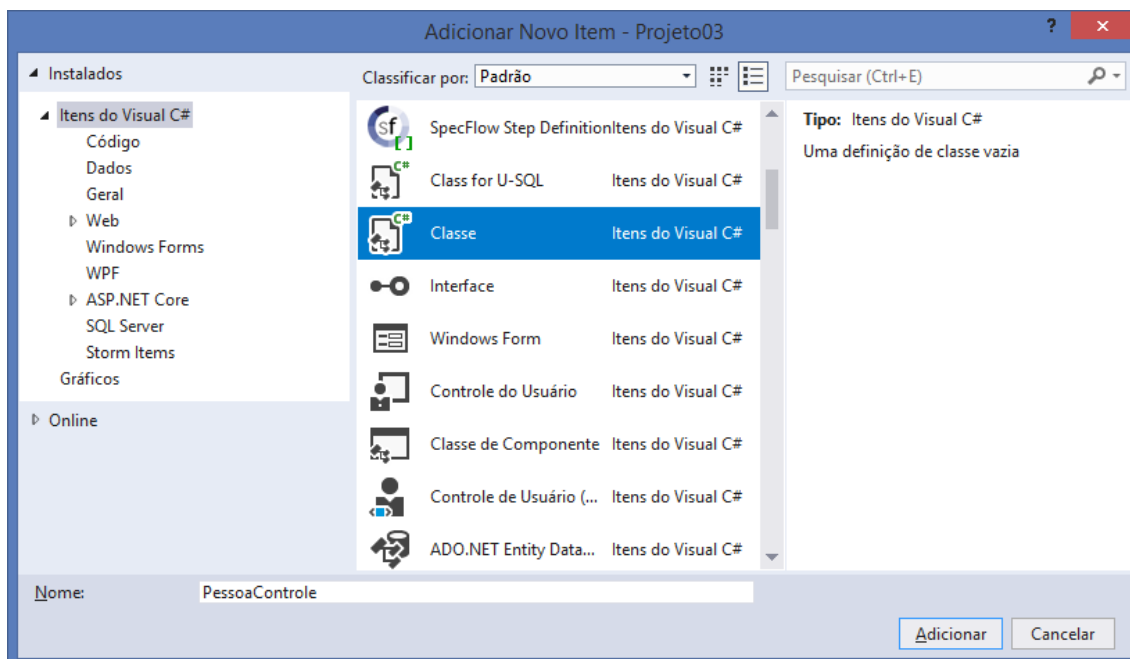
```
        // método abstrato..
```

```
        void ImprimirDados(T obj);
```

```
    }
```

```
}
```

## Implementando:

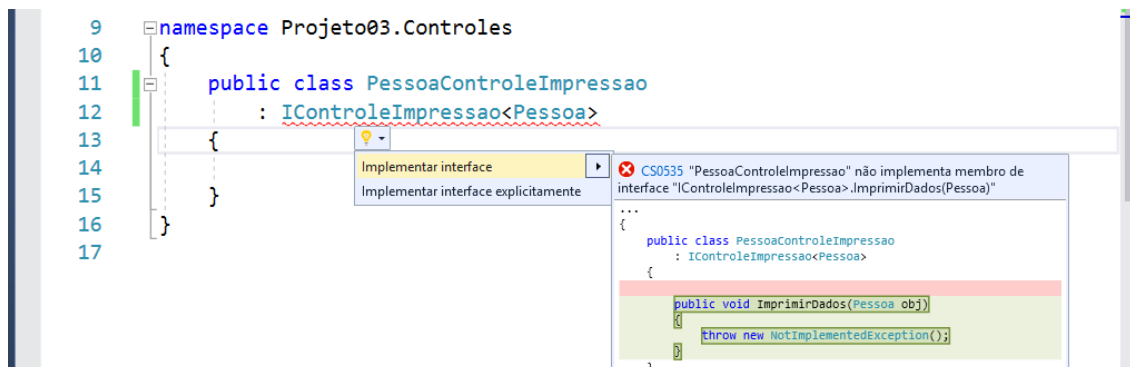


```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Projeto03.Contratos;
using Projeto03.Entidades;
```

```
namespace Projeto03.Controles
```

```
{
    public class PessoaControlImpressao
        : IControlImpressao<Pessoa>
    {
    }
}
```

## Implementando:

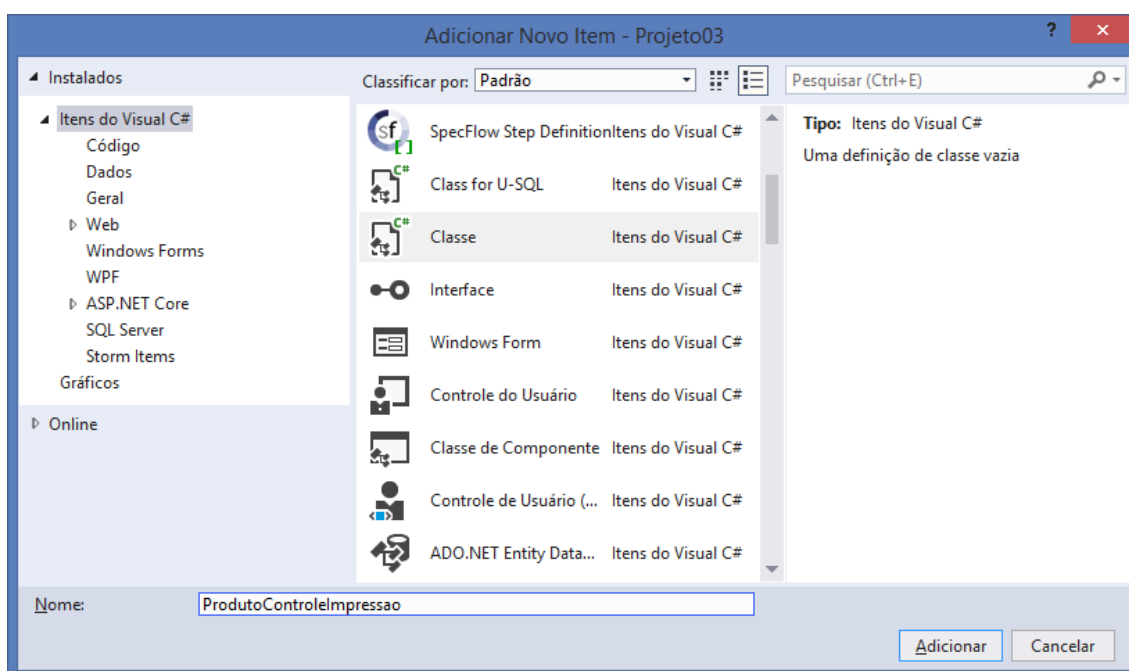




```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Projeto03.Contratos;
using Projeto03.Entidades;

namespace Projeto03.Controles
{
    public class PessoaControleImpressao
        : IControleImpressao<Pessoa>
    {
        public void ImprimirDados(Pessoa obj)
        {
            Console.WriteLine("\nDados de Pessoa:\n");

            Console.WriteLine("Id da Pessoa.: " + obj.IdPessoa);
            Console.WriteLine("Nome.....: " + obj.Nome);
        }
    }
}
```



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Projeto03.Entidades; //importando..
using Projeto03.Contratos; //importando..

namespace Projeto03.Controles
{
    public class ProdutoControleImpressao
        : IControleImpressao<Produto>
    {

```

```

        public void ImprimirDados(Produto obj)
        {
            Console.WriteLine("\nDados do Produto:\n");

            Console.WriteLine("Id do Produto...: " + obj.IdProduto);
            Console.WriteLine("Nome.....: " + obj.Nome);
            Console.WriteLine("Preço.....: " + obj.Preco);
        }
    }
}

```

## Executando:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Projeto03.Entidades;
using Projeto03.Controles;

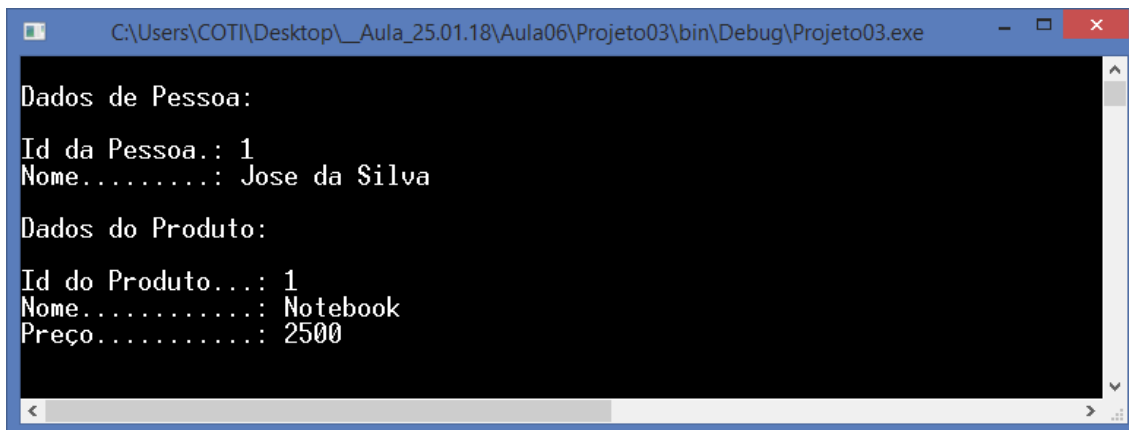
namespace Projeto03
{
    class Program
    {
        static void Main(string[] args)
        {
            Pessoa pessoa = new Pessoa(1, "Jose da Silva");
            Produto produto = new Produto(1, "Notebook", 2500);

            PessoaControleImpressao c1 = new PessoaControleImpressao();
            c1.ImprimirDados(pessoa);

            ProdutoControleImpressao c2 = new ProdutoControleImpressao();
            c2.ImprimirDados(produto);

            Console.ReadKey();
        }
    }
}

```



```

C:\Users\COTI\Desktop\_Aula_25.01.18\Aula06\Projeto03\bin\Debug\Projeto03.exe

Dados de Pessoa:
Id da Pessoa.: 1
Nome.....: Jose da Silva

Dados do Produto:
Id do Produto...: 1
Nome.....: Notebook
Preço.....: 2500

```