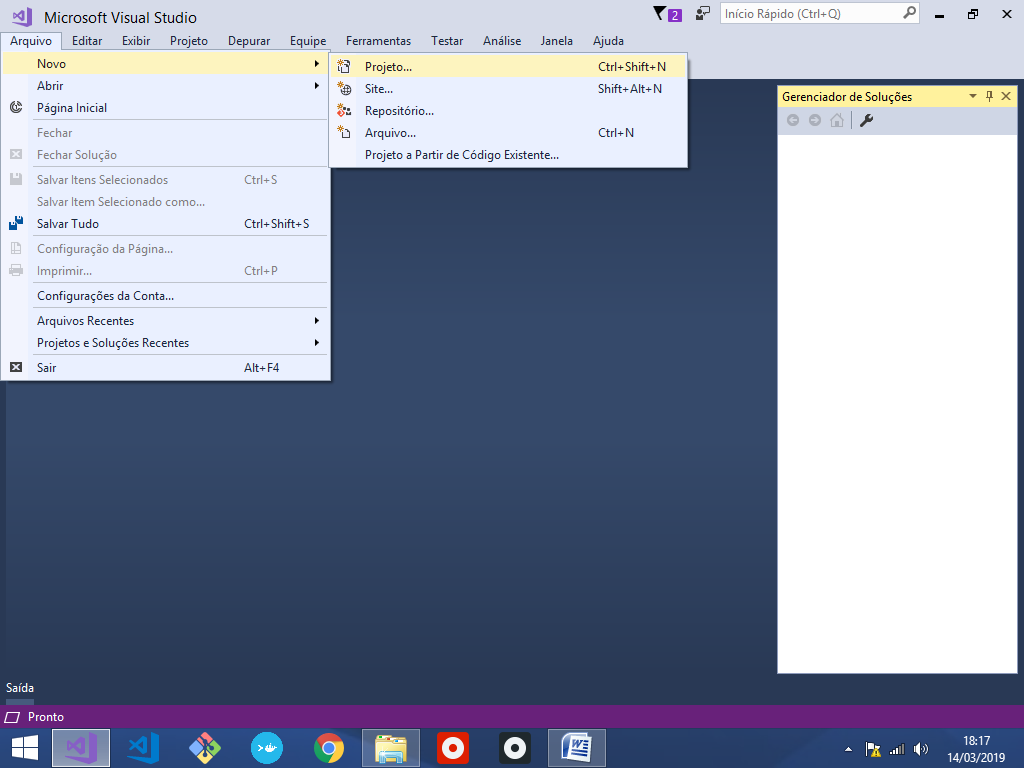
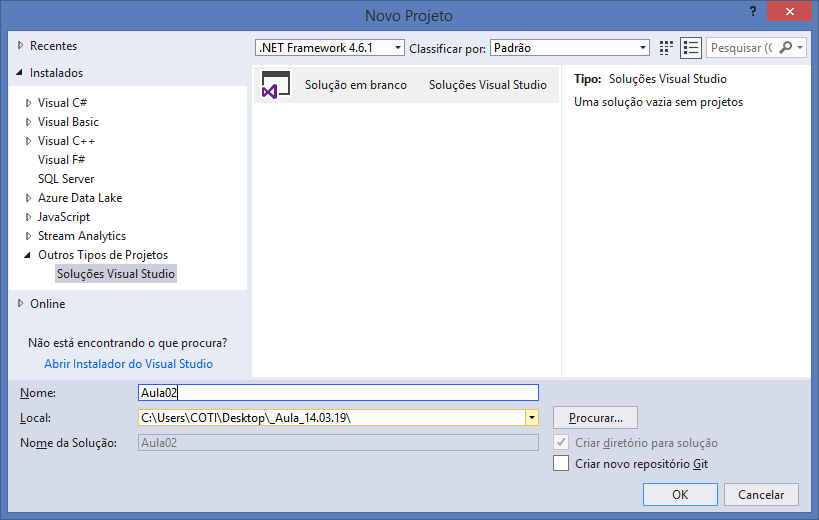
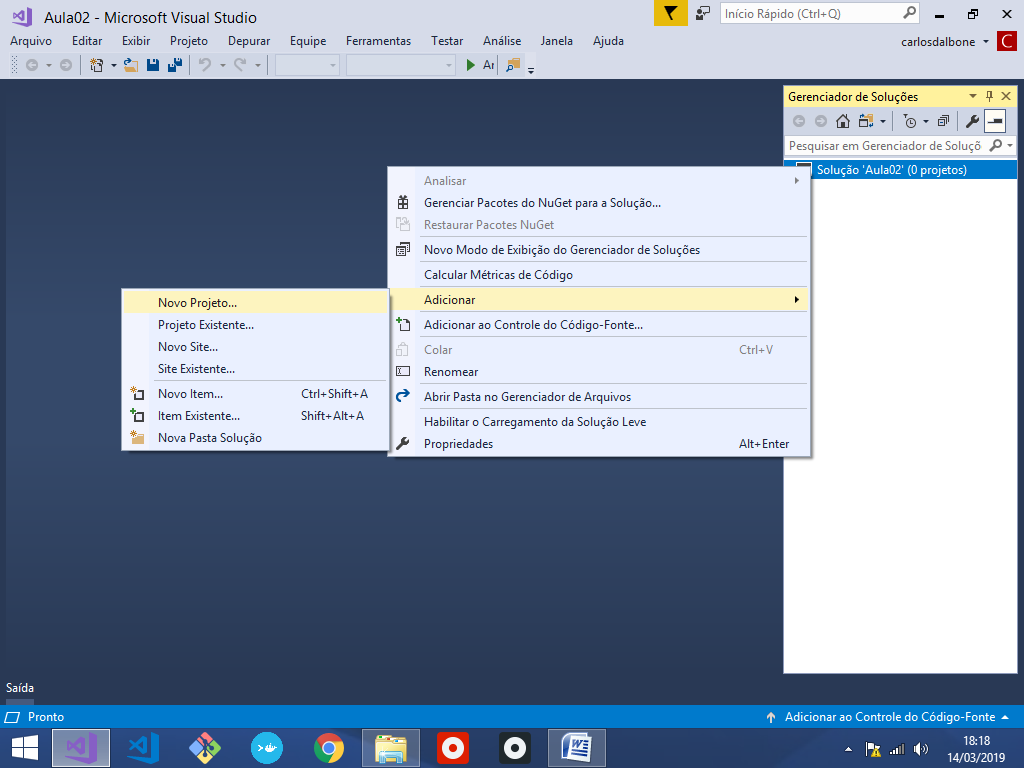
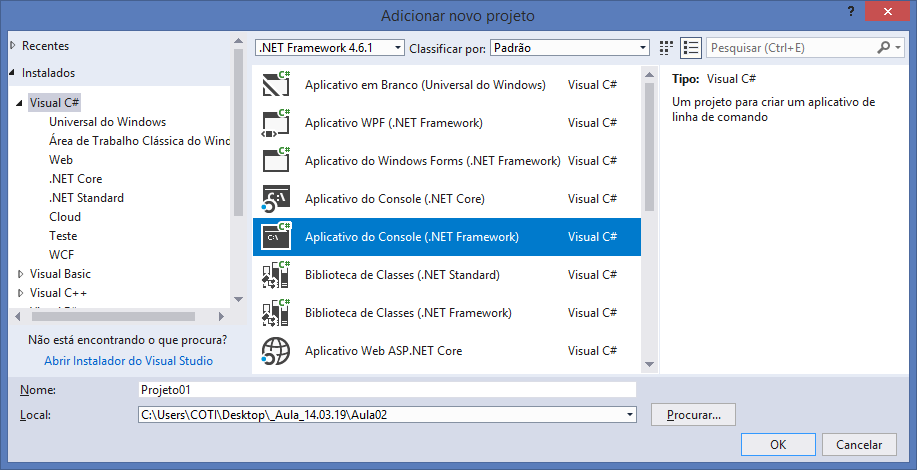
**Criando uma nova solution em branco:**





Criando um primeiro projeto **Console Application**



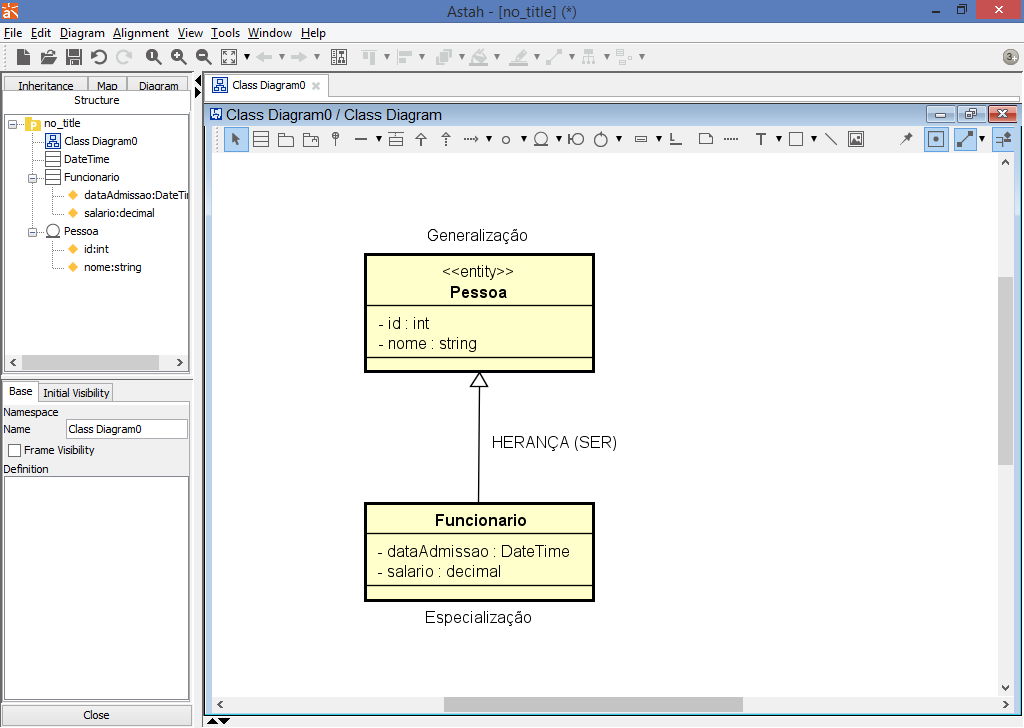


Relacionamentos entre classes

Qualquer classe em C# pode relacionar-se de 2 formas com outra classe: Por meio de **Herança (SER)** ou de **Associação (TEM)**.

Herança (SER)

Tipo de relacionamento que define uma hierarquia de superclasse e subclasses (generalização / especialização)





Encapsulamento implicito

É feito quando declaramos apenas a propriedade set/get sem a necessidade de declarar o atributo privado, ficando este implicito pelo compilador.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Projeto01.Entities

{

public class Pessoa

{

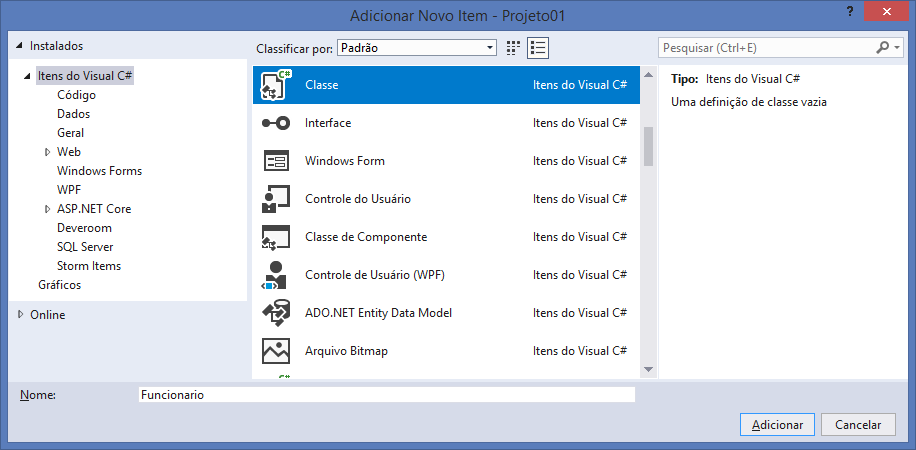
//prop + 2x[tab]

**public int Id { get; set; }**

**public string Nome { get; set; }**

}

}



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Projeto01.Entities

{

//Funcionário É-UMA Pessoa

**public class Funcionario : Pessoa**

{

**public DateTime DataAdmissao { get; set; }**

**public decimal Salario { get; set; }**

}

}

**Regras sobre herança:**

1) Não é permitido herança múltipla entre classes

Em C#, cada classe só pode ter 1 única superclasse.

**public class A**

**{**

**}**

**public class B**

**{**

**}**

**public class C ~~: A, B~~**

**{**

**}**

2) Classes definidas como "sealed" não podem ser herdadas (não podem ter subclasses)

O uso do "sealed" não permite que uma classe tenha subclasses

**public sealed class A**

**{**

**}**

**public class B ~~: A~~**

**{**

**}**

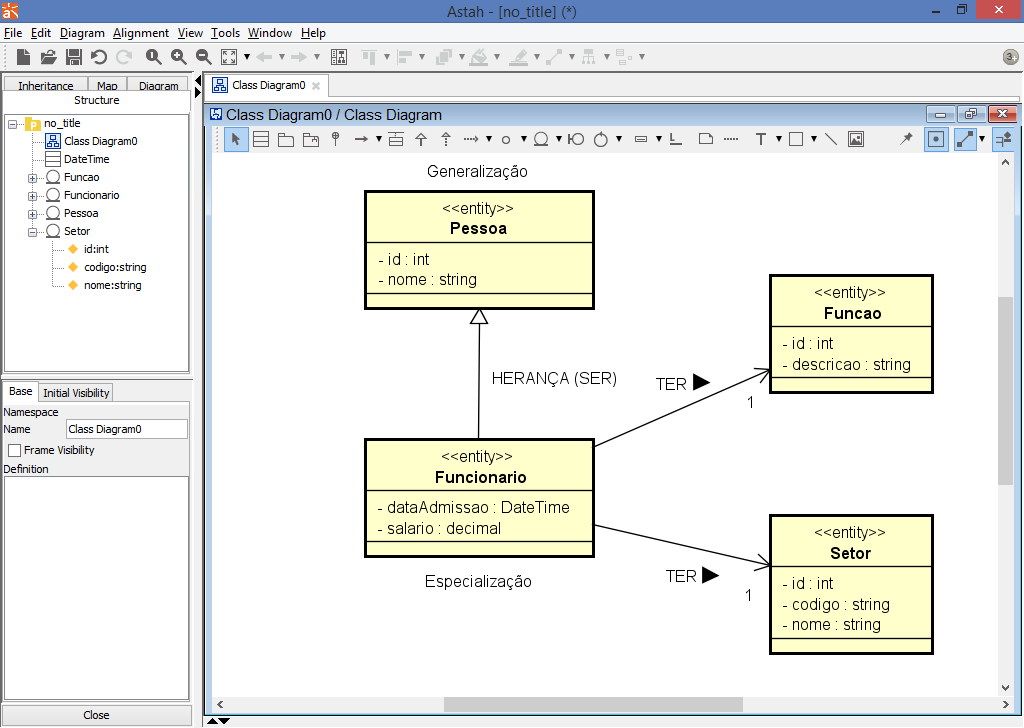
Associação (TER)

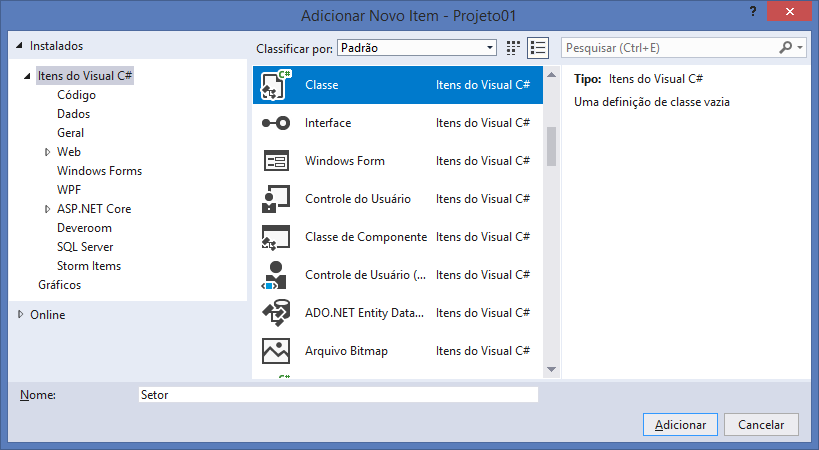
Tipo de relacionamento de utilização, por exemplo:

* Funcionario TEM Setor
* Funcionario TEM Funcao

Todo relacionamento de associação é acompanhado de uma multiplicidade, por exemplo:

* Funcionario TEM 1 Setor
* Funcionario TEM 1 Funcao





using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Projeto01.Entities

{

public class Setor

{

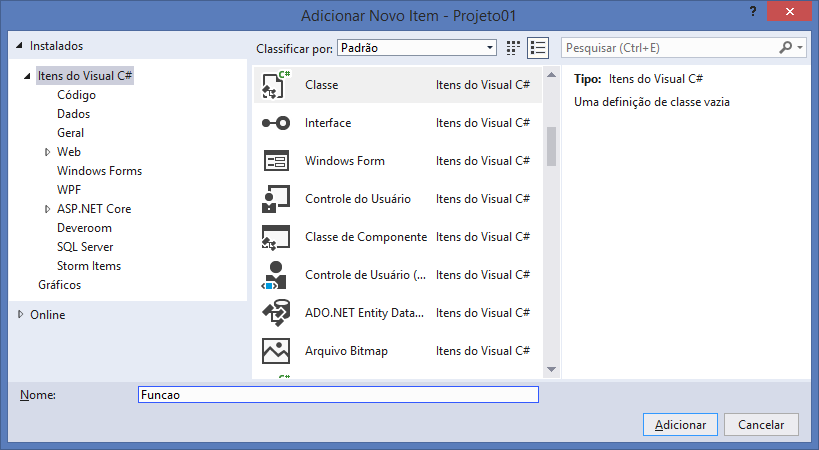
public int Id { get; set; }

public string Codigo { get; set; }

public string Nome { get; set; }

}

}



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Projeto01.Entities

{

public class Funcao

{

public int Id { get; set; }

public string Descricao { get; set; }

}

}

**Criando os relacionamentos de Associação:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Projeto01.Entities

{

//Funcionário É-UMA Pessoa

public class Funcionario : Pessoa

{

public DateTime DataAdmissao { get; set; }

public decimal Salario { get; set; }

**//Relacionamentos de Associação**

**public Funcao Funcao { get; set; } //TER-1**

**public Setor Setor { get; set; } //TER-1**

}

}

-----------------

**Exercício:**

* Crie um programa em C# que leia os seguintes   
  dados de um funcionário:

Id

Nome

Data de Admissão

Salário

Código do Setor

Nome do Setor

Descrição da Função

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto01.Entities; //importando

namespace Projeto01

{

class Program

{

static void Main(string[] args)

{

//criando e instanciando objetos

Funcionario funcionario = new Funcionario();

funcionario.Setor = new Setor();

funcionario.Funcao = new Funcao();

try

{

Console.WriteLine("\n - CADASTRO DE FUNCIONÁRIO - \n");

Console.Write("Informe o Id do Funcionário.....: ");

funcionario.Id = int.Parse(Console.ReadLine());

Console.Write("Informe o Nome..................: ");

funcionario.Nome = Console.ReadLine();

Console.Write("Informe a Data de Admissão......: ");

funcionario.DataAdmissao = DateTime.Parse(Console.ReadLine());

Console.Write("Informe o Salário...............: ");

funcionario.Salario = decimal.Parse(Console.ReadLine());

Console.Write("Informe a Descrição da Função...: ");

funcionario.Funcao.Descricao = Console.ReadLine();

Console.Write("Informe o Código do Setor.......: ");

funcionario.Setor.Codigo = Console.ReadLine();

Console.Write("Informe o Nome do Setor.........: ");

funcionario.Setor.Nome = Console.ReadLine();

}

catch(Exception e)

{

Console.WriteLine("Erro: " + e.Message);

}

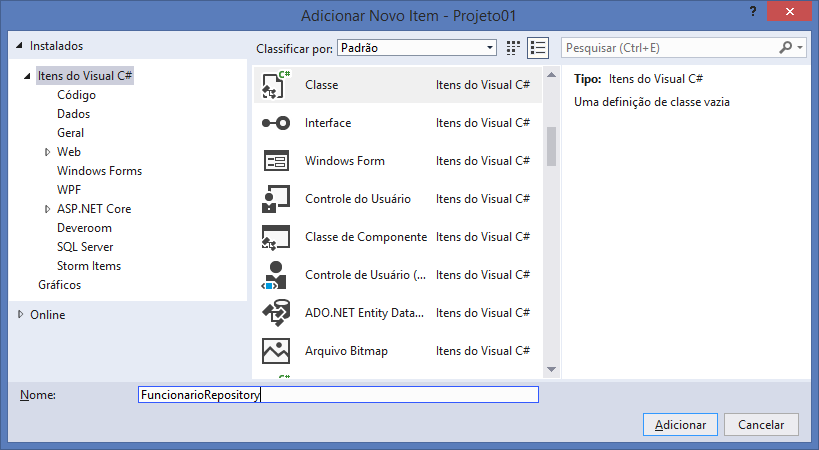
Console.ReadKey();

}

}

}

**Criando uma classe para exportar os dados   
de funcionário para um arquivo TXT**



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto01.Entities; //importando

using System.IO; //importando

namespace Projeto01.Repositories

{

public class FuncionarioRepository

{

//método para exportar os dados de um funcionário

//para arquivo de extensão TXT

public void ExportarParaTxt(Funcionario funcionario)

{

//criando uma variável para definir o nome do arquivo

string nomeArquivo = string.Format("funcionario\_{0}.txt",

DateTime.Now.ToString("ddMMyyyyHHmmss"));

//abrindo o arquivo para gravação..

using (StreamWriter writer = new StreamWriter

("c:\\temp\\" + nomeArquivo))

{

writer.WriteLine("Id.........: "

+ funcionario.Id);

writer.WriteLine("Nome.......: "

+ funcionario.Nome);

writer.WriteLine("Salário....: "

+ funcionario.Salario.ToString("c")); //currency

writer.WriteLine("Admissão...: "

+ funcionario.DataAdmissao.ToString("dd/MM/yyyy"));

writer.WriteLine("Função.....: "

+ funcionario.Funcao.Descricao);

writer.WriteLine("CodSetor...: "

+ funcionario.Setor.Codigo);

writer.WriteLine("Setor......: "

+ funcionario.Setor.Nome);

}

}

}

}

**Voltando ao método main:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto01.Entities; //importando

using Projeto01.Repositories; //importando

namespace Projeto01

{

class Program

{

static void Main(string[] args)

{

//criando e instanciando objetos

Funcionario funcionario = new Funcionario();

funcionario.Setor = new Setor();

funcionario.Funcao = new Funcao();

try

{

Console.WriteLine("\n - CADASTRO DE FUNCIONÁRIO - \n");

Console.Write("Informe o Id do Funcionário.....: ");

funcionario.Id = int.Parse(Console.ReadLine());

Console.Write("Informe o Nome..................: ");

funcionario.Nome = Console.ReadLine();

Console.Write("Informe a Data de Admissão......: ");

funcionario.DataAdmissao = DateTime.Parse(Console.ReadLine());

Console.Write("Informe o Salário...............: ");

funcionario.Salario = decimal.Parse(Console.ReadLine());

Console.Write("Informe a Descrição da Função...: ");

funcionario.Funcao.Descricao = Console.ReadLine();

Console.Write("Informe o Código do Setor.......: ");

funcionario.Setor.Codigo = Console.ReadLine();

Console.Write("Informe o Nome do Setor.........: ");

funcionario.Setor.Nome = Console.ReadLine();

**//gravar o arquivo..**

**FuncionarioRepository repository = new FuncionarioRepository();**

**repository.ExportarParaTxt(funcionario);**

**Console.WriteLine("\nDados gravados**

**em arquivo TXT com sucesso.");**

}

catch(Exception e)

{

Console.WriteLine("Erro: " + e.Message);

}

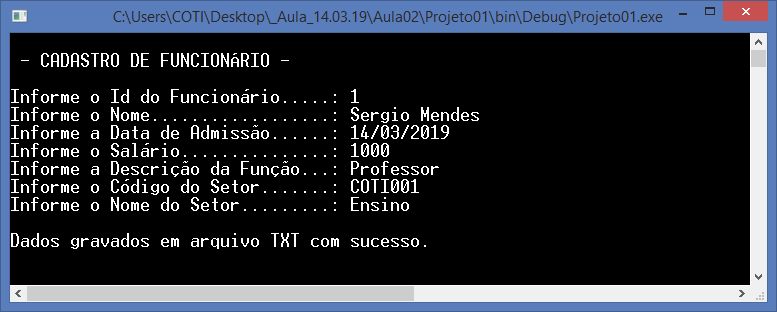
Console.ReadKey();

}

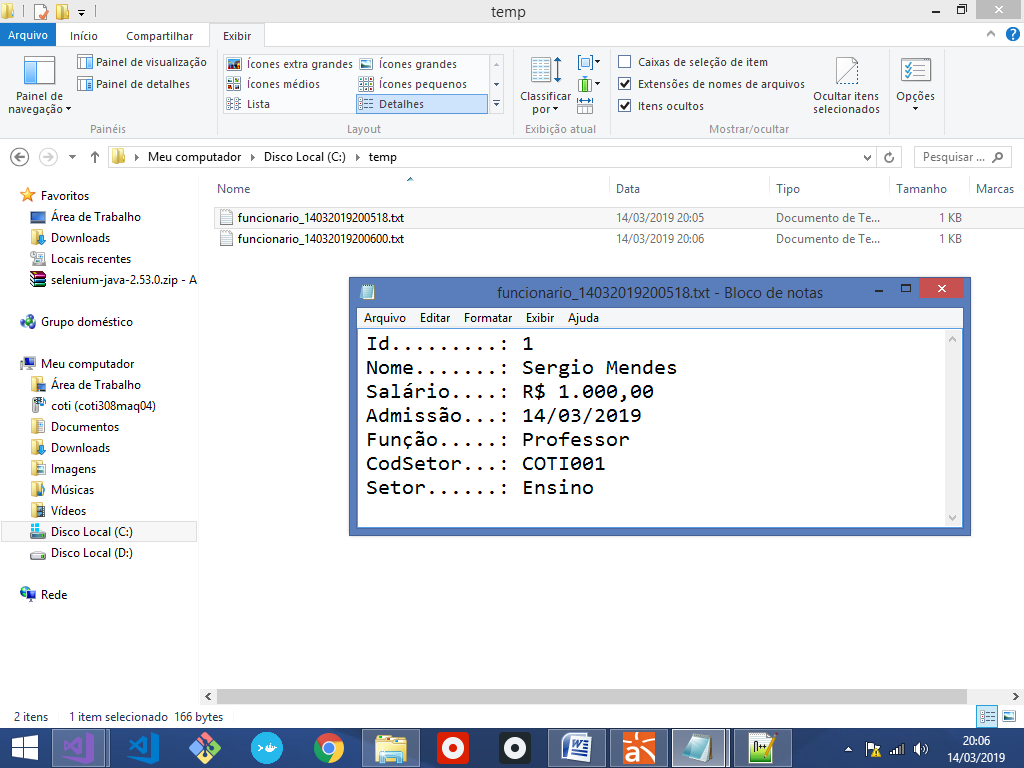
}

}

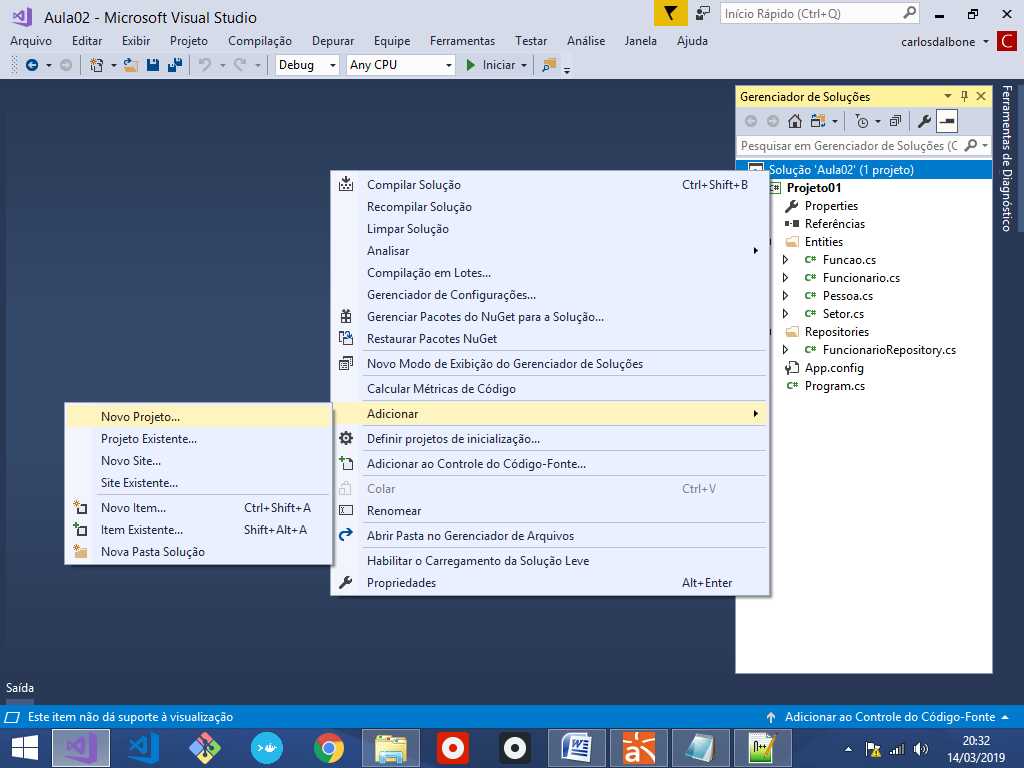
**Executando:**

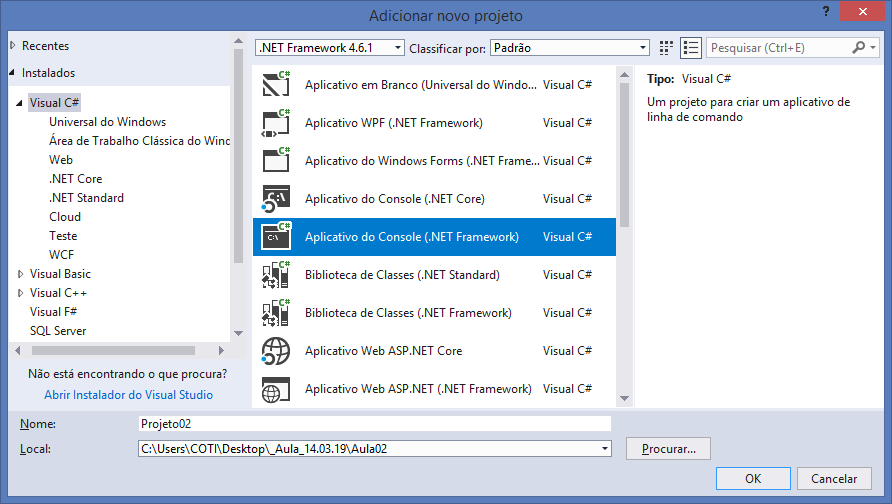


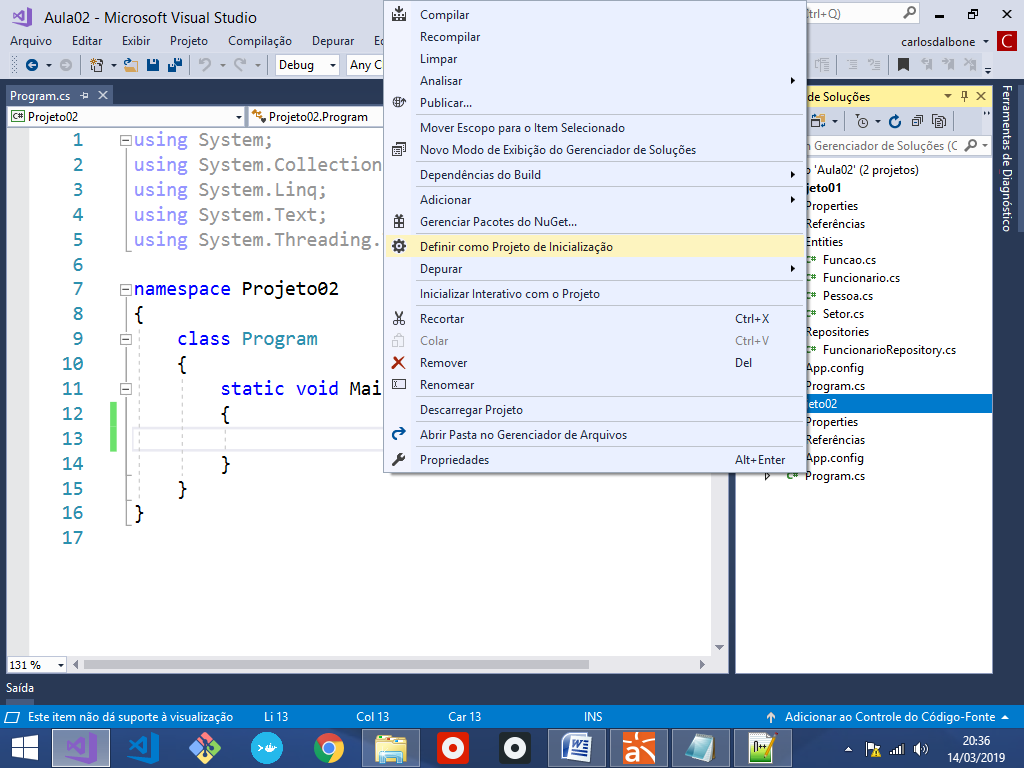
**Arquivos gerados:**



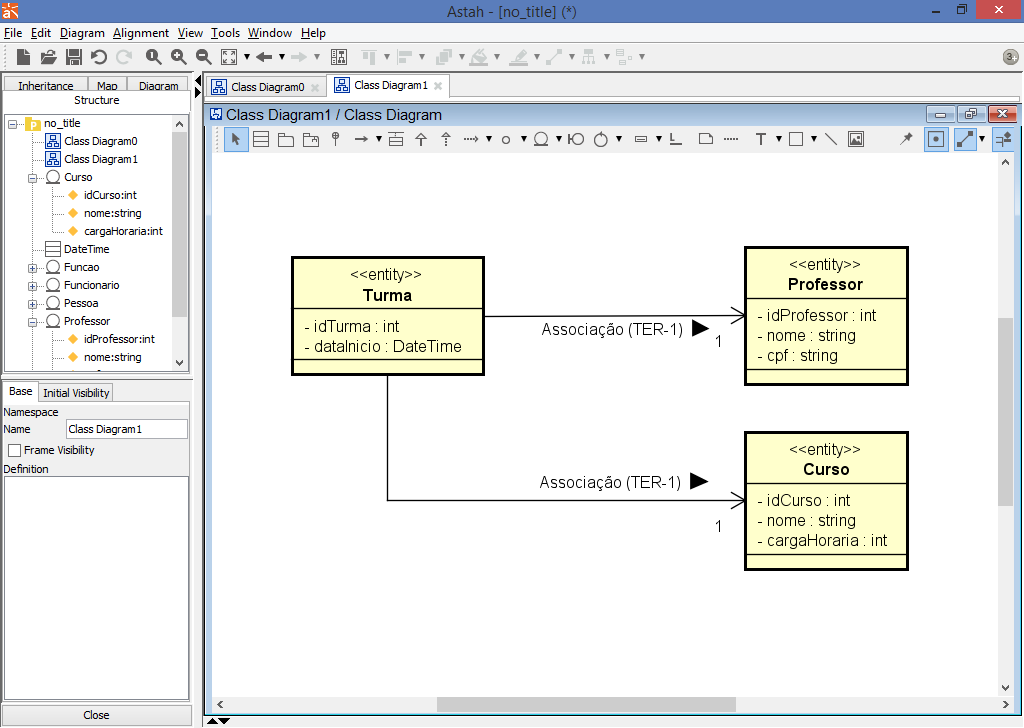
**Novo projeto:**







**Criando o modelo de entidades abaixo:**



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Projeto02.Entities

{

public class Curso

{

public int IdCurso { get; set; }

public string Nome { get; set; }

public int CargaHoraria { get; set; }

}

}

-------------

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Projeto02.Entities

{

public class Professor

{

public int IdProfessor { get; set; }

public string Nome { get; set; }

public string Cpf { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Projeto02.Entities

{

public class Turma

{

public int IdTurma { get; set; }

public DateTime DataInicio { get; set; }

public Professor Professor { get; set; }

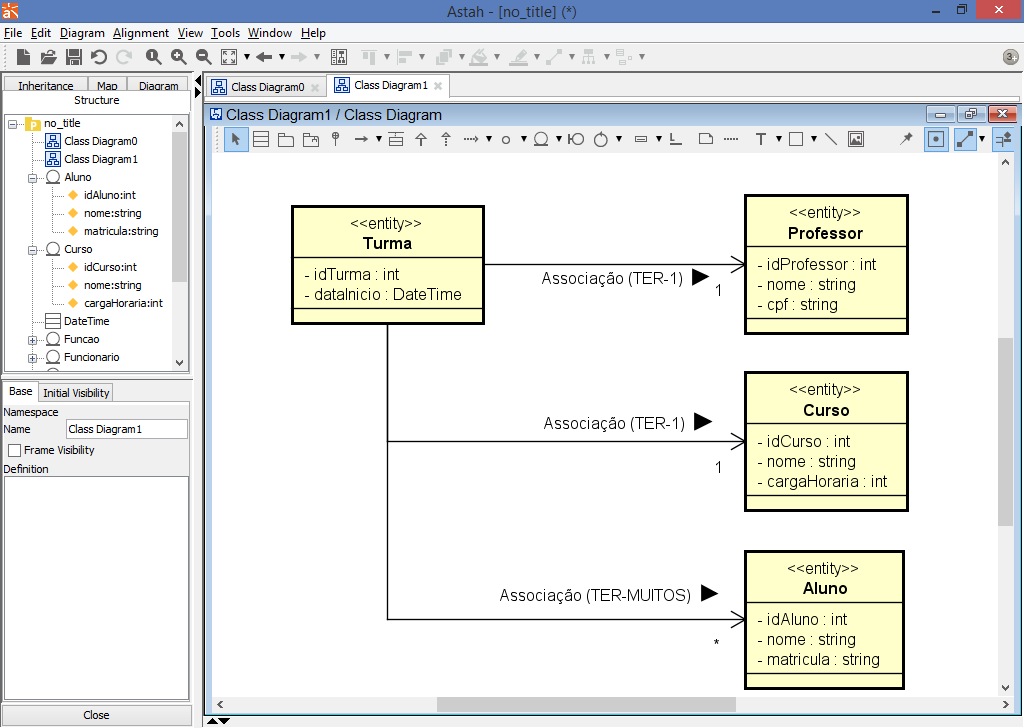
public Curso Curso { get; set; }

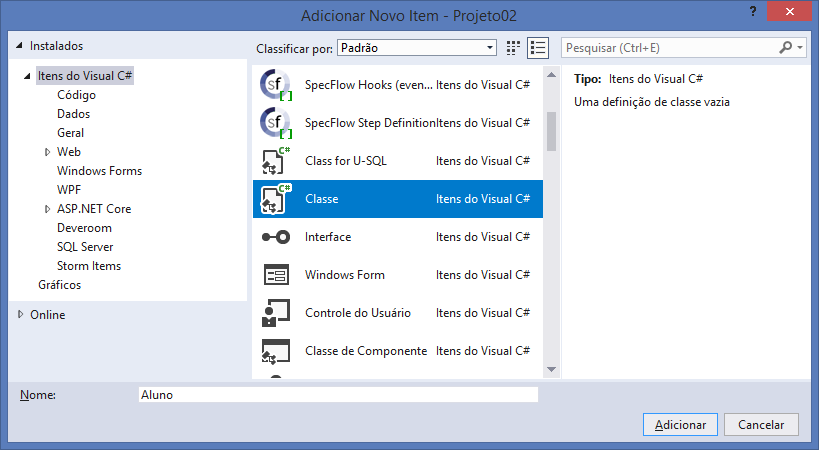
}

}

---------------------

Incrementando o modelo com um relacionamento de Associação TER-MUITOS





using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Projeto02.Entities

{

public class Aluno

{

public int IdAluno { get; set; }

public string Nome { get; set; }

public string Matricula { get; set; }

}

}

**Relacionando Turma com MUITOS Alunos**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Projeto02.Entities

{

public class Turma

{

public int IdTurma { get; set; }

public DateTime DataInicio { get; set; }

//Relacionamento TER 1 Professor

public Professor Professor { get; set; }

//Relacionamento TER 1 Curso

public Curso Curso { get; set; }

//Relacionamento TER MUITAS Turmas

**public List<Aluno> Alunos { get; set; }**

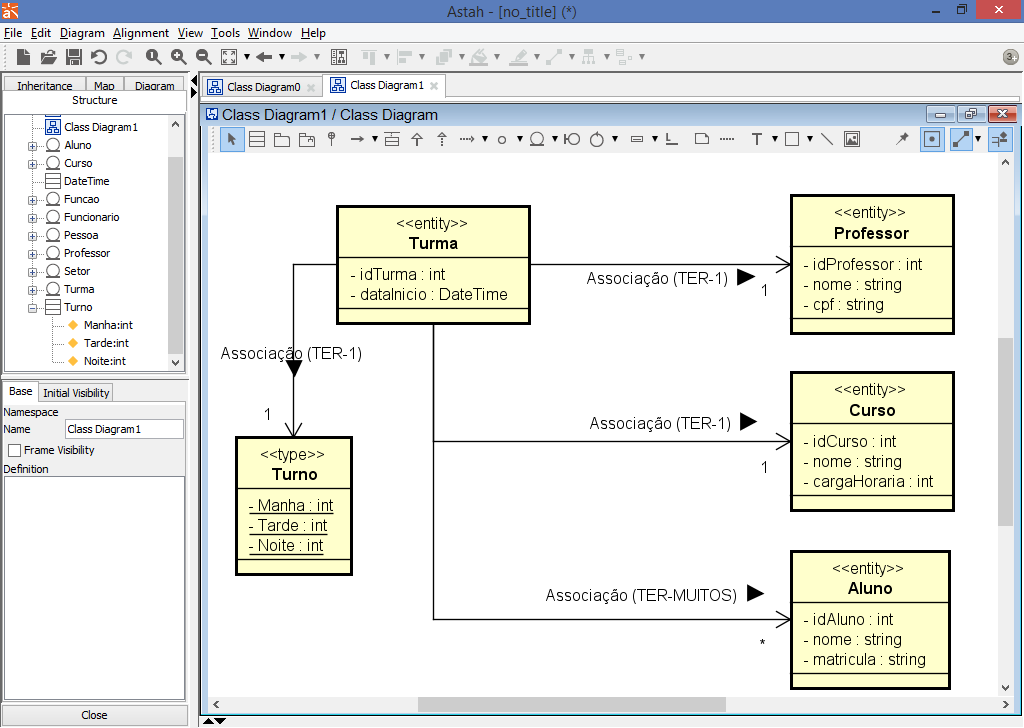
}

}

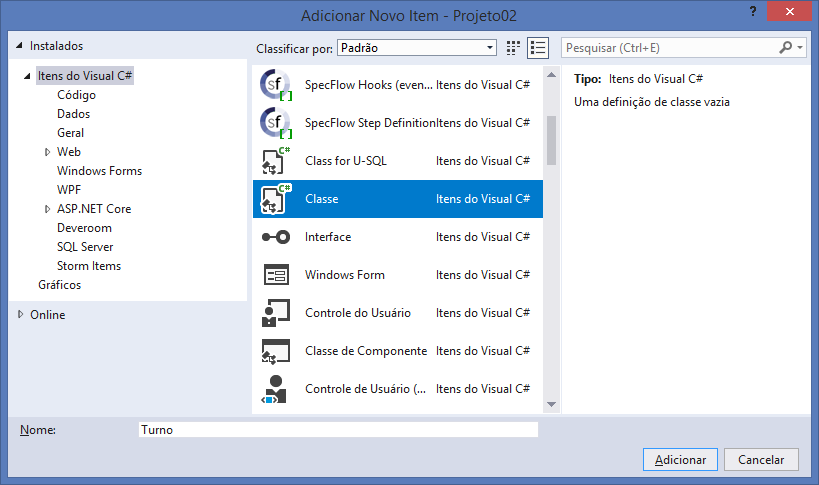
Enums (Types)

Classes simples utilizadas para representar tipos multivalorados.

Por exemplo:



**Criando a classe ENUM**



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Projeto02.Entities

{

public enum Turno

{

Manha = 1,

Tarde = 2,

Noite = 3

}

}

Relacionando Turma com Turno

Turma TEM 1 Turno

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Projeto02.Entities

{

public class Turma

{

public int IdTurma { get; set; }

public DateTime DataInicio { get; set; }

//Relacionamento TER 1 Professor

public Professor Professor { get; set; }

//Relacionamento TER 1 Curso

public Curso Curso { get; set; }

//Relacionamento TER MUITAS Turmas

public List<Aluno> Alunos { get; set; }

**//Relacionamento TER 1 Turno**

**public Turno Turno { get; set; }**

}

}

**Voltando para a classe Program (Método Main)**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto02.Entities; //importando..

namespace Projeto02

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("\n - CONTROLE DE TURMAS - \n");

try

{

Turma turma = new Turma();

turma.Professor = new Professor();

turma.Curso = new Curso();

turma.Alunos = new List<Aluno>();

Console.Write("Informe o Id da Turma..................: ");

turma.IdTurma = int.Parse(Console.ReadLine());

Console.Write("Data de início da Turma................: ");

turma.DataInicio = DateTime.Parse(Console.ReadLine());

Console.Write("Nome do Professor......................: ");

turma.Professor.Nome = Console.ReadLine();

Console.Write("Cpf do Professor.......................: ");

turma.Professor.Cpf = Console.ReadLine();

Console.Write("Nome do Curso..........................: ");

turma.Curso.Nome = Console.ReadLine();

Console.Write("Carga horária do Curso.................: ");

turma.Curso.CargaHoraria = int.Parse(Console.ReadLine());

}

catch(Exception e)

{

Console.WriteLine("Ocorreu um erro: " + e.Message);

}

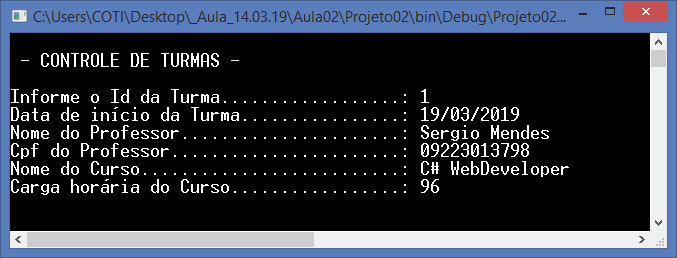
Console.ReadKey();

}

}

}

**Resultado parcial:**



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Projeto02.Entities; //importando..

namespace Projeto02

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("\n - CONTROLE DE TURMAS - \n");

try

{

Turma turma = new Turma();

turma.Professor = new Professor();

turma.Curso = new Curso();

turma.Alunos = new List<Aluno>();

Console.Write("Informe o Id da Turma..................: ");

turma.IdTurma = int.Parse(Console.ReadLine());

Console.Write("Data de início da Turma................: ");

turma.DataInicio = DateTime.Parse(Console.ReadLine());

Console.Write("Nome do Professor......................: ");

turma.Professor.Nome = Console.ReadLine();

Console.Write("Cpf do Professor.......................: ");

turma.Professor.Cpf = Console.ReadLine();

Console.Write("Nome do Curso..........................: ");

turma.Curso.Nome = Console.ReadLine();

Console.Write("Carga horária do Curso.................: ");

turma.Curso.CargaHoraria = int.Parse(Console.ReadLine());

Console.Write("Informe (1)Manha, (2)Tarde ou (3)Noite.: ");

turma.Turno = (Turno) int.Parse(Console.ReadLine());

Console.Write("Informe a quantidade de alunos.........: ");

int qtdAlunos = int.Parse(Console.ReadLine());

//for + 2x[tab]

for (int i = 0; i < qtdAlunos; i++)

{

Aluno aluno = new Aluno();

Console.WriteLine("\n\tMatricula............: ");

aluno.Matricula = Console.ReadLine();

Console.WriteLine("\tNome..............: ");

aluno.Nome = Console.ReadLine();

//adicionando o aluno na turma..

turma.Alunos.Add(aluno);

}

//imprimindo...

Console.WriteLine("\n - DADOS DA TURMA - \n");

Console.WriteLine("Id da Turma........: "

+ turma.IdTurma);

Console.WriteLine("Data de Início.....: "

+ turma.DataInicio);

Console.WriteLine("Professor..........: "

+ turma.Professor.Nome);

Console.WriteLine("CPF................: "

+ turma.Professor.Cpf);

Console.WriteLine("Curso..............: "

+ turma.Curso.Nome);

Console.WriteLine("Carga Horária......: "

+ turma.Curso.CargaHoraria);

Console.WriteLine("Turno..............: "

+ turma.Turno.ToString());

Console.WriteLine("Qtd de Alunos......: "

+ turma.Alunos.Count);

**//ler cada Aluno contido na lista**

**foreach(Aluno aluno in turma.Alunos)**

**{**

**Console.WriteLine("\n\tNome do Aluno..: " + aluno.Nome);**

**Console.WriteLine("\tMatricula......: " + aluno.Matricula);**

**}**

}

catch(Exception e)

{

Console.WriteLine("Ocorreu um erro: " + e.Message);

}

Console.ReadKey();

}

}

}

**Executando:**

