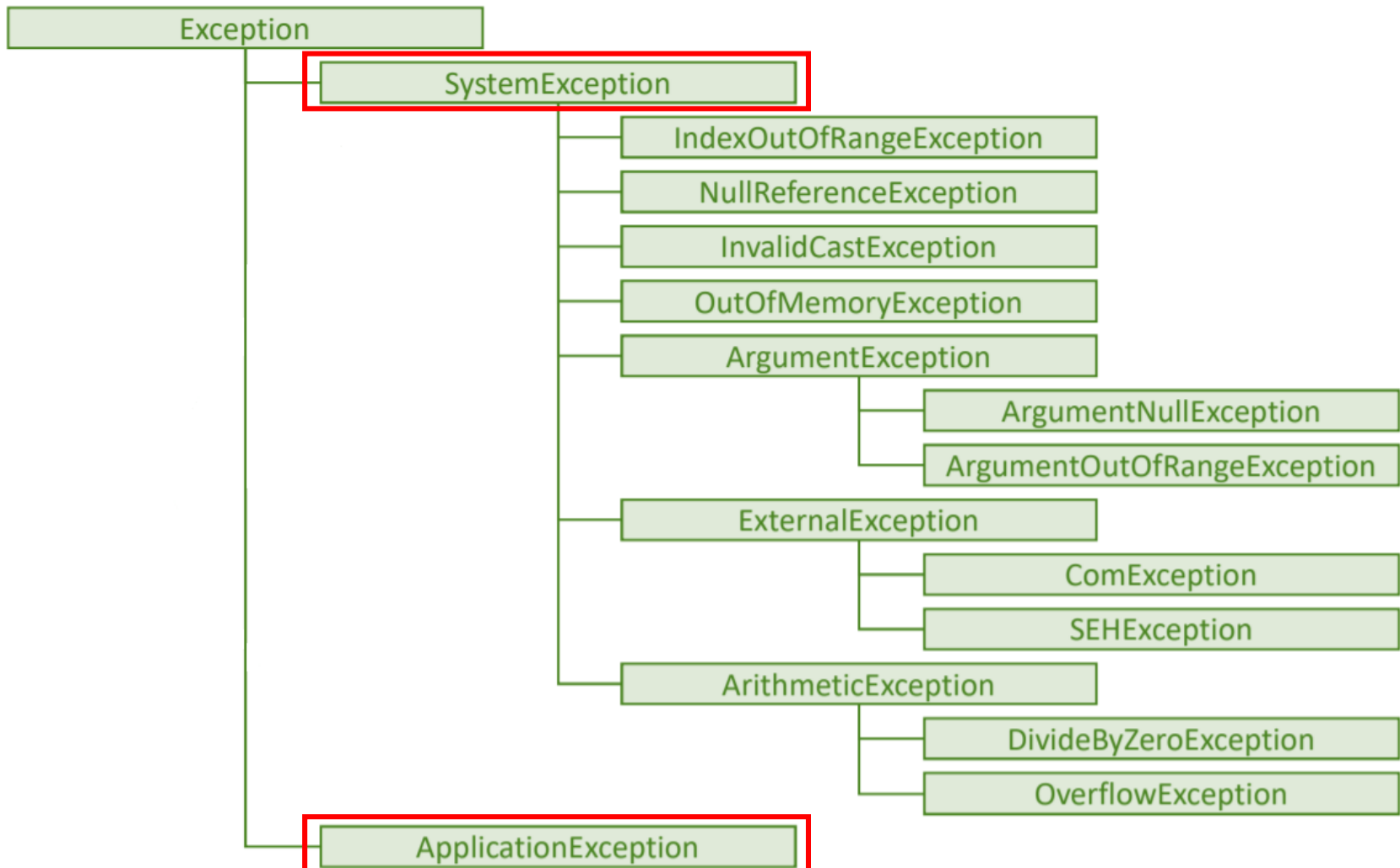


Tratamento de Exceções

Conceitos

- Qualquer condição de erro ou comportamento inesperado encontrado por um programa em execução
 - Quando lançada, uma exceção é propagada na pilha de métodos em execução até que seja capturada e tratada ou o programa seja encerrado
 - No **.Net Framework**, uma exceção é um objeto herdado da classe **System.Exception**
-

Classes



Vantagens

- Com o uso de tratamento de exceções, os erros são **tratados** de forma **consistente** e **flexível**, utilizando boas práticas de programação
- **Vantagens:**
 - Delega a lógica do erro para a classe ou método responsável por conhecer as regras de negócio que pode ocasionar o erro
 - Trata exceções de tipos diferentes de forma organizada (inclusive hierárquica)
 - A exceção pode carregar quaisquer tipos de dados

Estrutura try-catch

▪ Bloco try

- Código que representa a execução normal do trecho que pode acarretar em uma exceção

▪ Bloco catch

- Contém o código a ser executado caso uma exceção ocorra
- Deve ser utilizado o tipo de exceção a ser tratado (upcasting é permitido)

Upcasting é uma operação que cria uma referência da classe **base** a partir de uma referência da classe derivada. (subclasse → superclasse) (ex.: Gerente → Funcionário)

Sintaxe exemplo estrutura try-catch

```
try {  
    //código que pode gerar um exceção  
}  
catch (ExceptionType e) {  
    //Código a ser executado caso uma exceção ocorra  
}  
catch (ExceptionType e) {  
    //Código a ser executado caso uma exceção ocorra  
}  
catch (ExceptionType e) {  
    //Código a ser executado caso uma exceção ocorra  
}
```

Programa exemplo

```
try
{
    int num1 = int.Parse(Console.ReadLine());
    int num2 = int.Parse(Console.ReadLine());
    int result = num1 / num2;
    Console.WriteLine(result);
    Console.ReadKey();
}
catch (DivideByZeroException)
{
    Console.WriteLine("Divisão por zero não permitido!");
}
catch (FormatException e)
{
    Console.WriteLine($"Erro de formatação: {e.Message}");
}
Console.ReadKey();
```

Bloco finally

- Bloco que contém código que será executado **independentemente** de ter ou não **ocorrido** uma **exceção**
 - Exemplo clássico: **fechar** um **arquivo** ou uma **conexão** com o **banco de dados** ao final do processamento
-

Exemplo sintaxe bloco finally

```
try {  
}  
catch (ExceptionType e) {  
}  
finally {  
    //será executado independentemente de ter ou  
    não ocorrido uma exceção  
}
```

Programa exemplo

```
private static void ExcecoesFinally()
{
    FileStream fs = null;
    try
    {
        fs = new FileStream(@"C:\VS\teste.txt", FileMode.Open);
        StreamReader sr = new StreamReader(fs);
        string linha = sr.ReadLine();
        Console.WriteLine(linha);
    }
    catch (FileNotFoundException e)
    {
        Console.WriteLine("Arquivo não encontrado: ", e.Message);
    }
    finally
    {
        if (fs != null)
        {
            fs.Close();
        }
    }
}
```

Exceções Personalizadas

- Fazer um programa para ler os dados de uma matrícula em um curso de informática no Senac (número do laboratório, data de início e data de fim) e mostrar os dados da matrícula, inclusive sua duração em dias. Em seguida, ler novas datas de início e fim, atualizar a matrícula e mostrar novamente a matrícula com os dados atualizados.
- O programa não deve aceitar dados inválidos para a reserva, conforme as seguintes regras:
- Alterações de matrícula só podem ocorrer para datas futuras
- A data de fim deve ser maior que a data de início

Matricula

+ numeroLab: int

+ dataInicio: DateTime

+ dataFim: DateTime

+ DuracaoCurso: int

+ AtaulizaDataCurso: dataInicio: DateTime, dataFim: DateTime): void

Exceções Personalizadas

Saída em tela:

Número do laboratório: 1234

Data de início (dd/MM/yyyy): 23/09/2019

Data do encerramento (dd/MM/yyyy): 26/09/2019

Matrícula: Laboratório 1234, início: 23/09/2019, encerramento: 26/09/2019, 3 dias

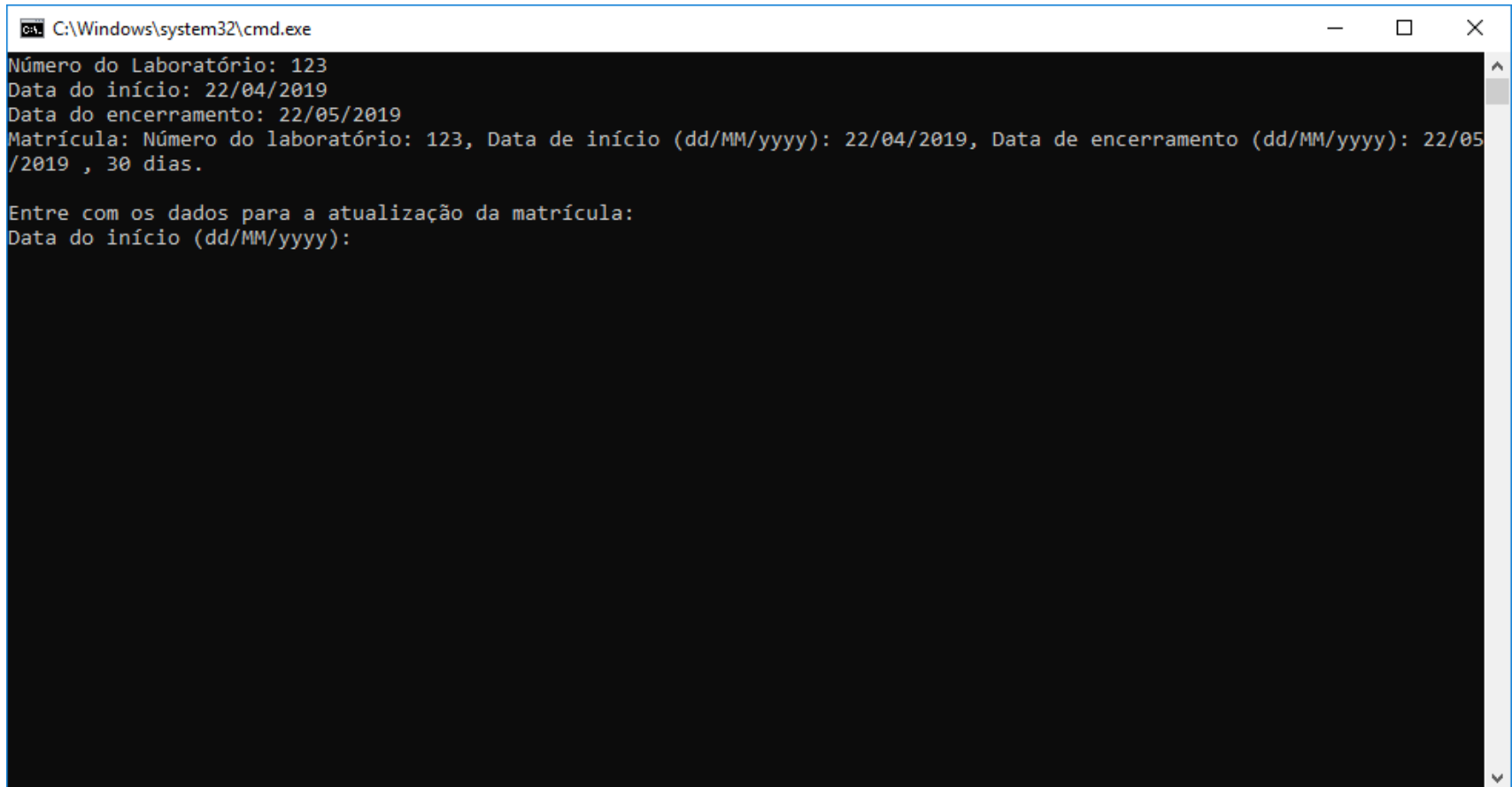
Digite os dados para atualização da Matrícula:

Data de início (dd/MM/yyyy): 24/09/2019

Data de encerramento (dd/MM/yyyy): 29/09/2019

Matrícula: Laboratório 1234, início: 24/09/2019, encerramento: 29/09/2019, 5 dias

Exemplo

A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The window has a black background with white text. The text displayed is as follows:

```
C:\Windows\system32\cmd.exe
Número do Laboratório: 123
Data do início: 22/04/2019
Data do encerramento: 22/05/2019
Matrícula: Número do laboratório: 123, Data de início (dd/MM/yyyy): 22/04/2019, Data de encerramento (dd/MM/yyyy): 22/05/2019 , 30 dias.

Entre com os dados para a atualização da matrícula:
Data do início (dd/MM/yyyy):
```

The window includes standard Windows window controls (minimize, maximize, close) in the top right corner and a vertical scrollbar on the right side.