Linguagem de Programação II

Análise e Desenvolvimento de Sistemas

2º semestre de 2019

Prof. Me. Renato Carioca Duarte

Exceção

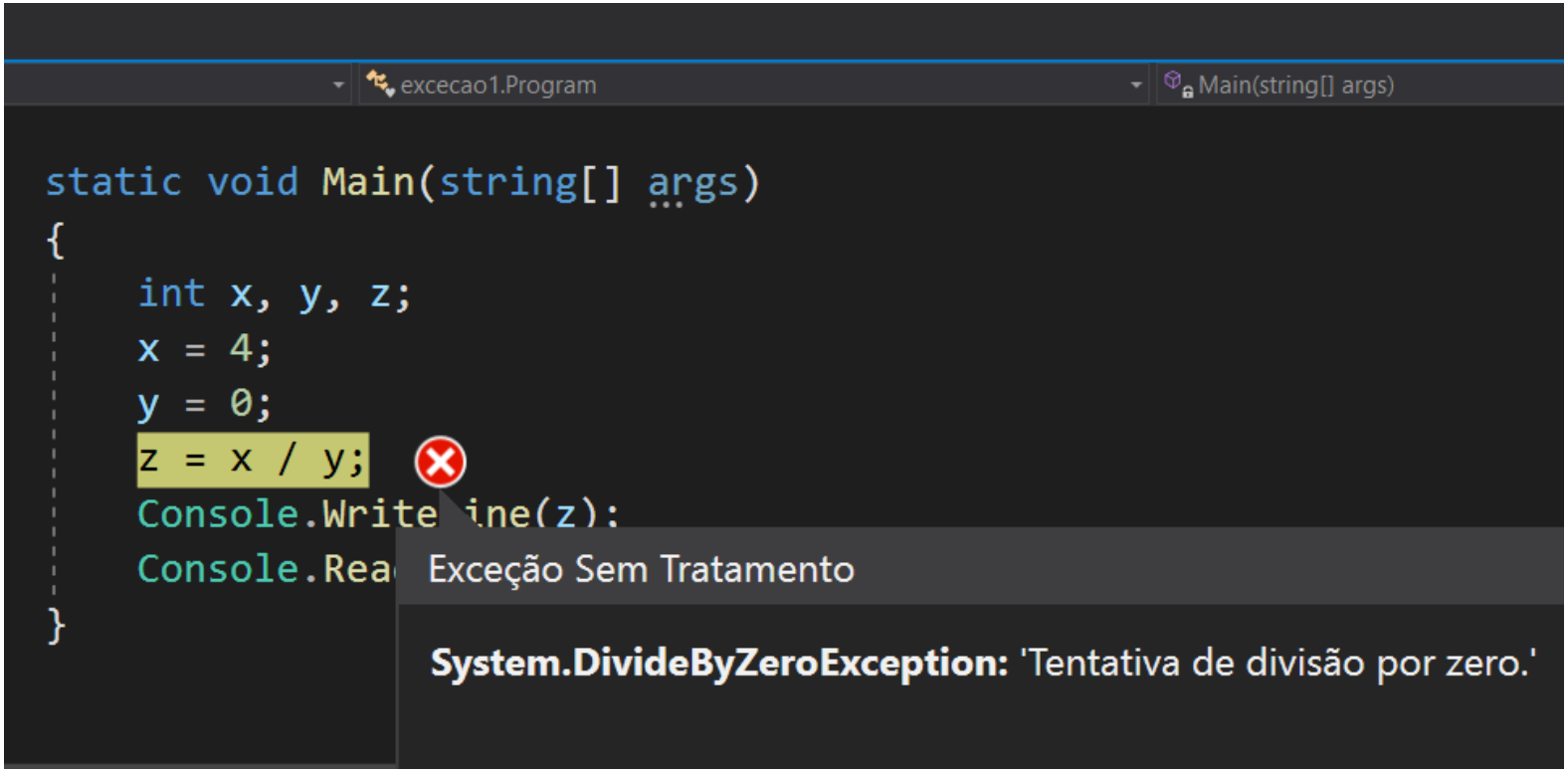
- Uma exceção é um erro em tempo de execução em um programa, que viola uma condição que não foi especificada para acontecer durante a operação normal.
- Um exemplo na prática é quando um programa tenta fazer a divisão por zero ou tenta escrever em um arquivo somente leitura.
- Quando isto ocorre, o sistema pega o erro e lança uma exceção.
- Se o programa não provê código para tratar uma exceção, o sistema irá parar o programa.
- Por exemplo, o seguinte código lança uma exceção quando o programa tenta dividir por zero.

Exceção

```
static void Main(string[] args)
{
    int x, y, z;
    x = 4;
    y = 0;
    z = x / y;
    Console.WriteLine(z);
    Console.ReadKey();
}
```

Exceção

- A exceção lançada quando este código é executado é a seguinte



TRY/CATCH

- A instrução try-catch consiste em um bloco try seguido por uma ou mais cláusulas catch, que especificam os manipuladores para diferentes exceções.Quando uma exceção é lançada, o programa procura a instrução catch que trata essa exceção.
- O comando try consiste de três seções.

```
Try

Catch

Finally
```

- O bloco de código do try contém o código que será validado e caso tenha uma exceção não será executado.
- O cláusula catch pode aparecer uma ou mais vezes abaixo na seção do try. Estes blocos são códigos que lançam a exceção dependendo de qual exceção é gerada pelo compilador.
- O finally contém o código a ser executado sobre qualquer circunstância. Tendo sido lançada ou não uma exceção o finally é executado.

TRY/CATCH

```
static void Main(string[] args)
{
    int x, y, z;
    x = 4;
    y = 0;
    try
    {
        z = x / y;
        Console.WriteLine(z);
    } catch (Exception e)
    {
        Console.WriteLine(e.Message);
    }
    finally
    {
        Console.ReadKey();
    }

}
```

TRY/CATCH

- O try e o catch são obrigatórios, porém, não é necessário ter mais de um catch para cada tratamento de exceção.
- O exemplo de código de divisão por zero causa uma exceção.
- O código pode ser modificado para que a exceção possa ser tratada utilizando o comando try e fornecendo uma simples cláusula catch.
- Quando a exceção é lançada, ela utiliza o código que está no bloco do catch para avisar o usuário o que está acontecendo de modo amigável.

Exemplo

```
static void Main(string[] args)
{
    int iMax = 0;
    Console.WriteLine("Digite um numero inteiro para ser o maximo a ser sorteado");
    try {
        iMax = int.Parse(Console.ReadLine());
        Random r = new Random();
        int iRand = r.Next(1, iMax);
        Console.WriteLine("O valor sorteado entre 1 e {1} é {0}", iRand, iMax);
    }
    catch (ArgumentException e) {
        Console.WriteLine("o numero não é valor valido");
    }
    catch (Exception e) {
        Console.WriteLine(e);
    }

    finally {
        Console.ReadKey();
    }
}
```

Lançamento de Exceção

- Até agora foram capturadas apenas exceções lançadas pelo próprio ambiente
- C# usa a palavra reservada throw para lançar uma exceção de execução, mas é possível lançar suas próprias exceções.
- Assim como no comando return o fluxo de execução é interrompido no comando throw, ou seja, nenhum comando subsequente será executado.
- O fluxo então segue como se fosse uma exceção lançada pelo ambiente.
- O bloco try mais próximo será inspecionado, se não o próximo bloco try até que algum trate a exceção ou chegue no tratador padrão.

Exemplo

- Considere a classe Conta com método sacar

```
class Conta
{
    double saldo;
    bool sacar(double valor)  {
        if (this.saldo < valor)  {
            return false;
        }
        else  {
            this.saldo -= valor;
            return true;
        }
    }
}
```

Exemplo

- Reescrevendo o método sacar de Conta para lançar uma exceção

Repare que não é mais **boolean**

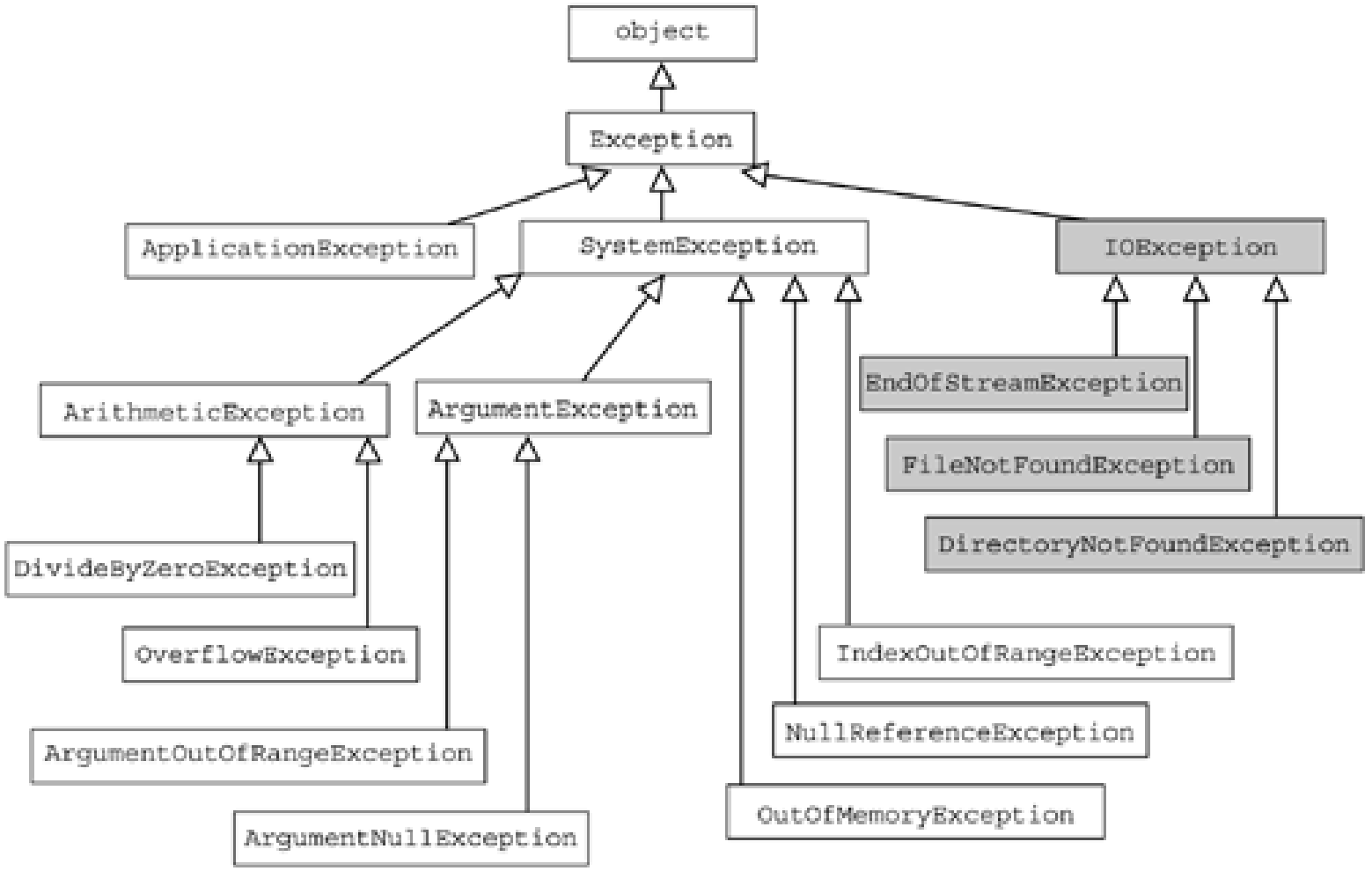
```
class Conta
{
    double saldo;

    void sacar(double valor)
    {
        if (this.saldo < valor)
            throw new Exception("Saldo Insuficiente");
        this.saldo -= valor;
    }
```

Exemplo

- Executando o método Saca

```
static void Main(string[] args)
{
    Conta cc = new Conta();
    cc.depositar(100);
    try
    {
        cc.sacar(100);
    }
    catch (Exception e)
    {
        Console.WriteLine(e.Message);
    }
```

Classes de exceção



Classes de exceção

ArgumentException	Um argumento não nulo que é passado para um método é inválido.
ArgumentNullException	Um argumento que é passado para um método é null.
ArgumentOutOfRangeException	Um argumento está fora do intervalo de valores válidos.
DirectoryNotFoundException	Parte de um caminho de diretório não é válida.
DivideByZeroException	O denominador em uma operação Decimal de divisão ou inteiro é zero.
DriveNotFoundException	Uma unidade não está disponível ou não existe.
FileNotFoundException	Um arquivo não existe.
FormatException	Um valor não está em um formato apropriado para ser convertido de uma cadeia de caracteres por um método de Parseconversão, como.
IndexOutOfRangeException	Um índice está fora dos limites de uma matriz ou coleção.
InvalidOperationException	Uma chamada de método é inválida no estado atual de um objeto.
KeyNotFoundException	A chave especificada para acessar um membro em uma coleção não pode ser encontrada.
NotImplementedException	Um método ou uma operação não está implementada.
NotSupportedException	Não há suporte para um método ou uma operação.
ObjectDisposedException	Uma operação é executada em um objeto que foi Descartado.
OverflowException	Uma operação de conversão ou aritmética resulta em um estouro.
PathTooLongException	Um caminho ou nome de arquivo excede o comprimento máximo definido pelo sistema.
PlatformNotSupportedException	A operação não tem suporte na plataforma atual.
RankException	Uma matriz com o número incorreto de dimensões é passada para um método.
TimeoutException	O intervalo de tempo alocado para uma operação expirou.
UriFormatException	É usado um Uniform Resource Identifier (URI) inválido.

Exercícios

1. Usando como base o código abaixo, reescreva-o para acrescentar o tratamento de erro caso o arquivo não exista no local definido.

```
static void Main(string[] args)
{
    string [] linhas = System.IO.File.ReadAllLines(
        @"C:\Users\renat\source\repos\leituraArquivo\arq2.txt");

    foreach (string x in linhas)
    {
        Console.WriteLine(x);
    }
    Console.ReadKey();
}
```

Exercícios

2. Faça um programa C# que recebe um numero e mostra uma mensagem na tela “Erro: deve digitar um número” caso digite um caracter não numérico.

3. Fazer um Form que recebe 2 números double (dividendo e divisor) e mostre uma MessageBox com o texto “o numero divisor não pode ser zero”, caso o seja digitado o numero zero no textbox do número divisor.