Erros, Validações e Correcções

? Exercício 1: Divisão por zero

```
using System;
namespace E1
{class DivisaoPorZero
 {static void Main(string[] args)
  {Console.Write("Dividendo? ");
   int X=Convert.ToInt16(Console.ReadLine());
   Console.Write("Divisor? ");
   int Y=Convert.ToInt16(Console.ReadLine());
   try
   {Console.WriteLine(X / Y);}
   catch (System.DivideByZeroException)
   {Console.WriteLine("O divisor tem de ser diferente de zero");}
}}}
```

? Exercício 2: Detecção de dois erros

```
using System;
namespace E2
{class DeteccaoDeDoisErros
 {static void Main(string[] args)
 {try
  {Console.Write("Dividendo? ");
   int X=Convert.ToInt16(Console.ReadLine());
   Console.Write("Divisor? ");
   int Y=Convert.ToInt16(Console.ReadLine());
   Console.WriteLine(X/Y);}
  catch (System.FormatException)
   { Console.WriteLine("Estamos à espera de números inteiros"); }
  catch (System.DivideByZeroException)
   {Console.WriteLine("O divisor tem de ser diferente de zero");}
}}}
```

? Exercício 3: Erro de conversão

```
using System;
namespace E3
{class ErrosDeConversao
 {public static void Main(string[] args)
```

```
  {double Desv=0;
   double[] D={5.78, 7, 6.54, 3.345, 10, 10.9};
   int[] I=new int[D.Length];
   for (int J=0; J<D.Length; J++)
    {try
    {I[J]=(int)D[J];
     if (I[J]!=D[J])
     throw new Exception();}
     catch
      {Console.WriteLine ("{0} foi truncado para {1}",D[J],I[J]);
       Desv += D[J] - I[J];}}
   Console.WriteLine("\n Somatório dos erros {0}", Desv);
}}}
```

? EXERCÍCIO 4: VALIDAÇÃO DE DADOS ALFANUMÉRICOS

```
using System;
namespace E4
{class ValidacaoString
 {public static void Main(string[] args)
  {const string Resp1="SIM", Resp2="NAO";
   const string Pergunta="Mais nomes? ";
   int I=-1, ContaA=0;
   string Rperg="";
   string[] Dados=new string[20];
   Boolean Er=true;
   do
    {try
     {I++;
      Console.Write("Nome ");
      Dados[I]=Console.ReadLine();
      Console.Write(Pergunta);
      Rperg=Console.ReadLine().ToUpper();
      if (Rperg != Resp1 && Rperg != Resp2)
      throw new Exception();}
     catch
      {Er=true;
       while (Er==true)
        {Console.WriteLine("Responda SIM ou NAO ");
         Console.Write(Pergunta);
         Rperg=Console.ReadLine().ToUpper();
         Er=(Rperg != Resp1 && Rperg != Resp2);}}
      finally
       {if (Dados[I].Substring(0, 1).ToUpper() == "A")
       ContaA++;}
   }while (Rperg == Resp1);
```

```
    Console.WriteLine("Digitou {0} nomes começados por A",
    ContaA);
}}}
```

? EXERCÍCIO 5: VALIDAÇÃO DE INTEIROS

```
using System;
namespace E5
{class ValidaIdade
 {static void Main(string[] args)
  {int X=0;
   bool Valida=false;
   while (Valida == false)
    {try
     {Console.Write("Digite a sua idade >");
      X=Convert.ToInt16(Console.ReadLine());
      Valida=X>=17 && X<=77;
      if (Valida==false)
       throw new Exception();}
     catch
    {Console.WriteLine ("A idade é um número inteiro igual
                         superior a 17 e inferior ou igual a
                         77");}}
  Console.WriteLine("Idade validada: {0} anos", X);
}}}
```

? EXERCÍCIO 6: VALIDAÇÃO DE CARACTERES

```
using System;
namespace E6
{class ValidaCateg
 {static void Main(string[] args)
  {Char Categ=' ' ;
   bool Valida=false;
   while (Valida == false)
    {try
     {Console.Write("Categoria profissional >");
      Categ=Convert.ToChar(Console.ReadLine());
      Valida=Categ >='C' && Categ<='E';
      if (Valida==false)
       throw new Exception();}
     catch
      {Console.WriteLine("Digite C, D ou E");}}
   Console.WriteLine("categoria validada: {0} ", Categ);
}}}
```

? EXERCÍCIO 7: ÍNDICE FORA DO LIMITE SUPERIOR

```
using System;
namespace E7
{class ErroIndice
 {static void Main(string[] args)
  {int[] A={ 4, 5, 6, 2, 8, 0 };
  int N=A.Length;
  bool Erro=false;
  for (int I=0; I<=N && Erro!=true; I++)
   {try
    {Console.WriteLine("{0,4:D}", A[I]);}
   catch (System.IndexOutOfRangeException)
    {Console.WriteLine("O índice ultrapassou o limite máximo");
     Erro=true;}}
  Console.WriteLine("Mas, continuamos o processamento");
}}}
```

? EXERCÍCIO 8: VALIDAÇÃO DE PALAVRAS-CHAVE

```
using System;
namespace E8
{class ValidaPassword
 {public static void Main(string[] args)
 {const int Limite=3;
  string Password="abcde";
  Console.Write("Password:");
  ConsoleKeyInfo C ;
  string Dig ;
  int Tenta=0;
  bool Invalida=true;
  while (Invalida == true && Tenta<Limite)
   {Dig="";
    try
     {for (int I=0; I < Password.Length; I++)
      {C=Console.ReadKey(true);
       Console.SetCursorPosition(10 + I, Console.CursorTop);
       Console.Write("*");
       Dig += C.KeyChar.ToString();}
       Invalida=(Password != Dig);
       if (Invalida)
       throw new Exception();}
    catch
     {Tenta++;
      Console.Clear();
```

```
        Console.Write("Password:");}}
   if (Tenta>=Limite)
    Console.WriteLine("\n Esgotou o número de tentativas");
    else
     Console.WriteLine("\nPalavra-chave válida");
}}}
```

? EXERCÍCIO 9: DETECÇÃO DE ERROS ENCADEADOS

```
using System;
namespace E9
{class DeteccaodeErrosEncadeados
 {static void Main(string[] args)
  {int N=4;
   int[] A=new int[N];
   for (int I=0; I<=N-1; I++)
    {try
     {Console.Write("Digite a[{0}] ", I);
      A[I]=Convert.ToInt16(Console.ReadLine());
      try
       {if (A[I] < 0)
        throw new Exception();}
       catch
        {Console.WriteLine("Negativo substituímos por 10");
         A[I]=10;}
       finally
        {Console.WriteLine("Multiplicámos por 100");
         A[I] *= 100; }}
      catch
       {Console.WriteLine("Carácter, string ou real-
       substituímos por 20");
        A[I]=20;}
      finally
       {Console.WriteLine("Multiplicámos por 5");
        A[I] *= 5;
        Console.WriteLine("a[{0}] passa a ser {1}",I, A[I]);}}
   Console.WriteLine("\n Vector:");
   foreach(int I in A)
    Console.WriteLine(I);
}}}
```

? EXERCÍCIO 10: HISTÓRICO DE ERROS

```
using System;
namespace E10
```

```
{class HistoricoErros
 {static void Main(string[] args)
  {string Historico="";
   int N=4;
   int[] A=new int[N];
   for (int I=0; I<=N - 1; I++)
    {try
     {Console.Write("Digite a[{0}] ", I);
      A[I]=Convert.ToInt16(Console.ReadLine());
      try
       {if (A[I] < 0)
        throw new Exception();}
      catch
       {Historico+="Negativo - substituímos por 10 \n";
        A[I]=10;}
      finally
       {Historico += "Multiplicámos por 100 \n";
       A[I] *= 100;}}
    catch
     {Historico+="Caracter, string ou real";
      Historico+=" - substituímos por 20 \n";
      A[I]=20;}
    finally
     {Historico+="Multiplicámos por 5";
      A[I] *= 5;
      Historico+=" A["+I+"] passa a ser "+A[I]+"\n";}}
   Console.WriteLine(Historico);
}}}
```