

Série Convergente 2

Mário Leite

...

Na postagem anterior foi comentado sobre séries convergentes e divergentes, apresentado um programa (codificado em Visualg) sobre a expressão da série $x(x+1)$ onde foi calculada a soma dos seus termos quando esse número de termos tende ao infinito. O resultado obtido mostrou que esse limite é **1**, onde a **figura 1** mostra a expressão desta série, indicando o seu limite.

$$\lim_{x \rightarrow \infty} [x/(x+1)]$$

Figura 2 - Expressão a ser calculada

Nesta presente postagem é mostrado o programa “**Serie2**”, codificado em C#, agora considerando duas situações: somando um *milhão* e um *bilhão* de termos., respectivamente A **figura 1a** mostra a entrada do programa para um milhão de termos e a **figura 1b** a saída do programa e o tempo de processamento com este número de termos. A **figura 2a** mostra a entrada para um *bilhão* de termos e a **figura 2b** a saída com o tempo gasto no processamento para este número alto de termos. O programa foi rodado nas mesmas condições de *hardware* utilizado na postagem anterior para um *milhão* de termos, quando foi codificado em Visualg. Note a brutal diferença de tempo de processamento: **3995,3 segundos** em Visualg e de apenas **48,9 segundos** quando codificado em C#. Considerando um *bilhão* de termos, o tempo de processamento foi um pouco mais de dez horas, o que dá em torno de **37312,5 segundos**, rodado nesta linguagem, no mesmo equipamento e nas mesmas condições de memória e espaço útil em disco.

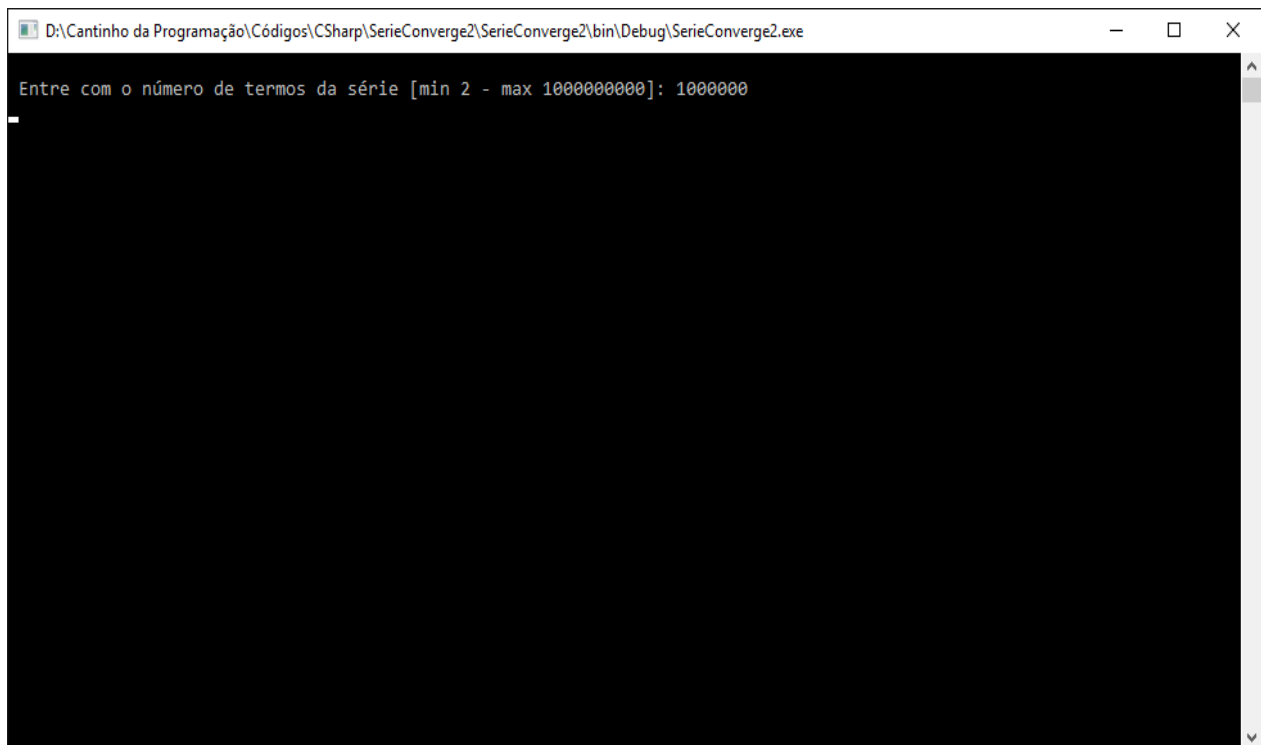


Figura 1a - Entrando com um milhão de termos da série

```
D:\Cantinho da Programação\Códigos\CSharp\SerieConverge2\SerieConverge2\bin\Debug\SerieConverge2.exe
Número de termos: 999977      Valor da série: 0,999998999978
Número de termos: 999978      Valor da série: 0,999998999979
Número de termos: 999979      Valor da série: 0,99999899998
Número de termos: 999980      Valor da série: 0,999998999981
Número de termos: 999981      Valor da série: 0,999998999982
Número de termos: 999982      Valor da série: 0,999998999983
Número de termos: 999983      Valor da série: 0,999998999984
Número de termos: 999984      Valor da série: 0,999998999985
Número de termos: 999985      Valor da série: 0,999998999986
Número de termos: 999986      Valor da série: 0,999998999987
Número de termos: 999987      Valor da série: 0,999998999988
Número de termos: 999988      Valor da série: 0,999998999989
Número de termos: 999989      Valor da série: 0,99999899999
Número de termos: 999990      Valor da série: 0,999998999991
Número de termos: 999991      Valor da série: 0,999998999992
Número de termos: 999992      Valor da série: 0,999998999993
Número de termos: 999993      Valor da série: 0,999998999994
Número de termos: 999994      Valor da série: 0,999998999995
Número de termos: 999995      Valor da série: 0,999998999996
Número de termos: 999996      Valor da série: 0,999998999997
Número de termos: 999997      Valor da série: 0,999998999998
Número de termos: 999998      Valor da série: 0,999998999999
Número de termos: 999999      Valor da série: 0,999999
Número de termos: 1000000      Valor da série: 0,999999000001

Início do processamento: 8/11/2022 - 12:16:40
Término do processamento: 8/11/2022 - 12:17:29

Tempo de processamento [hh:mm:ss:ms]: 00:00:48.8704799
```

Figura 1b - Saída do programa para um milhão de termos da série

```
D:\Cantinho da Programação\Códigos\CSharp\SerieConverge2\SerieConverge2\bin\Debug\SerieConverge...
Entre com o número de termos da série [min 2 - max 1000000000]: 1000000000
```

Figura 2a - Entrando com um bilhão de termos da série

```
D:\Cantinho da Programação\Códigos\CSharp\Serie2\Serie2\bin\Debug\Serie2.exe

Número de termos: 999999976      Valor da séries: 0,999999999000000
Número de termos: 999999977      Valor da séries: 0,999999999000000
Número de termos: 999999978      Valor da séries: 0,999999999000000
Número de termos: 999999979      Valor da séries: 0,999999999000000
Número de termos: 999999980      Valor da séries: 0,999999999000000
Número de termos: 999999981      Valor da séries: 0,999999999000000
Número de termos: 999999982      Valor da séries: 0,999999999000000
Número de termos: 999999983      Valor da séries: 0,999999999000000
Número de termos: 999999984      Valor da séries: 0,999999999000000
Número de termos: 999999985      Valor da séries: 0,999999999000000
Número de termos: 999999986      Valor da séries: 0,999999999000000
Número de termos: 999999987      Valor da séries: 0,999999999000000
Número de termos: 999999988      Valor da séries: 0,999999999000000
Número de termos: 999999989      Valor da séries: 0,999999999000000
Número de termos: 999999990      Valor da séries: 0,999999999000000
Número de termos: 999999991      Valor da séries: 0,999999999000000
Número de termos: 999999992      Valor da séries: 0,999999999000000
Número de termos: 999999993      Valor da séries: 0,999999999000000
Número de termos: 999999994      Valor da séries: 0,999999999000000
Número de termos: 999999995      Valor da séries: 0,999999999000000
Número de termos: 999999996      Valor da séries: 0,999999999000000
Número de termos: 999999997      Valor da séries: 0,999999999000000
Número de termos: 999999998      Valor da séries: 0,999999999000000
Número de termos: 999999999      Valor da séries: 0,999999999000000
Número de termos: 1000000000      Valor da séries: 0,999999999000000

Início do processamento: 10/11/2022 - 12:31:54
Término do processamento: 10/11/2022 - 22:53:47

Tempo de processamento [hh:mm:ss:ms]: 10:21:52.4909046
```

Figura 2b - Saída do programa para um bilhão de termos da série

Código da aplicação

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

namespace Serie2
{
    //Calcula o limite de uma série
    //Em C#
    //Autor: Mário Leite
    internal class Program
    {
        static void Main(string[] args)
        {
            const int MAX=1000000000;
            int n;
            double Sx;

            /* Define variáveis de espaços formatadas adequadamente */
            string strT = "      Número de termos: ";
            string strS = "      Valor da série: ";
            string strIni = "      Início do processamento: ";
            string strFim = "      Término do processamento: ";
        }
    }
}
```

```

/* Define instâncias temporais iniciais */
int diaHoje1 = int.Parse(DateTime.Now.Day.ToString());
int mesHoje1 = int.Parse(DateTime.Now.Month.ToString());
int anoHoje1 = int.Parse(DateTime.Now.Year.ToString());
string hora1 = DateTime.Now.ToString("HH:mm:ss tt"); //formato 24 horas
string data1 = diaHoje1 + "/" + mesHoje1 + "/" + anoHoje1;
Stopwatch ObjSw = new Stopwatch(); //cria objeto de tempo
ObjSw.Start(); //liga o cronômetro

Console.WriteLine(); //apenas salta uma linha

/* Valida entrada do número de termos da série */
n = 1;
while ((n < 2) || (n > MAX))
{
    Console.Write(" Digite número de termos da série [min 2 - max " +MAX + "]: ");
    n = int.Parse(Console.ReadLine());
} //fim da validação

Console.WriteLine();

/* loop para gerar os valores da Série */
for (int x = 1; x <= n; x++)
{
    Sx = (double)x / (x + 1); //expressão da Série
    string k = x.ToString(); //converte x para string
    double pot = Math.Pow(10, k.Length); //parcela de espaço ser adicionada
    int spc = (10 - k.Length) + k.Length; //espaço total ocupado pelo termo
    string termo = k.PadRight(spc, ' '); //preenche o termo com espaços em branco
    var Serie = String.Format("{0:0.0000000000000000}", Sx); //formata valor da Série
    Console.WriteLine(strT + termo + strS + Serie); //imprime linha do resultado
} //fim do loop de geração dos valores da série

Console.WriteLine();

/* Define instâncias temporais finais e mostra o tempo de processamento */
ObjSw.Stop(); //desliga o cronômetro
int diaHoje2 = int.Parse(DateTime.Now.Day.ToString());
int mesHoje2 = int.Parse(DateTime.Now.Month.ToString());
int anoHoje2 = int.Parse(DateTime.Now.Year.ToString());
string data2 = diaHoje2 + "/" + mesHoje2 + "/" + anoHoje2;
string hora2 = DateTime.Now.ToString("HH:mm:ss tt"); //formato 24 horas
Console.WriteLine(strIni + data1 + " - " + hora1);
Console.WriteLine(strFim + data2 + " - " + hora2);
Console.WriteLine();
TimeSpan tempo = ObjSw.Elapsed; //calcula o tempo de processamento
Console.WriteLine("          Tempo de processamento [hh:mm:ss:ms]: {0}", tempo);
Console.ReadKey();
} //fim do método principal
} //fim da classe
} //fim do programa

```

