You have been watching a video for some time. Knowing the total video duration find out what portion of the video you have already watched.

Example

For part = "02:20:00" and total = "07:00:00", the output should be  
videoPart(part, total) = [1, 3].

You have watched 1 / 3 of the whole video.

Input/Output

* **[execution time limit] 3 seconds (cs)**
* **[input] string part**

A string of the following format "hh:mm:ss" representing the time you have been watching the video.

* **[input] string total**

A string of the following format "hh:mm:ss" representing the total video duration.

* **[output] array.integer**
  + An array of the following format [a, b](where a / b is [a reduced fraction](keyword://reduced-fraction)).

**[C#] Syntax Tips**

// Prints help message to the console

// Returns a string

**string** **helloWorld**(**string** name) {

Console.Write("This prints to the console when you Run Tests");

**return** "Hello, " + name;

}

<https://app.codesignal.com/arcade/code-arcade/time-river/DKjM5qLz5ccK2ukhH/description>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

public class Fraction

{

/// <summary>

/// Class attributes/members

/// </summary>

long m\_iNumerator;

long m\_iDenominator;

/// <summary>

/// Constructors

/// </summary>

public Fraction()

{

Initialize(0, 1);

}

public Fraction(long iWholeNumber)

{

Initialize(iWholeNumber, 1);

}

public Fraction(long iNumerator, long iDenominator)

{

Initialize(iNumerator, iDenominator);

}

/// <summary>

/// Internal function for constructors

/// </summary>

private void Initialize(long iNumerator, long iDenominator)

{

Numerator = iNumerator;

Denominator = iDenominator;

ReduceFraction(this);

}

/// <summary>

/// Properites

/// </summary>

public long Denominator

{

get

{ return m\_iDenominator; }

set

{

if (value != 0)

m\_iDenominator = value;

else throw new Exception("error");

//throw new FractionException("Denominator cannot be assigned a ZERO Value");

}

}

public long Numerator

{

get

{ return m\_iNumerator; }

set

{ m\_iNumerator = value; }

}

public long Value

{

set

{

m\_iNumerator = value;

m\_iDenominator = 1;

}

}

/// <summary>

/// The function returns the current Fraction object as a string

/// </summary>

public override string ToString()

{

string str;

if (this.Denominator == 1)

str = this.Numerator.ToString();

else

str = this.Numerator + "/" + this.Denominator;

return str;

}

/// <summary>

/// The function returns GCD of two numbers (used for reducing a Fraction)

/// </summary>

private static long GCD(long iNo1, long iNo2)

{

// take absolute values

if (iNo1 < 0) iNo1 = -iNo1;

if (iNo2 < 0) iNo2 = -iNo2;

do

{

if (iNo1 < iNo2)

{

long tmp = iNo1; // swap the two operands

iNo1 = iNo2;

iNo2 = tmp;

}

iNo1 = iNo1 % iNo2;

} while (iNo1 != 0);

return iNo2;

}

/// <summary>

/// The function reduces(simplifies) a Fraction object by dividing both its numerator

/// and denominator by their GCD

/// </summary>

public static void ReduceFraction(Fraction frac)

{

try

{

if (frac.Numerator == 0)

{

frac.Denominator = 1;

return;

}

long iGCD = GCD(frac.Numerator, frac.Denominator);

frac.Numerator /= iGCD;

frac.Denominator /= iGCD;

if (frac.Denominator < 0) // if -ve sign in denominator

{

//pass -ve sign to numerator

frac.Numerator \*= -1;

frac.Denominator \*= -1;

}

} // end try

catch (Exception ex)

{

//throw new FractionException("Cannot reduce Fraction: " + exp.Message);

throw new Exception("error");

}

}

}

class Program

{

static int[] videoPart(string part, string total)

{

string[] input\_part = part.Split(':');

int horas\_part = int.Parse(input\_part[0]);

int min\_part = int.Parse(input\_part[1]);

int seg\_part = int.Parse(input\_part[2]);

string[] input\_total = total.Split(':');

int horas\_total = int.Parse(input\_total[0]);

int min\_total = int.Parse(input\_total[1]);

int seg\_total = int.Parse(input\_total[2]);

TimeSpan t1 = new TimeSpan(horas\_part, min\_part, seg\_part);

TimeSpan t2 = new TimeSpan(horas\_total, min\_total, seg\_total);

//

//Console.WriteLine(t1.TotalSeconds);

//Console.WriteLine(t2.TotalSeconds);

Fraction fraction = new Fraction((long)t1.TotalSeconds, (long)t2.TotalSeconds);

Fraction.ReduceFraction(fraction);

return new int[] { (int)fraction.Numerator, (int)fraction.Denominator };

}

static void Main(string[] args)

{

string part = "02:20:00";

string total = "07:00:00";

int[] res = videoPart(part, total);

Console.WriteLine(res[0] + "/" + res[1]);

Console.ReadLine();

}

}