# Math.Asin(Rational) Method

名前空間: WS.Theia.ExtremelyPrecise

アセンブリ: ExtremelyPrecise.dll

サインが指定数となる角度を返します。

public static WS.Theia.ExtremelyPrecise.Rational Asin(WS.Theia.ExtremelyPrecise.Rational sin);

## パラメーター

sin　Rational  
サインを表す数。-1以上1以下である必要があります。

## 戻り値

Rational  
-π/2 ≤θ≤π/2 の、ラジアンで表した角度 θ。 または sin < -1 または sin > 1、あるいは sin が NaN と等しい場合は、NaN。

# 例

次の例では、Asinを使用して台形の内角を計算しています。

/// <summary>  
/// The following class represents simple functionality of the trapezoid.  
/// </summary>  
using System;  
using WS.Theia.ExtremelyPrecise;  
  
namespace MathClassCS  
{  
 class MathTrapezoidSample  
 {  
 private Rational m\_longBase;  
 private Rational m\_shortBase;  
 private Rational m\_leftLeg;  
 private Rational m\_rightLeg;  
  
 public MathTrapezoidSample(Rational longbase, Rational shortbase, Rational leftLeg, Rational rightLeg)  
 {  
 m\_longBase = Math.Abs(longbase);  
 m\_shortBase = Math.Abs(shortbase);  
 m\_leftLeg = Math.Abs(leftLeg);  
 m\_rightLeg = Math.Abs(rightLeg);  
 }  
  
 private Rational GetRightSmallBase()  
 {  
 return (Math.Pow(m\_rightLeg,2.0) - Math.Pow(m\_leftLeg,2.0) + Math.Pow(m\_longBase,2.0) + Math.Pow(m\_shortBase,2.0) - 2\* m\_shortBase \* m\_longBase)/ (2\*(m\_longBase - m\_shortBase));  
 }  
  
 public Rational GetHeight()  
 {  
 Rational x = GetRightSmallBase();  
 return Math.Sqrt(Math.Pow(m\_rightLeg,2.0) - Math.Pow(x,2.0));  
 }  
  
 public Rational GetSquare()  
 {  
 return GetHeight() \* m\_longBase / 2.0;  
 }  
  
 public Rational GetLeftBaseRadianAngle()  
 {  
 Rational sinX = GetHeight()/m\_leftLeg;  
 return Math.Round(Math.Asin(sinX),2);  
 }  
  
 public Rational GetRightBaseRadianAngle()  
 {  
 Rational x = GetRightSmallBase();  
 Rational cosX = (Math.Pow(m\_rightLeg,2.0) + Math.Pow(x,2.0) - Math.Pow(GetHeight(),2.0))/(2\*x\*m\_rightLeg);  
 return Math.Round(Math.Acos(cosX),2);  
 }  
  
 public Rational GetLeftBaseDegreeAngle()  
 {  
 Rational x = GetLeftBaseRadianAngle() \* 180/ Math.PI;  
 return Math.Round(x,2);  
 }  
  
 public Rational GetRightBaseDegreeAngle()  
 {  
 Rational x = GetRightBaseRadianAngle() \* 180/ Math.PI;  
 return Math.Round(x,2);  
 }  
  
 static void Main(string[] args)  
 {  
 MathTrapezoidSample trpz = new MathTrapezoidSample(20.0, 10.0, 8.0, 6.0);  
 Console.WriteLine("The trapezoid's bases are 20.0 and 10.0, the trapezoid's legs are 8.0 and 6.0");  
 Rational h = trpz.GetHeight();  
 Console.WriteLine("Trapezoid height is: " + h.ToString());  
 Rational dxR = trpz.GetLeftBaseRadianAngle();

Console.WriteLine("Trapezoid left base angle is: " + dxR.ToString() + " Radians");  
 Rational dyR = trpz.GetRightBaseRadianAngle();  
 Console.WriteLine("Trapezoid right base angle is: " + dyR.ToString() + " Radians");  
 Rational dxD = trpz.GetLeftBaseDegreeAngle();  
 Console.WriteLine("Trapezoid left base angle is: " + dxD.ToString() + " Degrees");  
 Rational dyD = trpz.GetRightBaseDegreeAngle();  
 Console.WriteLine("Trapezoid left base angle is: " + dyD.ToString() + " Degrees");  
 }  
 }  
}

# 注釈

戻り値が正の場合、x軸のプラス方向から反時計回りの角度を示します。戻り値が負の場合、x軸のプラス方向から時計回りの角度を示します。戻り値に180/Math.PIを乗算する事でラジアンから度に変換できます。

# 適用対象

### .NET Core

2.0

### .NET Framework

4.6.1

### .NET Standard

2.0

### UWP

10.0.16299

### Xamarin.Android

8.0

### Xamarin.iOS

10.14

### Xamarin.Mac

3.8