Math.Acos(Rational) Method

名前空間: WS.Theia.ExtremelyPrecise

アセンブリ: ExtremelyPrecise.dll

コサインが指定数となる角度を返します。

```
public static WS.Theia.ExtremelyPrecise.Rational
Acos(WS.Theia.ExtremelyPrecise.Rational cos);
```

パラメーター

cos　Rational

コサインを表す数。-1 以上 1 以下である必要があります。

戻り値

Rational

$0 \leq \theta \leq \pi$ の、ラジアンで表した角度 θ。 または $\cos < -1$ または $\cos > 1$、あるいは cos が NaN と等しい場合は、NaN。

例

次の例では、Acos 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