PARTE “A”

```
using System ;
 using System.Collections.Generic ;
 using System.Linq ;
 using System.Text ;
 using System.Threading.Tasks ;
 namespace Operadores_aritmeticos
 {

class Program
   {

        static void Main ( string [ ] args )
        {
            // Operadores aritmeticos
            int num = 6 , num2 = 5 ;
        }
            Console.WriteLine (" El resultado de la suma es : + ( num - num2 ) ) ;
            Console.ReadKey();
   }
 }
```

```
using System ;
 using System.Collections.Generic ;
 using System.Linq ;
 using System.Text ;
 using System.Threading.Tasks ;
 namespace Operadores_aritmeticos
 {

class Program
   {

        static void Main ( string [ ] args )
        {
            // Operadores aritmeticos
           double num , pot , resultado ;
Console.WriteLine ( " Digite el numero que quiere elevar : " ) ;
num = Convert.ToDouble ( Console.ReadLine ( ) ) ;
Console.WriteLine ( " Digite a la potencia que quiere elevar : " ) ;
pot = Convert.ToDouble ( Console.ReadLine ( ) ) ;
resultado Math,Pow ( num , pot ));
Console.WriteLine  (" El resultado es : + resultado") ;
Console.ReadKey ( ) ;}}}
```

Ejemplos buscados.

```
/// <summary>
/// The following class represents simple functionality of the trapezoid.
/// </summary>
using System;

namespace MathClassCS
{
   class MathTrapezoidSample
   {
      private double m_longBase;
      private double m_shortBase;
      private double m_leftLeg;
      private double m_rightLeg;

      public MathTrapezoidSample(double longbase, double shortbase, double leftLeg, double
rightLeg)
      {
         m_longBase = Math.Abs(longbase);
         m_shortBase = Math.Abs(shortbase);
         m_leftLeg = Math.Abs(leftLeg);
         m_rightLeg = Math.Abs(rightLeg);
      }

      private double 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 double GetHeight()
      {
         double x = GetRightSmallBase();
         return Math.Sqrt(Math.Pow(m_rightLeg,2.0) - Math.Pow(x,2.0));
      }

      public double GetSquare()
      {
         return GetHeight() * m_longBase / 2.0;
      }

      public double GetLeftBaseRadianAngle()
      {
         double sinX = GetHeight()/m_leftLeg;
```

```
            return Math.Round(Math.Asin(sinX),2);
        }

        public double GetRightBaseRadianAngle()
        {
            double x = GetRightSmallBase();
            double 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 double GetLeftBaseDegreeAngle()
        {
            double x = GetLeftBaseRadianAngle() * 180/ Math.PI;
            return Math.Round(x,2);
        }

        public double GetRightBaseDegreeAngle()
        {
            double 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");
            double h = trpz.GetHeight();
            Console.WriteLine("Trapezoid height is: " + h.ToString());
            double dxR = trpz.GetLeftBaseRadianAngle();
            Console.WriteLine("Trapezoid left base angle is: " + dxR.ToString() + " Radians");
            double dyR = trpz.GetRightBaseRadianAngle();
            Console.WriteLine("Trapezoid right base angle is: " + dyR.ToString() + " Radians");
            double dxD = trpz.GetLeftBaseDegreeAngle();
            Console.WriteLine("Trapezoid left base angle is: " + dxD.ToString() + " Degrees");
            double dyD = trpz.GetRightBaseDegreeAngle();
            Console.WriteLine("Trapezoid left base angle is: " + dyD.ToString() + " Degrees");
        }
    }
}
```

```
using System;

public class Example
```

```
{
   public static void Main()
   {
      // Define several positive and negative dividends.
      int[] dividends = { Int32.MaxValue, 13952, 0, -14032,
                                   Int32.MinValue };
      // Define one positive and one negative divisor.
      int[] divisors = { 2000, -2000 };

      foreach (int divisor in divisors)
      {
         foreach (int dividend in dividends)
         {
            int remainder;
            int quotient = Math.DivRem(dividend, divisor, out remainder);
            Console.WriteLine(@"{0:N0} \ {1:N0} = {2:N0}, remainder {3:N0}",
                              dividend, divisor, quotient, remainder);
         }
      }
   }
}
// The example displays the following output:
//       2,147,483,647 \ 2,000 = 1,073,741, remainder 1,647
//       13,952 \ 2,000 = 6, remainder 1,952
//       0 \ 2,000 = 0, remainder 0
//       -14,032 \ 2,000 = -7, remainder -32
//       -2,147,483,648 \ 2,000 = -1,073,741, remainder -1,648
//       2,147,483,647 \ -2,000 = -1,073,741, remainder 1,647
//       13,952 \ -2,000 = -6, remainder 1,952
//       0 \ -2,000 = 0, remainder 0
//       -14,032 \ -2,000 = 7, remainder -32
//       -2,147,483,648 \ -2,000 = 1,073,741, remainder -1,648
```

PARTE “B”

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

 using  System.Collections.Generic ;
 namespace Operadores
   {
     class Program
     {

        static void Main ( string [ ] args )
        {
            // Operadores relaciones
            double peso ;
            Console.WriteLine ( " Digita tu peso : " ) ;
            peso = Convert.ToDouble ( Console.ReadLine ( ) ) ;
            if (peso > 100){
                Console.WriteLine("tu peso es normal");
            }
            Console.ReadKey();
        }
     }
 }
```