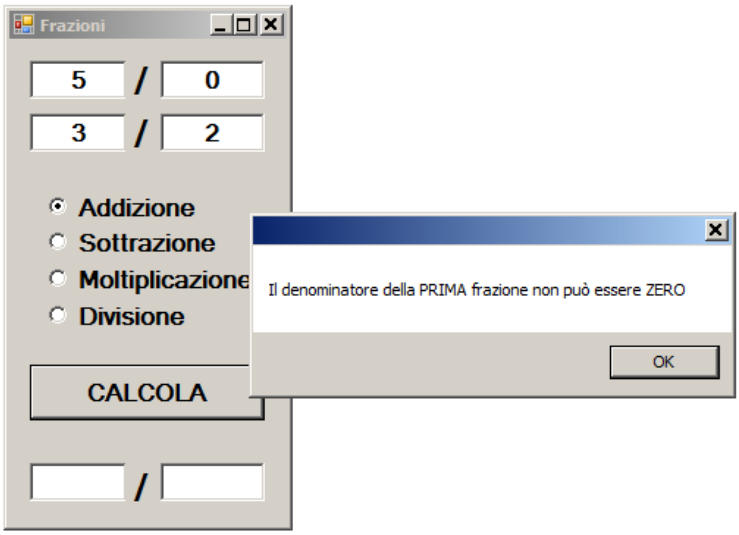
Esempio Pilota:

ClasseFrazioneConEventiInCSharp

Definizione della Form



Codice della Form

```
using System;
using System.Windows.Forms;

namespace ClasseFrazioneConEventiInCSharp
{
   public partial class frmOperazioniSulleFrazioni : Form
   {
      public frmOperazioniSulleFrazioni()
      {
         InitializeComponent();
      }


      /* questa variabile serve per far sì che, in caso di
        errore nelle frazioni, NON si proceda al calcolo */
      private bool SiEVerificatoUnErrore;


      // Dichiariamo gli oggetti frazione, a livello globale ...
      Frazione FA;
      Frazione FB;
      Frazione FR;


      private void frmOperazioniSulleFrazioni_Load
                                        (object sender, EventArgs e)
      {
         /* creiamo FA e associamo "GestisciDenominatoreNulloInFA"
           (gestore dell'evento) al suo evento "DenominatoreNullo" */
         FA = new Frazione();
         FA.DenominatoreNullo += GestisciDenominatoreNulloInFA;

         // ... stessa cosa per FB ...
         FB = new Frazione();
         FB.DenominatoreNullo += GestisciDenominatoreNulloInFB;

         // creiamo anche FR per il risultato ...
         FR = new Frazione();
      }

      // questo è il metodo che GESTISCE
        L'EVENTO "click" sul PULSANTE plsCalcola ...

      private void plsCalcola_Click(object sender, EventArgs e)
      {
         // inizializziamo a "false" la variabile che segnala errori ...
         SiEVerificatoUnErrore = false;

         // impostiamo Numeratore e Donominatore di FA ...
         FA.Numeratore = Convert.ToInt16(txtNumA.Text);
         FA.Denominatore = Convert.ToInt16(txtDenA.Text);

         // impostiamo Numeratore e Donominatore di FB ...
         FB.Numeratore = Convert.ToInt16(txtNumB.Text);
         FB.Denominatore = Convert.ToInt16(txtDenB.Text);

         // Procediamo al calcolo solo se non ci sono stati errori ...

         if  ( ! SiEVerificatoUnErrore)
         {
            if (radAddizione.Checked)
               FR.Somma(FA, FB);
            else if (radSottrazione.Checked)
               FR.Differenza(FA, FB);
            else if (radMoltiplicazione.Checked)
               FR.Prodotto(FA, FB);
            else if (radDivisione.Checked)
               FR.Quoziente(FA, FB);

            // visualizziamo Numeratore e Denominatore di FR ...
            txtNumRis.Text = FR.Numeratore.ToString();
            txtDenRis.Text = FR.Denominatore.ToString();
         }
      }

      // questo è il GESTORE per L'EVENTO "DenominatoreNullo" di FA ...

      private void GestisciDenominatoreNulloInFA
                                        (object sender, EventArgs e)
      {
         MessageBox.Show
               ("Il denominatore della PRIMA frazione non può essere ZERO");

         // riposizionati sul denominatore della prima frazione ...
         txtDenA.Focus();
         txtDenA.SelectAll();

         // ricordati che si è verificato un errore ...
         SiEVerificatoUnErrore = true;
      }

      // questo è il GESTORE per L'EVENTO "DenominatoreNullo" di FB ...

      private void GestisciDenominatoreNulloInFB
                                        (object sender, EventArgs e)
      {
         MessageBox.Show
              ("Il denominatore della SECONDA frazione non può essere ZERO");

         // riposizionati sul denominatore della prima frazione ...
         txtDenB.Focus();
         txtDenB.SelectAll();

         // ricordati che si è verificato un errore ...
         SiEVerificatoUnErrore = true;
      }
   }
}
```

Codice della Classe Frazione

```
using System;
using System.Windows.Forms;

namespace ClasseFrazioneConEventiInCSharp
{
   class Frazione
   {


      // Area Dati interna ...

      private int _Num;
      private int _Den;



      // Costruttori ...

      public Frazione()
      {
         _Num = 0;
         _Den = 1;
      }



      // Eventi ...

      // Definisco un evento e lo chiamo "DenominatoreNullo".
      // L'evento è sempre preferibile definirlo di tipo EventHandler ...

      public event EventHandler DenominatoreNullo;



      // Proprietà ...

      public int Denominatore
      {
         get { return _Den; }

         set
         {
            if (value == 0)
            {

// in questo punto "generiamo" l'evento, con i parametri (OBBLIGATORI):
//   - l'oggetto stesso (this) che sta generando l'evento
//   - un oggetto di tipo EventArgs, utile per passare inform. sull'evento

               DenominatoreNullo  (  this,  new EventArgs ( )  );
            }
            else
               _Den = value;
         }
      }



      public int Numeratore
      {
         get { return _Num; }
         set { _Num = value; }
      }


      // Metodi ...

      public void Somma(Frazione F1, Frazione F2)
      {
         Numeratore = F2.Denominatore * F1.Numeratore +
                      F1.Denominatore * F2.Numeratore;
         Denominatore = F1.Denominatore * F2.Denominatore;
         Semplifica();
      }

      public void Differenza(Frazione F1, Frazione F2)
      {
         Numeratore = F2.Denominatore * F1.Numeratore -
                      F1.Denominatore * F2.Numeratore;
         Denominatore = F1.Denominatore * F2.Denominatore;
         Semplifica();
      }

      public void Prodotto(Frazione F1, Frazione F2)
      {
         Numeratore = F1.Numeratore * F2.Numeratore;
         Denominatore = F1.Denominatore * F2.Denominatore;
         Semplifica();
      }

      public void Quoziente(Frazione F1, Frazione F2)
      {
         Numeratore = F1.Numeratore * F2.Denominatore;
         Denominatore = F1.Denominatore * F2.Numeratore;
         Semplifica();
      }

      // il metodo Semplifica richiama il sottoprogramma privato MCD ...

      public void Semplifica()
      {
         int M;

         M = MCD(Numeratore, Denominatore);

         Numeratore = Numeratore / M;
         Denominatore = Denominatore / M;
      }


      // sottoprogrammi Privati ...

      // il sottoprogramma privato MCD utilizza il metodo di Euclide
      // nella versione cosiddetta delle "sottrazioni successive" ...

      private int MCD(int N1, int N2)
      {
         if ((N1 == 0) || (N2 == 0))
            return 1;
         else
         {
            while (N1 != N2)
               if (N1 > N2)
                  N1 = N1 - N2;
               else
                  N2 = N2 - N1;

            return N1;
         }
      }
   }
}
```