Creare un Delegato scrivendo il nome della Funzione o dell'Azione

Grazie alla tecnica di REFLECTION, si vuole creare un Delegato al volo digitando in una casella di input il nome della funzione o della azione da applicare.

In sostanza si parte da una stringa contenente il nome della funzione, o della azione, che si vuole utilizzare.

Innanzitutto si crea un oggetto di tipo MethodInfo

```
MethodInfo m1 =    type.GetMethod("Funzione");
MethodInfo m2 =    type.GetMethod("Azione");
```

Poi si crea il delegato a partire dall'oggetto MethodInfo

```
Func<int,int> f = (Func<int,int>) Delegate.CreateDelegate(typeof(Func<int,int>), m1);
Action<int> a = (Action<int>) Delegate.CreateDelegate(typeof(Action<int>), m2);
```

A questo punto si può utilizzare il delegato

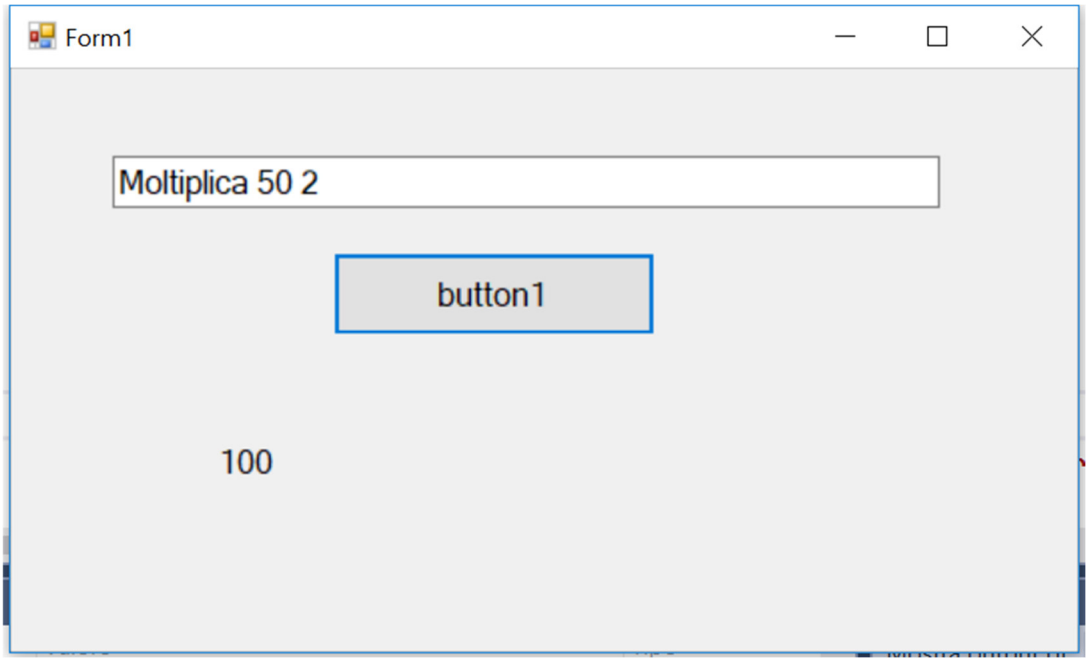
```
int x = 1;  // valore su cui applicare i calcoli e le azioni previste
int risultato = f( x );
azione( x );
```

I metodi statici da assegnare al delegato sono i seguenti:

```
public static void Azione(int n)
{
    Console.WriteLine(n);  // azione di esempio
}

public static int Funzione(int n)
{
    return n + 1;  // calcolo di esempio
}
```

Con questa tecnica si vuole scrivere un piccolo programma che interpreta ed esegue l'operazione scritta in una casella di input:



Il form del programma mostra l'espressione digitata dall'utente e il suo risultato

Il codice completo dell'applicazione è il seguente:

Form1.cs

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


namespace WindowsFormsApp3
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            // valuta espressioni del tipo
            // Moltiplica 3.4 5
            // Somma 4 5
            // Dividi 5 7
            // Sottrai 55 9

            // leggo l'espressione dal text box
            string espressione = textBox1.Text;
            string[] arr = espressione.Split(' ');

            // alternative equivalenti per ricavare il tipo della classe Form1
            //Type type = typeof(WindowsFormsApp3.Form1);
            Type type = Type.GetType("WindowsFormsApp3.Form1");
```

```
            // gli elementi dell'espressione
            string nomeOperazione = arr[0];
            double x = Convert.ToDouble(arr[1]);
            double y = Convert.ToDouble(arr[2]);

            // dal nome dell'operazione ottengo il corrispondente Delegato
            MethodInfo m = type.GetMethod(nomeOperazione);
            // creo il delegato
            Func<double,double,double> f = (Func<double,double,double>)
                     Delegate.CreateDelegate(typeof(Func<double,double,double>), m);

            // effettuo il calcolo sfruttando il delegato f
            double z = f(x, y);

            // scrivo il risultato
            label1.Text = z.ToString();
        }

        // Metodi Statici da assegnare ai delegati

        public static double Moltiplica(double x, double y)
        { return x * y; }

        public static double Dividi(double x, double y)
        { return x / y; }

        public static double Somma(double x, double y)
        { return x + y; }

        public static double Sottrai(double x, double y)
        { return x - y; }

    }
}
```