

I den foregående opgave udvidede du din regnemaskine med nye funktioner for plus, minus, gange og divider. Så nu er det en rigtig regnemaskine. Men vi skal lige have kigget lidt mere på koden bag ved.

For som det ser ud nu bruger du kode der ligner det herunder:

```
private double RegneFunktion(char regneTegn)
{
    double tal1 = 0;
    double tal2 = 0;
    double resultat = 0;

    try
    {
        tal1 = Convert.ToDouble(tbTal1.Text);
        tal2 = Convert.ToDouble(tbTal2.Text);
    }
    catch (Exception exc)
    {
        MessageBox.Show(exc.Message, "Hov...");
    }

    if (regneTegn == '+')
    {
        resultat = tal1 + tal2;
    }

    return resultat;
}
```

Og det er der egentlig ikke noget i vejen med. Det er dog lige det der '+' tegn. For her tager vi fat i et tegn. Og som du sikkert allerede ved, så er der altså 2^16 brugbare tegn i din windows. ASCII tabellen indeholder 255 tegn i sin grundform. Og 16 bit eller 65536 tegn i den udvidede tegnetabel, som din windows bruger...

Altså er det forholdsvis nemt at bruge et forkert tegn her! Hvis eks. Vis din kode skal distribueres imellem andre programmører. Hvordan ved de så hvilken tegn de er begrænset til at benytte???

Det ved de ikke. Og inden længe er der nogen der vil prøve modulus '%' og så går det galt!

Derfor skal vi i tilfælde som disse benytte noget der hedder en enumeration. Dette er i C# beskrevet som en bruger defineret type. Og ser i sin grundform sådant ud:

```
enum MyEnum
{
}
```

Du kan få visual studio til at lave den for dig, hvis du skriver enum og trykker to gange på "tab" tasten.

En enum, kan oprettes inden i din klasse, eller udenfor. Du bestemmer selv. Og for at du kan bruge den, skal du, som med en hver anden type, oprette den i din kode. Herunder har jeg vist hvordan dette kunne se ud når den er oprettet inden i klassen (husk på at i eksemplet her, er den privat for klassen!) :

```
public partial class MainWindow : Window
{
    MyEnum testEnum;

    public MainWindow()
    {
        InitializeComponent();
    }

    enum MyEnum
    {
    }
}
```

Og herefter er den så klar til at benytte. Problemet er bare at der ikke er nogle valgmuligheder i vores enum endnu. Så det skal vi have tilføjet! Jeg går også lige skridtet videre og omskriver vores enum, så navnet giver lidt mening, i forbindelse med vores regnemaskine:

```
public partial class MainWindow : Window
{
    Regnetegn regnetegn;

    public MainWindow()
    {
        InitializeComponent();
    }

    /// <summary>
    /// Dette er en enum der definere de gængse regnetegn
    /// </summary>
    enum Regnetegn
    {
        Plus,
        Minus,
        Divider,
        Gange
    }
}
```

Så er den klar til at benytte i koden. Men husk at det nu er en variabel af typen "Regne tegn" som du skal overfører til din regne funktion!

Altså som herunder:

```
regnetegn = Regnetegn.Plus;
RegneFunktion(regnetegn);

// Eller det direkte kald:

RegneFunktion(Regnetegn.Plus);
```

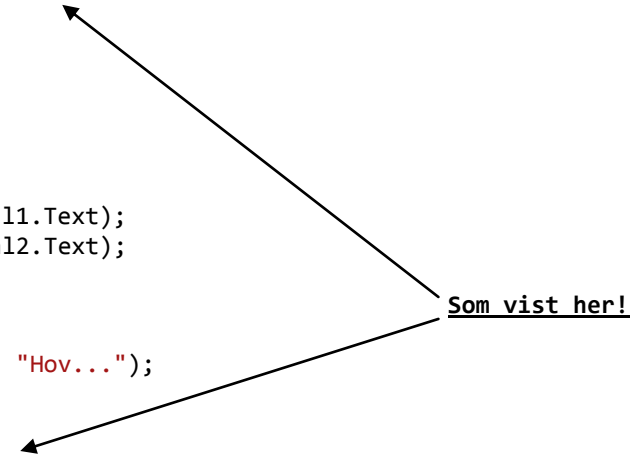
Og vores regne funktion skal så bygges om til at tage et andet input, som herunder:

```
private double RegneFunktion(Regnetegn regneTegn)
{
    double tal1 = 0;
    double tal2 = 0;
    double resultat = 0;

    try
    {
        tal1 = Convert.ToDouble(tbTal1.Text);
        tal2 = Convert.ToDouble(tbTal2.Text);
    }
    catch (Exception exc)
    {
        MessageBox.Show(exc.Message, "Hov...");
    }

    if (regneTegn == Regnetegn.Plus)
    {
        resultat = tal1 + tal2;
    }

    return resultat;
}
```



Som vist her!

Det var det. Nu har vi modificeret vores program, så det er let forståeligt for andre, samtidig med at vi har reduceret muligheden for "misbrug".

Afslutningen på opgaven bliver at erstatte samtlige if sætninger med en switch struktur!