TWEEDE DEELTENTAMEN IMPERATIEF PROGRAMMEREN
VRIJDAG 21 OKTOBER 2011, 11.00-13.00 UUR

- De lijst met standaardfuncties na afloop graag weer inleveren. De opgaven mag je houden (behalve als je heel vroeg vertrekt).
- Het tentamen bestaat uit 3 opgaven, die meetellen in de verhouding 30%:30%:40%. Als je een deel van een opgave niet weet, probeer dan toch zo veel mogelijk onderdelen op te schrijven!

---

1. Deze opgave bestaat uit een aantal tekstvragen.
   Houd het antwoord kort: een of twee zinnen per onderdeel kan al genoeg zijn.

   (a) Wat is een *null pointer*?
   En wanneer treedt er een *null pointer exception* op?

   (b) Bij het opbouwen van een userinterface roep je `Add` aan voor alle gemaakte knoppen, labels, tekstvelden enz. die op het window zichtbaar moeten worden. Hoe kan het dat `Add` parameters van verschillend type accepteert? Hoe is de parameter van `Add` blijkbaar gedeclareerd?

   (c) *Het is in deze opgave niet toegestaan om de methode* `ToLower` *uit de klasse* `String` *te gebruiken.*
   Schrijf een statische methode `HoofdKlein` die een hoofdletter in de range A–Z omzet in de overeenkomstige kleine letter a–z. Andere waarden van de parameter worden ongewijzigd teruggegeven.

   (d) Hoe kun je in een klasse een member maken die vanuit een andere klasse wel kan worden bekeken, maar niet worden veranderd?

   (e) Wat wordt er in het geheugen opgeslagen bij event-properties, zoals `Paint` in de klasse `Form` ?

2. In de klasse String zit een methode `Replace`. Deze methode levert een nieuwe string op, waarin elk voorkomen van het character dat als eerste parameter wordt meegegeven, is vervangen door het character dat als tweede parameter wordt meegegeven. Bijvoorbeeld:

```
"Utrecht".Replace('t','x')    geeft "Uxrechx"
"A+2+#@?".Replace('+','9')    geeft "A929#@?"
```

   Ook is er een methode `EndsWith`, die oplevert of een string eindigt met de string die als parameter wordt meegegeven. Bijvoorbeeld:

```
"Utrecht".EndsWith("recht")    geeft true
```

   Stel dat je de auteur van de klasse `String` bent. Veel andere members van die klasse zijn al geschreven (die mag je dus gebruiken), maar nog niet de `Substring` en `IndexOf` methoden (die mag je dus niet gebruiken).

   (a) Schrijf de methode `Replace`.

   (b) Schrijf de methode `EndsWith`.

   (c) Schrijf een statische methode met als parameter een array met strings, die de langste string van de array oplevert. Je mag aannemen dat de array minstens één string bevat. Als er meerdere strings zijn die even lang zijn mag je zelf kiezen welke je daarvan oplevert.

Zie vervolg op de achterkant

3. Bekijk het gegeven programma op pagina 3 (de klasse `Sterrenhemel` en pagina 4 (de klassen `Ster` en `Program`). Onderaan deze pagina staat een screenshot van het programma in werking.

   De gebruiker kan punten aanklikken in een window (maximaal 100 keer). Gecentreerd op die punten verschijnt een 9-puntige ster. Als de gebruiker meer dan 100 punten aanklikt gebeurt er niets (ook geen foutmelding). Als de gebruiker op de button 'Leeg' drukt, verdwijnen alle sterren en kan hij/zij aan 100 nieuwe sterren beginnen.

   De tweede button en de tekstbox spelen verderop in de opgave een rol.

   (a) In de klasse `Ster` worden de eigenschappen van een ster gemodelleerd: de positie van het midden en het aantal stralen (`tips`). De methode `LaatZien` laat zo'n ster zien op een graphics. Leg uit hoe de toekenningen aan `a` en `b` bijdragen in de vorming van de ster. (Je hoeft de wiskunde in de regels er boven niet uit te leggen).

   (b) Geef de drie ontbrekende declaraties in de klasse `Sterrenhemel`, compleet met hun initialisaties.

   (c) Schrijf de body van de methode `Teken` in de klasse `Sterrenhemel`.

   (d) Schrijf de body van de methode `MuisKlik` in de klasse `Sterrenhemel`.

   We willen nu dat de gebruiker behalve sterren ook zonnen kan tekenen. Een zon ziet er bijna hetzelfde uit als een ster, maar het verschil is dat de stralen van de ster nu een gekleurde cirkel als achtergrond krijgen.

   Bovendien kan de gebruiker het aantal stralen variëren, door een getal (alleen oneven getallen ≥ 3) in het tekstveld in te tikken.

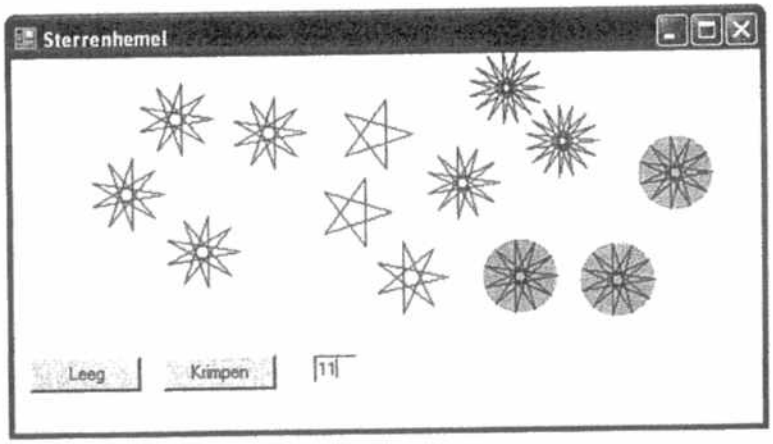
   (e) Schrijf een klasse `Zon` met daarin een constructormethode en een methode `LaatZien`. Er moeten zonnen van verschillende kleuren gemaakt kunnen worden.

   Vermijd hierbij overbodig schrijfwerk. Het is daarvoor echter wel nodig om nog een kleine wijziging aan te brengen in de klasse `Ster`. Welke?

   (f) Maak een nieuwe versie van de methode `MuisKlik` in de klasse `Sterrenhemel`. Als de gebruiker met de rechtermuisknop klikt, moet er een gele zon verschijnen in plaats van een ster.

   Bovendien moet nu ook het getal uit de tekstbox gebruikt worden voor het aantal stralen. Als er een oneven getal in de tekstbox staat op het moment dat er geklikt wordt, is dat het aantal stralen van de nieuwe ster. Als er niets, of een even getal of andere onzin (letters etc.) ingevuld zijn, is het aantal stralen standaard 9.

   (g) Als de gebruiker op de knop 'Krimpen' drukt, gaan alle sterren en zonnen langzaam naar de linker bovenhoek van het window bewegen: elke seconde vermindert de afstand met 10 procent. Schrijf de body van de methode `Beweeg` en de extra benodigde hulp-methode in `Sterrenhemel` die daarvoor nodig zijn. (De beweging hoeft niet meer gestopt te kunnen worden).



Links in het voorbeeld-plaatje vier standaard 9-puntige sterren. Daarnaast heeft de gebruiker ook 5-, 7-, 11- en 15-puntige sterren gemaakt, en drie 11-puntige zonnen.

```
using System;
using System.Drawing;
using System.Threading;
using System.Windows.Forms;

public class Sterrenhemel : Form
{
    // TODO: declaraties (opgave b)

    private static Button MaakKnop(string s, Point p, EventHandler eh)
    {   Button b = new Button();
        b.Text      = s;
        b.BackColor = Color.LightGray;
        b.Location  = p;
        b.Click    += eh;
        return b;
    }

    public Sterrenhemel()
    {   this.ClientSize = new Size(500, 250);
        this.Text = "Sterrenhemel";
        this.BackColor = Color.White;
        this.Paint      += this.Teken;
        this.MouseClick += this.MuisKlik;
        invoer.Location = new Point(200, 200);
        invoer.Size     = new Size(30, 20);
        this.Controls.Add(MaakKnop("Leeg",    new Point( 10, 200), this.Leeg  ));
        this.Controls.Add(MaakKnop("Krimpen", new Point(100, 200), this.Beweeg));
        this.Controls.Add(invoer);
    }

    public void Teken(object o, PaintEventArgs pea)
    {   // TODO (opgave c)
    }

    public void MuisKlik(object o, MouseEventArgs mea)
    {   // TODO (opgave d en f)
    }

    public void Leeg(object o, EventArgs ea)
    {   n = 0;
        this.Invalidate();
    }

    public void Beweeg(object o, EventArgs ea)
    {   // TODO (opgave g)
    }
}
```

Het voorbeeldprogramma gaat verder op de achterkant

```
public class Ster
{
    public Point plek;
    public int tips;
    public const int straal = 25;

    public Ster(Point p, int t)
    {   plek = p;
        tips = t;
    }

    public void Krimp()
    {   plek = new Point(plek.X * 9 / 10, plek.Y * 9 / 10);
    }

    public void LaatZien(Graphics g)
    {
        int a, b, t;
        a = 0;
        b = tips / 2;

        for (t = 0; t < tips; t++)
        {   int dx1 = (int)(Math.Cos(2*Math.PI*a/tips)*straal);
            int dy1 = (int)(Math.Sin(2*Math.PI*a/tips)*straal);
            int dx2 = (int)(Math.Cos(2*Math.PI*b/tips)*straal);
            int dy2 = (int)(Math.Sin(2*Math.PI*b/tips)*straal);

            g.DrawLine(Pens.Black, plek.X + dx1, plek.Y + dy1
                                 , plek.X + dx2, plek.Y + dy2 );
            a = b;
            b = (b + tips / 2) % tips;
        }
    }
}

class Program
{   static void Main()
    {   Application.Run(new Sterrenhemel());
    }
}

class Zon  // TODO (opgave e)
```

EINDE TENTAMEN