Uitwerkingen Tweede deeltentamen **Gameprogrammeren**
Vrijdag 15 oktober 2010, 11.00-13.00 uur

1. *(2 punten per deelvraag)* Deze opgave bestaat uit een aantal tekstvragen. Houd het antwoord kort: een of twee zinnen per onderdeel kan al genoeg zijn.

   (a) Wat is het verschil tussen `this` en `base`? Geef een voorbeeld van een situatie waarin het gebruik van `base` noodzakelijk is.

   Antwoord: `this` staat voor het object dat momenteel onder handen genomen wordt. `base` staat ook voor het object dat momenteel onder handen genomen wordt, met als belangrijk verschil dat `base` wijst naar het object alsof hij het type van de superklasse heeft. Als we een methode herdefiniëren en we willen in die methode de oorspronkelijke methode uit de superklasse aanroepen, dan moet dat met `base`. Ook moet `base` gebruikt worden als we de constructormethode van de superklasse willen aanroepen met parameters in de constructormethode van de subklasse.

   (b) Een `string` is in feite gewoon een array van waarden van het type `char`. Geef twee redenen waarom het toch handig is dat er een klasse `string` bestaat.

   Antwoord: String heeft een aantal extra methoden en properties die we kunnen gebruiken zoals `SubString` of `IndexOf`. Ook heeft string een dubbele quote-notatie om de string op te bouwen.

   (c) Wat betekent het dat een object *immutable* is? Geef een voorbeeld van een type object dat immutable is.

   Antwoord: Als een object immutable is, dan wordt de inhoud van het object nooit meer veranderd nadat het object is geconstrueerd. Een voorbeeld van een immutable object is een object van het type `string`.

   (d) Wat is het verschil tussen een klasse en een object?

   Antwoord: Een object is een groepje variabelen dat bij elkaar hoort, met methoden en properties. Een klasse is een beschrijving van de objecten met dezelfde opbouw en mogelijkheden.

   (e) Geef een voorbeeld van hoe een ééndimensionale array in games gebruikt kan worden. Geef ook een voorbeeld van hoe een multidimensionale array in games gebruikt kan worden.

   Antwoord: Een array kan gebruikt worden om de inventory van de speler bij te houden; om een lijst van alle spelers die het spel aan het spelen zijn bij te houden. Een multidimensionale array kan gebruikt worden om de tiles in een level te bewaren, of om een speelveld zoals bij schaken of dammen voor te stellen.

2. (a) *(5 punten)* Gegeven de volgende methodeheader:

   ```
   static bool IsWhiteSpace(string)
   ```

   Deze methode kijkt of een string bestaat uit alleen maar 'whitespace'. Onder 'whitespace' verstaan we spaties, tab-tekens en newline-tekens.

   Schrijf deze methode, *zonder* gebruik te maken van de bestaande `IsNullOrEmpty` en `IsNullOrWhiteSpace` methoden in de string-klasse.

   Antwoord:

   ```
   static bool IsWhiteSpace(string s)
   {
       for (int t=0; t<s.Length; t++)
           if (s[t]!=' ' && s[t]!='\t' && s[t]=='\n')
               return false;
       return true;
   }
   ```

   (b) *(5 punten)* In de klasse `String` zit onder andere de methode

   ```
   static int Compare(string, string)
   ```

   De methode `Compare` levert 0 op als de twee parameters precies gelijk zijn. Hij levert een negatief getal op (bijvoorbeeld -1, maar iets anders mag ook) als de eerste parameter kleiner is dan de tweede, en een positief getal (bijvoorbeeld 1) als die groter is. Met kleiner en groter wordt hier de woordenboek-ordening bedoeld: de eerste letter waar de strings verschillen bepaalt de ordening (volgens de Unicodes van die letters). Is de ene string een beginstuk van de andere, dan

is de kortste de kleinste. Spaties en leestekens tellen gewoon mee, die hoeven dus niet speciaal behandeld te worden.

Voorbeelden:

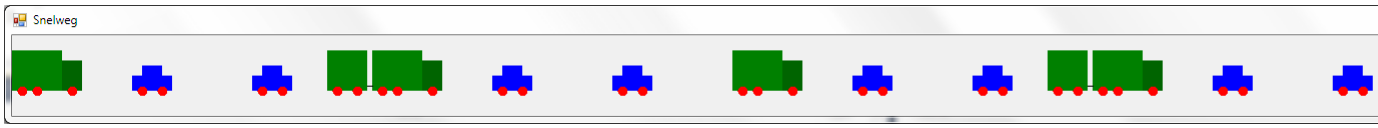
`String.Compare("aap", "noot")`	geeft een negatief getal, want `'a'<'n'`
`String.Compare("noot", "nieten")`	geeft een positief getal, want `'o'>'i'`
`String.Compare("niet", "nietmachine")`	geeft een negatief getal vanwege de lengte
`String.Compare("noot", "noot")`	geeft 0, want precies gelijk
`String.Compare("noot", "NOOT")`	geeft een positief getal, want `'n'>'N'`

Schrijf deze methode, *zonder* gebruik te maken van de bestaande `Compare` en `CompareTo` methoden.

Antwoord:

```
static int Compare(string s, string t)
{
    int m = s.Length;
    int n = t.Length;
    for (int i=0; i<Math.Min(m,n); i++)
    {
        char c = s[i];
        char d = t[i];
        if (c!=d) return c-d;
    }
    return m-n;
}
```

3. Bekijk het onderstaande programma. Het moet een file auto's op de snelweg tekenen, zoals in de screendump hieronder. Elke derde auto is een vrachtwagen, en elke tweede vrachtwagen is een combinatie met aanhanger.




```
public class Snelweg : Game
{   public SpriteBatch spriteBatch;
    public Texture2D wiel, auto, vrachtwagen, verbinding, aanhanger;
    public Snelweg()
    {   graphics = new GraphicsDeviceManager(this);
        graphics.PreferredBackBufferWidth = 1800;
        graphics.PreferredBackBufferHeight = 80;
        Content.RootDirectory = "Content";
        // TODO: ontbrekend deel van de constructor
    }
    protected override void LoadContent()
    {
        spriteBatch = new SpriteBatch(this.GraphicsDevice);
        wiel = Content.Load<Texture2D>("wiel");
        auto = Content.Load<Texture2D>("auto");
        vrachtwagen = Content.Load<Texture2D>("vrachtwagen");
        verbinding = Content.Load<Texture2D>("verbinding");
        aanhanger = Content.Load<Texture2D>("aanhanger");
    }
    protected override void Draw(GameTime gameTime)
    {   GraphicsDevice.Clear(Color.Grey);
        spriteBatch.Begin();
        for (int t = 0; t < rijbaan.Length; t++)
            rijbaan[t].Draw(this, t*120, 60);
        spriteBatch.End();
    }
    static void Main()
    {
        Snelweg spel = new Snelweg();
        spel.Run();
```

```
    }
}
class MotorVoertuig
{
    public void Draw(Game g, int x, int y)
    {
    }
}
class PersonenAuto : MotorVoertuig
{   public void Draw(Game g, int x, int y)
    {   g.spriteBatch.Draw(g.auto, new Vector2(x, y - 30), Color.Blue);
        g.spriteBatch.Draw(g.wiel, new Vector2(x + 5, y - 10), Color.Red);
        g.spriteBatch.Draw(g.wiel, new Vector2(x + 25, y - 10), Color.Red);
    }
}
class Vrachtwagen : MotorVoertuig
{   public void Draw(Game g, int x, int y)
    {   g.spriteBatch.Draw(g.vrachtwagen, new Vector2(x, y - 45), Color.Green);
        g.spriteBatch.Draw(g.wiel, new Vector2(x + 5, y - 10), Color.Red);
        g.spriteBatch.Draw(g.wiel, new Vector2(x + 20, y - 10), Color.Red);
        g.spriteBatch.Draw(g.wiel, new Vector2(x + 55, y - 10), Color.Red);
    }
}

class Combinatie : Vrachtwagen
{   public void Draw(Game g, int x, int y)
    {   // de vrachtwagen
        g.spriteBatch.Draw(g.vrachtwagen, new Vector2(x, y - 45), Color.Green);
        g.spriteBatch.Draw(g.wiel, new Vector2(x + 5, y - 10), Color.Red);
        g.spriteBatch.Draw(g.wiel, new Vector2(x + 20, y - 10), Color.Red);
        g.spriteBatch.Draw(g.wiel, new Vector2(x + 55, y - 10), Color.Red);
        // de aanhanger
        g.spriteBatch.Draw(g.verbinding, new Vector2(x - 5, y - 10), Color.Black);
        g.spriteBatch.Draw(g.aanhanger, new Vector2(x - 45, y - 45), Color.Green);
        g.spriteBatch.Draw(g.wiel, new Vector2(x - 40, y - 10), Color.Red);
        g.spriteBatch.Draw(g.wiel, new Vector2(x - 20, y - 10), Color.Red);
    }
}
```

(a) *(1 punt)* Er ontbreekt nog een declaratie. Schrijf deze declaratie, en geef aan waar die moet staan.
Antwoord: Boven in de klasse `Snelweg` komt de declaratie

```
MotorVoertuig[] rijbaan = new MotorVoertuig[25];
```

(b) *(2 punten)* Schrijf het ontbrekende deel van de constructor van `Snelweg`.
Antwoord:

```
for (int t = 0; t < rijbaan.Length; t++)
{   if (t % 3 != 0)
        rijbaan[t] = new PersonenAuto();
    else if (t % 6 == 0)
        rijbaan[t] = new Vrachtwagen();
    else rijbaan[t] = new Combinatie();
}
```

(c) *(2 punten)* In het gegeven programma zit nog een fout, waardoor er helemaal niets zichtbaar wordt. Hoe komt dat, en hoe kan de fout worden verbeterd?

Antwoord: Het element-type van de array is `MotorVoertuig`, dus wordt in de `Draw`-actie van `Snelweg` steeds de lege methode `Draw` van `MotorVoertuig` aangeroepen. Deze methode had `virtual` gedeclareerd moeten worden, en in de subklasse als `override`.

(d) *(1 punt)* De programmeur heeft een flink stuk code met copy&paste gedupliceerd. Waarom is dat geen goed idee?

Antwoord: Er ontstaat nu een versie-management probleem: als de vorm van de vrachtauto later verandert, moet dat op twee plaatsen worden aangepast.

(e) *(2 punten)* Hoe had het dupliceren van de code het beste vermeden kunnen worden?

Antwoord: In de klasse `Combinatie` had de trekker getekend kunnen worden met de aanroep `base.Draw(g,x,y)`.

(f) *(2 punten)* We willen de object-georiënteerde opzet van het programma nog verder doorvoeren, zo dat ook de concepten 'wiel' en 'aanhanger' met klassen worden gemodelleerd. Hoe kan dat netjes worden aangepakt? Zorg ervoor dat code-duplicatie zo veel mogelijk vermeden kan worden, en dat er nooit door een programmeerfout een losse aanhanger op de weg terecht kan komen.

Je hoeft dit niet helemaal uit te programmeren; geef alleen aan welke klassen er komen, hoe hun subklasse-relatie is, en welke declaraties er in (bestaande en/of nieuwe) klassen komen te staan.

Antwoord: Nieuwe klasse `Wiel`, los van alle andere klassen. Nieuwe klasse `Wagen` als superklasse van `MotorVoertuig`. Daarin een array met `Wiel`-objecten. Nieuwe klasse `Aanhanger` als subklasse van `Wagen`. In de klasse `Combinatie` een declaratie van een `Aanhanger`.

4. Beschouw de volgende klasse:

```
class GameObject
{
    protected Texture2D sprite;
    protected Vector2 positie, snelheid;

    public GameObject(Texture2D geladenSprite)
    {
        this.sprite = geladenSprite;
        this.positie = Vector2.Zero;
        this.snelheid = Vector2.Zero;
    }

    public virtual void Update()
    {
        this.positie += this.snelheid;
    }

    public virtual void Draw(SpriteBatch spriteBatch)
    {
        spriteBatch.Draw(this.sprite, this.positie, Color.White);
    }
}
```

In de volgende deelvragen gaan we een aantal methoden en properties toevoegen aan deze klasse. Schrijf alleen de opdrachten op die toegevoegd moeten worden, je hoeft niet steeds de hele klasse over te schrijven.

(a) *(3 punten)* Breid de `GameObject`-klasse uit met een property `Zichtbaar`. Deze property moeten we kunnen lezen en schrijven met een waarheidswaarde, en geeft aan of het game object getekend moet worden. Voeg membervariabelen toe indien nodig. Herschrijf ook de Draw-actie zodat rekening gehouden wordt met deze property.

Antwoord: We voegen een membervariabele `zichtbaar` toe:

```
protected bool zichtbaar = true;
```

De property is als volgt gegeven:

```
public bool Zichtbaar
{
    get { return this.zichtbaar; }
    set { this.zichtbaar = value; }
}
```

In de Draw-actie moeten we nu de waarde van de variabele testen:

```
public virtual void Draw(SpriteBatch spriteBatch)
{
    if (this.zichtbaar)
        spriteBatch.Draw(this.sprite, this.positie, Color.White);
}
```

(b) *(3 punten)* Schrijf een klasse `Diamant` die een subklasse is van de `GameObject`-klasse. Een speler kan met diamanten punten verdienen. Voeg een property `Punten` toe aan de `Diamant`-klasse waarmee de waarde van de diamant opgevraagd en veranderd kan worden. Let op dat de waarde van de diamant nooit kleiner dan 0 mag zijn. Voeg membervariabelen toe indien nodig. Standaard is een diamant 10 punten waard.

Antwoord:

```
class Diamant : GameObject
{
    protected int punten;

    public Diamant(Texture2D geladenSprite) : base(geladenSprite)
    {
        this.punten = 10;
    }

    public int Punten
    {
        get { return this.punten; }
        set { if (value >= 0) this.punten = value; }
    }
}
```

(c) *(2 punten)* In de `Diamant`-klasse willen we dat de Draw-actie ander gedrag vertoont. Namelijk als een diamant 50 punten of meer waard is, dan moet de sprite niet met een witte kleur (`Color.White`) getekend worden, maar met een rode kleur (`Color.Red`). Breid de `Diamant`-klasse uit zodat dit bewerkstelligd wordt.

Antwoord: We moeten de Draw-methode herdefiniëren, als volgt:

```
public override void Draw(SpriteBatch spriteBatch)
{
    Color kleur = Color.White;
    if (this.punten >= 50)
        kleur = Color.Red;
    if (this.zichtbaar)
        spriteBatch.Draw(this.sprite, this.positie, kleur);
}
```

(d) *(2 punten)* Zou je de membervariabelen van de `GameObject`-klasse ook als *private* in plaats van protected kunnen declareren? Zo nee: waarom niet? Zo ja: is dat een goed idee?

Antwoord: Ja dat zou kunnen. Het is echter geen goed idee, want dan kunnen eventuele subklassen van `GameObject`, zoals `Diamant` niet meer bij de membervariabelen komen.