Eerste deeltentamen Gameprogrammeren
Vrijdag 27 september 2013, 8.30-10.30 uur

Naam:	
Studentnummer:	

- Het tentamen bestaat uit 4 opgaven. Elke opgave levert 10 punten op. Je cijfer is het totaal aantal punten gedeeld door 4. Als je een deel van een opgave niet weet, probeer dan toch zo veel mogelijk op te schrijven!
- Het is niet toegestaan om boeken, aantekeningen of ander materiaal te gebruiken, met uitzondering van de lijst met standaardklassen, -methoden, en -properties. Deze lijst na afloop graag weer inleveren.

Veel succes!

---

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

   (a) *(2 punten)* Wat wordt er verstaan onder de *syntax* van een (programmeer)taal-constructie? En wat is de *semantiek* van een taal-constructie?

syntax:

semantiek:

   (b) *(3 punten)* Kruis aan welke van de volgende stellingen waar zijn:

   - ☐ Elk C# programma moet één klasse hebben met een statische methode die de naam Main draagt.
   - ☐ Een klasse is een groepje variabelen.
   - ☐ Als het woord **new** gebruikt wordt om een object te creëren, dan weten we dat het object als type een klasse heeft.
   - ☐ Het aanroepen van een methode met als resultaatwaarde **void** is een expressie.
   - ☐ Het Halting probleem is onoplosbaar.
   - ☐ Scripttalen zijn nooit declaratief.

(c) *(2 punten)* De vertaling van een C#-programma naar machinecode gebeurt in twee stappen. Eerst wordt het programma vertaald naar een intermediate language, en die wordt vervolgens weer vertaald naar machinecode. Noem twee voordelen van het vertalen in twee stappen.

1:

2:

(d) *(3 punten)* Hieronder staan zes programma-fragmenten. Je mag er vanuit gaan dat alle variabelen die gebruikt worden vantevoren gedeclareerd en van het type **int** zijn. Geef in elk van de gevallen aan *hoe vaak* de methode iets wordt aangeroepen. Licht het antwoord kort toe.

1	**for** (x=0; x<5; x++)
	  this.iets();
	for (y=0; y>5; y++)
	  this.iets();	

2	**for** (x=0; x<5; x++) **for** (y=x; y>0; y−−) **this**.iets();	

3	**for** (x=0; x<5; x++)
	  while (x % 3 == 0)
	  {
	  this.iets();
	  x++;
	  }	

4	**while** (**false**)
	  this.iets();
	if (**true**)
	  this.iets();	

5	**for** (x=0; x<x; x++) **this**.iets();	

6	**for** (x=20; x>=0; x/=3)
	  for (y=1; y<=x; y++)
	  this.iets();	

2. *($\frac{1}{2}$ punt per goed antwoord)* Hieronder staat 20 fragmenten uit een programma. Schrijf in de tabel hieronder achter elk programmafragment één daarbij passende letter, als volgt:

   - **T** als het programmafragment een **type** is
   - **E** als het programmafragment een **expressie** (maar geen constante) is
   - **O** als het programmafragment een **opdracht** is
   - **D** als het programmafragment een **declaratie** is
   - **C** als het programmafragment een **constante** (en dus ook een expressie) is
   - **X** als het programmafragment geen van bovenstaande dingen is

int i, j;		position.Zero;		Color.White		**true**=**false**	
(**float**)		'\\'		Game		**for**(;**false**;);	
Vector2 position;		**new** Random()		1 + 2 == 3		i = Color.Black.R;	
new Vector2(1, 2) * 3		i<5!=j>5		SpriteBatch		1E1	
while(**true**) **while**(**false**);		/*hallo		(**bool**)**true**		i = j++;	

Opgave 3 en 4 vragen een stukje programma. Kleine schrijffoutjes (hoofdletters, puntkomma's enz.) worden niet streng afgerekend, maar de elementen die de structuur van het programma bepalen (haakjes, accolades, aanhalingstekens enz.) zijn wel belangrijk. Schrijf die dus duidelijk en op de goede plaats op! Het is toegestaan (maar niet nodig) om C#-constructies die (nog) niet zijn behandeld toch te gebruiken. Je hoeft niet aan te geven welke **using***-opdrachten nodig zijn om de klassen te kunnen gebruiken.*

3. (a) *(3 punten)* Schrijf een methode IsPrimeNumber die, gegeven een geheel getal, aangeeft of het een priemgetal is. Een getal is een priemgetal als dat getal alleen maar deelbaar is door zichzelf en door één. Je mag er vanuit gaan dat altijd een positief getal meegegeven wordt aan de methode.

(b) *(5 punten)* Elk positief geheel getal kan uitgedrukt worden als een vermenigvuldiging van priemfactoren. Bijvoorbeeld:

$$102 = 2 \times 3 \times 17$$

$$712 = 2 \times 2 \times 2 \times 89$$

De priemfactoren kun je vinden door te beginnen met het delen door twee. Als dat lukt, dan is 2 de eerste priemfactor. Als het niet lukt, hoog de deler op en kijk of het daarmee kan, enzovoorts:

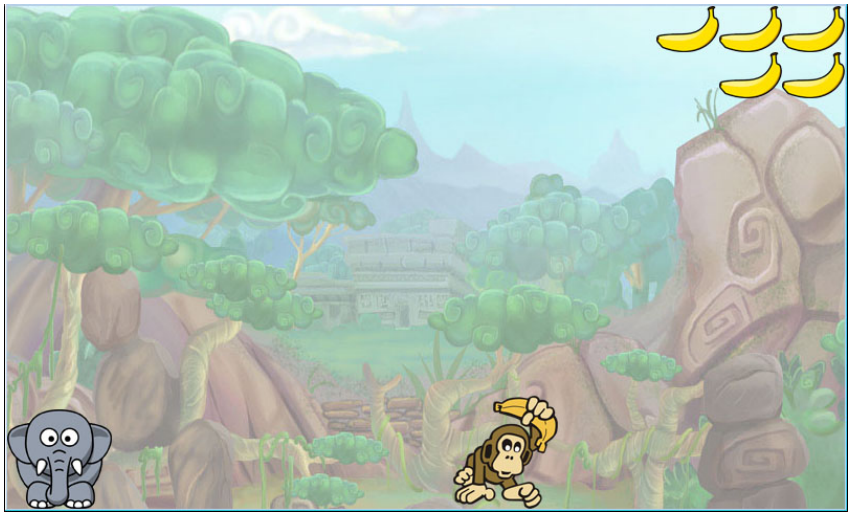
$$712/2 = 356 \qquad 356/2 = 178 \qquad 178/2 = 89 \qquad 89/89 = 1$$

Het getal 102 heeft 3 priemfactoren (2, 3, en 17), en het getal 712 heeft er vier (2, 2, 2, en 89). Schrijf een methode CountPrimeFactors die het aantal priemfactoren uitrekent van een geheel getal.

(c) *(2 punten)* Schrijf een stukje programma dat, door gebruik te maken van de CountPrimeFactors methode, uitrekent wat het kleinste positieve gehele getal is dat vijf priemfactoren heeft. Dit getal moet vervolgens op het scherm afgedrukt worden.

4. De maat is vol! De olifanten hebben er genoeg van dat de apen steeds hun bananen stelen en opeten. Ze hebben geprobeerd er met de apen over te praten, maar het enige antwoord dat ze kregen was "oeh oeh aah aah". De enige oplossing die overblijft is bovenop die vervelende apen te springen, dat zal ze leren!

   En zo begint het spel "Annoyed Elephants". Het doel van dit spel is om op zoveel mogelijk apen te springen. Dit doe je door in het scherm te klikken met je muis, waardoor de olifant springt. De plek waar je klikt bepaalt de snelheid en richting van de sprong. Elke keer als je op een aap springt, krijg je een banaan. De aap wordt dan op een andere willekeurige plek onderaan het scherm geplaatst en de olifant staat weer links onderin. Het aantal bananen dat je tot nu toe hebt verzameld wordt rechtsbovenin het scherm afgebeeld. In de appendix van deze toets staan de belangrijkste klassen die gebruikt worden in deze game. En dit is een screenshot van het spel:



   (a) *(2 punten)* Wat is de reden dat sommige membervariabelen in de AnnoyedElephants klasse **static** zijn?

Het aantal bananen dat je als speler verzameld hebt staat in de membervariabele points in de GameWorld klasse. Zoals je kunt zien in de screenshot worden er maximaal drie bananen naast elkaar getekend. Als je dus als speler bijvoorbeeld tot nu toe vijf bananen verzameld hebt, dan worden er twee rijen bananen getekend: één rij met drie bananen, en daaronder een rij met twee bananen.

(b) *(4 punten)* Het tekenen van de bananen gebeurt in de methode DrawBananas uit de GameWorld klasse. Werk deze methode uit (header en body). Let erop dat de bananen altijd netjes helemaal rechts in het scherm getekend worden, ook als de speler minder dan drie bananen verzameld heeft.

(c) *(2 punten)* Er mist nog een property Points in de GameWorld klasse. Werk deze property uit. Zorg ervoor dat het aantal verkregen punten nooit kleiner kan zijn dan nul.

(d) *(2 punten)* Als een olifant botst met een aap, dan wordt de aap op een willekeurige plek onderaan het scherm geplaatst. Echter, de x-positie van de aap mag nooit kleiner zijn dan 100, omdat de aap dan op de positie van de olifant zou kunnen komen. Het plaatsen van de aap gebeurt in de Reset methode. Werk de body van deze methode uit.

Appendix: klassen

AnnoyedElephants

```
class AnnoyedElephants : Game
{
    GraphicsDeviceManager graphics;
    SpriteBatch spriteBatch;
    InputHelper inputHelper;
    static Random random;
    static GameWorld gameWorld;
    static Vector2 screen;

    static void Main() {
        AnnoyedElephants game = new AnnoyedElephants();
        game.Run();
    }

    public AnnoyedElephants() {
        Content.RootDirectory = "Content";
        IsMouseVisible = true;
        graphics = new GraphicsDeviceManager(this);
        random = new Random();
        inputHelper = new InputHelper();
    }

    protected override void LoadContent() {
        spriteBatch = new SpriteBatch(GraphicsDevice);
        screen = new Vector2(GraphicsDevice.Viewport.Width, GraphicsDevice.Viewport.Height);
        gameWorld = new GameWorld(Content);
    }

    protected override void Update(GameTime gameTime) {
        inputHelper.Update();
        gameWorld.HandleInput(inputHelper);
        gameWorld.Update(gameTime);
    }

    protected override void Draw(GameTime gameTime) {
        GraphicsDevice.Clear(Color.White);
        spriteBatch.Begin();
        gameWorld.Draw(gameTime, spriteBatch);
        spriteBatch.End();
    }

    public static Vector2 Screen {
        get { return screen; }
    }

    public static Random Random {
        get { return random; }
    }

    public static GameWorld GameWorld {
        get { return gameWorld; }
    }
}
```

GameWorld

```
class GameWorld
{
    Texture2D background, banana;
    Elephant elephant;
    Monkey monkey;
    int points;

    public GameWorld(ContentManager Content)
    {
        background = Content.Load<Texture2D>("background");
        banana = Content.Load<Texture2D>("banana");
        elephant = new Elephant(Content);
        monkey = new Monkey(Content);
        points = 0;
    }

    public void HandleInput(InputHelper inputHelper)
    {
        elephant.HandleInput(inputHelper);
    }

    public void Update(GameTime gameTime)
    {
        elephant.Update(gameTime);
        monkey.Update(gameTime);
    }

    public void Draw(GameTime gameTime, SpriteBatch spriteBatch)
    {
        spriteBatch.Draw(background, Vector2.Zero, Color.White);
        DrawBananas(spriteBatch); // TO DO
        monkey.Draw(gameTime, spriteBatch);
        elephant.Draw(gameTime, spriteBatch);
    }

    public bool IsOutsideWorld(Vector2 position)
    {
        return position.X < 0 || position.X > AnnoyedElephants.Screen.X || position.Y > AnnoyedElephants.Screen.Y;
    }

    public Elephant Elephant
    {
        get { return elephant; }
    }

    public Monkey Monkey
    {
        get { return monkey; }
    }
}
```

Elephant

```
class Elephant
{
    Texture2D sprite;
    Vector2 position, velocity;

    public Elephant(ContentManager Content)
    {
        sprite = Content.Load<Texture2D>("elephant");
        Reset();
    }

    public void HandleInput(InputHelper inputHelper)
    {
        if (inputHelper.MouseLeftButtonPressed() && velocity == Vector2.Zero)
            velocity = (inputHelper.MousePosition - position) / 40;
    }

    public void Update(GameTime gameTime)
    {
        if (velocity != Vector2.Zero)
        {
            velocity.X *= 0.99f;
            velocity.Y += 0.1f;
        }
        position += velocity;
        if (AnnoyedElephants.GameWorld.IsOutsideWorld(position))
            Reset();
    }

    public void Reset()
    {
        position = new Vector2(0, AnnoyedElephants.Screen.Y - sprite.Height);
        velocity = Vector2.Zero;
    }

    public Vector2 CenterPosition
    {
        get { return position + new Vector2(sprite.Width, sprite.Height) / 2; }
    }

    public void Draw(GameTime gameTime, SpriteBatch spriteBatch)
    {
        spriteBatch.Draw(sprite, position, Color.White);
    }
}
```

Monkey

```
class Monkey
{
    Texture2D sprite;
    Vector2 position, velocity;

    public Monkey(ContentManager Content)
    {
        sprite = Content.Load<Texture2D>("monkey");
        Reset();
    }

    public void Update(GameTime gameTime)
    {
        Vector2 distanceVector = AnnoyedElephants.GameWorld.Elephant.CenterPosition - CenterPosition;
        if (distanceVector.Length() < 100)
        {
            AnnoyedElephants.GameWorld.Points++;
            AnnoyedElephants.GameWorld.Elephant.Reset();
            Reset();
        }
    }

    public void Reset()
    {
        // TODO
    }

    public Vector2 CenterPosition
    {
        get { return position + new Vector2(sprite.Width, sprite.Height) / 2; }
    }

    public void Draw(GameTime gameTime, SpriteBatch spriteBatch)
    {
        spriteBatch.Draw(sprite, position, Color.White);
    }
}
```

EINDE TOETS