Uitwerkingen tweede deeltentamen Gameprogrammeren
Vrijdag 18 oktober 2013, 11.00-13.00 uur

---

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

   (a) *(2 punten)* Gegeven de volgende methode:

```
// Neem aan dat x groter dan of gelijk aan y
int f(int x, int y)
{
    if (y == 0 || y == x)
        return 1;
    return f(x−1,y) + f(x−1, y−1);
}
```

   Wat is het resultaat van de aanroep f(f(4,2), 1)?
   Antwoord: 6

   (b) *(2 punten)* Stel dat de interface Demo is gedeclareerd, en een variabele test met die interface als type:

```
interface Demo
{
    public int Aantal();
}
Demo test;
```

   Het is niet mogelijk om deze variabele te initialiseren met

```
test = new Demo();
```

   i. Hoe kan deze variabele toch zinvol worden geïnitialiseerd (anders dan met **null**)?
   Antwoord: De variabele kan wijzen naar een object dat als type een klasse heeft die deze interface implementeert.

   ii. Waarom kan het handig zijn om test te declareren met Demo als type, in plaats van met het type van de expressie waarmee hij wordt geïnitialiseerd?
   Antwoord: Op die manier kan test gebruikt worden in methoden/properties die niet hoeven te weten welk type het object heeft waar test naar wijst. De interface definieert dan hoe het object gebruikt mag worden.

   (c) *(4 punten)* Beschouw de volgende instructie:

```
A obj = new A();
```

   Deze instructie bestaat uit een aantal onderdelen. Schrijf hieronder achter elk onderdeel wat het onderdeel precies doet:
   Antwoord:

   - declaratie: geeft het type en de naam van een variabele aan
   - **new** operator: reserveert het geheugen dat het object nodig heeft
   - constructor: kent waarden toe aan het geheugen gereserveerd door de **new** operator
   - toekenning: kent een verwijzing naar het geheugenadres van het object toe aan de variabele

(d) *(2 punten)* Beschouw de volgende instructie:

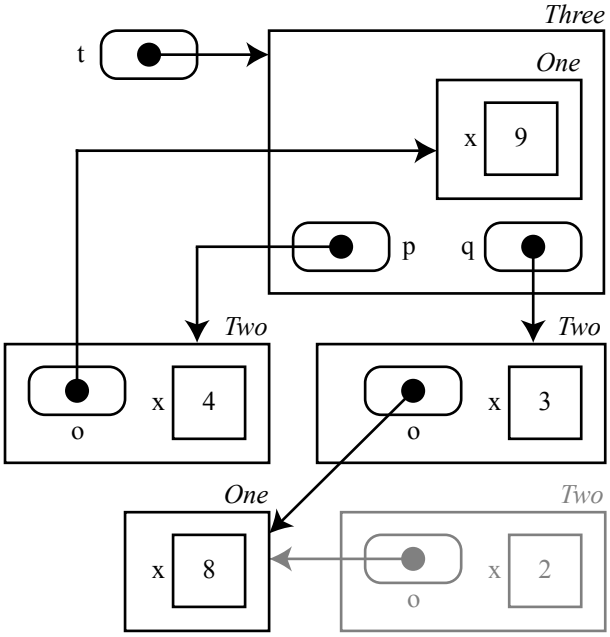
```
IList<GameObject> gameObjects = new List<GameObject>();
```

Kruis aan welke van de volgende stellingen waar zijn:

- ☐ Het aantal elementen in de lijst gameObjects mag niet gewijzigd worden, want het geheugen is reeds gereserveerd.
- X gameObjects mag objecten van het type GameObject bevatten, maar ook objecten die een subklasse van GameObject als type hebben.
- X gameObjects mag objecten van elk type bevatten waarvoor geldt dat GameObject de klasse is die aan de top van de overervingshiërarchie van dat type staat.
- ☐ Deze instructie compileert niet, want gameObjects is van het type IList<GameObject> terwijl we een object toekennen van het type List<GameObject>.

2. *(6 punten)*

Antwoord:



Opgave 3 vraagt 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.

3. (a) *(1 punt)* In de GameWorld klasse mist nog een declaratie. Welke declaratie is dit?
Antwoord: de volgende instructie mist nog (declaratie van grid):

```
MemoryCard[,] grid;
```

(b) *(3 punten)* In de HandleInput methode van de MemoryCard klasse moet gekeken worden of de speler op de memory kaart geklikt heeft. Is dit het geval, dan moet de toestand van de kaart veranderd worden naar 'open' (door middel van de variabele Open). Werk deze methode uit (zie volgende pagina voor het antwoordvak).
Antwoord:

```
if (!Visible)
    return;
Rectangle boundingBox = new Rectangle((int)Position.X, (int)Position.Y, cardFront.Width, cardFront.Height);
if (inputHelper.MouseLeftButtonPressed() && boundingBox.Contains((int)inputHelper.MousePosition.X,
                                                                 (int)inputHelper.MousePosition.Y))
    Open = true;
```

(c) *(3 punten)* **Antwoord:**

```
public void AddToGrid(MemoryCard m, int x, int y)
{
    Vector2 gridPos = (Memory.Screen − new Vector2(cols ∗ cellsize, rows ∗ cellsize)) / 2;
    m.Position = gridPos + new Vector2(x ∗ cellsize, y ∗ cellsize);
    grid[x, y] = m;
}
```

(d) *(8 punten)* In de constructor van de GameWorld klasse moet het grid gevuld worden met willekeurige memory kaarten. Schrijf de instructies die dit bewerkstelligen. Let op: van elke soort kaart moet er een even aantal kaarten in het grid zitten, want anders kun je niet alle kaarten bij elkaar zoeken. Daarnaast moeten de kaarten in een willekeurige volgorde in het grid zitten. Je mag er vanuit gaan dat de grid bestaat uit een even aantal elementen. *Hint: vul eerst het grid met de juiste kaarten, en 'schud' het grid daarna zodat de kaarten in willekeurige volgorde liggen.*
Antwoord:

```
int sheetIndex = 0;
for (int curr = 0; curr < rows ∗ cols; curr++)
{
    int col = curr % cols;
    int row = curr / cols;
    if (curr % 2 == 0)
        sheetIndex = Memory.Random.Next(8);
    AddToGrid(new MemoryCard(Content, sheetIndex), col, row);
}
Shuffle();
```

De methode Shuffle is als volgt gegeven:

```
public void Shuffle()
{
    for (int x = 0; x < cols; x++)
        for (int y = 0; y < rows; y++)
        {
            MemoryCard tmp = grid[x, y];
            int randomX = Memory.Random.Next(cols);
            int randomY = Memory.Random.Next(rows);
            AddToGrid(grid[randomX, randomY], x, y);
            AddToGrid(tmp, randomX, randomY);
        }
}
```

(e) *(5 punten)* In de Update methode van GameWorld moeten we, indien er twee kaarten open liggen kijken of de kaarten hetzelfde plaatje voorstellen. Is dit het geval, dan worden de kaarten onzichtbaar. Als de kaarten niet hetzelfde plaatje voorstellen, dan moeten ze weer omgedraaid worden. Schrijf de instructies op die dit bewerkstelligen.

Antwoord:

```
IList<MemoryCard> openCards = new List<MemoryCard>();
foreach (MemoryCard m in grid)
    if (m.Open)
        openCards.Add(m);
if (openCards.Count != 2)
    return;
openCards[0].Open = false;
openCards[1].Open = false;
if (openCards[0].SheetIndex == openCards[1].SheetIndex)
{
    openCards[0].Visible = false;
    openCards[1].Visible = false;
}
```

(f) *(2 punten)* De SpriteSheet klasse heeft een lees-schrijf property SheetIndex. Werk deze property uit. Zorg dat de sheet index nooit buiten het bereik van de sprite kan vallen.

Antwoord:

```
public int SheetIndex
{
    get
    {
        return this.sheetIndex;
    }
    set
    {
        if (value < sheetColumns * sheetRows && value >= 0)
            this.sheetIndex = value;
    }
}
```

(g) *(2 punten)* In de huidige implementatie van deze game gaat nog iets mis. De speler ziet namelijk nooit allebei de kaarten waarop hij/zij heeft geklikt. Waarom niet? Leg kort uit hoe je dit zou kunnen oplossen, je hoeft het niet daadwerkelijk uit te programmeren.

Antwoord: Als de speler op de tweede kaart klikt, dan worden de kaarten gelijk of omgedraaid, of onzichtbaar gemaakt. De speler ziet dus nooit de tweede kaart waarop hij/zij geklikt heeft. Een manier om dit op te lossen is door een timer te started zodra de speler twee kaarten opengelegd heeft, en pas de kaarten omdraaien of onzichtbaar maken als de timer afgelopen is.