Informatica: C# WPO 5

1. Inhoud

While-loop, do while, debuggen, graphics

2. Oefeningen

- Demo 1: Power of 2
- Demo 2: Tel totdat...
- Demo 3: Debug oplossing demo's 1 en 2
- A: Count down
- A: Random counting
- A: Faculteit
- A: Fibonacci
- A: Ad Fundum
- A: Notenbalk
- E: Kogelwerpen
- X: Teken sinus

2.1 Demo 1: Power of 2

In deze opgave worden de machten van 2 afgelopen. Om dit te bekomen wordt bij elke iteratie een getal vermenigvuldigt met 2. Voor deze iteratie uit totdat de opgegeven grens (door de gebruiker in te geven) bereikt is. Gebruik hiervoor een while-loop.

2.2 Demo 2: Tel totdat...

Schrijf een programma dat de gebruiker om 2 getallen vraagt. Tel vanaf het eerste getal af totdat het 2^{de} getal bereikt is. Doe dit eerst a.d.h.v. een for-loop. Bouw daarna de loop om tot een while-loop.

2.3 Demo 3: Debug oplossingen demo's 1 en 2

Ook hier wordt de debugger gebruikt om het verloop van de programma's in de voorgaande demo's te illustreren.

2.4 A: Count down

Print alle getallen van X t.e.m. 0 af. Doe dit eerst a.d.h.v. for-loops. Bouw daarna de loops om tot while-loops.

1. Print alle getallen af.
2. Print 1 getal op 3 af.
3. Print alleen de even getallen.
4. Print alleen de oneven getallen.

2.5 A: Random counting

Binnen een while-loop wordt telkens opnieuw een randomgetal gegenereerd. Dit randomgetal is begrensd tussen 0 en 100. De loop wordt gestopt wanneer het randomgetal groter is 97. Hoeveel randomgetallen zijn er geproduceerd voordat deze grens overschreden is? Gebruik onderstaand stukje code (do-while-loop) als basis voor je programma.

Codefragment 1: Randomgetallen genereren

```
// lokaal binnen de functie van een button
Random rnd = new Random();
int randomwaarde;
do
{
   randomwaarde = rnd.Next(0,100); // randomgetal genereren tussen 0 en 99
}
while (........);
```

Kan je je code omvormen tot een gewone while-loop?

2.6 A: Faculteit

De faculteit wordt veel toegepast binnen de kansrekening. Formule 1 geeft het principe hiervan weer.

$$n! = \prod_{i=1}^{i=n} i = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 6 \cdot ... \cdot n \tag{1}$$

Schrijf een programma dat de faculteit uitrekent totdat een bepaalde grens (door de gebruiker in te geven) overschreden wordt.

2.7 A: Fibonacci

De reeks van Fibonacci werd rond 1200 door Leonardo van Pisa voor het eerst beschreven. Deze rij wordt wiskundig als volgt beschreven:

$$f_n = f_{n-1} + f_{n-2} \tag{2}$$

met $n > 1$.
Deze rij start met de waarden 0 en 1. Het resultaat van deze rij wordt hieronder wergegeven:

$0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...$

Schrijf een programma dat de reeks uitrekent zolang het resultaat kleiner is dan een door de gebruiker aangegeven grens. De getallen worden in een aparte label of textbox geprint.

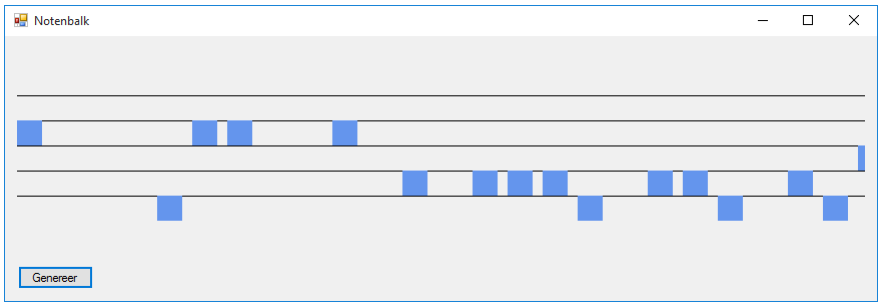
2.8 A: Ad Fundum

Op een PK-cantus nemen een zeer groot aantal leden deel. Als uitdaging gaan alle leden het volgende aan. De cantor neemt het voortouw door zichzelf een liter bier uit te schenken. Door een bierschaarste moeten de schachten het doen met minder bier. Als regel wordt aangenomen dat een schacht de halve hoeveelheid krijgt van zijn voorganger. Op die manier heeft de cantor 1 liter bier, schacht 1 een halve liter, schacht 2 een vierde van een liter, enz.

- Hoeveel liter bier worden er voor deze uitdaging uitgeschonken? Beredeneer dit getal!
- Ga dit na door dit te programmeren met een while-loop. Vanaf dat de uitgeschonken hoeveelheid bier kleiner is dan 1 cl wordt de loop gestopt. Hoeveel schachten hebben bier gekregen?

2.9 A: Notenbalk

Teken een notenbalk op een canvas. Elke noot wordt voorgesteld a.d.h.v. een klein vierkantje van 25 bij 25 pixels. De kleur van de noten is vrij te kiezen. De notenbalk bestaat uit 5 horizontale balken (horizontale strepen). Om de notenbalk in te vullen, kan gebruik gemaakt worden van een randomgenerator (van 0 t.e.m. 7). Indien het bekomen getal tussen 0 en 4 ligt, wordt een noot getekend op de passende balk. Getallen 5 en 6 leveren geen noot op. Ga door met het genereren en tekenen van de noten zolang de breedte van de canvas niet bereikt is.



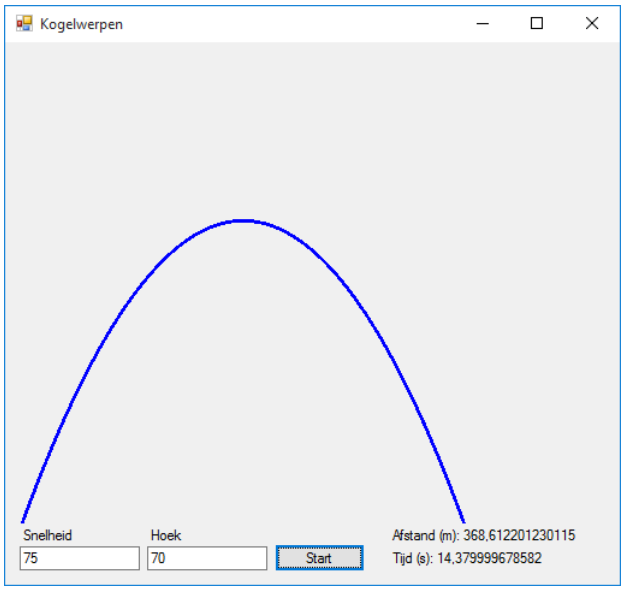
Figuur 1: Voorbeeld notenbalk. In deze afbeelding is telkens 10 pixels ruimte gelaten tussen de noten om het concept te verduidelijken.

2.10 E: Kogelwerpen

In een partijtje kogelwerpen worden massa's geworpen. De kromme die de massa's volgen wordt beschreven als:

$$\begin{cases} x(t) = v \cdot cos(\alpha) \cdot t \\ y(t) = v \cdot sin(\alpha) \cdot t - \frac{1}{2} \cdot g \cdot t^2 \end{cases} \tag{3}$$

Plot de baan die deze kogel aflegt. De beginsnelheid en de afschiethoek worden ingegeven worden door de gebruiker. Let hierbij op dat het mogelijk is dat de plot niet op het scherm past. Wanneer en op welke afstand raakt de kogel de grond? Ga ervan uit dat de kogel vanaf de grond afgeschoten wordt, en maak in het programma gebruik van een do-while lus.



Figuur 2: Voorbeeld kogelwerpen

2.11 X: Teken sinus

Plot een sinusfunctie in een canvas. Zorg ervoor dat de periode instelbaar is door de gebruiker. Om ervoor te zorgen dat je een continue plot bekomt, ga je in verschillende stappen te werk. Maak een array aan met een lengte dat gelijk is met het aantal horizontale pixels van de canvas (width-property). Vul deze array met de y-waarden die je bekomt uit de sinus-functie. Vergeet hierbij niet dat de ingebouwde sinus-functie van C# in radialen rekent. Je berekent de y-waarden door de gepaste x-waarde in de sinus in te vullen. Dit kan via onderstaande formule.

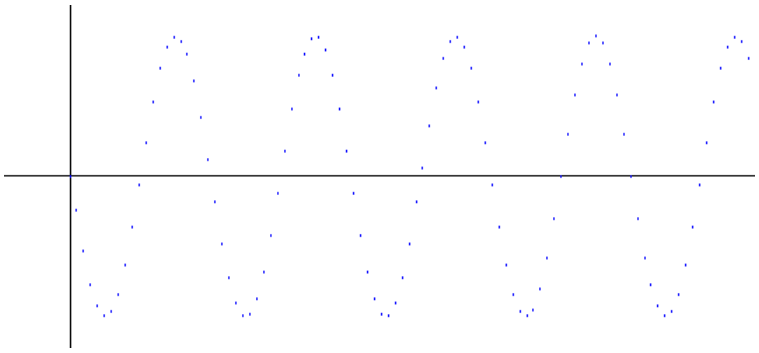
$$y = A \cdot sin(\frac{2\pi}{180} rx) \tag{4}$$

In deze formule stelt y de huidige y-waarde voor, x de huidige pixel, en r de hoeksnelheid (in te stellen door de gebruiker. Door deze punten als pixels (lijn met lengte 1) te plotten, wordt een grafiek zoals in figuur 3 bekomen. Om een vloeiende beweging te bekomen (figuur 4), worden deze punten tijdens het tekenen met elkaar verbonden. Gebruik hiervoor een lijn ("drawLine"), dat het huidige punt met het voorgaande verbindt (let hierbij op dat je loop-index vanaf 1 start

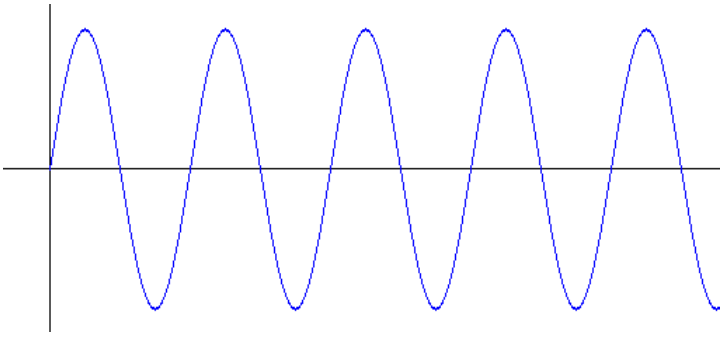
in de array, niet vanaf 0, waarom?).
Hint 1: Teken de plot met een verticale offset (een constante dat opgeteld wordt t.o.v. berekende y-waarde). Vergeet niet dat je volgens de verticale gespiegeld tekent! Als offset kan de halve hoogte van de canvas genomen worden. In dat geval wordt de getekende y-waarde:

$$y_{plot} = \frac{Height_{canvas}}{2} - y \tag{5}$$

Hint 2: Neem voor de amplitude van de sinus een waarde tussen de 25 en 250 (afhankelijk van de canvashoogte).



Figuur 3: Sinus op basis van punten.



Figuur 4: Continue sinus.