Objectgeoriënteerd Programmeren: WPO 3

1. Inhoud

Eenvoudige (enkelvoudige) overerving, override, ToString(), base, private, public, protected, virtual

2. Oefeningen

- Demo: Scheepvaart
- A: Polygon
- A: Rekenmachine
- A: Digitale logica
- E: Infection
- E: Matrices

2.1 Demo: Scheepvaart

In de scheepvaart kan men verschillende type schepen terugvinden. Zo bestaan er vrachtschepen, commerciële passagierschepen, duikboten en politieboten. Alle boten die hier vermeld zijn hebben als gemeenschappelijk kenmerken dat ze allemaal een bepaalde afmeting hebben (lengte en breedte), een positie (x en y), een snelheid (vx en vy) en een leeggewicht. De positie en snelheid van de boten worden hierbij als protected aangeduid zodat deze ook rechtstreeks via overerving aangesproken kunnen worden. Alle eigenschappen kunnen via de nodige getters en setters (properties) aangesproken worden. De verschillende schepen hebben echter elk een unieke eigenschap:

- Vrachtschepen hebben naast hun leeggewicht ook een maximale cargo die ze kunnen vervoeren. Deze wordt ook in gewicht uitgedrukt.
- Commerciële passagierschepen kunnen een maximaal aantal passagiers meenemen.
- Duikboten hebben een aantal torpedo's aan boord die ze kunnen afvuren.
- Politieschepen hebben naast een gewone snelheid ook een Fast mode waardoor ze 5 keren sneller kunnen bewegen.

Voorzie voor elk van de schepen een methode ToString() (override) waarin de eigenschappen van elk type schip worden teruggegeven. Voorzie voor elk van de klassen een tekenmethode die toelaat om het schip op en canvas weer te geven.

2.2 A: Polygon

Schrijf een klasse polygon waarin je volgende methoden, velden en eigenschappen (properties) voorziet:

- een lijst voor de x-coördinaten,
- een lijst voor de y-coördinaten,
- een methode Draw met als argumenten de canvas en een kleur,
- properties die toelaten om de coördinaten op te vragen (alleen lezen),
- een methode AddPoint waarin je een nieuw punt toevoegt aan de polygon (x en y). Hier zijn er dus 2 argumenten.

Deze klasse is een algemene klasse en laat dus toe om alle mogelijke vormen te tekenen. Echter is het ook mogelijk om gespecialiseerde klassen te schrijven die bv. een rechthoek, driehoek en vierkant tekenen. Erf de klasse polygon over in de klassen rectangle, square en triangle. In deze klassen komen onderstaande elementen.

- De klasse square heeft als argumenten voor de constructor het startpunt (x,y) en de grootte van het vierkant.
- De klasse rectangle heeft als argumenten voor de constructor het startpunt (x,y), tesamen met de hoogte en de breedte van het vierkant.
- De klasse triangle heeft als argumenten voor de constructor 3 punten (telkens x en y, dus 6 argumenten).

Voor deze klassen voorzie je de nodige properties en/of methodes. Merk op dat de superklasse zelf de coörindaten bijhoudt en ze niet kan aanpassen. Voorzie dus geen methodes in de kindklassen die hieraan tegenstrijdig zijn. Alle waarden die gebruikt worden om de objecten aan te maken mogen in de code zelf geschreven zijn (main window). Dit vereenvoudigt de opgave.

2.3 A: Rekenmachine

Schrijf een programma waarin je een bewerking zal uitvoeren op 2 ingegeven getallen. Schrijf eerst de superklasse mathoperation waarin je 2 getallen bijhoudt (kan aanpassen en lezen). Voorzie hierbij de nodige methoden/properties om de getallen te kunnen aanpassen indien nodig. Erf deze klasse over in de kindklassen multiplication, division, addition en subtraction. Elk van deze klasse zal een specifieke methode implementeren (add, multiply, subtract en divide) die een getal retourneert. Gebruik de 2 getallen van de superklasse om de bewerking op uit te voeren. De velden van de superklasse (de 2 getallen) zijn getalA en getalB en zijn beide protected. Ga ervan uit dat de bewerking altijd getalA (operatie) getalB is. De getallen geef je in via 2 textboxen. Via de gepaste knoppen (4 voor de bewerkingen) maak je het juiste object aan waarmee je bewerking uitvoert.

In principe zou je de bewerking in het main programma kunnen omzetten naar een string. Kan je de bewerking in elk van de klassen omzetten naar een string via de methode ToString()? Het resultaat van de ToString zou bv. moeten zijn:

- "3 + 5 = 8",
- "5 * 9 = 45",
- "17 / 5 = 3.4".

2.4 A: Digitale logica

Schrijf een programma waarin je digitale logica kan emuleren. Schrijf een klasse logicgate waarin je volgende methoden, velden en eigenschappen voorziet:

- inputgate1 (integer dat ofwel 0 of 1 is),
- inputgate2 (integer dat ofwel 0 of 1 is) en
- de velden (protected) die beide inputs bevat.

Merk op dat de waarden nooit anders mogen zijn dan 0 of 1!

Erf deze klasse over in de klassen orgate, andgate, nandgate en xorgate. Schrijf de methoden Or, And, Nand en Xor in respectievelijk de klassen orgate, andgate, nandgate en xorgate. Elk van deze methoden retourneert een integer (0 of 1) met als resultaat de gepaste operatie. Deze methoden nemen geen argumenten op. Overschrijf hierbij ook de methode ToString() zodat relevante informatie rechtreeks vanuit de klasse zelf geprint kan worden. Bv. "1 AND 0 = 0", "1 OR 1 = 1" en "1 XOR 1 = 0".

2.5 E: Infection

In dit programma ga je een simulatie bouwen van een virale besmetting. De simulatie bevat bloedlichamen en virussen. Beide hebben een aantal elementen gemeenschappelijk zoals de positie (binnen de canvas) en de verplaatsing. Voorzie de nodige properties om de positie en verplaatsing (respectievelijk x en y, dx en dy) te kunnen aanpassen. Voorzie ook een methode Update die het object kan verplaatsen bij elke timer-tick. Zorg ervoor dat de positie nooit buiten de canvas komt. Dit kan je doen door ook maximale x- en y-waarden mee te geven aan het object (constructor). Deze klasse herbergt de functionaliteiten van een voorgaande opgave: biljarttafel. Je kan deze klasse eerst uittesten door deze als een grijze cirkel op de canvas uit te tekenen. Deze klasse wordt door de klassen virus en bloedlichaam overgeërfd. De specifieke eigenschappen van deze klassen worden hieronder weergegeven.

De klasse hemellichaam beweegt en gaat dood als deze 10 virussen heeft opgenomen. Een bloedlichaam wordt getekend als een grote rode schijf van 25 pixels in diameter. Voorzie dus in de klasse een teller die bijhoudt hoeveel virussen er zijn tekengekomen. De klasse voorziet zelf een methode om getekend te kunnen worden (argument: canvas). De klasse beschikt ook over een methode die de afstand met een virus bepaalt (argument: virus, return boolean). Indien de virus zich op minder dan 5 pixels van het bloedlichaam bevindt, wordt de virus door het

bloedlichaam opgenomen (virussen teller ophogen). In dat geval geeft deze methode de waarde true terug. De virus verdwijnt dan ook. Bij het sterven van het bloedlichaam verandert deze naar een paarse kleur en kan deze ook niet meer bewegen. Virussen die vanaf dan nog in aanraking komen met bloedlichaam hebben dan geen effect meer.

De klasse virus beweegt en gaat dood wanneer deze door een bloedlichaam opgenomen wordt (zie hierboven). Een virus tekent zichzelf met een methode die als argument de canvas heeft. De virus is een groen vierkantje van 10 pixels. Een virus doet naast het bewegen en infecteren van bloedlichamen verder niets.

Maak in het main programma een lijst aan voor de bloedlichamen en een andere lijst voor de virussen. Telkens de canvas hertekend wordt, kijken alle bloedlichamen na of deze niet in aanraking komen met een virus. Is dit het geval, dan wordt de betrokken virus uit de lijst van virussen verwijderd. Merk op dat je hierbij best de lijst van virussen achterstevoren afhandelt. Maak bij het opstarten 100 virussen en 10 bloedlichamen aan. De posities worden op een randomwijze bepaakt. De verplaatsingen worden ook random bepaald, en liggen tussen 1 en 5. Gebruik een timer.

2.6 E: Matrices

Matrices en vectoren zijn handige wiskundige elementen die toelaten om snel en efficiënt berekeningen geprogrammeerd te krijgen. In deze opgave ga je zelf matrices en vectoren en de bewerkingen programmeren. Schrijf een klasse matrix waarin je een 2D array bijhoudt. De constructor van de klasse matrix neemt als argumenten de hoogte en de breedte van de matrix. De dimensies van de matrix kunnen na het aanmaken van het object niet meer gewijzigd worden. Voorzie volgende methoden:

- een methode Multiply(matrix m1) waar matrix m1 vermenigvuldigt wordt met de huidige matrix. Deze methode retourneert een nieuwe matrix met de juiste waarden,
- een methode Add(matrix m1) waard matrix m1 opgeteld wordt bij de huidige matrix.
- een static methode Identity(int grootte) die een vierkantsmatrix retourneert met een 1 op de diagonaal.

Erf de matrix over in de klassen kolomvector en rijvector. Beide nemen in de constructor 1 argument op: de lengte of de hoogte. Het andere argument dat doorgegeven wordt naar de constructor van de matrix (superklasse) wordt hier stilzwijgend op 1 gezet. Voor de rijvector voorzie je de methode Add(rijvector r1) waarbij de waarden uit r1 opgeteld worden aan de huidige rijvector. Voor de kolomvector voorzie je volgende methoden:

- Add(kolomvector c1) waarbij de waarden uit c1 aan de huidige kolomvector opgeteld worden,
- Multiply(matrix m1) waarbij de matrix m1 rechtsvermenigvuldigt wordt met de huidige kolomvector. Deze methode retourneert een kolomvector met de juiste waarden.

Alle waarden uit de matrices moeten ook ingevuld kunnen worden. Voorzie daarom in de klasse matrix de nodige eigenschappen zodat de waarden ingevuld en uitgelezen kunnen worden. Voordat een operatie uitgevoerd wordt, dient men ook na te gaan of de dimensies

van de matrices/vectoren correct zijn. Zorg ervoor dat de hoogte en breedte van een matrix opgevraagd kan worden (properties). Deze zijn uiteraard enkel uit te lezen (er naar toe schrijven is dus niet toegelaten).