Programmeren in C#

Hoofdstuk 1: De achtergronden van C#

- C# is een objectgeoriënteerde taal, afgeleid van Java en C++.
- De Microsoft .Net-omgeving is een zeer krachtig softwareproduct en bevat met C#, Visual Basic, C++ en F# enkele geavanceerde hedendaagse programmeertalen.
- Een programma is een lijst van instructies die automatisch worden opgevolgd door een computer.
- Objectgeoriënteerd programmeren (OOP) blijft erg belangrijk in het hedendaagse programmeren; C# maakt er volop gebruik van.

Hoofdstuk 2: De C#-ontwikkelomgeving

- Een belangrijk deel van de programmeertaak bestaat eruit besturingselementen op een formulier te zetten en hieraan bepaalde initiële properties (eigenschappen) mee te geven. Met de faciliteiten van de IDE van C# is dat een koud kunstje, al moet je er wel veel mee oefenen.

Hoofdstuk 3: Objecten en methoden voor eenvoudige graphics

- Statements (opdrachten) worden altijd netjes van boven naar beneden achter elkaar uitgevoerd, tenzij we expliciet om iets anders vragen.
- C# heeft een arsenaal tekenmethoden dat je kunt aanroepen om plaatjes te tekenen.
- Het grafisch systeem is gebaseerd op coördinaten.
- We kunnen bij het aanroepen van methoden de bijbehorende argumenten meegeven.
- Objecten aangemaakt via C# en XAML zijn gelijkwaardig.

Hoofdstuk 4: Variabele en bewerkingen

- Variabelen worden gebruikt om waarden op te slaan. Variabelen behouden hun waarde totdat we ze expliciet veranderen (bijvoorbeeld door een andere toekenningsopdracht).
- Operatoren werken op waarden.
- Een expressie is een berekening die een waarde oplevert. Deze kan in allerlei situaties gebruikt worden, ook als het rechterlid van een toekenning of als het argument van een methodeaanroep.

Hoofdstuk 5: Methode en argumenten

- Methoden voeren deeltaken van een programma uit.
- We kunnen argumenten doorgeven aan parameters.
- Een methode gebruiken wordt het aanroepen van een methode genoemd.
- Functiemethoden retourneren een resultaat.

Hoofdstuk 6: Objecten

- Het .Net-framework voorziet in een klassenbibliotheek met een enorm aantal klassen dat je kunt (en moet) gebruiken. Naast de besturingsklassen uit de toolbox zijn er nog vele andere klassen die in je programma kunnen worden ingebouwd met behulp van de using-opdracht en de bijbehorende constructor. Je zal daarnaast natuurlijk ook steeds zelf klassen maken en toevoegen aan je project. Dit zal het onderwerp vormen van hoofdstuk 10.

Programmeren in C#

Hoofdstuk 7: Beslissingen: if en switch

- If-opdrachten stellen de programmeur in staat om de volgorde van handelingen te regelen door het programma een test uit te laten voeren. Na afloop van de test voert de computer, afhankelijk van het testresultaat, verschillende rijen handelingen uit.
- Er zijn twee varianten van de if-opdracht:
  If
  If...else
- Aan een variabele van het type bool kan de waarde true of de waarde false worden toegekend. Een booleaanse variabele kan worden test met een if-opdracht.
- Met de switch-opdracht kan op een gemakkelijke manier een hele reeks tests worden uitgevoerd, maar de switch-opdracht kan alleen op integers en strings worden toegepast.

Hoofdstuk 8: Herhaling: for, while en do

- Een herhaling in een programma heet een lus. In C# zijn er drie instructies om de computer een lus te laten uitvoeren: for, while en do.
- Gebruik for als je van tevoren weet hoeveel herhalingen er moeten worden uitgevoerd.
- Gebruik while wanneer je van tevoren niet weet hoeveel herhalingen er moeten worden uitgevoerd.
- Do wordt als alternatief gebruikt voor while en wanneer een voorwaarde aan het eind van een lus getest moet worden.

Hoofdstuk 9: Debuggen

- Debuggen is het opsporen en vervolgens herstellen van fouten in een programma.
- De geïntegreerde ontwikkelomgeving van C# beschikt over een debugger-programma.
- Een breakpoint is een punt, gemarkeerd door een stip in de rand van de editor, waarop het programma tijdelijk stopt.
- Singe stepping is het volgen van het pad dat het programma volgt tijdens het uitvoeren.
- De actuele waarden van variabelen kunnen zowel bij breakpoints als gedurende single stepping getoond worden.

Programmeren in C#

Hoofdstuk 10: Klassen schrijven

- Een object is een instantie van een klasse. De klasse definieert een blauwdruk door middel van een verzameling gegevens met de daarbij behorende handelingen, methoden en properties die op de gegevens kunnen werken. C#-programma's worden geconstrueerd als een verzameling objecten.
- Er is een methode, met dezelfde naam als de klasse, die de initialisatie van een nieuw gemaakt object uitvoert. Deze methode wordt de constructormethode genoemd en wordt aangeroepen met new.
- Items in een klasse kunnen als private of als public worden gedeclareerd. Naar een private-item kan alleen van binnen de klasse worden verwezen. Naar een public-item kan door elke methode, property enzovoort worden verwezen (zowel binnen als buiten de klasse). Bij het ontwerpen van een C#-programma wordt het aantal public-items meestal tot een minimum beperkt om zo veel mogelijk gebruik te maken van het verbergen van informatie. Vooral gegevensitems worden meestal private gehouden.
- De beschrijving static betekent dat de variabele, property of methode tot de klasse behoort en niet bij bepaalde objecten? Een static-methode kan rechtstreeks worden aangeroepen, zonder dat het nodig is een instantie van de klasse met new aan te roepen. Een static-methode of -property is nuttig wanneer een methode niet bij een bepaald object hoort, of wanneer er handelingen met betrekking tot de klasse als geheel moeten worden uitgevoerd.

Hoofdstuk 11: Overerving

- Het overerven (uitbreiden) van de faciliteiten van een klasse is een goede manier om gebruik te maken van bestaande programmadelen (klassen).
- Een subklasse erft de faciliteiten van haar directe superklasse en van alle superklassen daarboven in de overervingsboom.
- Een klasse heeft slechts een directe superklasse waar ze van kan erven. Dit wordt in OOP-jargon enkelvoudige overerving genoemd.
- Een klasse die kan de faciliteiten van een bestaande klasse uitbreiden door een of meer van de volgende voorzieningen te bieden:
    - Aanvullende methoden en/of propertiers;
    - Aanvullende variabelen;
    - Methoden en/of properties die methoden en/of properties in de superklasse overschrijven (vervangen).
- Een variabele, methode of property kan op drie manieren toegankelijk zijn:
    - Public: toegankelijk vanuit elke klasse.
    - Private: alleen toegankelijk vanuit de klasse waarin de methode, property of variabele gedefinieerd is.
    - Protected: toegankelijk vanuit de klasse waarin de property, methode of variabele gedefinieerd is en vanuit elke subklasse hiervan.
- Een klassediagram is een boomstructuur die de overervingsrelaties laat zien.
- Naar de naam van de superklasse van een klasse wordt verwezen met het woord base.
- Een abstracte klasse wordt beschreven met abstract. Een dergelijke klasse kan niet worden gebruikt om een object te creëren, omdat deze onvolledig is. Een dergelijke abstracte klasse biedt nuttige variabelen, properties en methoden, die door subklassen worden geërfd.

Programmeren in C#

Hoofdstuk 12: Berekeningen

- Getallen kunnen als int of als double worden gerepresenteerd. Deze verschillende representaties hebben verschillende bereiken en nauwkeurigheden.
- Een variabele kan worden gedeclareerd als const. De waarde van een dergelijke variabele kan in de uitvoeringsfase van het programma niet worden veranderd.
- Bibliotheekfuncties bieden de bekende wiskundige functies, zoals de e-macht en de sinus van een hoek.
- De programmeur moet altijd alert zijn op exceptions die tijdens de berekeningen zouden kunnen optreden.

Hoofdstuk 13: Gegevensstructuren: lijsten en ListBox

- Een ListBox is een GUI-box die een lijst items bevat.
- Elk item in een ListBox is eenduidig vastgelegd door een integer, die een index genoemd wordt. Indexwaarden worden niet opgeslagen. Indexwaarden beginnen altijd bij 0.
- Een programma kan aan het einde van een ListBox items toevoegen, een item verwijderen, een item veranderen of een item op een willekeurig punt in een ListBox invoegen.
- Een ListBox gebruikt een Ilist-object om de informatie van de lijst op te slaan. Dit lijstobject ondersteunt de methoden om items aan een listBox toe te voegen of daaruit te verwijderen.
- Een lijst bevat een verzameling gegevenswaarden of objecten. Een dergelijke lijst groeit of krimpt, afhankelijk van de hoeveelheid gegevens die hij bevat.
- Elk item in een lijst is uniek en wordt bepaalde door een integer, een index.
- Bij het creëren van een lijst bepaalt de programmeur welk type objecten de lijst zal bevatten.
- Tot methoden die werken op lijsten behoren Add, Clear, Contains, IndexOf, Insert, RemoveAt, Remove.

Hoofdstuk 14: Eendimensionale arrays

- Een array is een verzameling gegevens. De programmeur geeft de array een naam. Alle items in de array moeten van hetzelfde type zijn (bijvoorbeeld allemaal int).
- Een array wordt samen met de andere variabelen gedeclareerd zoals in:
    - Int[] harry = new int[25];

  Waarin 24 de waarde van de grootste index is. De array heeft 25 elementen.

- Naar een individueel element in de array wordt verwezen met een met een integerwaarde, bijvoorbeeld:
    - harry[12] = 45;
- Indices hebben waarden die bij nul beginnen en doorlopen tot een bepaalde maximale waarde.

Programmeren in C#

Hoofdstuk 15: Tweedimensionale arrays

- Een tweedimensionale array is een verzameling gegevens in een tabel, met rijen en kolommen.
- De naam van een array wordt door de programmeur bepaald.
- Een array wordt samen met andere variabelen gedeclareerd zoals met:
    - int[,] alice = new [24, 30];

  waarin 23 de grootste index van de rijen in de array is en 29 de grootste index van de kolommen.
- Naar een individueel element van de array wordt verwezen door indices met een integerwaarde, bijvoorbeeld:
    - alice[12, 3] = 45;
- Jagged arrays zijn ook tweedimensionale rijen, maar kunnen per rij verschillen in dimensie. Je herkent ze aan het gebruik van [][] in de plaats van [,].

Hoofdstuk 16: Bewerkingen met strings

- Instanties van de klasse String bevatten een opeenvolging van tekens. Het eerste teken staat op positie 0.
- String-instanties kunnen bijvoorbeeld als volgt worden gedeclareerd en gecreëerd:
    - string s;
    - string name = “Mike”;
- De belangrijkste voorzieningen voor stringbewerking zijn de volgende methoden en properties:
    - Strings wijzigen
        - ToUpper
        - ToLower
        - Trim
        - Insert
        - Remove
    - Strings onderzoeken
        - Length
        - Substring
        - IndexOf
        - LastIndexOf
        - StartsWith
        - EndsWith
        - Split
    - Typeconversie
        - Convert.ToString
        - Convert.ToInt32
        - Convert.ToDouble

De klasse StringBuilder is een hulpklasse voor het manipuleren van strings. Het gebruik ervan is vaak aangewezen bij toepassingen die veel stringobjecten moeten verwerken en aanpassen, omdat daardoor de efficiëntie van je applicatie enorm kan verbeteren.

Programmeren in C#

Hoofdstuk 17: Exceptions

- Een exception is een ongebruikelijke situatie.
- Exceptions zijn instanties van klassen, die gecreëerd worden met new en opgegooid met throw.
- try-blokken worden gebruikt rond code die een exception zou kunnen opgooien.
- Een catch-blok kan een type exception opvangen of een klasse die verschillende exceptions bevat.
- De overeveringsboom van de klasse Exception (figuur 17.3) toont je de belangrijkste exceptions die je moet afhandelen en de klassen waarin deze zich bevinden.
- Je kunt zelf exceptions ontwerpen door een klasse te schrijven die erft van ApplicationException.

Hoofdstuk 18: Bestanden

- Bestanden kunnen worden geopend om er gegevens naar weg te schrijven of gegevens uit te lezen en daarna weer gesloten worden.
- De klasse Directory biedt toegang tot de inhoud van een directory. De methoden uit deze klasse kunnen worden gebruikt om de namen van subdirectory's en bestanden te manipuleren.

Hoofdstuk 19: Consoleprogramma's

- Consoletoepassingen kunnen tekstregels op het scherm zetten en ook inlezen wat de gebruiker typt.
- Ze kunnen argumenten inlezen die via de opdrachtregel zijn ingevoerd.
- Ze kunnen op verschillende manieren worden opgestart.
- Hun uitvoer kan verlegd worden naar een bestand met behulp van > en >>
- Batchbestanden kunnen opdrachten bevatten om programma's te draaien.

Hoofdstuk 20: Objectgeoriënteerd ontwerp

De objectgeoriënteerde ontwerptaak bestaat uit het vaststellen van de juiste objecten en klassen. In dit hoofdstuk zijn de volgende stappen in de aanpak van het OOOD aanbevolen:

1. Het bestuderen en indien nodig verduidelijken van de specificatie.
2. Objecten, properties en methoden afleiden van de specificatie, zodat het ontwerp fungeert als een model voor de toepassing. De werkwoorden zijn methoden en de zelfstandige naamwoorden objecten.
3. De objecten naar klassen generaliseren.
4. Controleren of er bibliotheekklassen of ander bestaande klassen kunnen worden hergebruikt door waar mogelijk compositie en overerving te gebruiken. Een 'is-een' en 'heeft-een'-analyse helpen om te bepalen of het om een overeverings- of compostie relatie gaat.
5. Gebruik richtlijnen voor klassenontwerp om een ontwerp te verfijnen.

Programmeren in C#

Hoofdstuk 21: Programmeerstijl

- Een goede programmeerstijl is belangrijk voor het bevorderen van de leesbaarheid, zodat debuggen en onderhoud gemakkelijker kunnen worden uitgevoerd. Een actuele verwijzing naar de stijlregels die Microsoft adviseert (en waarop dit hoofdstuk gebaseerd is), vind je terug op de website van dit boek (www.pearsonmylab.nl)
- Tot de richtlijnen voor goede programma-lay-out behoren het gebruik van geschikte namen, inspringingen, witregels en commentaar.
- C# heeft een nuttige voorziening om van de juiste gegevensitems constanten te maken.
- Klassen moeten een duidelijke, samenhangende doelstelling hebben.
- Geneste ifs, lussen en ingewikkelde voorwaarden moeten altijd zorgvuldige bekeken worden.
- Goede documentatie is altijd de moeite waard.

Hoofdstuk 22: Testen

- Testen is een techniek om te bereiken dat een programma zo veel mogelijk vrij van fouten is.
- Uitputtend testen is onmogelijk, omdat er te veel verschillende gevallen zijn.
- De testgegevens voor black-boxtests zijn uitsluiten gebaseerd op de specificatie van het programma.
- De testgegevens voor white-boxtests zijn gebaseerd op kennis van de werking van het programma.
- Inspectie betekent het bestuderen van de programmacode om fouten te vinden.
- Het met een debugger door de code heen stappen (walkthrough) kan een waardevolle bijdrage leveren aan het testen van een programma.
- Door stapsgewijs te ontwikkelen, houd je het ontwikkelen van grote programma's relatief simpel.

Hoofdstuk 23: Interfaces

- Interfaces worden gebruikt om de door een klasse geleverde diensten te beschrijven.
- Interfaces zijn handig om de structuur van het programma te beschrijven. Die beschrijving kan gecontroleerd worden door de C#-compiler.
- Interfaces kunnen worden gebruikt om te garanderen dat een klasse beantwoordt aan een bepaalde interfacebeschrijving. Dit bevordert de onderlinge aansluiting.

Hoofdstuk 24: Polymorfie

De principes van polymorfie zijn:

1. Een object behoudt altijd de identiteit van de klasse van waaruit het gemaakt is. Een object kan dus nooit worden geconverteerd naar een object van een andere klasse.
2. Wanneer een methode of property op een object wordt gebruikt, wordt automatisch de juiste methode of property geselecteerd.

Polymorfie versterkt het principe van het verbreden van informatie en draagt bij aan het breed toepasbeer maken van stukken code.