Volgende aanpassingen/fouten zijn na de druk van het boek gedetecteerd. Ze werden reeds verbeterd in de online versie van het handboek (kleine typfouten behandelen we hier niet)

Pagina 17

Volgende tip werd nog toegevoegd: "Je kan gratis op Memrise deze cursus dagelijks instuderen, de ideale manier om snel essentiele C# begrippen voor altijd te onthouden. De cursus is beschikbaar via : app.memrise.com/course/6382184/zie-scherp-scherper-programmeren-in-c-deel-1/."

Onderaan pagina 78:

Haakje teveel in de tweede lijn code.

Foute lijn:

```
(double)((tempGisteren + tempVandaag)) / 2); //geeft 22
```

Moet zijn:

```
(double)((tempGisteren + tempVandaag) / 2); //geeft 22
```

Bovenaan pagina 167:

De dag-array bestaat niet en moet de regen-array zijn:

Foute lijn:

```
double gemiddelde = (dag[0]+dag[1]+dag[2]+dag[3]+dag[4]+dag[5]+dag[6])/7.0;
```

Moet zijn:

```
double gemiddelde = (regen[0]+regen[1]+regen[2]+regen[3]+regen[4]+regen[5]+regen[6])/7.0;
```

Onderaan pagina 199:

We hebben Length nodig, niet GetLength wanneer we werken met jagged arrays (dank aan Erik Speelman hiervoor).

De foute zin: "Uiteraard moet je er wel rekening mee houden dat niet eender welke index binnen een bepaalde sub-array zal werken, het is dan ook aangeraden om zeker de `GetLength`-methode te gebruiken om de sub-arrays op hun lengte te bevragen."

Werd vervangen door: "Uiteraard moet je er wel rekening mee houden dat niet eender welke index binnen een bepaalde sub-array zal werken, het is dan ook aangeraden om zeker de `Length`-methode te gebruiken om de sub-arrays op hun lengte te bevragen. Wanneer je `.Length` bevraagt van de `tickets` array dan zal je 3 als antwoord krijgen, daar deze 2D array uit 3 sub-arrays bestaat.

Wil je vervolgens de lengte kennen van de middelste sub-array (met dus index 1) dan gebruik je `tickets[1].Length`."

Pagina 233:

De klasse persoon bevat instantievariabelen waardoor we in deze klasse nooit de voornaam en achternaam zullen kunnen instellen. Dit voorbeeld is veel duidelijker met auto-properties:

We maken van de klasse:

```
class Persoon
{
    public string Voornaam {get;set;}
    public string Achternaam {get;set;}
}
```

Vervolgens wordt de hele klasse:

```
class Persoon
{
       public string Voornaam {get;set;}
       public string Achternaam {get;set;}
       public string VolledigeNaam
       {
           get
           {
                   return $"{Voornaam} {Achternaam}";
           }
       }
        public string Email
       {
           get
           {
               return $"{Voornaam}@ziescherp.be";
           }
    }
}
```

Halverwege pagina 240:

De eerste zin over TimeSpan klopt niet, je kan DateTime objecten enkel van elkaar aftrekken, optellen gaat niet.

De zin: "Je kan DateTime objecten ook bij elkaar optellen en aftrekken."

Werd herschreven als: "Je kan DateTime objecten ook van elkaar aftrekken (optellen gaat niet!)."

Bovenaan pagina 257:

Dit is een subtiele bug (dank Erik Speelman!). Het resetten van een object dat je meegeeft via een methode, kan niet zoals het voorbeeld toont . De methode `VoegMetingToeEnVerwijder` moet herschreven worden als

```
public void VoegMetingToeEnVerwijder(Meting inMeting)
{
   Temperatuur += inMeting.Temperatuur;
   inMeting.Temperatuur = 0;
   inMeting.OpgemetenDoor =  "";
}
```

Ter info. Wil je wél je object kunnen resetten in de methode m.b.v. `new` dan zal je het `ref`-keyword moeten gebruiken (zie appendix pagina 406). De methode wordt dan herschreven als: `java public void VoegMetingToeEnVerwijder(ref Meting inMeting) { Temperatuur += inMeting.Temperatuur; inMeting.Temperatuur = 0; inMeting.OpgemetenDoor = ""; }`

Hier wordt de aanroep in plaats van `m1.VoegMetingToeEnVerwijder(m2);` dan `m1.VoegMetingToeEnVerwijder(ref m2);`.

Halverwege pagina 373:

Een nieuwe term **upcasting** werd nog toegevoegd aan volgende zin:

Originele zin: "Kortom, polymorfisme laat ons toe om referenties naar objecten van een child-type, toe te wijzen aan een variabele van het parent-type."

Nieuwe zin: Kortom, polymorfisme laat ons toe om referenties naar objecten van een child-type, toe te wijzen aan een variabele van het parent-type (**upcasting**)."

Onderaan pagina 391:

De klasse `MinisterVanMilieu` kan nog eenvoudiger en bevat code die weg moest. De nieuwe klasse wordt:

```
class MinisterVanMilieu:IMinister
{
    public void Adviseer()
    {
        VerhoogBosSubsidies();
        OpenOnderzoek();
        ContacteerGreenpeace();
    }

    private void VerhoogBosSubsidies(){ ... }
    private void OpenOnderzoek(){ ... }
    private void ContacteerGreenpeace(){ ... }
```

```
    }
}
```