	Sterk (3)	Voldoende (2)	Matig (1)	Zwak (0)
Klassendiagram omzetten	Het klassendiagram en de code komen exact overeen.	Het klassendiagram en de code komen in grote lijnen overeen. De klassen en relaties zijn juist, maar enkele methoden en attributen zijn niet aanwezig.	De klassen komen overeen, maar de relaties zijn niet overal juist, enkele methoden en attributen zijn niet aanwezig.	Klassen verschillen of zijn niet aanwezig.
Overerving	Overerving is compleet en juist geïmplementeerd.	Overerving is niet compleet geïmplementeerd, en/of methodes en constructoren roepen niet de base klasse aan wanneer mogelijk.	Overerving is geïmplementeerd, maar methodes en/of constructoren ontbreken.	Er is geen overerving toegepast.
Sorteren	Sorteren is zowel met IComparable als IComparer juist toegepast.	Sorteren is met IComparable en IComparer toegepast, maar de invulling is niet geheel correct.	Sorteren is maar met 1 van de 2 interfaces toegepast.	Sorteren is niet met de gevraagde interfaces toegepast.
Streams	Er wordt correct gebruik gemaakt van de juiste stream om een klasse of bestand op te slaan.	De juiste stream wordt gebruikt, maar niet alles wordt correct opgeslagen.	Er wordt een stream gebruikt, maar niet de juiste.	Er worden geen streams gebruikt.
Exceptions	Risicovolle code wordt netjes afgevangen met exception handling; de gebruiker wordt wanneer relevant over de fout geïnformeerd.	Risicovolle code wordt netjes afgevangen met exception handling; de gebruiker krijgt geen duidelijke melding.	Een enkele keer wordt exception handling toegepast, maar op veel plaatsen ontbreekt deze; of exception handling wordt overdadig veel gebruikt.	Er wordt geen exception handling toegepast.
Interfaces	Interfaces worden op de juiste manier gemaakt, gebruikt en geïmplementeerd.	Interfaces worden op de juiste manier gemaakt en/of gebruikt, maar de implementatie is niet geheel correct.	Er is een interface aangemaakt maar niet gebruikt, of een gebruikte interface is niet geïmplementeerd.	Er is geen gebruik gemaakt van interfaces.
Unit testing	Unit tests zijn zinnig, specifiek en opgesplitst op basis van individuele functies; functies zijn volledig gedekt door de tests.	Unit tests zijn breed van opzet en testen slechts een deel van de mogelijkheden van de geteste functies.	Er zijn weinig unit tests aangemaakt, of deze zijn erg elementair en bieden weinig meerwaarde bij het ontwikkelproces.	Er zijn geen unit tests geïmplementeerd.
Functionaliteit	De gehele user interface werkt probleemloos; event handling is correct geïmplementeerd.	Nagenoeg de hele user interface is functioneel, maar een enkele knop heeft een onvolledige of onjuiste functie; niet overal is event handling correct toegepast.	Gegevens worden getoond bij het opstarten van de applicatie, en/of 1 of 2 knoppen werken naar verwachting.	Er kan geen functionaliteit gebruikt worden via de user interface.
Codeerstijl	Code is juist ingedeeld in klassen; functies en blokken code zijn voorzien van commentaar; duidelijke verantwoordelijkheid voor iedere functie.	Functies zijn aangemaakt om dubbele code te voorkomen; functies zijn voorzien van beschrijvend commentaar; code heeft weinig nesting; functies hebben een duidelijke verantwoordelijkheid.	Variabelen hebben beschrijvende namen; af en toe is commentaar toegevoegd; veel dubbele code.	Er is geen aandacht besteed aan de codeerstijl: variabelen hebben onduidelijke namen; uitlijning is inconsistent; veel dubbele code; overdadige nesting.
Architectuur	De applicatie heeft een duidelijke scheiding tussen de aanwezige lagen en er kan zonder problemen nieuwe functionaliteit toegevoegd worden.	Er is een scheiding tussen de lagen, maar de code in deze lagen staat niet op de goede plek. Het uitbreiden van de applicatie betekent dat bestaande code gewijzigd moet worden.	Code is onderverdeeld in klassen, maar veel functionaliteit is nog altijd in de event handlers te vinden.	Er is geen rekening gehouden met de structuur van de applicatie: heel veel code staat direct in de event handlers.