**Naam student: Kars Christian van Velzen**

**Feedback op de oplossing van: Tibo Verreycken**

**Wat vond je van deze oplossing?**

Deel A: STELLINGEN

*Zet achter elke stelling een kruisje in het voor jou passende hokje. Licht je antwoord toe. Feedbackformulieren waarin geen toelichtingen gegeven worden gaan je medestudent niet vooruit helpen en worden dan ook als niet afgegeven beschouwd.*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Volledig akkoord | Akkoord | Neutraal | Niet akkoord | Helemaal niet akkoord | Toelichting |
| Het design laat toe de implementatie vlot te gebruiken. |  | X |  |  |  | Het idee van een archief-reservaties is aan te raden. |
| Het design is modulair, de modules hebben duidelijke contracten. |  | X |  |  |  | Iedere klasse is apart & eenvoudig neergeschreven. +punt:Overzicht |
| In het design wordt gesproken over WAT, niet HOE (geen verwijzingen naar implementatie). |  | X |  |  |  | Er is een duidelijke & eenvoudige abstractie gemaakt van het probleem |
| Duidelijke naamkeuze (van variabelen, constanten, functies, …) |  | X |  |  |  | Er is 1 duidelijke taalkeuze, sommige methoden hebben een iets langere naam door het Nederlandse. Duidelijke variabelnamen. |
| Zijn er methodes voorzien om makkelijk van ADT te wisselen? Je wil bv wisselen van 2-3-boom naar een hashmap. |  |  | X |  |  | Er zijn geen methode die een switch implementeren naar een andere ADT maar de implementatie is zodanig opgesteld dat je van ADT kunt wisselen. |
| Deze oplossing komt in aanmerking als voorbeeld voor de andere studenten |  | X |  |  |  | Als ik naar mijn eigen contract kijk is dit een veel duidelijker & gestructureerder voorbeeld |

Deel B: OPEN VRAGEN

*Beantwoord kort onderstaande open vragen. Probeer in je antwoord zo concreet mogelijk te zijn.*

Flexibiliteit

*Een design is pas flexibel als het goed tegen wijzigingen kan. Stel nu dat we de volgende wijzigingen doen. Geef de klassen die* ***toegevoegd*** *moeten worden (enkel bij naam, geen methodes of attributen) en de klassen die vermoedelijk moeten* ***aangepast*** *worden in dit ontwerp. Baseer je dus enkel op de contracten van dit ontwerp. Als je dat niet kan, duid dan goed aan waar het niet duidelijk is wat er moet gebeuren en/of waar er moet aangepast worden.*

|  |
| --- |
| We implementeren een AVL-boom en willen die in het reservatiesysteem gebruiken om de vertoningen op te slaan. |
| Geen aanpassingen vereist |

|  |
| --- |
| We willen dat de vertoningen in de boom zitten met als key het aantal vrije plaatsen. |
| *Geen verandering is nodig, enkel de key input verschilt.* |

|  |
| --- |
| Er zijn nu ook ochtendvoorstellingen, m.a.w. er komt een slot bij om 11u. |
| *Het systeem zou nogsteeds moeten blijven werken* |
| TOPS - volgende onderdelen van de oplossing vond ik bijzonder goed: |
| *- Overzicht, eenvoud, eenduidig taalgebruik*  *- Reservatie-archief*  *- Zaal getplaatsen methode is slim, scheelt een extra variabel in vertoning mbt het uitrekenen van vrije plaatsen - Compact & Simpel (KISS)* |

|  |
| --- |
| TIPS - welke wijzigingen (qua structuur, inhoud, vormgeving, klasses die je mist, methodes die je mist, …) die je nog niet vermeldde bij één van de bovenstaande vragen, kan je suggereren? Licht toe. |
| - Hoe een uniek id maken? Kunnen we hier geen formule voor maken? (Anders checken of het ID al dan niet bestaat)  - Vertoning: start/stop variabel (boolean) om na te gaan of een vertoning daadwerklijk bezig is  - Tijdslots in seconden plaatsen lijkt eenvoudiger |