**Feedback op de oplossing van groep:**

Groep: Cedric Hollanders, Christoph Ronken

Opdracht: 3

**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 |
| De design laat toe de implementatie vlot te gebruiken. |  | x |  |  |  | Contracten zijn voor de hand liggend, duidelijk wat waar gebruikt zal worden. |
| De design is modulair, de modules hebben duidelijke contracten |  | x |  |  |  |  |
| In de design wordt gesproken over WAT, niet HOE (geen verwijzingen naar implementatie) |  |  | x |  |  | Soms wat onduidelijk over hoe het geïmplementeerd gaat worden. |
| Duidelijke naamkeuze (van variabelen, constanten, functies, …) | x |  |  |  |  |  |
| De geavanceerde ADT's (2-3 boom, ...) maken in het design gebruik van 1 datastructuur (bv Honing, Marshmallow, ...). |  |  |  |  |  | Niet van toepassing |
| De geavanceerde ADT's (2-3 boom, ...) werken voor alle datastructuren (bv Honing, Marshmallow, ...). |  |  |  |  |  | Niet van toepassing |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Zijn er methodes voorzien om makkelijk van ADT te wisselen? Je wil bv wisselen van 2-3-boom naar een hashmap. |  |  |  |  |  | nvt |
| De gegevens van een ADT zijn enkel via de ADT operaties bereikbaar. |  |  | x |  |  |  |
| Gegevens die niet buiten het ADT moeten/mogen bekend zijn, worden verborgen. |  |  |  |  |  | nvt |
| Er is object-georienteerd gewerkt | x |  |  |  |  |  |
| Deze oplossing komt in aanmerking als voorbeeld voor de andere studenten |  | x |  |  |  | Naast de onduidelijkheden over de implementatie was alles goed te verstaan. |

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 gebruiken om de chocoladeshots op te slaan. |
| *De class die hiervoor aangepast moet worden is de winkelclass,*  *-er moet een nieuwe wrapper geschreven worden, de stock class moet ook gewijzigd worden, het resupply contract moet aangepast worden.* |

|  |
| --- |
| We willen dat bestellingen nu gesorteerd worden op gebruikersid. |
| *De bestellingclass wordt gewijzigd, de createOrder en makeOrder ook.* |

|  |
| --- |
| Er zijn nu ook light chocoladeshots, met minder suiker, maar enkel voor de witte en melkchocolade. |
| *Stock moet aangepast worden, de bestellingclass en de chocolademelk class ook.* |
| TOPS - volgende onderdelen van de oplossing vond ik bijzonder goed: |
| *In het algemeen zijn de contracten vrij duidelijk, de grotere methods waren goed uitgebreid.* |

|  |
| --- |
| 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. |
| Het contract was niet nauwkeurig genoeg, de kleinere classes hadden geen methods, het was niet duidelijk wat er ging gebeuren. |