Programmeeropdracht PO_Paradigma versie 2019-2020

1 Inleiding

Een van de competenties van een HBO afgestudeerde is de vaardigheid om zichzelf een nieuwe programmeertaal aan te leren. Binnen de opleiding is er slechts ruimte voor enkele programmeertalen, veel minder dan er gebruikt worden in het werkveld. Daarnaast is het belangrijk om in te zien dat er verschillende programmeerparadigma's bestaan.
De cursus APP is te kort om je volledig te bekwamen in de verschillende programmeerparadigma's. In deze opdracht ga je daarom voor een taal en paradigma die je interessant lijken wat meer de diepte in. Je gaat in deze opdracht zelf een programmeeruitdaging formuleren, die vervolgens uitwerken in een voor jou nieuwe programmeertaal en daarover rapporteren in blogvorm.
Deze opdracht biedt je bewust weinig handvatten, het is echt de bedoeling om jezelf te verdiepen en op zoek te gaan naar geschikte literatuur. Dit is in lijn met de uitgangspunten van het ASD semester: zelfstandig en verdiepend. Begin dan ook op tijd aan deze opdracht.

2 Doel

Het eerste doel van deze opdracht is het zelfstandig leren van een nieuwe programmeertaal en daarmee je software craftmanship te vergroten. Het tweede doel is om kennis op te doen van een tot dusver voor jou onbekend programmeerparadigma. Het derde doel is de kennis die je daarbij opdoet op een toegankelijke manier te delen met collega's.

3 Instructies

De opdracht bestaat concreet uit drie onderdelen:
Het kiezen van een taal/paradigma en je erop oriënteren met behulp van (online) literatuur en/of tutorials
Het definiëren van een eigen uitdaging en die uitprogrammeren in de gekozen taal.
Het schrijven van een reeks blogposts over je ervaringen met de nieuwe taal.

3.1 Kiezen en oriënteren taal/paradigma

Omdat de imperatieve en object-georienteerde paradigma's al uitgebreid aan bod zijn gekomen, zijn programmeertalen die hierop gebaseerd zijn niet toegestaan. Voorbeelden van **niet** toegestane programmeertalen zijn: Java, C#, C, C++, Ruby, Perl, PHP, Python, Bash, LUA, Javascript (en afgeleiden).

Talen die expliciet wel zijn toegestaan zijn o.a.:

- Scala
- Erlang
- Clojure
- Haskell
- Prolog
- Assembler(*) voor een specifieke CPU

(*) Strikt genomen is Assembler een imperatief paradigma, maar vanwege de verdieping in hardware die benodigd is staan we deze taal toe.

Kies je een andere taal? Dat mag. Toon dan aan dat deze taal een ander paradigma dan imperatief of OO implementeert. Je dient **vooraf** goedkeuring te krijgen van je docent.

Zoek nu literatuur, (online) tutorials, coding challenges etc op en bekwaam jezelf in de basics van de taal. Maak hiertoe opgaven die de gevonden literatuur je biedt om de taal te leren. **Leg je vorderingen vast in blogvorm (zie hoofdstuk 3.3).**

Als optionele literatuur bij dit vak staat het boek ”Seven Languages in Seven Weeks”. Je kunt voor een aantal talen de introductie hoofdstukken uit dit boek volgen. Dit boek is echter sterk verouderd en veel opgaven / tools uit dit boek werken daarom niet meer goed. Je zult deze problemen dan zelf moeten “overwinnen”.

3.2 Het definiëren en uitwerken van een eigen uitdaging

Formuleer zelf een programmeeruitdaging die je gaat uitwerken in de door jouw gekozen programmeertaal. **Beschrijf je uitdaging en leg deze ter goedkeuring voor aan de docent.**

Je uitdaging moet:

- uitdagend zijn. Geef duidelijk aan wat jij denkt dat moeilijk gaat worden en eventueel welke onderdelen eenvoudiger.
- passen bij het gekozen paradigma.
- SMART geformuleerd zijn. Het moet helder zijn wanneer je de uitdaging succesvol hebt uitgevoerd.

Werk jouw uitdaging naar beste kunnen uit. Het kan tijdens het uitwerken blijken dat je uitdaging veel te moeilijk of makkelijk blijkt. Stel in dat geval je uitdaging op tijd bij. De uitdaging is niet het belangrijkste maar wel een goede analyse. Daarover meer in de volgende paragraaf.

3.3 Vastleggen vorderingen in blogvorm

Doe in een een blogpost of een serie blogposts verslag van je aanpak en welke interessante aspecten aan de taal je bent tegengekomen (denk aan: syntaxis, semantiek, pragmatiek en paradigma’s) in vergelijking met jou al bekende talen zoals Java.
Leg je vorderingen van het zoeken van literatuur en het maken van tutorials vast (met code voorbeelden). Leg uit wat de uitdaging was, wat je geprobeerd hebt en waarom. Geef naast het beschrijven van wat je gedaan hebt vooral ook je mening. Als je tussentijds je uitdaging hebt moeten bijstellen, beschrijf je dat ook in je blogpost(s).
Geef aan waar het knelpunt lag en wat je aangepast hebt, of geef juist aan hoe je de lat voor jezelf hoger hebt gelegd. Je schrijft de blogpost(s) niet voor een opdrachtgever of docent, maar voor je collega’s. De tekst moet geschreven zijn als een IT-blog. Neem bijvoorbeeld codevoorbeelden op in de tekst, verwijs naar regels daarin met uitleg, en plaats in de lopende tekst links naar relevante online bronnen. Ook kun je schermafdrukken of andere figuren opnemen. Wat je gebruikt om de blog te maken maakt niet uit, zolang je je docent maar doorgeeft hoe hij/zij erbij kan.

4 Beoordeling

De opdracht wordt als voldoende of onvoldoende beoordeeld.

Beoordelingscriteria:

- Alle door jou gemaakte code en, wanneer van toepassing, de uitvoer ervan, is inzichtelijk voor de docent.
- De programmeeruitdaging is SMART geformuleerd.
- Je beargumenteert waarom deze uitdaging kan passen bij de gekozen taal en het paradigma.
- Je (eventueel bijgestelde) uitdaging is moeilijk genoeg om je in de taal te verdiepen.
- Je maakt gebruik van de eigenschappen van het gekozen paradigma.
- Je maakt gebruik van de mogelijkheden van een blog (inline code, directe links, media etc).
- Je legt niet alleen uit wat je gedaan hebt, maar geeft ook je mening.
- De tekst is geschreven voor de doelgroep collega-software ontwikkelaars.