Samen op weg naar jouw succes!

Stichting Empower Yourself

ICT Empowerment Jr. App Tester: opdrachten

Auteur:	René Steetskamp (steets@otech.nl)
Opdrachtgever:	Stichting Empower Yourself (Margarita González Saavedra)
Datum:	25 augustus 2019
Status:	Concept
Kenmerk:	20190825-OT-SEY-RS-1
Versie:	0.1

Inleiding

Doelstelling van dit document

Dit document bevat de opdrachten die horen bij de ICT Empowerment omscholingscursus tot Junior App Tester.

Doelgroep van dit document

Dit document is gericht aan de deelnemers aan deze cursus. De deelnemers zijn ingedeeld in groepen, waarbinnen ze de opdrachten in dit document uitvoeren.

Gebruik van dit document

De opdrachten in dit document zijn bedoeld om zelfstandig door de deelnemers uitgevoerd te worden. De voorbereiding en de individuele opdrachten dienen deelnemers voor zich zelf uit te voeren. De groepsopdrachten voer je samen met je groep uit.

Van elke les dient per groep een korte presentatie te worden gemaakt en vóór de volgende les ingeleverd via Slack in het kanaal van de betreffende les. Voor de presentatie kan gebruik worden gemaakt van bijvoorbeeld Google Docs of LibreOffice. Voorafgaand aan de volgende les zal van één deelnemer worden gevraagd om de resultaten van zijn groep te presenteren.

Verdere aanwijzingen:

- De opdrachten hebben tot doel om de deelnemers aan te zetten om zelf informatie op te zoeken op het Internet en deze informatie binnen hun groep te bespreken.
- Achter elke opdracht staat een indicatie van de tijd die aan de opdracht dient te worden besteed (exclusief pauzes). Deze tijdsindicatie is bedoeld als timebox:
    - Probeer die tijd er ook aan te besteden: het gaat er niet om de opdrachten zo snel mogelijk af te hebben, maar om zoveel mogelijk kennis te vergaren.
    - Als je de opdracht niet af krijgt in de aangegeven tijd, stop dan toch met deze opdracht en ga verder met de volgende.

Centrale case

Naast diverse andere voorbeelden, kijken we bij elke les kijken we naar een centrale case: een text adventure zoals Dunnet. Deze case vormt de rode draad door de hele cursus.

Les 1: Software Engineering

Voorbereiding

Opdracht 1.1: Dunnet (60min)

Ontdek de text adventure Dunnet.

Individuele opdrachten

Opdracht 1.2: Werkgebieden (45min)

In de presentatie die bij dit onderwerp hoort, zijn een aantal werkgebieden van Software Engineering genoemd. Onderzoek drie van deze werkgebieden en beantwoord de volgende vragen:

1. Zoek een korte beschrijving van het werkgebied.
2. Zoek een screenshot van een toepassing in dat werkgebied er uit ziet.
3. Zoek uit of/hoe Python in dit werkgebied wordt gebruikt.

Opdracht 1.3: Decompositie (45min)

Naast software kun je ook hardware in functionele brokjes opdelen. Denk aan een PC die je kan opdelen in scherm, processor, toetsenbord, etc. Beschrijf tenminste 5 functionele onderdelen van een auto. Geef naast de naam van het onderdeel ook een beschrijving van wat het onderdeel doet en hoe het samenwerkt met andere onderdelen.

Opdracht 1.4: Wenselijke systeemeigenschappen (30min)

In de presentatie wordt een aantal wenselijke systeemeigenschappen genoemd. Zoek en beschrijf nog tenminste twee andere voorbeelden.

Opdracht 1.5: Agile principes (30min)

De presentatie benoemt 12 agile principes. Kies 3 van deze principes die je het meest aanspreken en omschrijf waarom je deze hebt gekozen.

Opdracht 1.6: Agile voordelen (30min)

Beschrijf drie voordelen van Agile ten opzicht van het watervalmodel.

Opdracht 1.7: Agile risico's (60min)

Naast veel voordelen, brengt Agile ook risico's met zich mee. Als je deze risico's kent, kan je je er beter tegen wapenen ('een gewaarschuwd man, telt voor twee'). Zoek op het internet zo veel mogelijk nadelen en risico's op, en beschrijf de vijf belangrijkste.

Groepsopdrachten

Opdracht 1.8: Agile (90min)

Bediscussieer de voor- en nadelen van Agile ten opzicht van het watervalmodel, en vat deze samen in een presentatie (1 sheet voor de voordelen, 1 sheet voor de nadelen).

Opdracht 1.9: functionele beschrijving (120min)

Beschrijf drie gewenste functies van een text adventure zoals Dunnet. Beschrijf elke functie op één sheet van de presentatie:

1. Eén in gewoon Nederlands.
2. Eén als Scrum user story.
3. Eén als Gherkin scenario.

Als voorbeeld staan hieronder drie gewenste functies van een cursussysteem op die manieren beschreven:

1. Het systeem moet cursusinformatie kunnen opslaan in een database.

2. Als trainer
   Wil ik huiswerk van deelnemers kunnen bekijken,
   Zodat ik kan zien of de deelnemer de stof heeft begrepen

3. Scenario: Gemiddeld cijfer
   Gegeven een deelnemer met cijfers 4, 6 en 8
   Dan heeft de deelnemer een gemiddeld cijfer van 6
   En staat de deelnemer op “voldoende”

Les 2: Scrum

Individuele opdrachten

Opdracht 2.1: nalezen (90min)

Bestudeer http://www.scrumguides.org/scrum-guide.html.

en http://scrumguide.nl/

Opdracht 2.2: Scrum board in Trello (45min)

Schrijf je in op https://trello.com/ en maak een nieuw board aan waarin je je voortgang van de opdrachten in deze omscholing bijhoudt.

Groepsopdrachten

Opdracht 2.3: Scrum Daily Standup (45 min)

Vanaf nu is het de bedoeling dat jullie de dag als groep beginnen met een Scrum Daily Standup.

Bestudeer de tips en valkuilen op:
https://agilescrumgroup.nl/5-tips-goede-daily-stand-up-meeting/

Bespreek met je groep manieren om de valkuilen binnen jullie groep te vermijden.

Opdracht 2.4: Team Scrum board (45min)

Maak een Scrum board aan waarin je als groep kunt samenwerken. Gebruik dit Scrum board bij toekomstige opdrachten voor de samenwerking binnen je groep.

Opdracht 2.5: INVEST-vuistregel (30min)

Bespreek in hoeverre de user story van opdracht 1.9 voldoet aan de INVEST-vuistregel. Verbeter de functiebeschrijving waar mogelijk.

Groepsopdrachten (verdieping)

Opdracht 2.6: DevOps (90 min)

Naast Scrum komt DevOps erg vaak voor als Agile methode in vacatures. Zoek met je groep op wat DevOps betekent en wat de overeenkomsten / verschillen zijn met Scrum. Verwerk dit in een presentatie (2 à 3 dia's).

Opdracht 2.7: Continuous delivery (90 min)

Om de korte iteraties van agile methoden mogelijk te maken, zijn technieken voor Continuous Integration ontwikkeld. Zoek de volgende begrippen op en verwerk ze in een presentatie (1 dia per begrip):

- Continuous integration
- Continuous delivery
- Continuous testing
- Rolling release

Geef in jullie presentatie ook aan wat de relaties zijn tussen deze begrippen.

Besteedt in een aparte sheet aandacht aan de rol van deze technieken in een agile project.

Voorbereiding les 3: Testen

We hebben nu gekeken naar software engineering en projecten in het algemeen. Voor deze omscholing concentreren we ons op testen

Opdracht 3.1: lezen (90min)

Bestudeer hoofdstukken 0 t/m 3 van
https://www.scrum.as/academy.php?show=2&chapter=1.

Les 3: Testen

Individuele opdrachten

Opdracht 3.1: vacatures (30min)

Zoek vacatures voor software testers op het Internet. Onderzoek welke vaardigheden er gevraagd worden. Maak onderscheid naar persoonlijke/sociale vaardigheden en technische vaardigheden.

Opdracht 3.2: SWOT-analyse(45min)

Bij een SWOT-analyse (Strengths, Weaknesses, Opportunities, Threats) zet je de sterke/zwakke punten en kansen/bedreigingen op een rijtje. Zoek op hoe je een SWOT-analyse uitvoert en pas dit toe op jezelf ten opzichte van de gevraagde vaardigheden uit de opdracht 3.1. Dus welke vaardigheden heb je al, welke vaardigheden moet je nog leren, etc.

Groepsopdrachten

Opdracht 3.3: SMART-vuistregel (45min)

Bekijk in hoeverre de functiebeschrijvingen van opdracht 1.9 voldoen aan de SMART-vuistregel. Verbeter de functiebeschrijving waar mogelijk.

Opdracht 3.4: Test documentatie (120 minuten)

In IEEE 829 wordt een aantal begrippen genoemd die binnen een project worden gebruikt om het testen in goede banen te leiden:

- (Master) Test Plan
- Test Design Specification
- Test case
- Test scenario / script
- Incident report

Zoek van elk van bovenstaande begrippen een definitie en een voorbeeld/template.

Opdracht 3.5: vacatures (30min)

Bespreek jullie resultaten van opdracht 3.1.

Voorbereiding volgende les

Volgt later

Les 4: Testen met Python

Individuele opdrachten

Opdracht 4.1: installeren (60 min)

Installeer Python, Visual Studio Code en Git.

Verken deze tools en kijk wat je er nu al mee kunt. Installeer in Code ook de Python plugin.

Opdracht 4.2: lezen (60 min)

Lezen: https://cgoldberg.github.io/python-unittest-tutorial/

Bestuderen: https://docs.python.org/3/library/unittest.html

Zorg dat je antwoord kan geven op vragen zoals:

- Wat is het verschil tussen setUp en setUpClass?
- Wat is het nut van tearDown?
- Welke assertions zijn er en hoe gebruik je die?

Zoek andere bronnen als je de antwoorden niet kunt vinden in de twee artikelen hierboven.

Opdracht 4.3: bash (30 min)

Oefen met `bash`. Voer bijvoorbeeld alle voorbeelden uit van https://docs.python.org/3/library/unittest.html. Zorg dat je weet hoe je naar een directory kunt gaan, hoe je de inhoudt van een directory kunt onderzoeken, hoe je een bestand kunt maken, etc.

Opdracht 4.4: alternatieven (60 min)

WIj gebruiken Python, (Visual Studio) Code en (Git) Bash. Maar er zijn veel alternatieven, waatonder (selectie):

Programmeertaal	**Editor**	**CLI**
PHP Java C#	Visual Studio (grote broer) UltraEdit Wing IDE	cmd powershell

Bekijk deze alternatieven en onderzoek de verschillen en overeenkomsten, voor- en nadelen.

Duo-opdrachten

Voer onderstaande opdrachten uit met één van je teamleden. Voer de opdrachten wel allebei uit op je eigen laptop.

Opdracht 4.5: test case (60 min)

Voer het uitgebreidere voorbeeld uit die tijdens de les behandeld is:

1. plaats `BasicFunction` in het bestand `basicfunction.py`
2. plaats de test case in het bestand `basicfunction_test.py`
3. voer `basicfunction_test.py` uit

Opdracht 4.6: test case uitbreiden (45 min)

Breidt de class `BasicFunction` van de vorige opdracht uit met een methode "`clear_state`" die de state weer op 0 zet.

Voeg een test toe om deze nieuwe methode te testen **.**

Opdracht 4.7: maximum

Python biedt standaard de functie max die het maximum van twee getallen bepaalt. Zie https://docs.python.org/3/library/functions.html#max

Schrijf zelf een functie die hetzelfde doet.

Schrijf in een apart bestand unit tests (in een test case) die controleren dat jouw functie hetzelfde doet als de standaardfunctie. Test in elk geval voor de volgende waarden:

- 0 en 1
- 3 en -5
- 6.7 en 3.8
- 4 en "tekst"

Opdracht 4.8: boetes uitdelen

Stel we hebben een flitspaal bij een weg binnen de bebouwde kom. Voor elke 5 kilometer die je te hard rijdt, krijg je een boete van €10. Als je meer dan 30km te hard rijdt, wordt je rijbewijs ingenomen.

Schrijf een functie die

- "OK" oplevert als je je aan de maximum snelheid van 50kmh houdt,
- die "rijbewijs ingenomen" teruggeeft als je meer dan 30km te hard rijdt, en
- anders het bedrag van je boete teruggeeft.

Schrijf een test die deze functie controleert. Test eerst de snelheden 45, 65 en 90.

Welke waarden verwacht je bij de snelheden 0, 50 en 80? schrijf hier ook tests voor.

Opdracht 4.9: raadspelletje I

De website https://www.practicepython.org/ biedt een lijst met opdrachten (met oplossingen) om te oefenen met Python.

Bestudeer oefeningen 1, 2 en 8, en probeer oefening 9 zelf te doen.

Opdracht 4.10: raadspelletje II

Bestudeer oefeningen 21 en 22, en probeer oefening 30 zelf te doen.

Opdracht 4.11: Fibonacci

Bestudeer oefening 13. Bij deze oefening worden twee oplossingen gegeven.

Sla de oplossingen op in een .py-bestand op je eigen laptop en schrijf in een apart bestand een test case die van beide oplossingen test of ze werken.

Test in elk geval of de oplossingen werken met een negatief getal, met een groot getal (bijvoorbeeld 1000). Wat verwacht je wat er gebeurt al je een tekst invoert? Schrijf hiervoor ook een test.

Les 5: versiebeheer met git en github

Individuele opdrachten

Opdracht 5.1: oefening (30min)

Doe de oefeningen op https://try.github.io/.

Opdracht 5.2: zelfstudie (150 min)

Bestudeer https://www.learnenough.com/git-tutorial en doe de opdrachten.

Let op: in paragraaf 1.1 staat dat je op een Unix-machine moet werken. Dit is *niet* nodig. In plaats daarvan gebruik maken van `git bash`.

Opdracht 5.3: lokale git repository (60 min)

Maak een nieuwe git repository in een nieuwe map. Plaats je broncode van opdracht 4.8 in deze map en voeg ze toe aan je repository. Als je opdracht 4.8 nog niet gemaakt had, probeer het dan nu nog een keer.

Verbeter de broncode aan op basis van de nieuwe dingen die je vanmorgen hebt geleerd over programmeren. Probeer dat zo te doen dat de tests nog steeds slagen (refactoring). Commit de wijzigingen.

Opdracht 5.4: centrale repository (120 min)

Maak een repository aan op github. En push hier de lokale repository van opdracht 5.3 naar toe.

Onderzoek hoe op github de issues en de wiki werken. Maak een issue en een wiki pagina.

Onder de tab 'projects' kan je kan je een Scrum-board maken. Onderzoek de verschillen (voor- en nadelen) in vergelijking met Trello.

Duo-opdrachten

Opdracht 5.5: samenwerken (120 min)

Ruil met een groepsgenoot: make een clone van elkaars repository.

Maak een wijziging (bij voorkeur een verbetering) in de code van je groepsgenoot. Commit deze wijziging en push hem naar de centrale repository. Hiervoor moet je groepsgenoot wel eerst schrijfrechten (*commit rights*) geven. Zoek samen uit hoe dit moet.

Maak een issue aan in de repository van je groepsgenoot en wijzig een wiki-pagina. Onderzoek ook het Scrum-board onder 'Projects'.