Nota: Schrijf je antwoorden kort en bondig in de daartoe voorziene velden. Elke theorie-vraag staat op 2 punten, elke oefening op 8 punten. Het geheel staat op 40 punten.

Vraag 1. ..[..../2]
Definieer de termen traceerbaarheid (traceability) en correctheid (correctness).

a) Traceerbaarheid: .. *kunnen we afleiden welke componenten impact ondervinden van bepaalde veranderingen?* .....

b) Correctheid:. *bouwen we het juiste product op de juiste manier?* .....

.....

Vraag 2. ..[..../2]
Wat is een kritisch pad (critical path)?

Het pad waarop vertraging op één taak resulteert in vertraging van het hele project. .....

Waarom is het belangrijk om dit pad te kennen?

Om taken te identificeren die geen vertraging mogen oplopen. .....

.....

Vraag 3. ..[..../2]
Use cases passen goed binnen een iteratief en incrementeel proces. Geef een voorbeeld en licht zowel het iteratieve als het incrementele aspect toe.

Joris Van Geet 21-1-08 11:59
Comment:
1pt voor iteratief
1pt voor incrementeel
niets voor een voorbeeldje

- *use cases verfijnen dmv scenarios* .....
- *specifieren v system boundries door iteratief actoren en use cases te identificeren*
- *alternatieve paden en extensies toevoegen aan scenarios* .....
- *project plan specifieren door iteratie over 'cost estimate' en 'priority assignment'*

Vraag 4. ..[..../2]
Geef één voordeel en één nadeel van functionele decompositie?

Voordeel: *clear problem decomposition strategy* .....

.....

Nadeel: *naive of maintainability of interoperability* .....

.....

Vraag 5. ..[..../2]

Hoe bepaal je dat je voldoende getest heb? Geef en bespreek één techniek.

statistical testing: testen tot 'failure rate' onder 'risk treshold' .....

.....

.....

Vraag 6. ..[..../2]

Waarom zijn *redundant checks* geen goede manier om *design by contract* te ondersteunen? Geef drie redenen.

Joris Van Geet 21-1-08 12:05
Comment:
,5 per juist antwoord
+,5 als alle drie juist

1) *extra complexiteit* .....

2) *performance penalty* .....

3) *of foute context: provider kan eventuele fouten niet inschatten/oplossen* .....

Vraag 7. ..[..../2]

a) Wat is het verband tussen *Clean Room Development* en formele specificaties?

Cleanroom is voornamelijk gebaseerd op formele specificaties (aangevuld met testen). .....

b) Waarom is het noodzakelijk om *sequence diagrams* aan te vullen met *state charts*?

een state chart is een specificatie van alle mogelijke en onmogelijke scenarios (sequences) .....

Vraag 8. ..[..../2]

Joris Van Geet 21-1-08 12:02
Comment:
0,5
0,5
1

Wat is *coupling*?

een maat voor hoe sterk een component gekoppeld is met een andere component via een connector .....

Wat is *cohesion*?

de mate waarin de onderdelen van een component samenhangen .....

.....

Hoe kunnen we ze gebruiken als criterium voor een goed design?

minimise coupling and maximise cohesion .....

.....

Vraag 9. ..[..../2]

Kan een correct werkend stuk software toch een slechte kwaliteit hebben? Waarom (niet)?

Ja, er zijn nog andere kwaliteitsfactoren dan correctheid, (zoals onderhoudbaarheid, efficiëntie, ...) .....

.....

Vraag 10..[..../2]

Wat is het verschil tussen *mean time to failure* en *avarage time between failures*?

MTTF is een hypothetische metriek gebaseerd op een kansverdeling .....

ATBF is een beschrijvende metriek gebaseerd op reële metingen .....

.....

Wat is het verband tussen deze twee metrieken?

ABTF wordt gebruikt om de kansverdeling van MTTF te kallibreren .....

.....

Vraag 11..[..../2]

Geef vier activiteiten die tools moeten ondersteunen tijdens refactoring.

1) *refactoring (de transformaties zelf)*.....

2) *regression testing* .....

3) *configuration- and version management* .....

4) *fast edit/compile/run cycles*.....

Vraag 12..[..../2]

Noem twee belangrijke ethische regels en verklaar ze kort.

1)..... : .....

.....

.....

2) ..... : .....

.....

.....

Vraag 13..[..../8]
In volgende oefening mag je uitgaan van een design by contract taal die naast de standaard programmeerconstructies ook volgende termen bevat:

- "∀ <elementNaam> <collectie>:", voor elk element in collectie/array geldt
- "∃" <elementNaam> <collectie>:", er bestaat een element waarvoor geldt
- "⇒", als ... dan geldt ...
- pre.<object>, waarde van een object voor methode oproep
- return, waarde die teruggegeven wordt door een methode

1. Orden de drie mogelijke postcondities van zwak naar sterk.

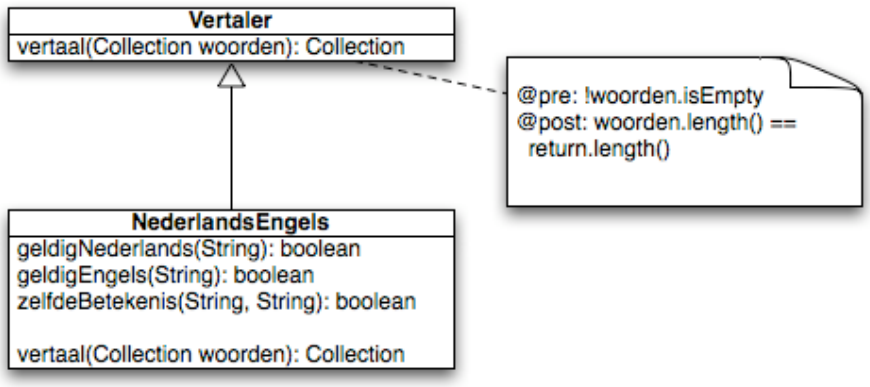
@pre: x >= 9 x := x + 5; @post: x >= 4 @post: x >= 9 @post: x >= 14

antwoord: a -> b -> c = 1punt

redenering: 1punt

1) eenvoudiger aan te voldoen omdat subset is, x => b,
2) {true} zwakste conditie, {false} sterkste, a ligt dichter bij false

2. Schrijf een contract voor de vertaal-methode uit de klasse NederlandsEngels. De methode vertaalt de lijst van meegegeven woorden van het Nederlands naar het Engels. Alle methodes uit het diagram mogen gebruikt worden. De geldig-methodes controleren of een woord bestaat in een bepaalde taal, zelfdeBetekenis controleert of twee woorden een zelfde betekenis hebben, eventueel zelfs over verschillende talen.



@pre: zelfde als super = geen punten .....

@pre: geen vernauwing van pre of commentaar dat dit niet mag = 1punt

```
public Collection vertaal(Collection woorden){
}
```

@post: zelfde als super OR zwakke conditie = geen punten

@post: Voor alle: zelfdeBetekenis() = 1punt

Voor alle: geldigEngels() = 1punt

3. Schrijf een contract voor volgende methode. Zorg er hierbij voor dat een deel van de huidige verantwoordelijk van de supplier verschuift naar de client van de methode.

   De methode koppelt twee woorden aan elkaar. Tussen beide woorden komt er een door de client opgegeven karakter te staan. Als er geen karakter werd meegegeven wordt een koppelteken gebruikt. Woorden mogen enkel letters bevatten (gecontroleerd door isLetter(char)).

@pre: woord1.size !=0 & woord2.size != 0 =1punt

@pre: woord1 & woord2 bestaan uit letters = 1punt

```
public char[] combine(char[] woord1, char[] woord2, char marker){
        if ( marker == '' ) marker :=  '-';
        if ( woord1.size == 0 ) throw Exception;
        if ( woord2.size == 0 ) throw Exception;
        char[] returnValue = new char(woord1.size + woord2.size + 1);
        for (int i = 0; i < woord1.size; i++){
                returnValue[i] := woord1[i];
        }
        for (int i = 0; i < woord2.size; i ++){
                returnValue[i+woord1.size+1] := woord2[i];
        }
        returnValue[woord1.size] := marker;
        return returnValue;
}
```

@post: Er bestaat marker= '-' of marker=pre.marker OF

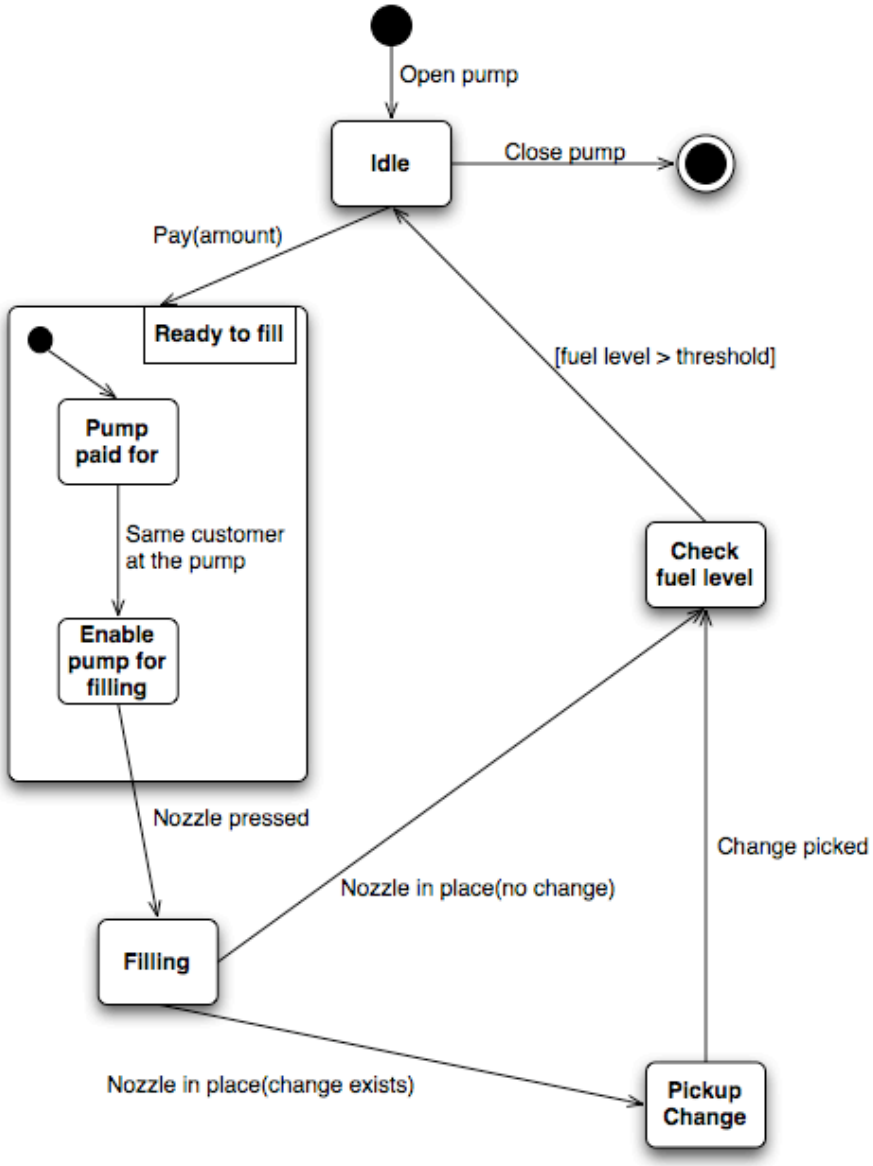
result.size=woord1.size+woord2.size+1 = 1punt

@post: zwakke conditie = geen punten

foute conditie = -0,5

Vraag 14. [..../8]
In deze vraag is het de bedoeling dat jullie het gedrag van een dieselpomp in een tankstation beschrijven met behulp van een *statechart*. De specificaties zijn als volgt:

a) Een klant moet eerst betalen. Er wordt enkel cash aanvaard. Indien de betaling gebeurd is, zal de pomp beschikbaar zijn om de toegestane hoeveelheid diesel te tanken
b) Op elk moment kan slechts één persoon de pomp gebruiken.
c) Wanneer een klant toegelaten wordt om te betalen en te tanken zal, zoals afgedwongen in punt g, er steeds voldoende diesel voorhanden zijn
d) De klant zal niet meer kunnen tanken indien de meter aangeeft dat het krediet waarvoor betaald werd opgebruikt is.
e) Elke kraan heeft een sensor die het tanken afbreekt indien de tank vol is.
f) Indien de klant meer betaald heeft dan het bedrag nodig om de tank te vullen, dan zal hij de overschot terugbetaald krijgen. Ondertussen kan een nieuwe klant niet betalen voor de pomp en hem dus ook niet gebruiken.
g) Een nieuwe klant zal pas kunnen betalen indien het brandstofniveau in de opslagtank boven een bepaalde threshold uitkomt. Indien niet zal de pomp moeten wachten tot wanneer de threshold opnieuw overschreden wordt. Gebruik *guards* om dit te modeleren.
h) Maak één van je staten *composite* om volgende conditie af te dwingen: wanneer een klant betaalt, dan mag niemand anders beginnen tanken aan de pomp.


Open pump
Idle
Close pump
Pay(amount)
Ready to fill
[fuel level > threshold]
Pump paid for
Same customer at the pump
Check fuel level
Enable pump for filling
Nozzle pressed
Change picked
Nozzle in place(no change)
Filling
Nozzle in place(change exists)
Pickup Change