Naam: MODELANTWOORD . . . . . .

# Theorie

Beantwoord onderstaande vragen (11 vragen, elk op 2 punten) door
- de antwoordzinnen KORT aan te vullen
- de fout(e) alternatief(ven) te schrappen (evt. met motivatie)

1. Wat wordt bedoeld met *traceerbaarheid* (“traceability”)?

   *Impact* van veranderingen op product componenten. . . . . .

   (Zowel tussen requirements en code als vice versa) . . . . . .

2. Wat is het verschil tussen een fout, een faling en een defect (“error”, “failure”, “fault”)?

   Fout (“error”) Een *manifestatie* van een defect. . . . . .

   . . . . . .

   Faling (“failure”) Een *afwijking* tussen specificatie en draaiend systeem. . .

   . . . . . .

   Defect (“fault”)Een*vergissing*inontwerpofcode(“Kan”abnormaalgedragveroorzaken)

   . . . . . .

3. Wanneer noemen we een pre-conditie *redelijk* ? Wat heeft redelijkheid tot gevolg?

   Een pre-conditie is redelijk als de conditie *verantwoord* kan worden in termen van de specificaties (en *alleen* de specificaties).

   . . . . . .

   Bij een redelijke pre-conditie kunnen oproepers van de overeenkomstige

   operatie de pre-conditie *verifieren/nagaan* en *vervullen* (“satisfy”)... . . . .

   . . . . . .

4. Waarom moet er altijd één acteur (“actor”) zijn die wint bij een “use case” ?

   Hij zal *argumenteren* om de use case in de behoeftenspecificatie te houden.

   Dat is belangrijk omdat de behoeftenspecificatie *zo klein mogelijk* wordt gehouden / om het *onderhandelingsproces* doorzichting te houden . . . . . .



Naam: . . . . . .

5. [Inde context van CRC kaarten.] Wat is een *verantwoordelijkheid* (“responsability”)? Wat is een *samenwerking* (“collaboration”)?

   Een verantwoordelijkheid (“responsability”) is een *publieke dienst* die een object levert aan derden (i.e., zijn kennis + zijn acties)

   . . . . . .

   Een samenwerking (“collaboration”) is den lijst van *andere objecten* die nodig zijn om een verantwoordelijkheid te vervullen.

   . . . . . .

6. Kun je drie symptomen geven van code die verbeterd kan worden d.m.v. “refactoring”?

   1. code duplicatie. . . . . .
   2. geneste condities . . . . . .
   3. grote klasses / methodes 4. “verkeerd” overerven . . . . . .

7. Wat is een patroon (“pattern”)? Waarom is dat nuttig bij het beschrijven van een software architectuur?

   Een patroon (“pattern”) is de *essentie* van een oplossing voor een terugkerend *probleem* in een welbepaalde *context*

   . . . . . .

   Dat is nuttig expertendiscussies / documentatie van bestaande ervaring / kennis van wanneer (en wanneer niet) toe te passen / kennis van de voor- en nadelen .

8. Kun je 3 argumenten *tegen* formele methodes opsommen? Is er eventueel een corresponderend argument *voor*?

   1 (tegen) dure specificatie . . . . . .

   2 (tegen) hoge training. . . . . .

   3 (tegen) moeilijk voor eindgebruikers . . . . . .

   1 (voor) goedkope implementatie . . . . . .

   2 (voor) simpele wiskunde/logica . . . . . .

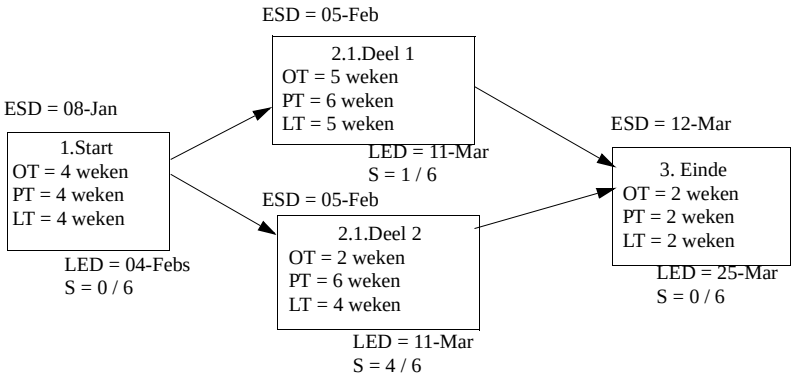
   3 (voor) hangt af van de gebruiker + animatie . . . . . .

9. Als kwaliteitscontrole geen kwaliteit kan garanderen, waarom hechten we er dan toch zoveel belang aan ?

   Omdat zo het toeval vermeden wordt / kans op kwaliteit toeneemt . . . . . .

   . . . . . .

10. Gegeven het volgende PERT diagram en een kalender.




Legende: OT = optimistische tijd; PT = pessimistische tijd; LT = waarschijnlijke ("likely") tijd

| week | startdatum | week | startdatum | week | startdatum |
|---|---|---|---|---|---|
| 2 | 08-Jan | 7 | 12-Feb | 12 | 19-Mar |
| 3 | 15-Jan | 8 | 19-Feb | 13 | 26-Mar |
| 4 | 22-Jan | 9 | 26-Feb | 14 | 2-Apr |
| 5 | 29-Jan | 10 | 5-Mar | 15 | 9-Apr |
| 6 | 05-Feb | 11 | 12-Mar | 16 | 16-Apr |

Het project zal aanvangen op 8 januari 2001. Bereken (a) de waarschijnlijk einddatum van het project (op basis van LT); (b) het kritisch pad op basis van LT; (c) het pad met het grootste risico op afwijking. (*Toon de tussenberekeningen op het PERT diagram !*)

(a) Waarschijnlijk eindatum = 25 maart . . . . . .

(b) Kritisch pad = 1 + 2.1 + 3 . . . . . .

(c) Pad met grootste risici = 1 + 2.2 + 3

want = SQRT ( 0 + $(4 / 6)^2$ + 0) > SQRT ( 0 + $(1 / 6)^2$ + 0) . . . . . .

11. Wat bedoelen we met de uitspraak 'de koppeling metriek ("coupling metric") voldoet niet aan de respresantatie conditie'?

Twee componenten A en B kunnen sterker gekoppeld zijn dan twee andere C en D, en toch kan de gemeten koppeling tussen A en B kleiner zijn dan de gemeten koppeling van C en D





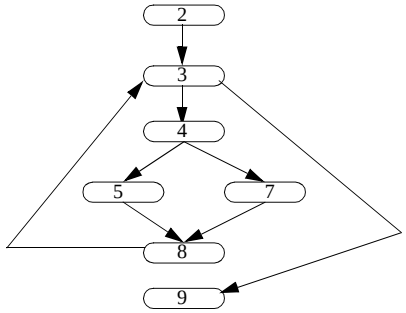
# Oefeningen

1. Oefening 1: Basic Path Testing

Beschouw onderstaande C++ methode ggd(a, b). Gevraagd wordt A) om de overeenkomstige flow chart te tekenen, B) de cyclomatische complexiteit te berekenen, C) een set van paden op te stellen voor basic path testing en D) overeenkomstige testcases te bepalen.

```
1.  int ggd(int a, int b) {
2.        assert (a > 0 && b > 0);
3.        while (a != b) {
4.              if (a > b)
5.                    a = fmod(a-1,b) + 1;
6.              else
7.                    b = fmod(b-1,a)+1;
8.        }
9.        return a;
10. };
```

OPM: fmod(a, b) geeft de rest weer van de gehele deling a/b. Vb: fmod(15,7) = 1




B) 8 pijlen - 7 knopen + 2 = 3 = aantal binaire condities + 1 = aantal vlakken

C) {2, 3, 9} {2, 3, 4, 5, 8, 3, 9} {2, 3, 4, 7, 8, 3, 9}

D) Testcase1: ggd(1, 1) = 1

Testcase2: ggd(2, 1) = 1

Testcase3: ggd(1, 2) = 1

Naam: . . . . . .

2. Oefening 2: Z, pre- en postcondities

Beschouw een access controle systeem dat bestaat uit een set machines (hosts). Elk van deze machines staat aan of uit. Er is 1 bijzondere machine, genaamd *local*, die staads aan staat. Op deze machine staat een verzameling van documenten, waarbij elk document een uniek nummer heeft. Tevens houdt de machine *local* een lijst bij van machines die deze documenten mogen raadplegen (opvragen).

Gebruikt het type *Host* voor de machines, het type *DocNr* voor een document nummer, het type *Document* voor het eigenlijke document en het type *Status* voor de toestand van een machine:

*Status* ::= on | off

a) Geef een Z specificatie van het systeem
b) Geef een schema dat een document opvraag specifieert

c) Geef aan wat de preconditie(s) en postconditie(s) van het schema in b) zijn

a)

System

$$
\begin{array}{l}
machines: \mathbb{P}\ Hosts \\
status\_mach: Hosts \pfun Status \\
docs: Document \pfun DocNr \\
allowed: \mathbb{P}\ Hosts \\
\hline
local \in machines \\
dom\ status\_mach = machines \\
local \mapsto on \in status\_mach \\
allowed \subset machines \\
x \mapsto y \in docs \land z \mapsto y \in docs \Rightarrow x = z
\end{array}
$$

[ Hosts, DocNr, Document, Status ]

b) & c)

DocVraag

$$
\begin{array}{ll}
\Xi\ System & \leftarrow \text{pre \& postconditie} \\
docnr? : DocNr & \\
machine? : Hosts & \\
document! : Document & \\
\hline
machine? \in allowed & \leftarrow \text{preconditie} \\
machine? \mapsto on \in status\_mach & \leftarrow \text{preconditie} \\
document! = docs^{-1}\ (|\ docnr?\ |) & \leftarrow \text{postconditie}
\end{array}
$$