Tentamen Computerondersteund Probleemoplossen

Docent: Kurt Lust

do. 17 april 2008, 9.00-12.00 uur

Omdat ik op 6 dagen twee tentamens en een doctoraatsverdediging heb, kan het zijn dat het tentamen niet op tijd kan nagekeken worden.

Het tentamen bestaat uit 4 opgaven. Alle antwoorden moeten goed gestructureerd en gemotiveerd zijn. De gevolgde werkwijze moet duidelijk zijn. Het gebruik van een rekenmachine is toegestaan.

Wanneer je code moet schrijven, doe dit dan zorgvuldig en volgens de regels van de kunst zoals besproken in de cursus. Een correct programma is niet voldoende. Een programma moet ook leesbaar zijn en gedocumenteerd zijn waar nodig!

Het tentamen wordt eerst gequoteerd op 100 punten. 10 punten krijg je zowiezo, de resterende 90 moet je verdienen. De punten staan aangegeven bij elke vraag. Het resultaat van dit tentamen telt voor 2/3 van de eindscore, het practicumresultaat voor 1/3.

Bij het tentamen krijg je ook een evaluatieformulier. Je moet dat apart indienen. Stop het dus niet in de envelop met je antwoorden!

Gelieve **op ieder blad** Uw naam, studentennummer en studierichting te vermelden. Begin iedere vraag op een nieuwe bladzijde en maak geen puzzel van Uw oplossing. Veel succes!

Vraag 1 20 punten. Nederlandse banken, behalve de postbank, gebruiken negencijferige rekeningnummers. In de code zit een foutendetecterende mogelijkheid ingebouwd. Deze nummers moeten voldoen aan de zogenaamde *elfproef*. Deze proef werkt als volgt: het laatste cijfer wordt vermenigvuldigd met 1, het op een na laatste met 2, enzovoort, tot en met het eerste cijfer, dat dus wordt vermenigvuldigd met 9. Vervolgens maak je de som van deze producten. Deze som moet deelbaar zijn door 11. Bijvoorbeeld, voor het rekeningnummer 563098961 zegt dit criterium dat

$$5 \cdot 9 + 6 \cdot 8 + 3 \cdot 7 + 0 \cdot 6 + 9 \cdot 5 + 8 \cdot 4 + 9 \cdot 3 + 6 \cdot 2 + 1 \cdot 1 = 231$$

deelbaar moet zijn door 11.

a) Neem aan dat het rekeningnummer gewoon als een getal gegeven is, bijvoorbeeld

```
nr = 563098961;
```

Schrijf een stukje code om na te gaan of dit een correct rekeningnummer is. De uitvoer van je programma is een zin in de vorm

```
Het rekeningnummer 563098961 is een correct rekeningnummer.
Het rekeningnummer 563098961 is een fout rekeningnummer.
```

b) Stel nu dat de vaak gebruikte schrijfwijze met puntjes gebruikt wordt, en het rekeningnummer dus als een string gegeven is, bijvoorbeeld

```
nr = '56.30.98.961';
```

Schrijf opnieuw een stukje code dat de geldigheid van dit rekeningnummer verifieert. De uitvoer van je programma is nu een zin in de vorm

De opgave beslaat 4 bladzijden

```
Het rekeningnummer 56.30.98.961 is een correct rekeningnummer.
Het rekeningnummer 56.30.98.961 is een fout rekeningnummer.
```

Je mag aannemen dat de computer de ASCII conventie gebruikt voor het voorstellen van karakters, wat betekent dat 0, 1, 2, ... opeenvolgende karaktercodes hebben. Je mag in deel b) van deze opgave eventueel onderdelen van je antwoord voor deel a) herbruiken, maar dat hoeft niet. Een mogelijkheid om een string in een getal om te zetten, is de functie **str2num** (invoerargument de string, uitvoerargument het getal), maar er zijn andere zeker even geschikte mogelijkheden.

Vraag 2 20 punten. We willen een programma schrijven dat voor een niet-negatief geheel getal berekent uit hoeveel cijfers het getal bestaat (dus hoeveel cijfers er afgedrukt moeten worden als we dat getal willen afdrukken). Welke van de volgende programma's zijn correct? Geef voor een foutief programma aan wanneer het fout loopt. Zoek voor de correcte programma's een lusinvariant, plaats uitgebreid commentaar in de code en verbeter de stijl van de code ook op andere punten volgens de principes die we in de cursus besproken hebben.

a) Eerste programma:

```
function b = aantalcijfers(a)
c = 0;
d = a;
while d > 0
c = c + 1;
d = floor(d/10);
end
end
```

b) Tweede programma:

```
function b = aantalcijfers(a)
b = 0;
c = a;
while c >= 10
b = b+1;
c = floor(c/10);
end
b = b+1;
end
```

c) Derde programma:

```
function b = aantalcijfers(a)
b = 0;
d = a;
while d >= 0;
b = b + 1;
d = floor(d/10);
end
end
```

Vraag 3 20 punten. Conditie en stabiliteit.

a) Leg in enkele regels de betekenis van de begrippen conditie en stabiliteit uit. Het is niet nodig hierbij een voorbeeld te geven of lange afleidingen te maken.

b) Het is niet zo moeilijk om aan te tonen dat de omtrek van een regelmatige m-hoek waarvan de eenheidscirkel de ingeschreven cirkel is, gelijk is aan

$$2m \tan \frac{\pi}{m}.$$

Wanneer m steeds groter wordt, convergeert deze uitdrukking naar 2π, de omtrek van de eenheidscirkel. Stel nu dat P_n de helft is van de omtrek van de regelmatige 2^n-hoek. Met behulp van de betrekking

$$\tan(2x) = \frac{2\tan x}{1 - \tan^2 x}$$

is het dan mogelijk om een recursiebetrekking af te leiden voor P_n. (Het volstaat $x = P_n/2^n$ en $2x = P_{n-1}/2^{n-1}$ te nemen, maar je moet dit niet uitwerken.) Zo kan men de wiskundig equivalente recursiebetrekkingen

$$\text{(a)} \begin{cases} P_2 &= 4, \\ P_n &= 2^n \frac{\sqrt{2^{2(n-1)} + P_{n-1}^2} - 2^{n-1}}{P_{n-1}}, \ n \in \mathbb{N} \text{ en } n > 2, \end{cases}$$

en

$$\text{(b)} \begin{cases} P_2 &= 4, \\ P_n &= \frac{2P_{n-1}}{1+\sqrt{1+\left(\frac{P_{n-1}}{2^{n-1}}\right)^2}}, \ n \in \mathbb{N} \text{ en } n > 2, \end{cases}$$

afleiden. Het is duidelijk uit de definitie van P_n dat P_n naar π moet convergeren als n steeds groter wordt. Kan je zonder een uitgebreide wiskundige uitwerking besluiten of de ene betrekking te verkiezen is boven de andere? Zo ja, welke is te verkiezen en waarom? En heeft je argument met stabiliteit of met conditie te maken (en waarom)?

Vraag 4 30 punten (10+5+15). We laten een bal los op een hoogte van 2m en willen numeriek de beweging simuleren. We nemen aan dat we de bal geen rotatie meegeven, zodat hij verticaal op en neer blijft bewegen. In deze opgave verwaarlozen we ook de luchtweerstand. Bijgevolg wordt de beweging van de bal beschreven door de tweede-orde differentiaalvergelijking

$$y''(t) = -g$$

met $g = 9.81 m/s^2$. Omgezet naar een stelsel van eerste-orde differentiaalvergelijkingen krijgen we dus

$$\begin{cases} y'(t) &= v(t) \\ v'(t) &= -g. \end{cases}$$

Wanneer de bal op de grond botst, botst hij terug. De botsing is echter niet volledig elastisch: bij iedere botsing gaat een beetje energie verloren. Dat betekent dat de snelheid waarmee de bal terugbotst, kleiner is dan de snelheid waarmee hij de grond raakt:

$$y'(b+) = -r y'(b-)$$

waarbij $y'(b-)$ de snelheid is waarmee de bal de grond raakt (negatief dus), $y'(b+)$ de snelheid is waarmee hij terugkaatst, en r de restitutiecoëfficiënt is. Neem $r = 0.9$, maar zorg ervoor dat dit eenvoudig gewijzigd kan worden.

a) Stel dat je een functie

$$[\text{T, Y}] = \text{simuleer\_bal\_tot\_event}(\ \text{y0},\ \text{tstap},\ \text{Teinde}\ )$$

hebt die de bewegingsvergelijking voor de bal numeriek oplost tot de eerste gebeurtenis, waarbij "gebeurtenis" het bereiken van de opgegeven eindtijd kan zijn of het op de grond komen van de bal. De invoerargumenten van deze functie zijn een kolomvector met de beginpositie en -snelheid, een (maximale) staplengte en de lengte van het tijdsinterval waarover de differentiaalvergelijking opgelost moet worden. Uitvoerargumenten zijn een kolomvector T met tijdstippen waarop een benadering berekend werd, en een matrix Y met twee *rijen* zodat Y(1,i) de positie is van de bal op tijdstip T(i) en Y(2,i) de snelheid is van de bal op tijdstip T(i).

Schrijf een programma om de beweging van de bal te simuleren voor een gegeven tijdsinterval, inclusief het terugkaatsen van de bal bij een botsing. Gebruik hiervoor de hierboven gegeven functie simuleer_bal_tot_event . De in- en uitvoerargumenten van de te schrijven functie zijn dezelfde als voor simuleer_bal_tot_event . Zorg ervoor dat de tijdstippen waarop een botsing plaatsvindt, als twee opeenvolgende elementen voorkomen in T, de eerste keer met de inkomende snelheid in Y en de tweede keer met de snelheid waarmee de bal terugkaatst. *In deze opgave moet je geen implementatie maken van simuleer_bal_tot_event . Je mag ervan uitgaan dat je deze functie hebt. In deel c) van de vraag implementeren we dan deze functie.*

b) Geef de bevelen die je nodig hebt om met je programma de beweging van de bal te simuleren gedurende 10 seconden, waarbij je de bal uit stilstand loslaat op een hoogte van 2 meter. Geef ook het bevel (de bevelen) nodig om een figuur te maken van de hoogte van de bal in functie van de tijd.

c) Rest ons nog de functie

$$[\mathrm{T},\ \mathrm{Y}] = \mathrm{simuleer\_bal\_tot\_event}(\ \mathrm{y0},\ \mathrm{tstap},\ \mathrm{Teinde}\ )$$

uit deel (a) verder uit te werken. Schrijf deze functie, waarbij je gebruik maakt van de voorwaartse Euler methode. Je mag hierbij aannemen dat de bal vanop een geldige positie vertrekt (dus beginpositie ≥ 0) en niet in rust op de grond gelegd wordt (dus een beginsnelheid $\neq 0$ als de beginhoogte 0 is). Je hoeft niet te testen of de invoerargumenten geldige waarden bevatten.

Je mag voor een of meerdere stappen een kleinere staplengte hanteren wanneer dat nodig is. Zorg er wel voor dat er maar op één plaats in je programma wijzigingen doorgevoerd moeten worden wanneer we de bewegingsvergelijking willen aanpassen (bijvoorbeeld, luchtweerstand toevoegen).

Deze opgave is minder eenvoudig dan het lijkt. Tijdens een tijdstap kunnen er immers meerdere "gebeurtenissen" plaats vinden: Je kan in één tijdstap zowel de grond raken als de eindtijd overschrijden, en je programma moet stoppen bij de eerste gebeurtenis!

Hint: Je kan een matrix zonder kolommen maar met twee rijen aanmaken met het bevel **zeros**(2,0). Afhankelijk van de oplossing waarvoor je kiest, kan dat eventueel handig zijn in deel a) van deze vraag.