TECHNISCHE UNIVERSITEIT DELFT
Faculteit Elektrotechniek, Wiskunde en Informatica

TW1090 Inleiding Programmeren
Tentamen
16 maart 2017, 9:00 – 12:00 uur

Normering: Opgave 1 t/m 3 ieder 6 punten. Score: totaal aantal punten + 2. Cijfer: score / 2

Voorbereiding

Lees eerst de instructies op het aparte Instructie blad (Inloggen, IDLE starten, pdf van het boek openen).

Gebruik bij de implementatie van de opgaven alleen de modules / bibliotheken die nu al door middel van imports worden geïmporteerd. Voeg dus geen import statements toe. Verander ook de hoofdprogramma's niet, indien al een hoofdprogramma is gegeven. Voeg dan alleen implementaties toe van de gevraagde functies en test deze functies in het hoofdprogramma. Vul je naam en studienummer ook in het commentaar van de programma's in (zie de inhoud van de .py files). Open in de IDLE omgeving de .py file die je wilt gaan editen. Zorg ervoor dat je steeds maar 1 .py file tegelijk geopend hebt. Save je .py files regelmatig.

Opgave 1 **(punten: a: 1, b: 2, c: 3)**

Gebruik voor deze opdracht het programma in de file `A1.py`.

Punten en vectoren implementeren we met behulp van het datatype tuple (met 3 gehele getallen). Het punt $P=(1,2,3)$ of de vector $\mathbf{p}=<1,2,3>$ wordt dus in Python in een variabele p opgeslagen door middel van het statement `p = (1, 2, 3)`
We willen een computerprogramma schrijven dat voor drie punten P, Q en R in $\mathbb{R}^3$ een vergelijking berekent van het vlak door deze drie punten. De berekening wordt uitgevoerd zoals in het volgende voorbeeld met $P=(1,2,3)$, $Q=(-1,0,2)$, $R=(2,1,0)$
Bereken eerst de vectoren $\mathbf{q}-\mathbf{p}=<-2,-2,-1>$ en $\mathbf{r}-\mathbf{p}=<1,-1,-3>$
Bereken vervolgens een normaalvector $\mathbf{n}$ van het vlak met behulp van een uitwendig product
$\mathbf{n}=(\mathbf{q}-\mathbf{p})\times(\mathbf{r}-\mathbf{p})=<5,-7,4>$
De vlakvergelijking heeft nu de vorm $5x-7y+4z=d$ waarbij d gevonden kan worden door punt P in de vlakvergelijking in te vullen. In dit voorbeeld $d=5-14+12=3$
De vlakvergelijking wordt dus $5x-7y+4z=3$

Het uitwendig product van de vectoren $\mathbf{p}=<p_1,p_2,p_3>$ en $\mathbf{q}=<q_1,q_2,q_3>$ wordt op de volgende manier berekend: $\mathbf{p}\times\mathbf{q}=<p_2q_3-p_3q_2, p_3q_1-p_1q_3, p_1q_2-p_2q_1>$

Om vlakberekeningen zoals in het voorbeeld hierboven mogelijk te maken in het programma implementeren we drie functies.

a) `dif(a, b):`
Deze functie geeft het verschil $\mathbf{a}-\mathbf{b}$ terug van twee vectoren **a** en **b** in $\mathbb{R}^3$.

b) `uitproduct(a, b):`

Deze functie geeft het uitwendig product $\mathbf{a}\times\mathbf{b}$ terug van twee vectoren $\mathbf{a}$ en $\mathbf{b}$ in $\mathbb{R}^3$.

c) `vlak(p, q, r):`

Deze functie berekent een vlakvergelijking van het vlak waar de punten P, Q en R in liggen (met plaatsvectoren resp. $\mathbf{p}$, $\mathbf{q}$ en $\mathbf{r}$). De functie geeft een string terug.

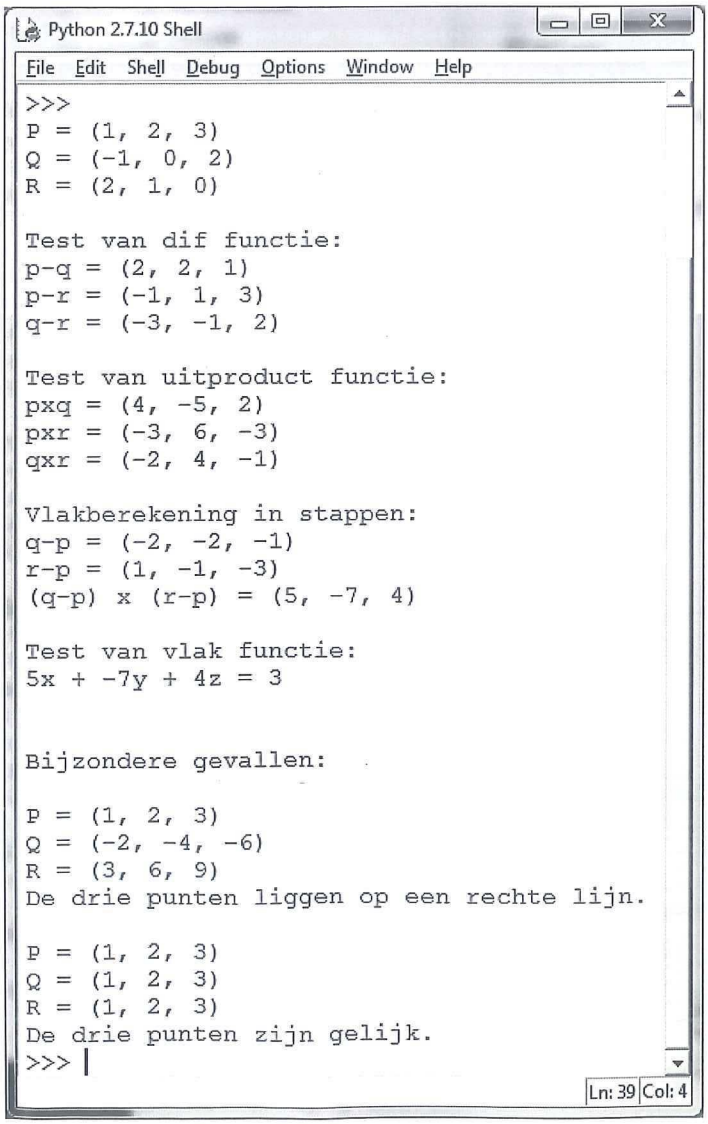
Voorbeelden van strings die de functie kan teruggeven zijn:

```
5x + -7y + 4z = 3                    0x + -7y + 4z = 0
10x + -14y + 8z = -6                 -1x + 0y + 0z = 7
-5x + 7y - 4z = -3
```

De linkse drie voorbeelden stellen allemaal hetzelfde vlak voor. Het maakt niet uit welke van de drie strings de functie in dit geval teruggeeft.

Indien er geen vlakvergelijking berekend kan worden moet de functie (afhankelijk van de situatie) de string `'De drie punten liggen op een rechte lijn.'` of `'De drie punten zijn gelijk.'` teruggeven.

Na implementatie van de functies moet het testprogramma de volgende uitvoer geven:



Opgave 2 **(punten: a: 3, b: 3)**

Gebruik voor deze opdracht het programma in de file `A2.py`.

a) `count_larger(lst):`
Deze functie bepaalt hoeveel paren opeenvolgende elementen in een lijst van gehele getallen in dalende volgorde staan.
Voorbeeld: `lst = [2, 5, 3, 8, 12, 7]`
5 > 3 en 12 > 7, dus 2 paren staan in dalende volgorde. De overige paren staan in stijgende volgorde (2 < 5, 3 < 8 en 8 < 12). De functie geeft in dit geval 2 terug. Zie het Python shell window aan het eind van deze opgave voor de gewenste uitvoer.

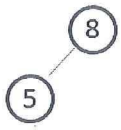
b) `is_in(bst, n):`
Deze functie bepaalt of getal `n` voorkomt in de binaire zoekboom (binary search tree) `bst`.

Een binaire zoekboom is een binaire boom met eigenschappen die ervoor zorgen dat een waarde snel gevonden kan worden. In een binaire zoekboom heeft elke knoop maximaal twee verwijzingen naar een andere knoop. Verder heeft elke knoop in de boom de eigenschap dat alle waarden in de linker subboom kleiner zijn dan de waarde in de knoop en alle waarden in de rechter subboom groter dan de waarde in de knoop.
We zullen binaire zoekbomen opslaan in geneste lijsten. Hieronder staat een voorbeeld waarbij begonnen wordt met een lege binaire zoekboom (bst). In iedere volgende bst is er een knoop aan toegevoegd op de juiste plaats.

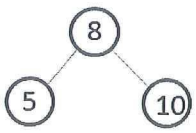
```
bst = []                                        # lege binaire zoekboom
```

```
bst = [[], 8, []]                               # 8 is toegevoegd
```

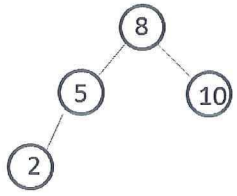

```
bst = [[[], 5, []], 8, []]                      # 5 is toegevoegd
```




```
bst = [[[], 5, []], 8, [[], 10, []]]            # 10 is toegevoegd
```




```
bst = [[[[], 2, []], 5, []], 8, [[], 10, []]]   # 2 is toegevoegd
```



Opgave 3 **(punten: a: 2, b: 2, c: 2)**

Gebruik voor deze opdracht het programma in de file `A3.py`.

Vul de implementatie van de klasse `Curve` aan. De klasse moet na implementatie van de ontbrekende methoden voldoen aan het klassendiagram hieronder:

class Curve(object)

Instance attributes

a	float
b	float
c	float
d	float
n	int

Methods

__init__	(constructor)
__str__	(overload str function)
generate_points()	-> x, y (numpy arrays)
draw()	genereert een plot van de kromme

Een instance van de klasse `Curve` bevat 5 attributen a, b, c, d en n en stelt de vectorfunctie $f(t) =< a\cos(bt), c\sin(dt) >$ voor. De functie wordt benaderd met n+1 punten in het interval $[-\pi, \pi]$.

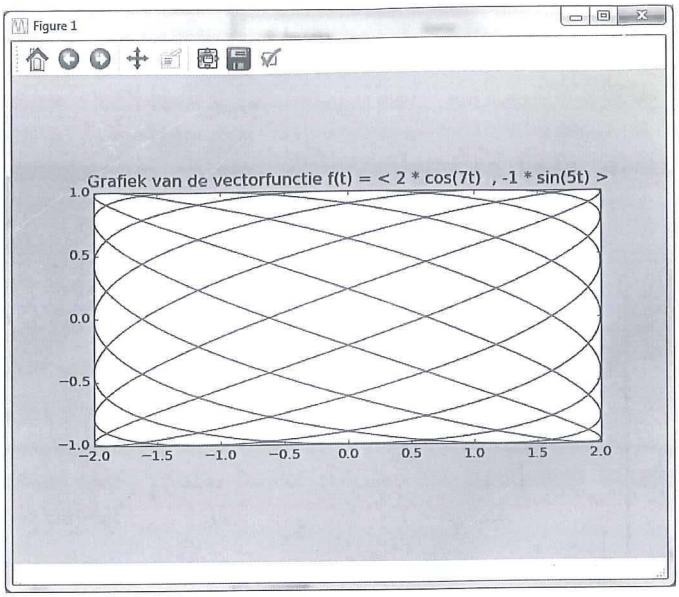
a) Overload de standaard functie `str` voor Curve objecten. Na `c = Curve(1, 2, 3, 4, 100)` is `c` van type `Curve` en moet `str(c)` de volgende string teruggeven:
`'f(t) = < 1 * cos(2t) , 3 * sin(4t) >'`

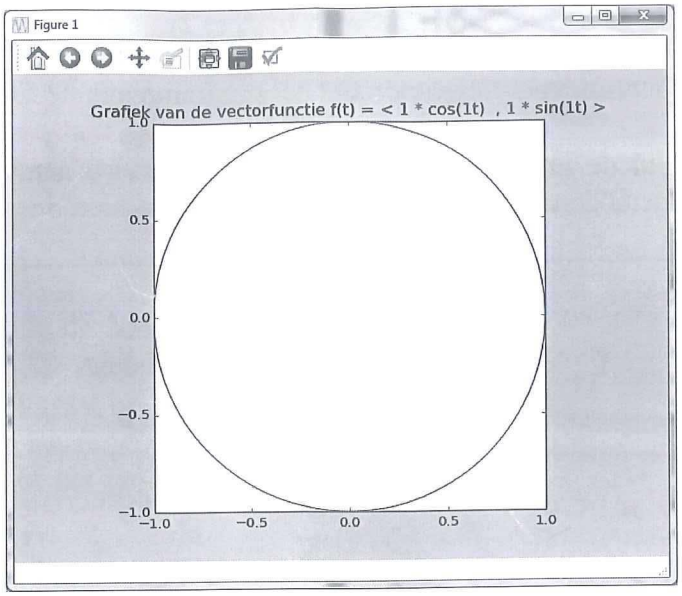
Implementeer tevens de methoden:

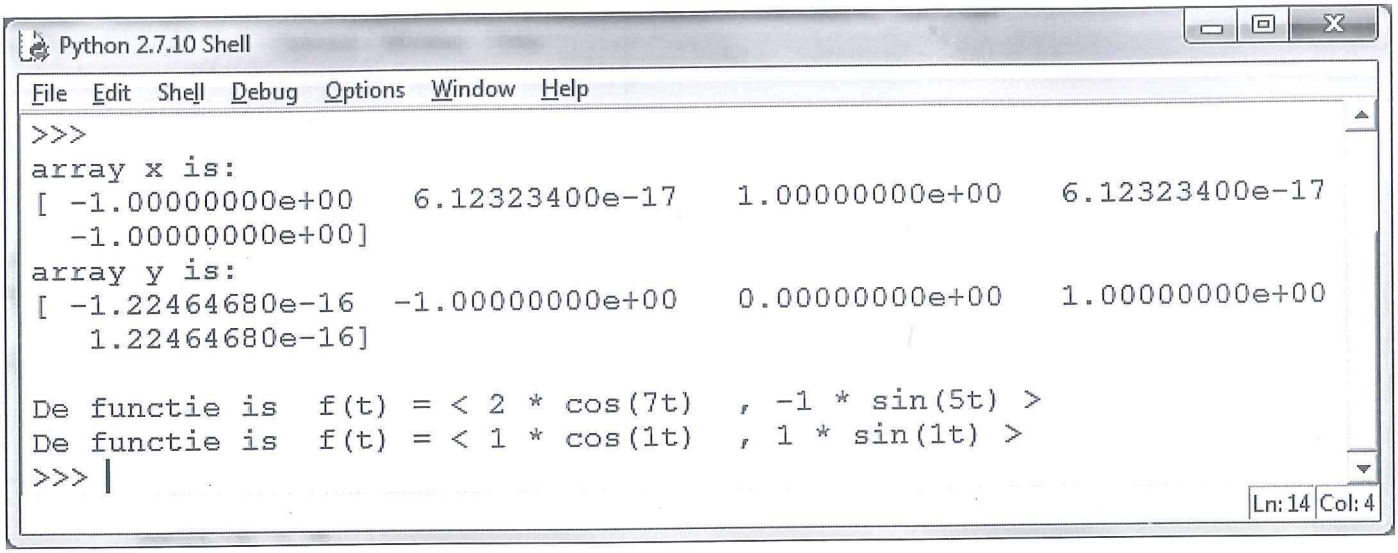
b) `generate_points():`
Deze methode geeft een array met x-coördinaten en een array met y-coördinaten terug behorend bij het Curve object waarvoor de methode wordt aangeroepen. De arrays bevatten in totaal n+1 x- en y-coördinaten van punten op de kromme voor waarden van t in het interval $[-\pi, \pi]$. De waarden van t verdelen het interval $[-\pi, \pi]$ in n subintervallen $[t_i, t_{i+1}]$ voor $i = 0, 1, 2, .., n-1$ van gelijke grootte.

c) `draw():`
Deze methode plot de grafiek van het `Curve` object waarvoor de methode `draw()` wordt aangeroepen (met n+1 punten). De schaal op de assen moet gelijk zijn zodat de cirkel ook echt als cirkel wordt getoond i.p.v. een ellips. In de titel van de plot moet de tekst worden getoond zoals in de plots op de volgende bladzijde.

Zorg ervoor dat de werking van de methoden die je implementeert **precies** overeenkomt met de beschrijving in het eerder gegeven klassendiagram. Test de methoden door runnen van het testprogramma en aanroepen van de te testen methoden in de Python shell na runnen van de klasse. Na implementatie van de methoden moet het testprogramma de volgende plots en uitvoer in de Python shell opleveren:







Afsluiten

Sluit alle .py bestanden. Controleer in IDLE door middel van Kies File | Open nog een keer extra of je in je directory **Homedrive(H:)** drie .py bestanden hebt staan met in de naam jouw naam en studienummer. **Deze drie bestanden vormen het werk dat je inlevert.**

Druk nu op het sluithokje rechts bovenin het IDLE window. Sluit op deze manier alle Editor en Python Shell windows van IDLE. Sluit ook de pdf van het boek. Kies tenslotte Start | Exit | OK. Je wordt nu uitgelogd.

AAN DE HAND VAN UITSLUITEND DE DRIE .PY BESTANDEN MET DE JUISTE NAMEN WORDT JE WERK BEOORDEELD.

INDIEN DE .PY BESTANDEN NIET OP HOMEDRIVE(H:) STAAN MAAR OP DE DESKTOP, DAN HEB JE GEEN WERK INGELEVERD.

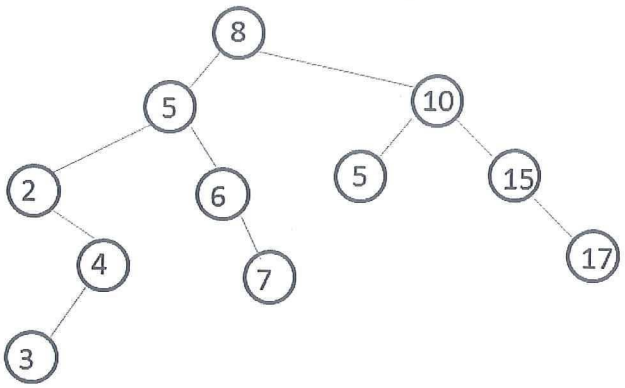
einde opgaven

Een binaire zoekboom is dus:

- een lege lijst of
- een lijst met 3 elementen:
  - een binaire zoekboom
  - een geheel getal
  - een binaire zoekboom

De functie `is_in(bst, n)` start bovenin de boom. Als de boom leeg is geeft de functie `False` terug. Als het getal dat gezocht wordt (het getal `n`) gelijk is aan het getal boven in de boom, dan geeft de functie `True` terug. Als het getal dat je zoekt kleiner is dan het getal boven in de boom dan zoek je verder in de linker tak, anders zoek je verder in de rechter tak. Deze keuze wordt op ieder niveau in de boom opnieuw uitgevoerd totdat beslist is dat het getal niet voorkomt of het getal gevonden is. Implementeer de functie **recursief**.

De binaire zoekboom waarmee getest wordt ziet er als volgt uit:



Na implementatie van de functies moet het testprogramma de volgende uitvoer geven: