Departement Informatica en Informatiekunde, Faculteit Bètawetenschappen, UU.
In elektronische vorm beschikbaar gemaakt door de TBC van A–Eskwadraat.
Het college INFOIMP werd in 2005/2006 gegeven door Jeroen Fokker.

Imperatief Programmeren, eerste deeltentamen (INFOIMP) 28 september 2005

Opgave 1 en 2 zijn tekst-vragen.
Houd het antwoord kort: een of twee zinnen per onderdeel kan al genoeg zijn.

Opgave 1 *(20 punten)*

a) Wat is het verschil tussen een *compiler* en een *interpreter*?

b) Wat is het verschil tussen een *expressie* en een *opdracht*?

c) De aanroep van een methode vormt soms een expressie, en soms een opdracht. Hoe kunt je dat zien aan de *header* van de methode?

d) Hoe kun je dat zien aan de *body* van de methode?

e) In veel klassen zit een methode met dezelfde naam als de klasse: de zogeheten *constructormethode*. Zo'n constructormethode wordt op een bijzondere manier aangeroepen. Hoe? En wat gebeurt er dan?

Opgave 2 *(20 punten)*

a) Waarvoor dient een declaratie?

b) Een declaratie kan op drie wezenlijk verschillende plekken in een programma staan.

Geef van elk van deze drie mogelijkheden aan:

- Op welke plek staat deze soort declaratie?
- Waar wordt een declaratie op deze plek voor gebruikt?
- Waar en hoe krijgt de variabele die zo wordt gedeclareerd zijn waarde?
  (in totaal dus drie keer drie (korte) antwoorden geven)

Opgave 3 en 4 vragen een stukje programma.
Kleine schrijffoutjes (hoofdletters, puntkomma's enz.) worden niet streng afgerekend, maar de elementen die de structuur van het programma bepalen (haakjes, accolades, aanhalingstekens enz.) zijn wel belangrijk. Schrijf die dus duidelijk en op de goede plaats op!
Het is toegestaan (maar niet nodig) om Java-constructies die (nog) niet zijn behandeld toch te gebruiken.

Opgave 3 *(30 punten)*

Een benadering van de logaritme van een (reëel) getal a kun je als volgt uitrekenen:
Bereken eerst $x = a - 1$. Bereken vervolgens

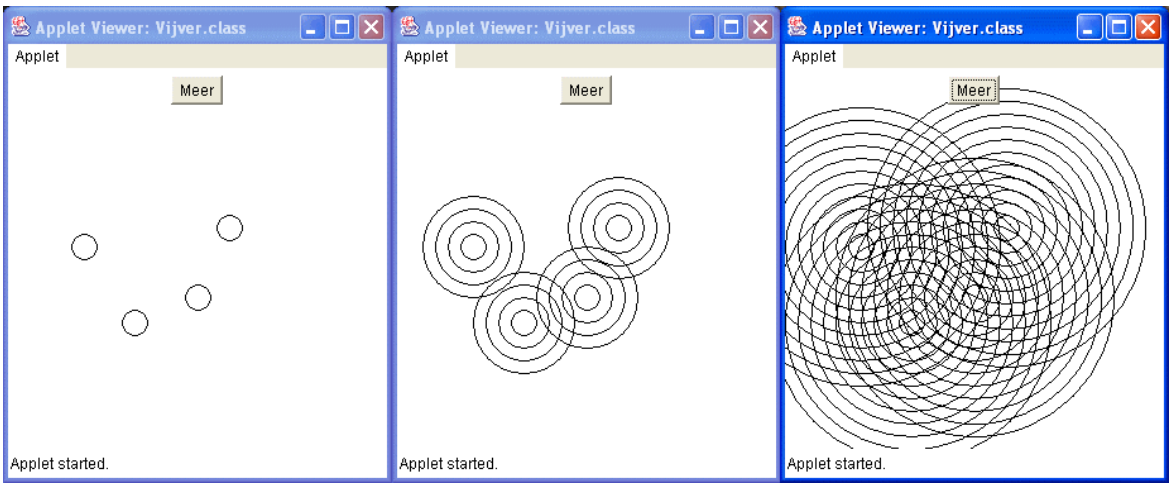
$$\frac{x}{1} - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \frac{x^5}{5} - \frac{x^6}{6} + \ldots$$

Let op: de termen worden dus afwisselend bij het resultaat opgeteld en afgetrokken. Het resultaat is ook weer een reëel getal.
(Deze formule werkt alleen als $0 < a < 2$, maar daarop hoef je niet te controleren).
Schrijf een methode `logaritme` die deze benadering door 20 van deze termen te sommeren, en dat als resultaat oplevert.

Het is hierbij niet toegestaan om de bestaande methodes, zoals `log`, uit de klasse `Math` te gebruiken. Ook is het niet toegestaan om alle 20 termen helemaal uit te schrijven.
Je mag wel (maar hoeft niet) extra hulp-methoden definiëren.

Opgave 4 *(30 punten)*

Schrijf een complete applet met de volgende werking. Je hoeft alleen de Java-file te schrijven, en je mag daarin de import-regels weglaten. De HTML-file is hieronder gegeven.
Op het scherm is een button zichtbaar, en vier cirkels. De cirkels stellen de kringen voor die vier in een vijver gegooide steentjes verspreiden. Het centrum van de vier cirkels bevindt zich respectievelijk op positie $(60, 140)$, $(100, 200)$, $(150, 180)$ en $(175, 125)$. Het eerste plaatje hieronder laat de beginsituatie zien.
Elke keer als de gebruiker op de knop drukt, breiden de kringen zich uit. In het tweede plaatje heeft de gebruiker driemaal op de knop gedrukt, en zijn er rond elk steentje dus vier kringen zichtbaar. In het derde plaatje heeft de gebruiker tienmaal op de knop gedrukt.



De HTML-file waarmee deze applet wordt gestart is als volgt:

```
<HTML>
  <APPLET code=Vijver.class width=300 height=300>
    <PARAM name=afstand value=10>
  </APPLET>
</HTML>
```

De waarde die in de PARAM genaamd afstand wordt gespecificeerd (in dit voorbeeld is dat 10) moet in de applet gebruikt worden als afstand tussen de kringen.
Het is in dit programma niet toegestaan om de code voor zo'n groep cirkels viermaal helemaal uit te schrijven. (Bespaar je de moeite om het toch te proberen: dit levert geen extra punten op). Gebruik in plaats daarvan een mechanisme dat in Java beschikbaar is om dit soort situaties compact te formuleren.