



Faculteit Economie en Bedrijfskunde

MET ANTWOORDEN

Op dit voorblad vindt u belangrijke informatie omtrent het tentamen.

Voordat u met het tentamen begint: lees dit voorblad!

Tentamen: Programmeren (235P05)

Tentamendatum & -tijd: vrijdag 10 juni 2011, 9.00 uur – 12.00 uur.

Duur van het tentamen: 3 uren.

U dient zich te legitimeren met uw UvA legitimatiebewijs met foto of uw UvA collegekaart samen met paspoort of rijbewijs of een ander geldig legitimatiebewijs (voor studenten) met foto.

Vermeld uw naam en studentnummer op elk apart blad.

Waarschuwing tegen fraude: Fraudeer niet! Bij fraude staat u als maximale straf de uitsluiting van alle tentamens voor een jaar te wachten.

Uw mobiele telefoon dient uitgeschakeld en opgeborgen in uw tas te zijn. Uw tas dient gesloten links van uw tafel op de vloer te zijn geplaatst.

Tijdens het tentamen is toiletbezoek niet toegestaan (tenzij het bij wijze van uitzondering door de hoofdsurveillant uitdrukkelijk wordt toegestaan).

Toegestane hulpmiddelen: potlood, pen, gum, liniaal. NIET TOEGESTAAN: rekenmachine!

Specifieke toelichtingen voor dit tentamen: Het tentamen bevat 7 opgaven (4 pagina's inclusief deze). Er kan in totaal 100 punten behaald worden. Het cijfer wordt bepaald door het behaalde punten door 10 te delen.

Licht je antwoorden zoveel mogelijk toe. Tenzij anders is vermeld, mag u altijd gebruik maken van de voorgedefinieerde functies en procedures uit Math.

De uitslag worden uiterlijk 13 werkdagen na de tentamendatum bekend gemaakt.

Tentamen inzage: Uiteraard! Gaarne een afspraak maken via e-mail: J.A.M.Hontelez@uva.nl

Succes!

Opgave 1 (20 punten)

Gegeven is het volgende Pascal-programma:

```
program Opgave_1;
{$APPTYPE CONSOLE}
uses SysUtils;

var a,b,n : integer;

function f1(var a,b : integer): integer;
begin
  a:=a+b;
  b:=b+a;
  writeln(a:4,b:4);
  f1:=a+b;
end;

function f2(var a,b : integer): integer;
var s:integer;
begin
  n:=1;
  a:=f1(b,a);
  while n<3 do
  begin
    n:=n+1;
    b:=b+n*a;
  end;
  writeln(a:4,b:4);
  f2:=f1(a,b);
  writeln(a:4,b:4);
end;

(* hoofdprogramma *)
begin
  a:=2; b:=3;

  n:=1;
  repeat
    b:=f2(b,a);
    n:=n+1;
  until n>3;
  writeln(a:4,b:4);

  readln;
end.
```

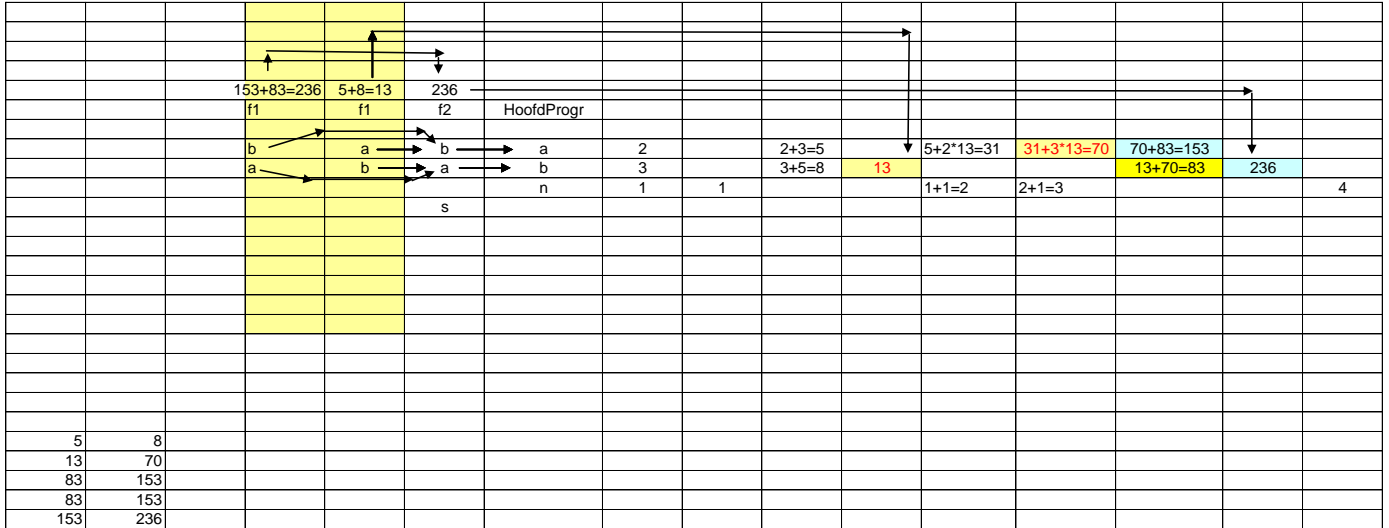
- 1a) Wat wordt er afgedrukt (denk aan de formattering en uitvoer in de functies zelf)? Geef hierbij uiteraard uitleg.
- 1b) Hetzelfde als vraag 1a (dus geef de uitvoer), maar nu met alle var's weggelaten in de formele parameterdeclaratie van de functies, d.w.z.:

```
function f1(a,b : integer): integer;
function f2(a,b : integer): integer;
```

Antwoord Opgave 1:

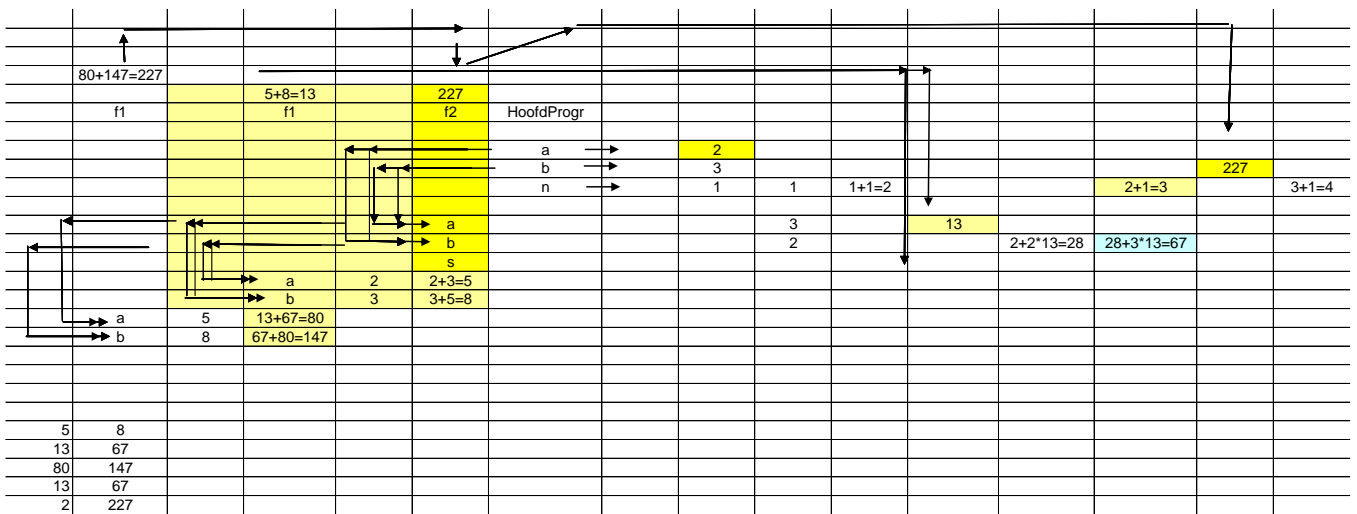
a (met var):

5 8
13 70
83 153
83 153
153 236



b (zonder var):

5 8
13 67
80 147
13 67
2 227



Opgave 2 (20 punten):

Omdat: $\cos(0)=1$ en $\cos\left(\frac{\pi}{2}\right)=0$ en $\cos(x)$ een continue en dalende functie is op het interval $\left(0, \frac{\pi}{2}\right]$,

is er precies één waarde x tussen 0 en $\frac{\pi}{2}$ waarvoor geldt: $\cos(x) = x$.

Gevraagd wordt nu een algoritme te maken dat deze waarde x bepaalt:

Bepaal x zodanig dat : $|x - \cos(x)| < \varepsilon$, $x \in \left[0, \frac{\pi}{2}\right]$, $\varepsilon = 0.000001$.

Hint: begin met het interval $[un, up] = \left[0, \frac{\pi}{2}\right]$ en verklein dit telkens z.d.d. de te bepalen x in het

interval $[un, up]$ blijft liggen. Doe dit zolang totdat $|x - \cos(x)|$ klein genoeg is geworden.

Opgemerkt zij, dat de functie $\cos(x)$ bekend is in Pascal (met unit Math) en dus gewoon gebruikt kan worden.

Antwoord opgave 2:

Met $un = 0$ en $up = \frac{\pi}{2}$ als start, weten we dus dat de waarde $x = \cos(x)$ in ieder geval daartussen ligt.

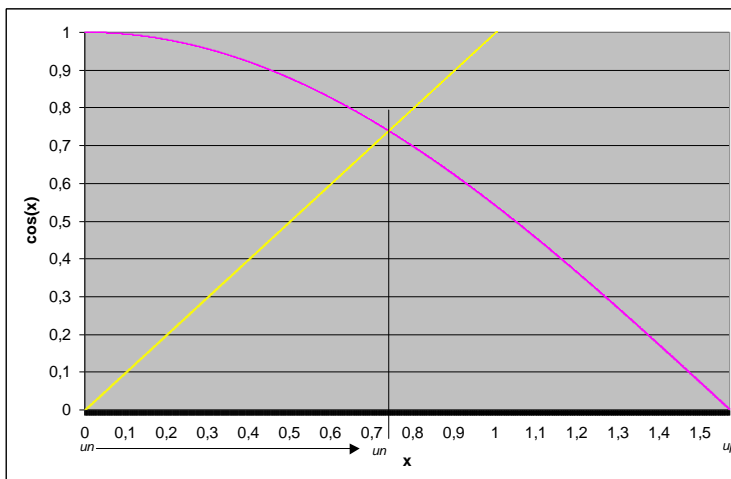
Verder geldt dus: $\cos(un) = \cos(0) = 1$ en dus $un < \cos(un)$ en $up > \cos(up)$.

We willen nu het interval verkleinen: neem daartoe: het midden van het interval: $m = \frac{up + un}{2}$ en ga na of $m < \cos(m)$, danwel $m > \cos(m)$:

Indien $m < \cos(m)$ pas dan ligt x in het interval $[m, up]$ en kunnen de ondergrens gelijk trekken aan m : $un := m$. Evenzo:

Indien $m > \cos(m)$ pas dan: $x \in [un, m]$ en pas dus de bovengrens aan: $up := m$

We hebben hiermee het interval waarin x ligt, gehalveerd. Dit moeten we net zo lang blijven doen, totdat $|\cos(x) - x| < \varepsilon$.



Hiermee is het algoritme bepaald:

$un := 0; up := \frac{\pi}{2};$

$x := \frac{un + up}{2};$

Repeat

If $x > \cos(x)$ **then** $up := x$ **else** $un := x$;

$x := \frac{un + up}{2};$

Until $|\cos(x) - x| < \varepsilon$

Of, in Pascal-taal (is niet nodig voor het tentamen):

```
un:=0; up:=pi/2;
```

```
repeat
```

```
  if (up+un)/2 > cos((up+un)/2)
```

```
  then up:=(up+un)/2
```

```
  else un:=(up+un)/2;
```

```
  x:=(up+un)/2;
```

```
until abs(cos(x)-x) < eps;
```

Opgave 3 (15 punten)

Gegeven is het volgende lineair onafhankelijke stelsel vergelijkingen:

$$\begin{pmatrix} a_{11} & 0 & \cdots & 0 \\ a_{21} & a_{22} & 0 & 0 \\ \vdots & a_{32} & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn-1} & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

Schrijf een procedure:

```
Procedure AxIsb (A:MatrixNN; b:VectorN; var x: vectorN);
```

die de onbekenden x_1, x_2, \dots, x_n berekent.

De definitie van de types die in de formele parameterlijst zijn gebruikt, zijn:

```
MatrixNN = array[1..n, 1..n] of real;  
VectorN = array[1..n] of real;
```

Antwoord opgave 3:

De oplossing zullen we eerst “wiskundig” uitschrijven:

Voor de j^e rij geldt: $\sum_{k=1}^j a_{jk}x_k = b_j \quad j=1,2,\dots,n$

Dus is $a_{jj}x_j = b_j - \sum_{k=1}^{j-1} a_{jk}x_k \quad j=1,2,\dots,n$, ofwel:

$$x_j = \frac{b_j - \sum_{k=1}^{j-1} a_{jk}x_k}{a_{jj}} \quad j=1,2,\dots,n$$

Merk op dat x_j te berekenen is als x_1, x_2, \dots, x_{j-1} bekend zijn!

We moeten dus met $j=1$ starten en zo doorgaan naar $j=n$.

Het algoritme voor bepaling x_j is verder eenvoudig: bepaal eerst de som $\sum_{k=1}^{j-1} a_{jk}x_k$:

som:=0;

for $k:=1$ **to** $j-1$ **do** som:=som + $a_{jk}x_k$;

en vervolgens: $x_j = \frac{b_j - \text{som}}{a_{jj}}$

en dit moet voor elke $j=1,2,\dots,n$, te beginnen met $j=1$ uitgevoerd worden. Hiermee is de procedure eenvoudig te maken:

```
Procedure AxIsb(A:MatrixNN; b:VectorN; var x: vectorN);
var i, j, k : integer;
    som      : real;

begin
  for j:=1 to n do
    begin
      som:=0;
      for k:=1 to j-1 do som:=som+A[j, k]*x[k];
      x[j] := (b[j]-som)/A[j, j];
    end;
  end; (* AxIsb *)
```

Opgave 4 (15 punten)

Gegeven is een rij b_i , $i=1,2,3,\dots,N$ van gehele getallen die niet-dalend is: $b_i \leq b_{i+1}$, $i=1,2,3,\dots,N-1$

gevraagd wordt nu te bepalen welke waarde het vaakst voorkomt en hoe vaak.

(Dit wordt ook wel het plateau-probleem genoemd, omdat een rijtje gelijke waarden kan worden gezien als een 'plateau' in een verder stijgende omgeving. Het eindantwoord kan dan ook geformuleerd worden in de trant van: 'p' is de lengte (hoe vaak) en 'q' de hoogte (de waarde) van het langste plateau van $b[1..N]$.)

Maak een procedure:

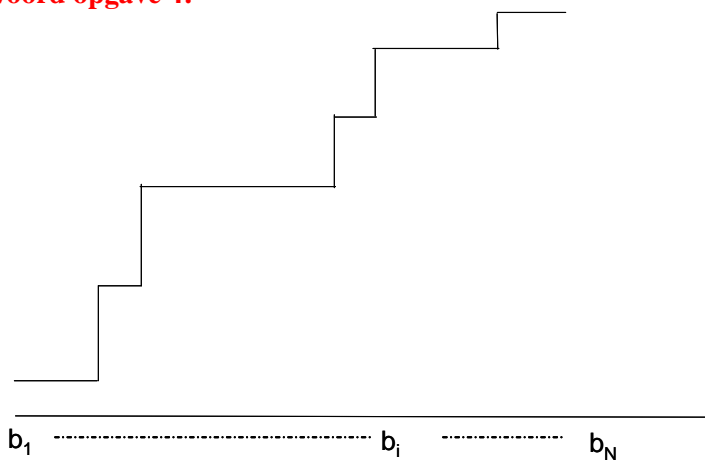
```
procedure PlateauBepalen (b:rijgetallen; var lengte,hoogte: integer);
```

die de lengte en hoogte bepaalt van het langste plateau van de rij niet-dalende getallen $b[1..N]$.
Ofwel: "hoogte" is de waarde die het vaakst voorkomt in de rij $b[1..N]$ en deze waarde komt "lengte" keer voor in de rij $b[1..N]$.

Hierbij is:

type rijgetallen = **array**[1..N] of **integer**; (en N is als constante gedefinieerd).

Antwoord opgave 4:



Omdat de rij b_1, b_2, \dots, b_N gesorteerd is, kunnen we volgens het bovenstaande plaatje de waarden b_i aflopen van links naar rechts en bijhouden hoe lang het plateau is en bijhouden wat het langste plateau tot nu toe is:

- De lengte, resp. hoogte van het tot nu toe langste plateau geven we aan met *lengte*, resp. *hoogte*.
- De lengte en hoogte van het huidige plateau geven (waartoe b_i behoort) geven we aan met *ll*, resp. *hh*.

Start/initialisatie: We starten met het eerste getal b_1 van de rij:

$i := 1$;

Dan is dus: $ll := 1$; $hh := b_i$;

En: $lengte := ll$; $hoogte := hh$;

$i := i + 1$:

Als we nu naar het volgende getal in de rij gaan kijken: $i := i + 1$; dan kunnen de volgende situaties zich voordoen:

1. $b_i = hh$: in dat geval wordt de lengte van het huidige plateau 1 groter: $ll := ll + 1$;
2. $b_i > hh$: in dat geval beginnen we aan een nieuw plateau: $ll := 1$; $hh := b_i$;

Hierna moeten we nagaan of het huidige plateau groter is dan het grootste tot nu toe:

- Indien $ll > lengte$ dan: $lengte := ll$; $hoogte := hh$; Uiteraard hoeft dat alleen in situatie 1: als $b_i = hh$.

Dit moet herhaald worden tot we alle getallen hebben bekeken, dus tot en met b_N .

Hiermee is het algoritme duidelijk (in pseudo code):

```
 $i := 1$ ;  
 $ll := 1$ ;  $hh := b_i$ ;  
 $lengte := ll$ ;  $hoogte := hh$ ;  
while  $i < N$  do  
begin  
  if  $b_i = hh$   
  then  
     $ll := ll + 1$ ;  
    If  $ll > lengte$  then begin  $lengte := ll$ ;  $hoogte := hh$ ; end;  
  else  $ll := 1$ ;  $hh := b_i$ ;  
end;
```

In Pascal-taal wordt dat dan (op tentamen niet nodig zo precies in Pascal te formuleren – mag wel uiteraard):

```
 $i := 1$ ;  
 $lengte := 1$ ;  $hoogte := b[i]$ ;  
 $ll := lengte$ ;  $hh := hoogte$ ;  
while  $i < N$  do  
begin  
   $i := i + 1$ ;  
  if  $b[i] > hh$   
  then begin  
     $ll := 1$ ;  $hh := b[i]$ ;  
  end  
  else begin  
     $ll := ll + 1$ ;  
    if  $ll > lengte$  then begin  $lengte := ll$ ;  $hoogte := hh$ ; end;  
  end;  
end;
```

Opgave 5 (10 punten)

Pas het algoritme van Opgave 4 aan voor het geval de rij getallen $b[1..N]$ niet niet-dalend is, maar een willekeurige rij van gehele getallen is; dus bepaal de waarde die het vaakst voorkomt ('hoogte') in de rij en geef ook aan hoe vaak ('lengte').

Antwoord opgave 5:

Het enige wat aangepast moet worden is dat voorafgaand aan het algoritme zoals dat in opgave 4 is aangegeven, de rij gesorteerd moet worden van laag naar hoog. Dit kan bijvoorbeeld met het Bubblesort-algoritme:

```
for i:=1 to N-1 do
  for j:=1 to N-i do
    if  $b_j > b_{j+1}$  then
      begin
         $h := b_j; b_j := b_{j+1}; b_{j+1} := h;$ 
      end;
```

Opgave 6 (5 punten)

Schrijf een pascal-functie

```
function middel(x,y,z:real):real;
```

die het middelste in grootte van de drie verschillende getallen x, y en z teruggeeft.

Voorbeeld: bij 3,7,2 is 3 het resultaat.

Antwoord opgave 6:

```
function middel(x,y,z:real):real;
begin
  if ((x<y) and (y<z)) or ((z<y) and (y<x)) then middel:=y else
  if ((y<x) and (x<z)) or ((z<x) and (x<y)) then middel:=x else middel:=z;
end;
```

Opgave 7 (15 punten)

Geef een algoritme dat toetst of een positief getal x perfect is. Dit is het geval als de som van de delers van dat getal gelijk is aan dit getal zelf. Voor deze opgave worden de delers van een positief geheel getal gedefinieerd als de positieve getallen waardoor het getal een geheel aantal malen deelbaar is, exclusief het getal zelf.

De delers van bijvoorbeeld 30 zijn dan: 1, 2, 3, 5, 6, 10 en 15. Voorbeelden van perfecte getallen zijn 6 (delers: 1, 2 en 3) en 28 (delers: 1, 2, 4, 7 en 14).

Antwoord opgave 7:

Uitleg in pseudocode/invariant:

Bepaling of een getal i een deler is van x :

De som van de delers moet worden bepaald, dus:
som:=som+i;

if (x mod i =0) **then** i is deler van x

for i:=1 **to** (x div 2) **do** **if** (i=deler) **then**

Hiermee is het algoritme bepaald:

som:=0;

for i:=1 **to** (x div 2) **do**

do **if** (x mod i =0) **then** som:=som+i;

controle of x perfect is:

if som=x **then** x is perfect.

Dit leidt dan tot onderstaand programma (programma is niet expliciet gevraagd):

```
program Vraag7;
{$APPTYPE CONSOLE}
uses SysUtils;
var x,som,i:integer;
begin
  Write('Geef getal: ');
  Readln(x);
  som:=0;
  for i:=1 to (x div 2) do
    if x MOD i=0 then som:=som+i;

  if som=x then Writeln('Dit getal is perfect.')
    else Writeln('Dit getal is niet perfect.');
```

 Readln;

end.