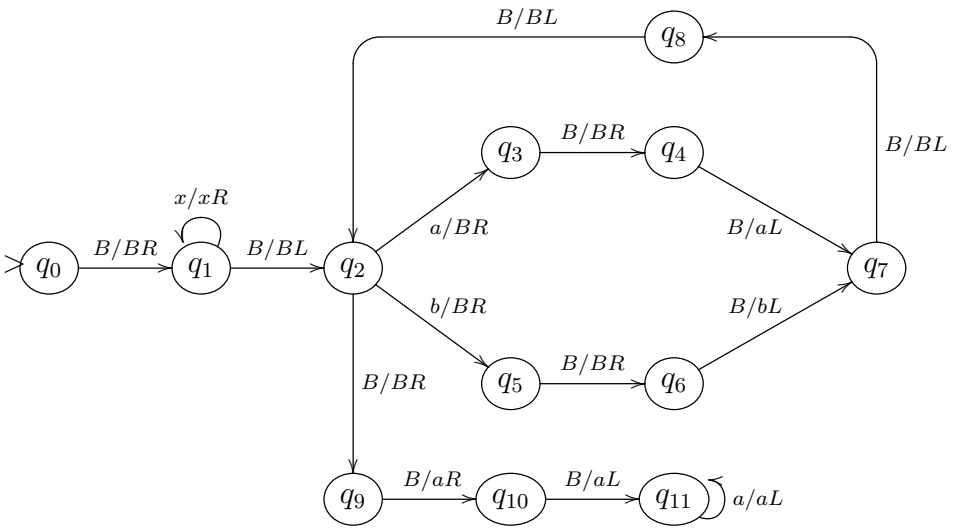
Berekenbaarheid 2015
Uitwerkingen Tentamen
5 november 2015

1. Definieer een standaard Turing-machine M_1 met input alfabet $\Sigma = \{a, b\}$ die twee a's voor zijn input plakt, dus met $M_1(w) = aaw$ voor alle $w \in \{a, b\}^*$.



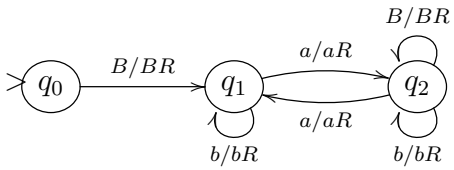
$x \in \{a, b\}$

2. Definieer een standaard Turing-machine M_2 met input alfabet $\Sigma = \{a, b\}$ die de taal

$$L_2 = \{w \in \{a, b\}^* \mid w \text{ bevat een even aantal } a\text{'s}\}$$

herkent *door stoppen*.

Merk op dat nul ook een even getal is.

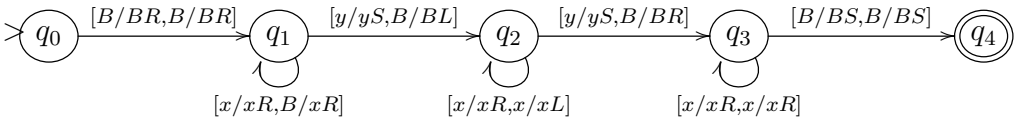


3. Definieer een nondeterministische twee-tape Turing-machine M_3 die de taal

$$L_3 = \{uu^R u \mid u \in \{a,b\}^*\}$$

herkent. Zorg er voor dat een correcte input van lengte n wordt herkend in hoogstens $n+4$ stappen.

Merk op dat het lege woord ook in deze taal zit.



$x \in \{a,b\}$
$y \in \{a,b,B\}$

Het aantal stappen om $uu^R u$ te herkennen is precies (ieder lusje duurt $|u|$ stappen)

$$3|u| + 4 = |uu^R u| + 4$$

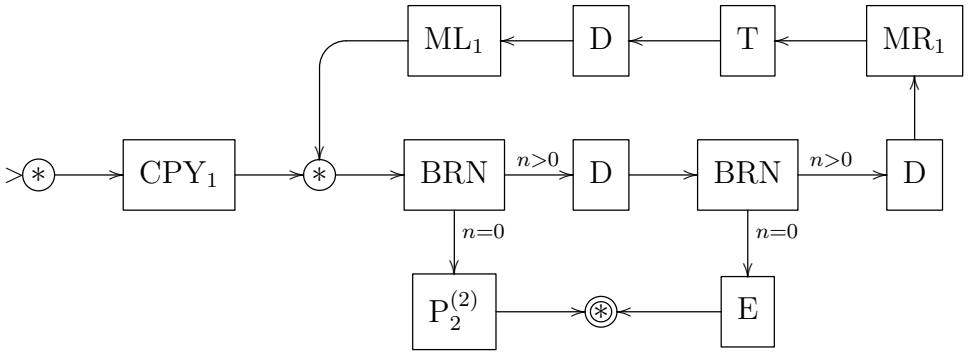
4. Definieer een numerieke Turing-machine M_4 die de functie

$$f_4(n) = \begin{cases} n/2 & \text{als } n \text{ even is} \\ \uparrow & \text{als } n \text{ oneven is} \end{cases}$$

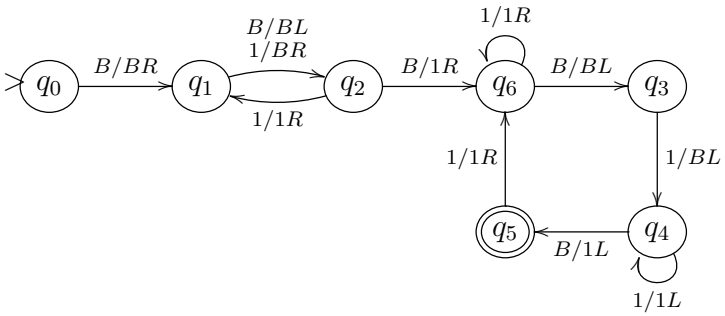
uitrekent. Zorg ervoor dat de machine aan de regels van een macro voldoet.

Je mag in je machine de macro's op pagina 10 gebruiken. Je mag in de machine ook expliciet toestanden en transities opnemen, dus je hoeft niet alles uitsluitend met macro's te doen.

Een oplossing met alleen macro's:



Een oplossing met alleen toestanden:



5. Heeft de code $R(U)$ van een universele Turing-machine U altijd

$$001011101$$

als substring? Verklaar je antwoord.

Zie pagina 11 voor een relevant stukje over codes van Turing-machines uit het boek van Sudkamp.

Ja, $R(U)$ heeft altijd de substring 001011101.

Een universele Turing machine termineert niet voor iedere input (bijv. als zijn input niet met een correcte code begint, termineert hij niet).

Dat betekent dat er vanuit de begintoestand q_0 een overgang moet zijn die hoort bij het symbool waar de machine op staat als hij wordt aangezet, namelijk B, en dat die overgang een stap naar rechts moet doen (want anders termineert hij voor iedere input abnormaal, wat als non-terminatie geldt, en dat is bij U ook niet het geval).

Deze transitie is dus van de vorm

$$\delta(q_0, B) = [q_i, x, R]$$

voor zeker symbool $x \in \{B, 0, 1\}$ (dat hoeft niet B te zijn). De code van deze transitie is

$$10111011^i011^x011$$

(waarbij als $x = B$ dan $1^x = 11$), en deze begint dus met de string

$$1011101$$

Deze code wordt of voorafgegaan door de drie nullen aan het begin van $R(U)$ of door de twee nullen waardoor hij van de code van de vorige transitie wordt gescheiden. In beide gevallen bevat $R(U)$ dus de substring

$$00\ 1011101$$

en dat was wat we moesten laten zien.

Merk op dat je echt moet gebruiken dat het hier om U gaat, er zijn ook codes van Turing machines zonder deze substring. Evenwel zullen die machines dan altijd direct termineren voor iedere input.

6. (a) Het probleem P_6 is gegeven als:

   input: een code $R(M)$ van een Turing-machine M
   vraag: stopt de Turing-machine M met als input 001011101?

   Laat zien dat dit probleem onbeslisbaar is.

   De vraag is equivalent aan de vraag of $001011101 \in L(M)$. Dit is een eigenschap van de taal van de machine die de input van het probleem is, dus als we laten zien dat het probleem niet-triviaal is volgt uit de stelling van Rice dat het probleem onbeslisbaar is.

   Het probleem geeft antwoord 'ja' voor de machine

   $$M_{\text{ja}}: \quad > (q_0)$$

   want die stopt voor *iedere* input, dus zeker voor input 001011101, en 'nee' voor de machine

   $$M_{\text{nee}}: \quad > (q_0) \; {}_{x/xR} \qquad x \in \Gamma$$

   want die stopt voor geen enkele input, dus ook niet voor de input 001011101. Omdat beide antwoorden kunnen voorkomen, is het probleem niet-triviaal.

   (b) Leg uit of dit probleem onder de stelling van Rice valt of niet.

   Ja, dit valt onder de stelling van Rice, zie de uitleg hierboven.

7. (a) Het probleem P_7 is gegeven als:

   input: een code $R(M)$ van een Turing-machine M
   vraag: bevat de taal $L(M)$ meer woorden dan er toestanden zijn in M?

Merk op dat als $L(M)$ oneindig is, het antwoord op deze vraag 'ja' zal zijn.

Laat zien dat dit probleem onbeslisbaar is.

(Sommige studenten merkten op dat aan de code van een machine niet het aantal toestanden is te zien, omdat toestanden zonder transitie niet in de code terug te vinden zijn. De bedoeling van de opgave was dat er met 'toestanden in M' de *minimale* verzameling toestanden is bedoeld die bij de code past. Ook als er helemaal geen transities met q_0 zijn zit daar wel altijd ook q_0 in.)

Dit is onbeslisbaar, want het blank tape probleem B reduceert naar dit probleem P_7.

Hiertoe moeten we bij iedere code $R(M)$ als input van B een code $R(M')$ als input van P_7 maken zodat de volgende equivalentie geldt:

$M(\lambda)\downarrow \iff L(M')$ bevat meer woorden dan M' toestanden heeft

Een constructie die deze eigenschap heeft, is door voor M' te nemen:

- wis tape
- doe M

Dat bovenstaande equivalentie voor iedere M geldt is makkelijk te zien.

Als M stopt met als input de lege tape, stopt M' op iedere input, dus is $L(M') = \{0,1\}^*$ en dus is dan $L(M')$ oneindig en bevat meer woorden dan er toestanden zijn in M'.

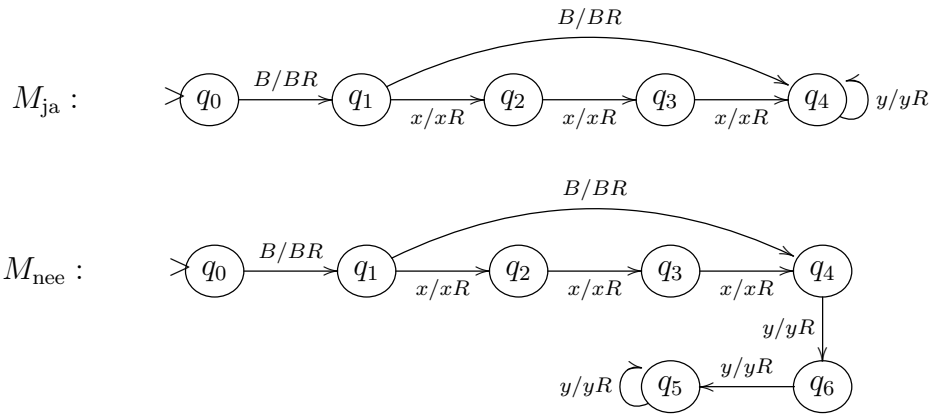
Als M niet stopt met als input de lege tape, stopt M' voor geen enkele input, dus is dan $L(M') = \emptyset$ en dus bevat $L(M')$ niet méér woorden dan er toestanden zijn in M'. (Zelfs als er nul toestanden zouden zijn, wat dus niet kan, zou dat nog niet het geval zijn).

(b) Leg uit of dit probleem onder de stelling van Rice valt of niet.

Nee, dit valt niet onder de stelling van Rice.

Er zijn machines M_{ja} en M_{nee} die dezelfde taal herkennen, maar waarvoor het antwoord van dit probleem verschillend is, dus het probleem vraagt niet naar een eigenschap van de taal van de machine die de input van het probleem is.

Het was niet nodig voor een correct antwoord om expliciet zulke M_{ja} en M_{nee} te geven, maar voor de volledigheid van deze uitwerking hier even wel:



Beide met $x \in \{0, 1\}$ en $y \in \{B, 0, 1\}$.

Dan hebben we

$$L(M_{\text{ja}}) = L(M_{\text{nee}}) = \{0, 1, 00, 01, 10, 11\}$$

dus beide machines herkennen een taal met 6 woorden, maar M_{ja} heeft 5 toestanden, terwijl M_{nee} 7 toestanden heeft.

8. (a) Geef numerieke functies f_8 en g_8 met $f_8 \circ g_8 = e$ en $g_8 \circ f_8 \neq e$. Hierin is e de 'lege' functie, die voor geen enkele input gedefinieerd is. Verklaar je antwoord.

Neem bijvoorbeeld:

$$f_8(x) = \begin{cases} \uparrow & \text{als } x = 0 \\ 0 & \text{als } x > 0 \end{cases}$$
$$g_8(x) = 0$$

In dit geval is $f_8 \circ g_8 = e$ maar $g_8 \circ f_8 = f_8 \neq e$.

(b) Geef de ariteiten van f_8 en g_8.

Beide functies hebben één argument, dus beide hebben ariteit 1.

9. (a) Gegeven de recursievergelijkingen voor een recursieve definitie van een numerieke functie f_9:

$$f_9(x, 0) = 1$$
$$f_9(x, y + 1) = f_9(x, y) \cdot (x + y)$$

Met deze functie kun je bijvoorbeeld definiëren:

$$x! = f_9(1, x)$$
$$\binom{x}{y} = \frac{f_9(x - y + 1, y)}{f_9(1, y)}$$

Bereken de waarde van $f_9(3, 4)$.

We hebben

$$f_9(x, y) = x \cdot (x + 1) \cdot (x + 2) \cdot \ldots \cdot (x + y - 1)$$

dus

$$f_9(3, 4) = 3 \cdot 4 \cdot 5 \cdot 6 = 360$$

(b) Geef numerieke functies g_9 en h_9 met

$$f_9 = \textsf{primrec}(g_9, h_9)$$

en die corresponderen met de recursievergelijkingen.

In de definitie van deze twee functies kun je gebruik maken van de functies op pagina 11.

Het recursieschema is

$$f_9(x, 0) = g_9(x)$$
$$f_9(x, y + 1) = h_9(x, y, f_9(x, y))$$

Dus de recursievergelijkingen corresponderen met dit schema met

$$g_9(x) = 1$$
$$h_9(x, y, w) = w \cdot (x + y)$$

(c) Geef de ariteiten van f_9, g_9 en h_9.
Pas op dat je van geen van deze drie functies vergeet de ariteit op te schrijven.
f_9 heeft ariteit 2, g_9 heeft ariteit 1, h_9 heeft ariteit 3.

(d) Schrijf deze functies g_9 en h_9 expliciet als compositie van functies uit de lijst op pagina 11.
Er geldt:

$$g_9 = c_1^{(1)}$$
$$h_9 = \mathsf{mult} \circ (p_3^{(3)}, \mathsf{add} \circ (p_1^{(3)}, p_2^{(3)}))$$

En we hebben dus:

$$f_9 = \mathbf{primrec}(c_1^{(1)}, \mathsf{mult} \circ (p_3^{(3)}, \mathsf{add} \circ (p_1^{(3)}, p_2^{(3)})))$$

10. (a) De Carmichael-functie f_{10} berekent voor iedere positief natuurlijk getal x de kleinste positieve n zodat

$$a^n \equiv 1 \mod x$$

voor *alle* a die geen gemeenschappelijke delers hebben met x.
Zo is bijvoorbeeld $f_{10}(10) = 4$, want

$$\begin{aligned} 1^4 &= 1 \equiv 1 \mod 10 \\ 3^4 &= 81 \equiv 1 \mod 10 \\ 7^4 &= 2401 \equiv 1 \mod 10 \\ 9^4 &= 6561 \equiv 1 \mod 10 \end{aligned}$$

maar

$$3^3 = 27 \not\equiv 1 \mod 10$$

We definiëren $f_{10}(0)\uparrow$.
Laat zien dat deze functie f_{10} een μ-recursieve functie is.
Je kunt de Carmichael-functie schrijven als

$$f_{10}(x) = \mu n.\Big(x \cdot n \cdot \prod_{a=2}^{x} \big[\big(\sum_{y=2}^{a} \mathsf{divides}(a,y) \cdot \mathsf{divides}(x,y)\big) + \mathsf{eq}(\mathsf{rem}(\mathsf{exp}(a,n),x),1)\big]\Big)$$

Uit deze schrijfwijze blijkt dat f_{10} μ-recursief is.

(De $x \cdot n$ aan het begin zijn nodig om te zorgen dat $f_{10}(0)\uparrow$, en om te voorkomen dat de μn al meteen stopt bij $n = 0$.)

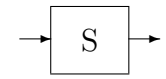
(b) Is f_{10} ook primitief recursief? Verklaar je antwoord.

Nee. Deze functie is niet totaal want $f_{10}(0)\uparrow$, en primitief recursieve functies zijn altijd totaal.

(Als we hadden gedefinieerd dat $f_{10}(0)\downarrow$, dan was de functie wel primitief recursief geweest, maar om dat in te zien moet je Eulers totiëntstelling kennen.)

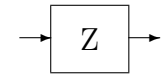
Je mag voor beide onderdelen van deze opgave gebruiken dat de functies op pagina 11 primitief recursief zijn.

Macro's voor Turing-machines voor numerieke berekeningen



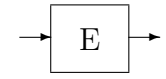
$\underset{\uparrow}{B}\,\overline{n}\quad BB$

$\underset{\uparrow}{B}\,\overline{n+1}\,B$



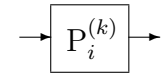
$\underset{\uparrow}{B}\,\overline{n}\quad B$

$\underset{\uparrow}{B}\,\overline{0}\ldots B$



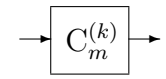
$\underset{\uparrow}{B}\,\overline{n}\,B$

termineert niet



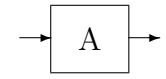
$\underset{\uparrow}{B}\,\overline{n_1}\,B\ldots B\,\overline{n_k}\,B$

$\underset{\uparrow}{B}\,\overline{n_i}\;B\ldots\quad\ldots B$



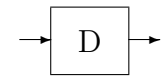
$\underset{\uparrow}{B}\,\overline{n_1}\,B\ldots B\,\overline{n_k}\,B$

$\underset{\uparrow}{B}\,\overline{m}\;B\ldots\quad\ldots B$



$\underset{\uparrow}{B}\,\overline{n}\,B\,\overline{m}\quad B$

$\underset{\uparrow}{B}\,\overline{n+m}\,BB$



$\underset{\uparrow}{B}\,\overline{n}\qquad B$

$\underset{\uparrow}{B}\,\overline{n \dot{-} 1}\ldots B$

$\underset{\uparrow}{B}\,\overline{n}\,B\,\overline{m}\quad B$

$\underset{\uparrow}{B}\,\overline{n \dot{-} m}\ldots B$

$\underset{\uparrow}{B}\,\overline{n}\,B\,\overline{m}\,B\ldots$

$\underset{\uparrow}{B}\,\overline{n \cdot m}\,B\ldots$

$\underset{\uparrow}{B}\,\overline{n_1}\,B\ldots B\,\overline{n_k}\,B$

$B\,\overline{n_1}\,B\ldots B\,\overline{n_k}\,\underset{\uparrow}{B}$

$B\,\overline{n_1}\,B\ldots B\,\overline{n_k}\,\underset{\uparrow}{B}$

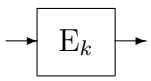
$\underset{\uparrow}{B}\,\overline{n_1}\,B\ldots B\,\overline{n_k}\,B$

$\underset{\uparrow}{B}B^i\,\overline{n}\,B\qquad (i \geq 0)$

$B^i\,\underset{\uparrow}{B}\,\overline{n}\,B$

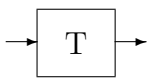
$B\,\overline{n}\,B^i\underset{\uparrow}{B}\qquad (i \geq 0)$

$\underset{\uparrow}{B}\,\overline{n}\,B^iB$



$\underset{\uparrow}{B}\,\overline{n_1}\,B\ldots B\,\overline{n_k}\,B$

$\underset{\uparrow}{B}\ldots\qquad\ldots B$



$\underset{\uparrow}{B}B^i\,\overline{n}\,B\qquad (i \geq 0)$

$\underset{\uparrow}{B}\,\overline{n}\,B^iB$

$\underset{\uparrow}{B}\,\overline{n}\,B\,\overline{m}\,B$

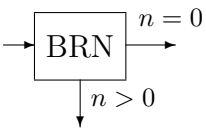
$\underset{\uparrow}{B}\,\overline{m}\,B\,\overline{n}\,B$

$\underset{\uparrow}{B}\,\overline{n_1}\,B\ldots B\,\overline{n_k}\,B\ldots\qquad\ldots B$

$\underset{\uparrow}{B}\,\overline{n_1}\,B\ldots B\,\overline{n_k}\,B\,\overline{n_1}\,B\ldots B\,\overline{n_k}\,B$

$\underset{\uparrow}{B}\,\overline{n_1}\,B\ldots B\,\overline{n_k}\,B\,\overline{n_{k+1}}\,B\ldots B\,\overline{n_{k+i}}\,B\ldots\qquad\ldots B$

$\underset{\uparrow}{B}\,\overline{n_1}\,B\ldots B\,\overline{n_k}\,B\,\overline{n_{k+1}}\,B\ldots B\,\overline{n_{k+i}}\,B\,\overline{n_1}\,B\ldots B\,\overline{n_k}\,B$



$\underset{\uparrow}{B}\,\overline{n}\,B$

$\underset{\uparrow}{B}\,\overline{n}\,B$

Codering van transities

Symbol	Encoding
0	1
1	11
B	111
q_0	1
q_1	11
$\vdots$	$\vdots$
q_n	1^{n+1}
L	1
R	11

Let $en(x)$ denote the encoding of a symbol x. A transition $\delta(q_i, x) = [q_j, y, d]$ is encoded by the string

$$en(q_i)0en(x)0en(q_j)0en(y)0en(d).$$

Primitief recursieve functies

$$\begin{aligned}
\mathsf{id}(x) &= x \\
z(x) &= 0 \\
s(x) &= x + 1 \\
p_i^{(k)}(x_1, \ldots, x_k) &= x_i \\
c_n^{(k)}(x_1, \ldots, x_k) &= n
\end{aligned}$$

$$\begin{aligned}
\mathsf{pred}(y) &= y \dot{-} 1 \\
\mathsf{add}(x, y) &= x + y \\
\mathsf{mult}(x, y) &= x \cdot y \\
\mathsf{sub}(x, y) &= x \dot{-} y \\
\mathsf{exp}(x, y) &= x^y \\
\mathsf{fact}(x) &= x! \\
\mathsf{sg}(x) &= \text{als } x \neq 0 \text{ dan 1 anders 0} \\
\mathsf{cosg}(x) &= \text{als } x \neq 0 \text{ dan 0 anders 1} \\
\mathsf{lt}(x, y) &= \text{als } x < y \text{ dan 1 anders 0} \\
\mathsf{gt}(x, y) &= \text{als } x > y \text{ dan 1 anders 0} \\
\mathsf{le}(x, y) &= \text{als } x \leq y \text{ dan 1 anders 0} \\
\mathsf{ge}(x, y) &= \text{als } x \geq y \text{ dan 1 anders 0}
\end{aligned}$$

$$\begin{aligned}
\mathsf{eq}(x, y) &= \text{als } x = y \text{ dan 1 anders 0} \\
\mathsf{ne}(x, y) &= \text{als } x \neq y \text{ dan 1 anders 0} \\
\mathsf{max}(x, y) &= \text{het maximum van } x \text{ en } y \\
\mathsf{min}(x, y) &= \text{het minimum van } x \text{ en } y \\
\mathsf{quo}(x, y) &= \text{als } y \neq 0 \text{ dan } \lfloor x/y \rfloor \text{ anders 0} \\
\mathsf{rem}(x, y) &= \text{als } y \neq 0 \text{ dan } x \bmod y \text{ anders } x \\
\mathsf{divides}(x, y) &= \text{als } y \neq 0 \text{ en } y \,|\, x \text{ dan 1 anders 0} \\
\mathsf{even}(x) &= \text{als } x \text{ even is dan 1 anders 0} \\
\mathsf{prime}(x) &= \text{als } x \text{ priem is dan 1 anders 0} \\
\mathsf{pn}(x) &= \text{het } x\text{-de priemgetal} \\
&\quad (\text{dus } \mathsf{pn}(0) = 2,\ \mathsf{pn}(1) = 3, \text{ etc.})
\end{aligned}$$