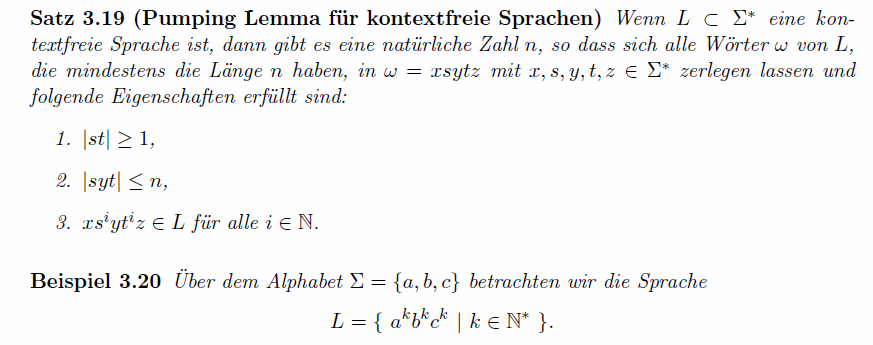
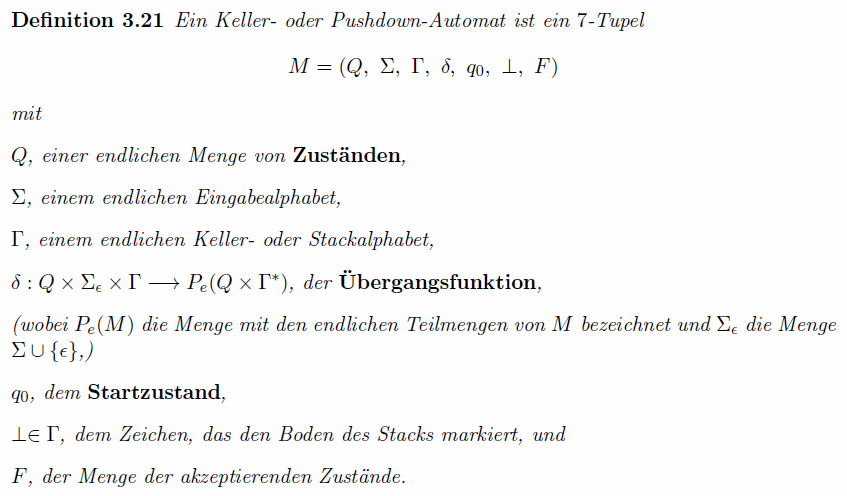
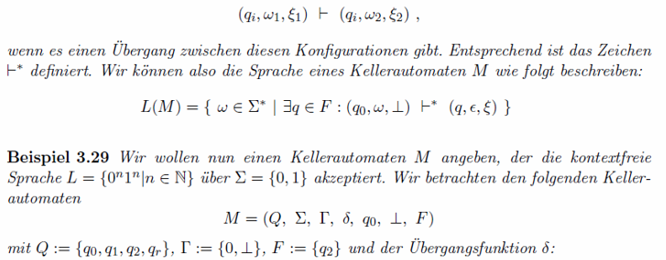
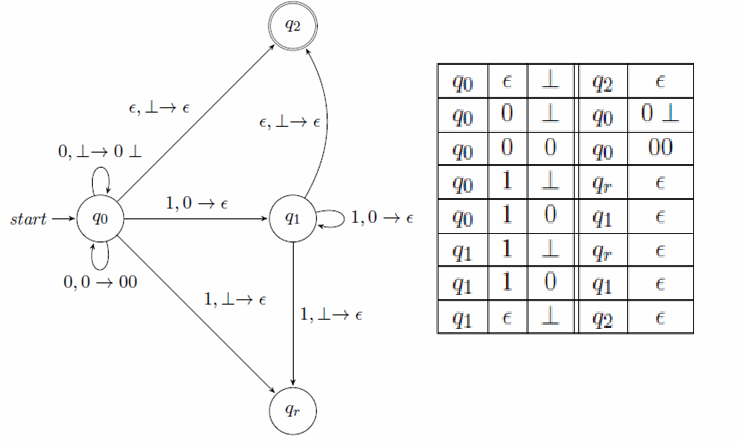
Pumping Lemma für kontextfreie Sprachen



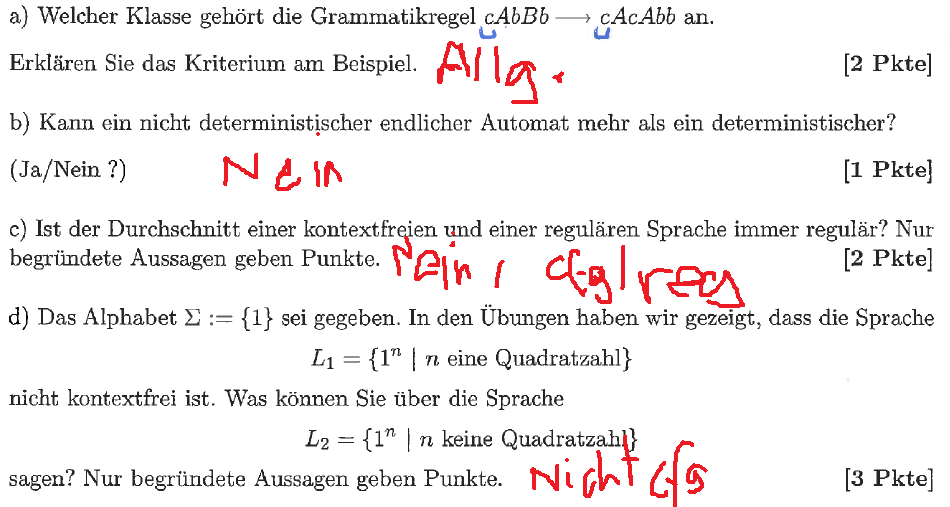
Kellerautomat



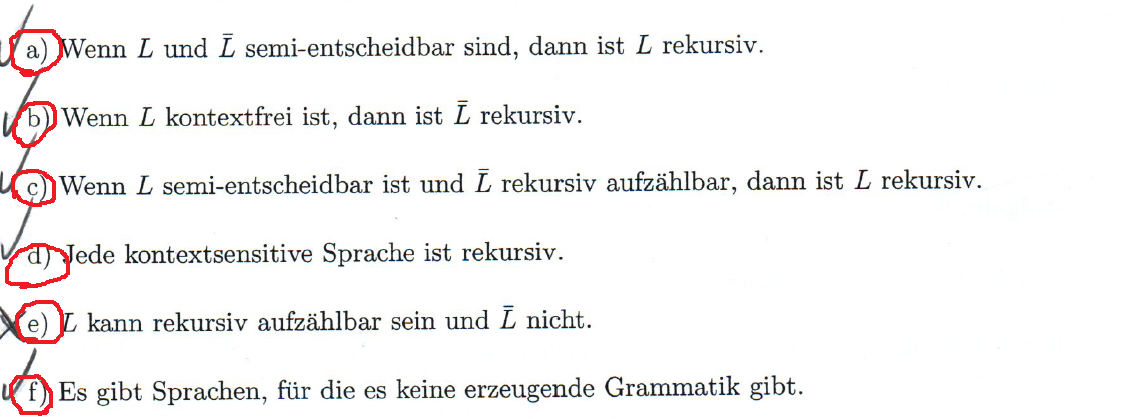




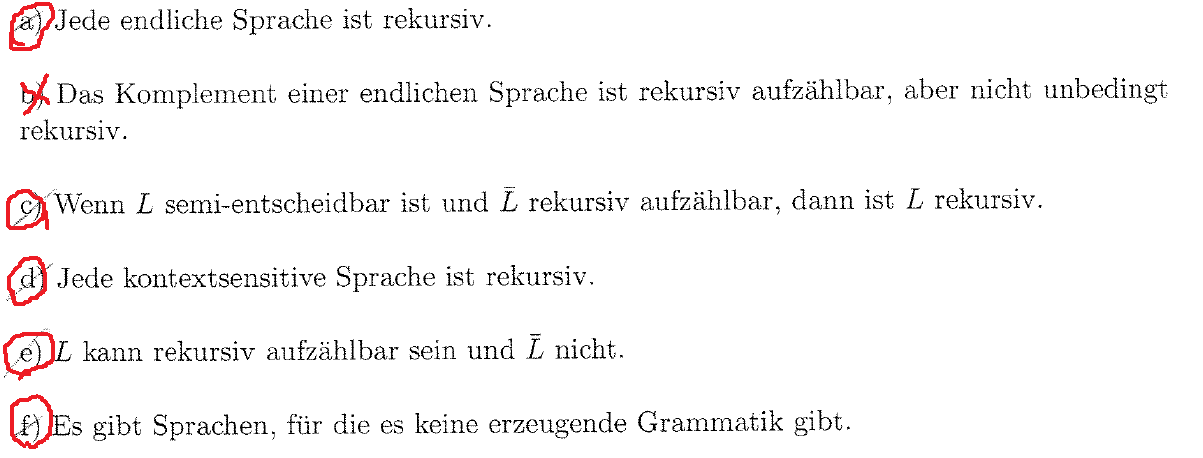
Fragen Theorie



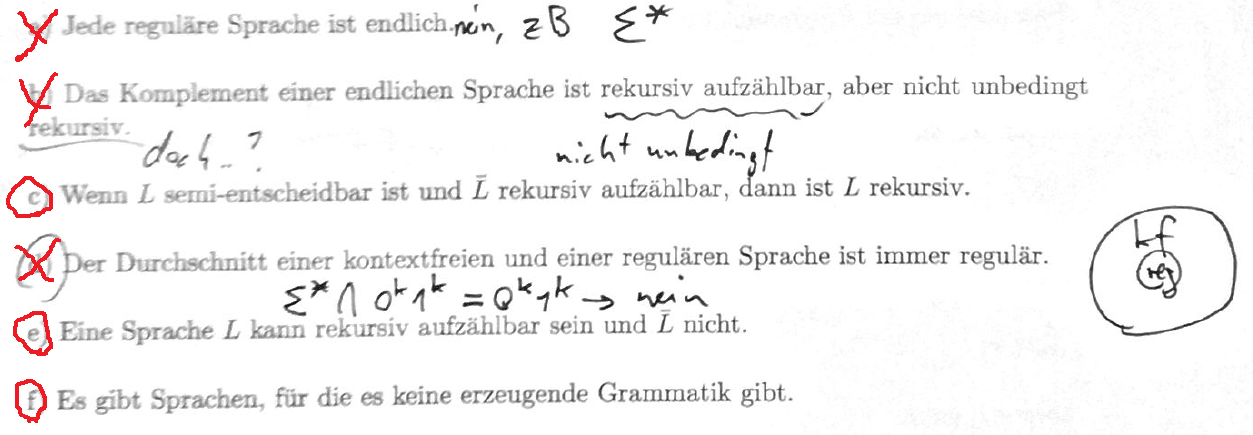
--------------------------------------------------



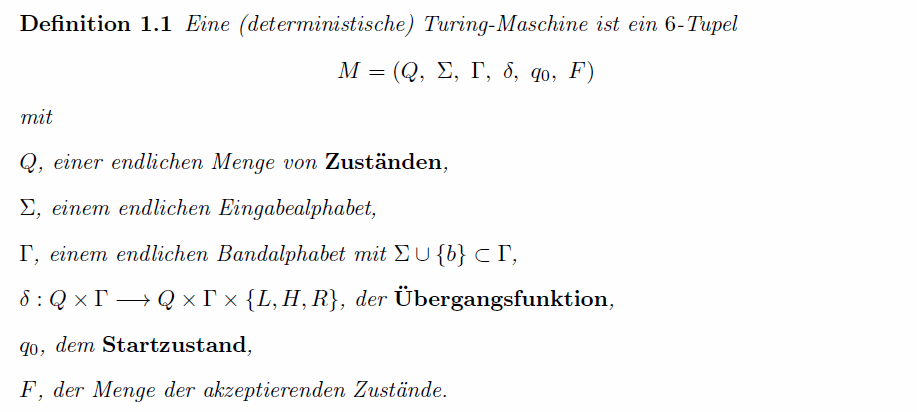
--------------------------------------------------

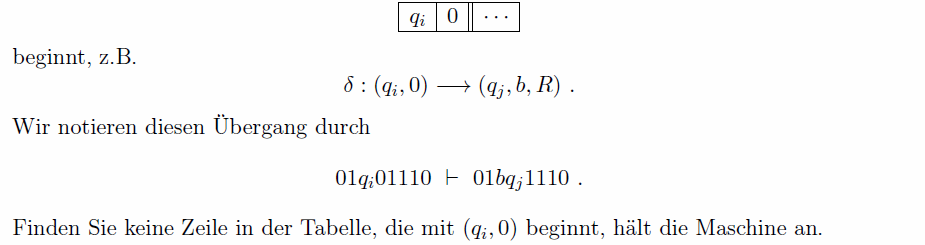


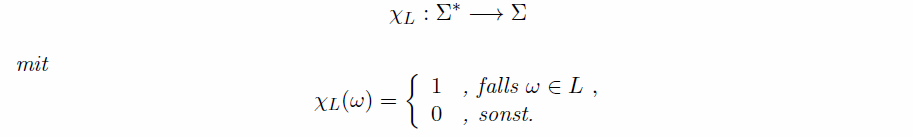
--------------------------------------------------

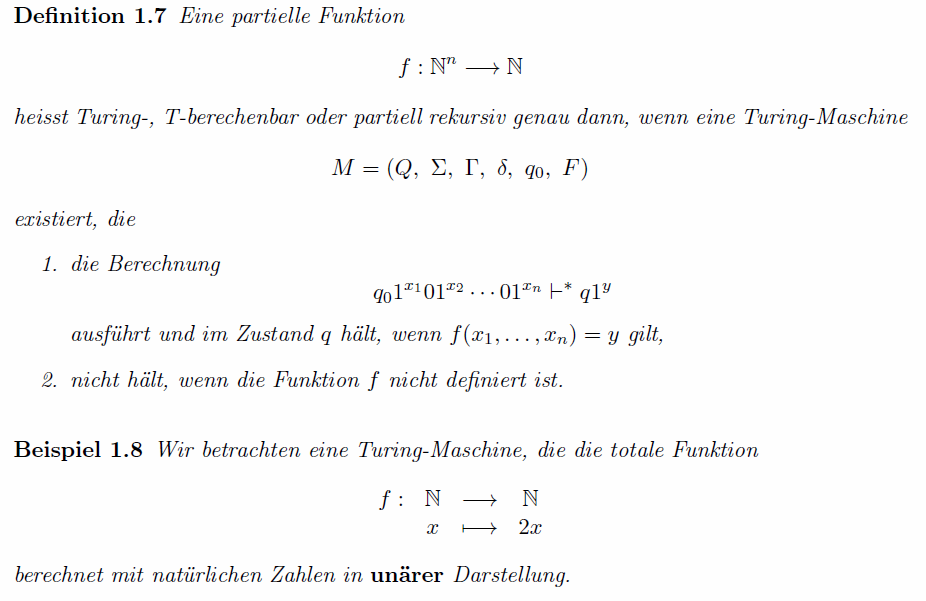


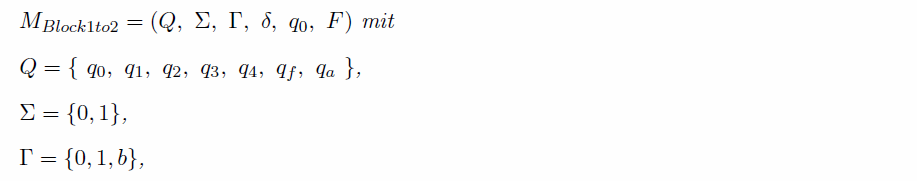
Turing Maschine

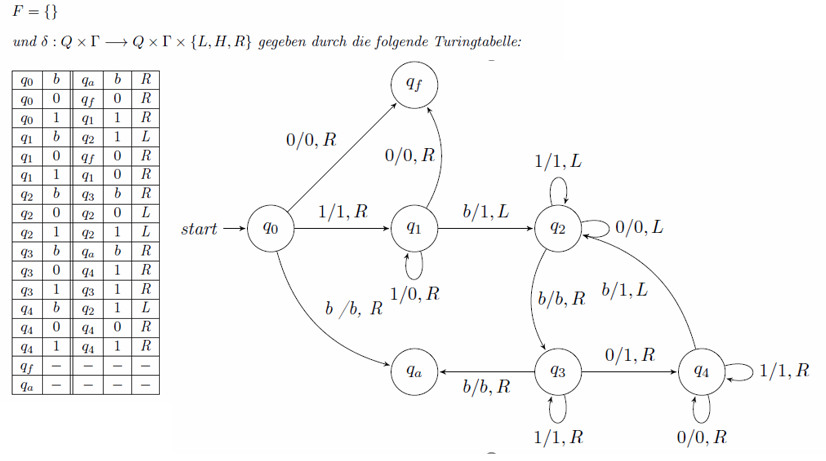




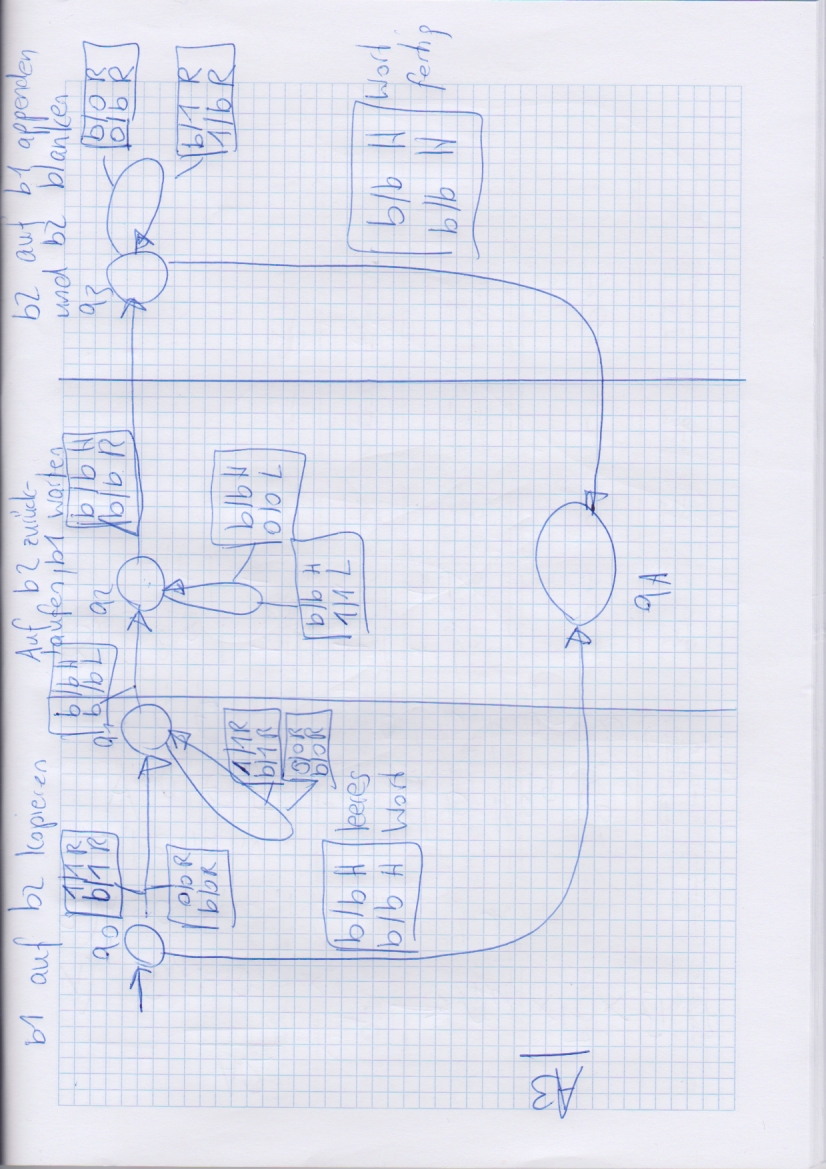






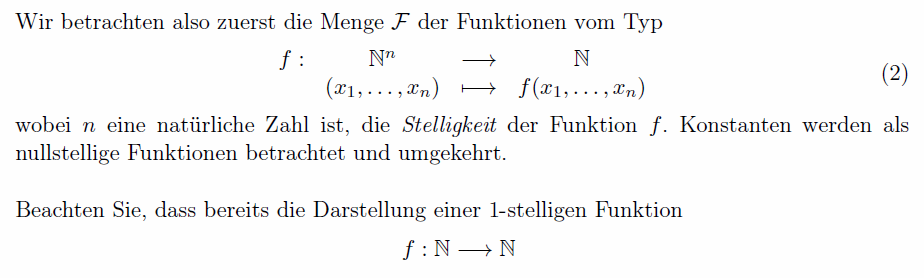


|  |  |
| --- | --- |
| static int Substract(int x1, int x2) {  int x000 = x1;  loop x2 do:  P(x000)  od  x1 = x000 // Return x1  }  static int Equals(int x1, int x2) {  // x1 = x2  int x100 = GenerateValue(1);  // x1 > x2  int x101 = Substract(x1, x2);  loop x101 do:  x100 = GenerateValue(0);  od  // x1 < x2  int x102 = Substract(x2, x1);  loop x102 do:  x100 = GenerateValue(0);  od  x1 = x100 // Return x1  } | static int FunctionF(int x1) {  // 1, sonst  int x200 = GenerateValue(1);  // 0, falls x = 0  int x201 = Equals(x1, 0);  loop x201 do:  x200 = GenerateValue(0);  od    // 2, falls x = 1  int x202 = Equals(x1, 1);  loop x202 do:  x200 = GenerateValue(2);  od  // 0, falls x = 2  int x203 = Equals(x1, 2);  loop x203 do:  x200 = GenerateValue(0);  od  x1 = x200 // Return x1  } |

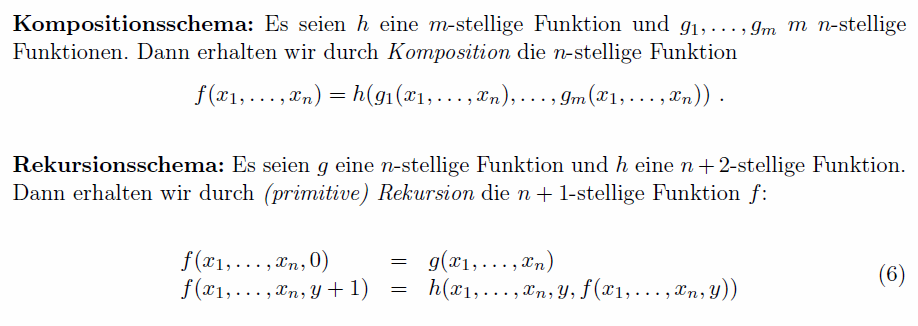


Rekurisve Funktionen

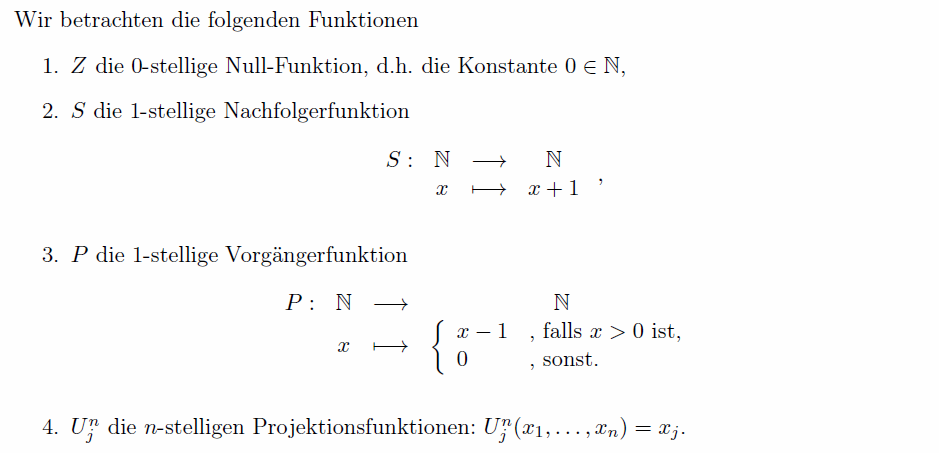
Definition



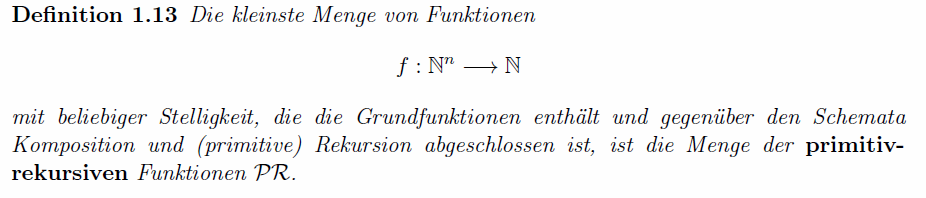
Primitiv rekursiv

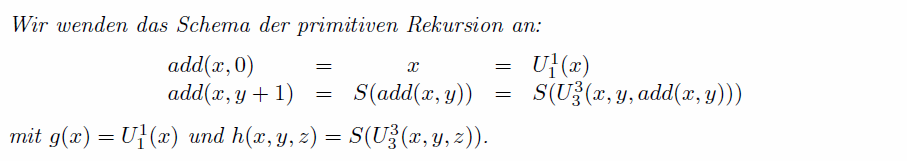


Grundfunktionen

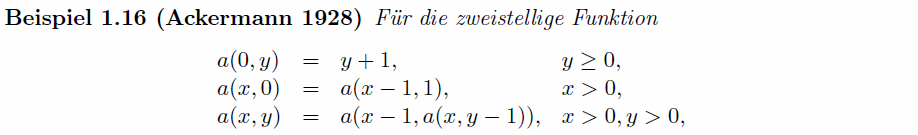


Primitiv rekursiv

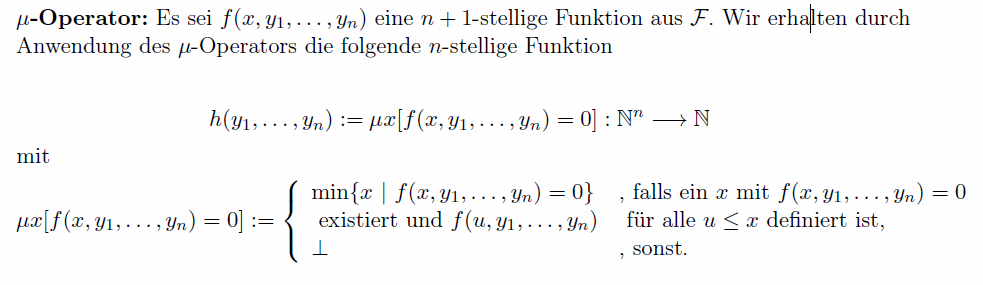


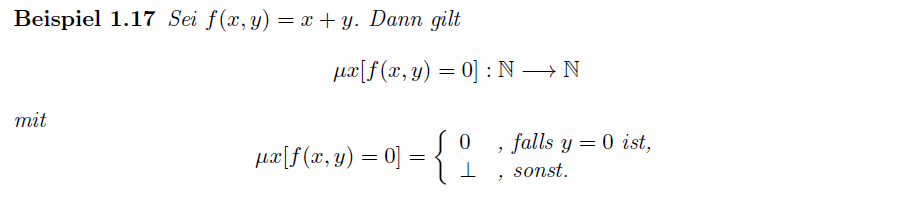


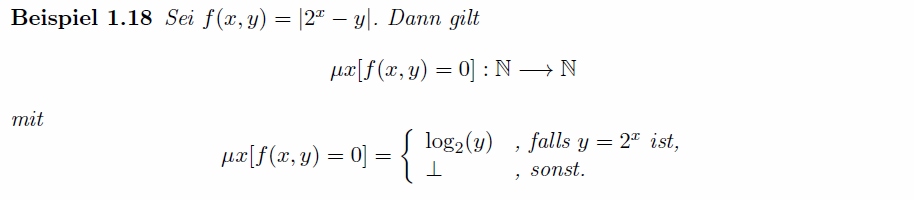
Ackermann Funktion

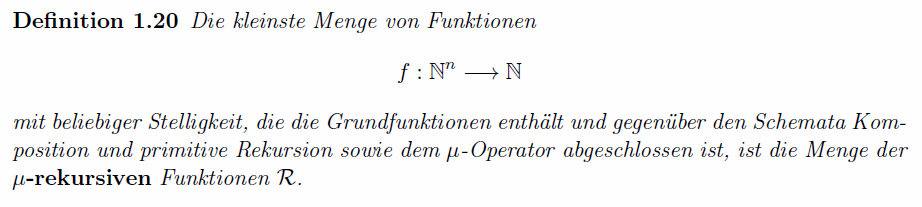


U rekursiv

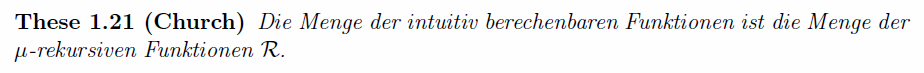






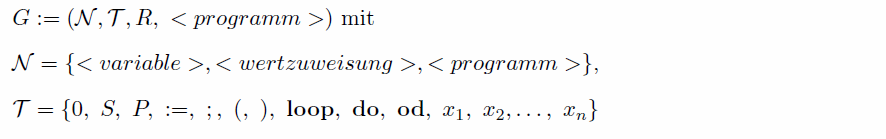


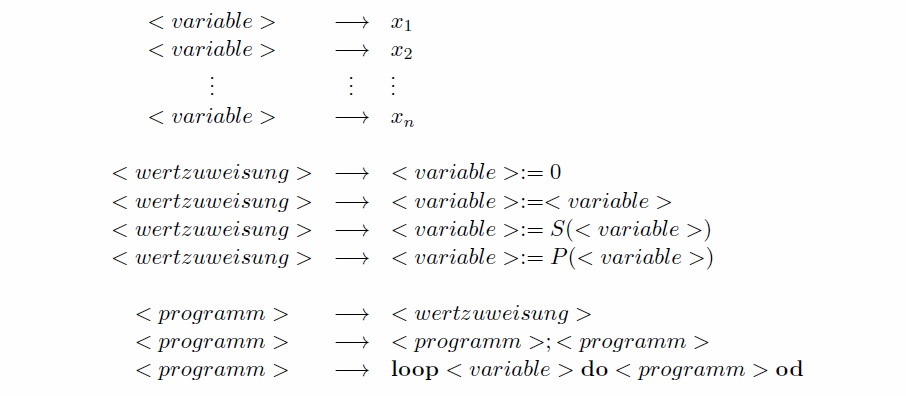
Church These

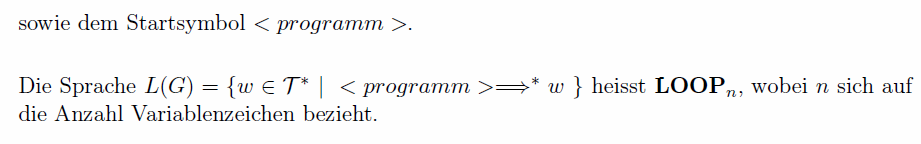


GOTO, WHILE und LOOP Programme

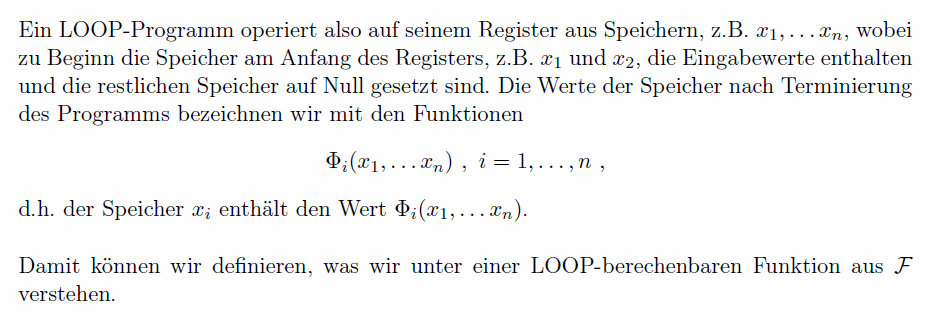
Definition der Sprache



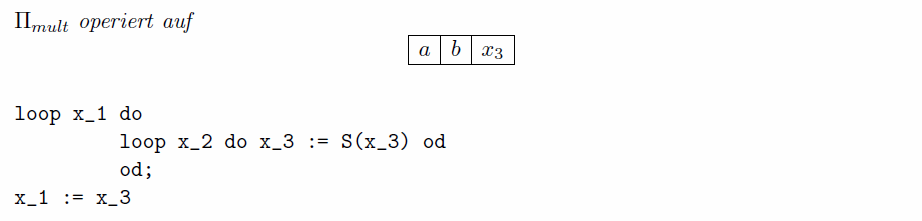




--------------------------

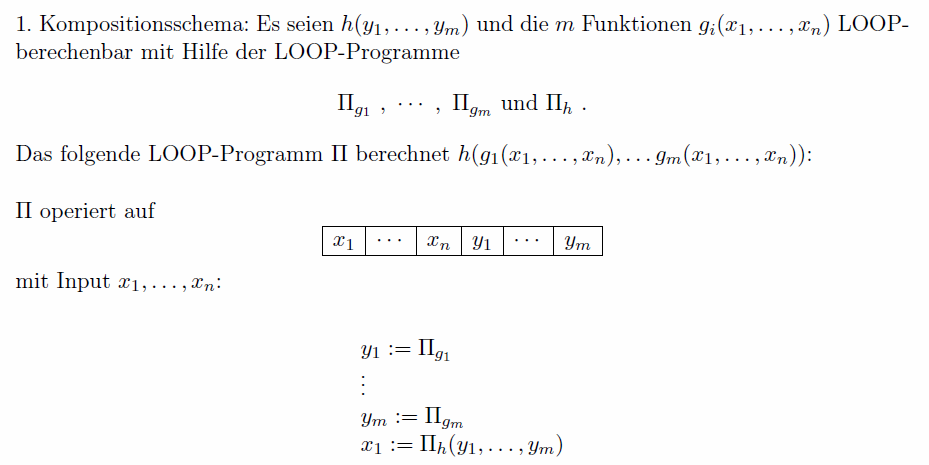


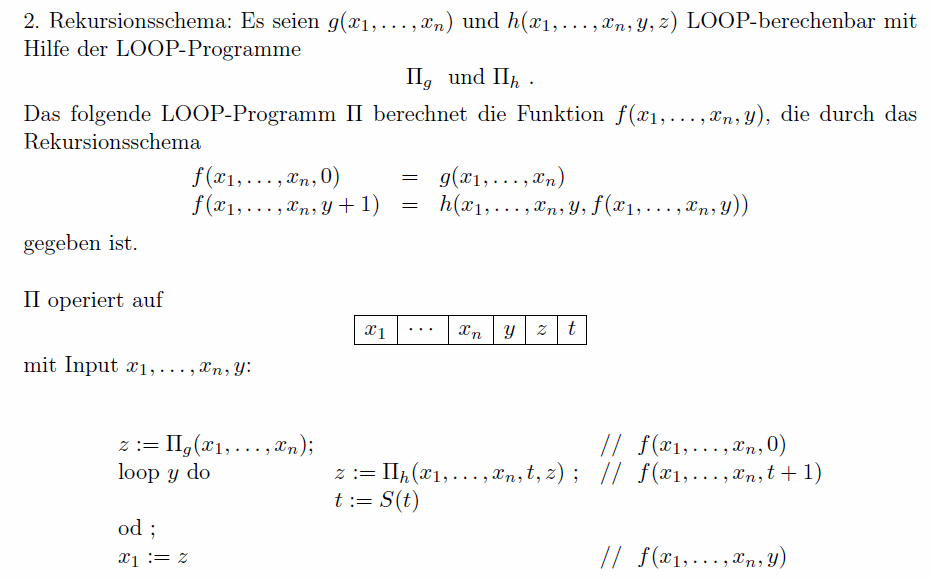
--------------------------



Länge: Zuweisung = 1, Loop = 1 + Programm

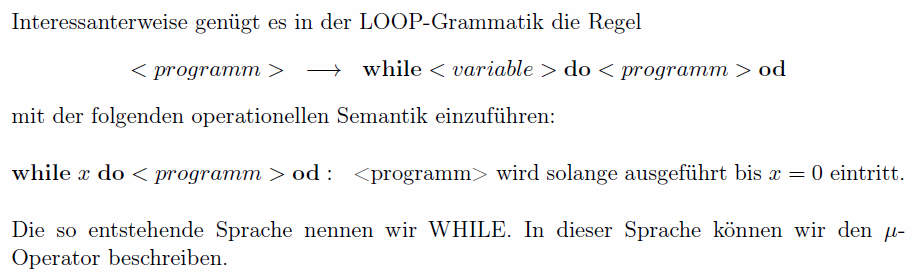
--------------------------





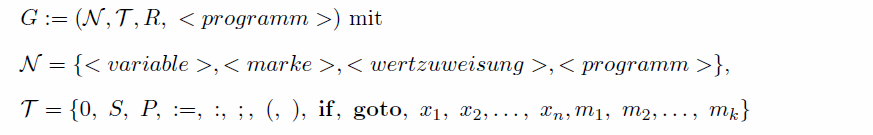
--------------------------

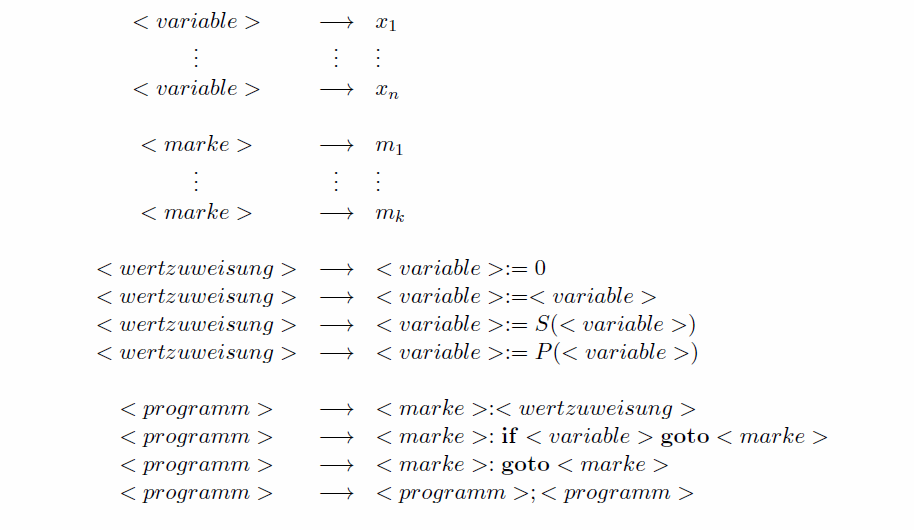
While für u rekursiv

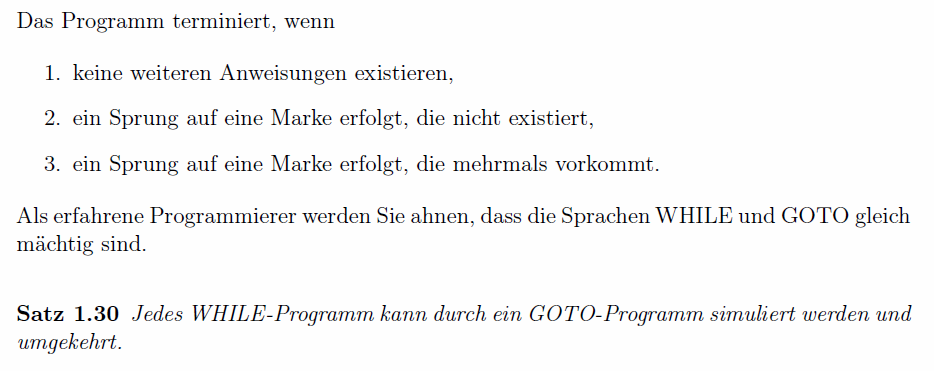


--------------------------

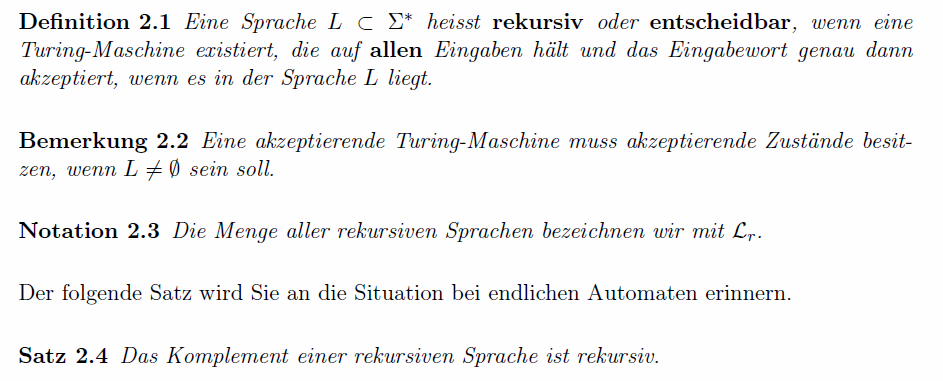
Goto Befehl

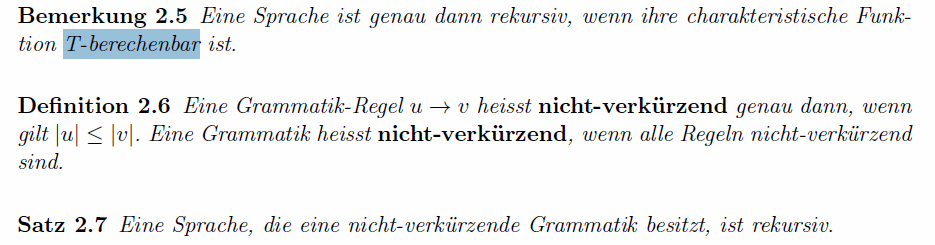


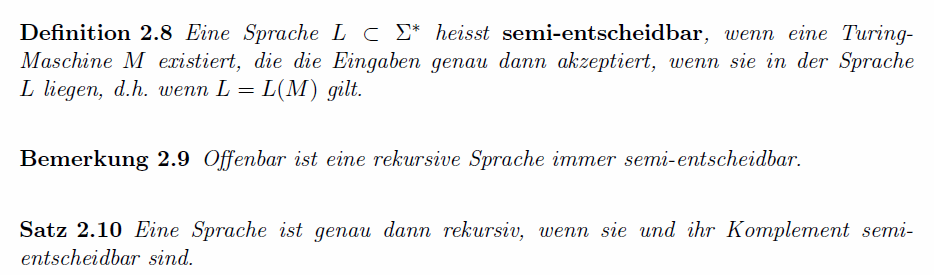




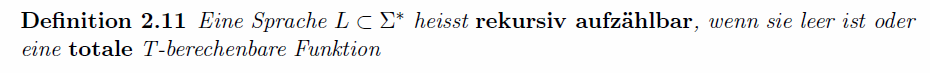
Definition rekursiv oder entscheidbar sowie das Komplement





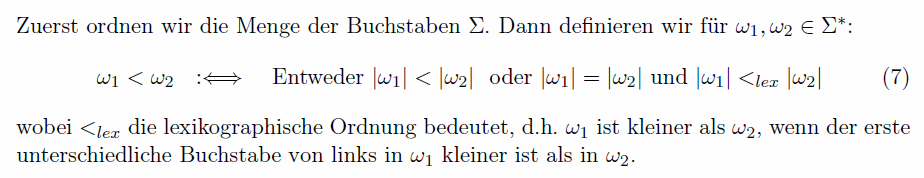


Definition rekursiv aufzählbar





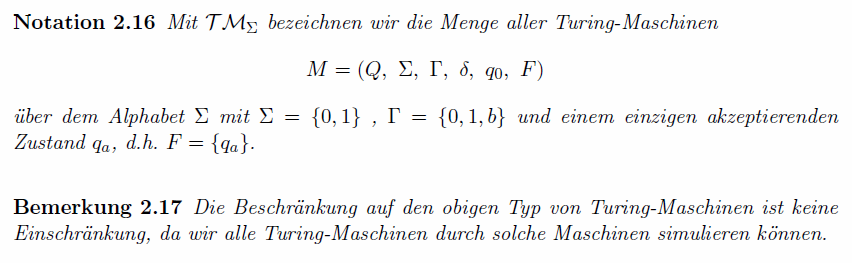
Definition Wortordnung

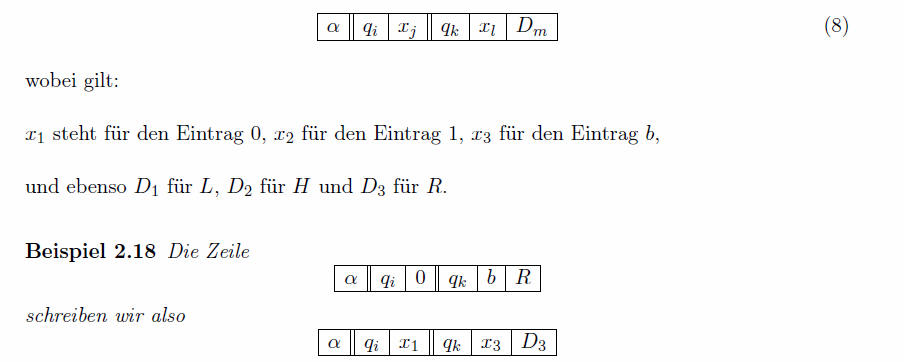


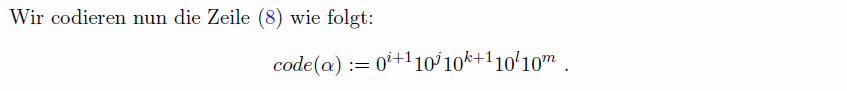
Notizen

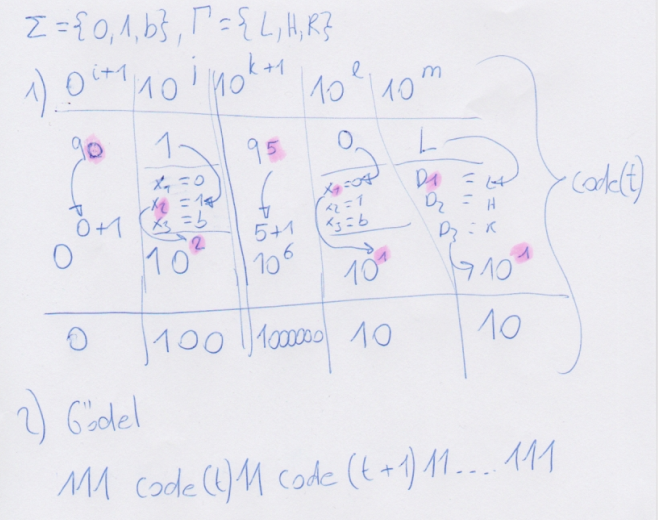
* Turing/T-berechenbar oder partiell rekursiv
  + Eine partielle Funktion heisst so, wenn eine Turing Maschine existiert, die beim Akzeptieren hält und sonst nicht hält/endlos läuft, wenn die Funktion nicht definiert ist
* Primitiv berechenbar PR
  + In «not too long» Zeit mit totalen (Grund-)Funktionen berechenbar 🡪 Ackermann braucht zu lange
* Mü/Mikro berechenbar R
  + f(y) = wurzel(y) 🡪 y = 9, x = 3
  + h(x, y) = 0
  + y – was mal x = 0
  + Bei x = 0 🡪 9, bei x = 1 9 – 1 = 8, bei x = 2 9 – 4 = 5 und bei x = 3 = f(9) ist 9 -9 = 0
* Entscheidbar / Rekursiv (Sprache Lr)
  + Eine Menge/Sprache A heisst entscheidbar, falls die charakteristische Funktion A berechenbar ist.
  + 1, falls w Element von A, 0 falls w nicht Element von A
  + Immer Ja oder Nein auf allen Input
  + Komplement einer rekursiven Sprache ist auch rekursiv (Immer eines, nie keines/beides)
  + Sprache ist rekursiv, falls ihre charakteristische Funktion T berechenbar ist
  + Sprache ist nicht-verkürzend/rekursiv, falls es keine verkürzende Grammatik gibt
* Semientscheidbar
  + Eine Menge A heisst semi-entscheidbar, falls die halbe charakteristische Funktion von A berechnenbar ist
  + 1, falls w Element von A, undefiniert falls w nicht Element von A
  + Auf ja immer ja, auf nein undefiniert/Endlosschleife
  + Eine Sprache ist entscheidbar/rekursiv, wenn A und A Komplement semi entscheidbar sind
* Rekursiv Aufzählbar (Sprache Lre)
  + Eine Sprache A heisst rekursiv aufzählbar, wenn A = {} oder falls es eine totale (T-)berechenbare Funktion gibt
  + Funktion f für A zählt A rekursiv auf --> Generator für unendlich Elemente
  + Beispiel Funktion f(x) = a^x oder totale Funktion
  + Eine Sprache A ist rekursiv aufzählbar genau dann, wenn A semi-entscheidbar ist (oder leer)
  + Jede Sprache, die eine Grammatik besitzt, ist rekursiv aufzählbar
* Unentscheidbare Probleme
  + Das spezielle Halteproblem für M wenn w ok, sonst M Halter stop -> Semi-entscheidbar

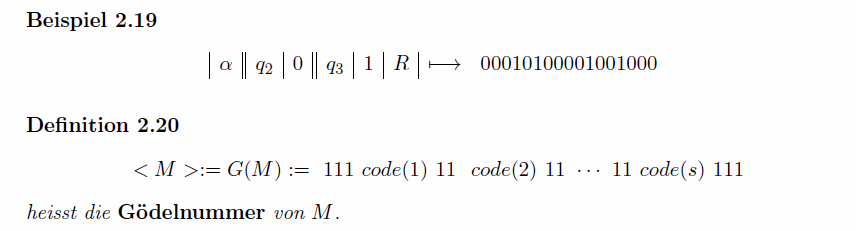
Gödelisierung











Die Berechnungstheorie muss injektiv und T-berechenbar sein, die Umkehrfunktion ebenfalls T-berechenbar

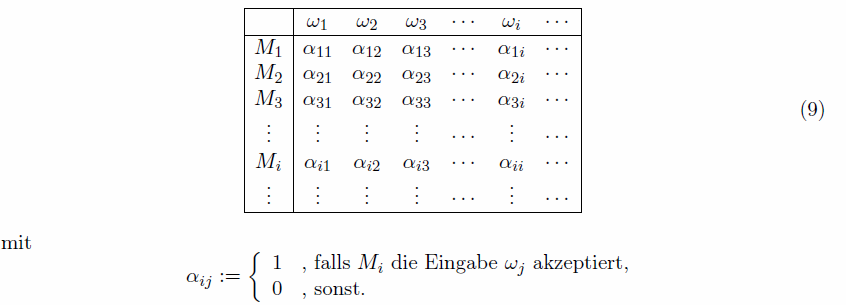
Ablauf

1. <M>w steht auf dem ersten der drei Bänder 🡪 Syntaxcheck
2. <M> wird auf das zweite Band kopiert und auf dem ersten Band gelöscht. Auf das dritte Band wird 0 = q0 als Startzustand geschrieben
3. Schritt bis zur Terminierung
   1. Q auf dem dritten Band und der passende Buchstabe von w auf dem ersten band lesen
   2. Auf dem zweiten Band die Regel dafür suchen
   3. Abarbeiten
      1. Wechsel auf den neuen Zustand auf dem dritten Band + schreiben auf Band
      2. Auf dem ersten Band den Buchstaben löschen und mit b ersetzen
      3. Auf dem ersten Band einen Schritt nach links gehen
      4. Zeiger des zweiten und dritten Bandes wieder auf den Anfang setzen

Mengenlehre und rekursive Aufzählbarkeit

* Wenn M eine Menge ist und P(M) die Potenzmenge, dann ist P(M) immer mächtiger 🡪 Es gibt Sprachen, die nicht rekursiv aufzählbar sind, eben z.B. die P(M)

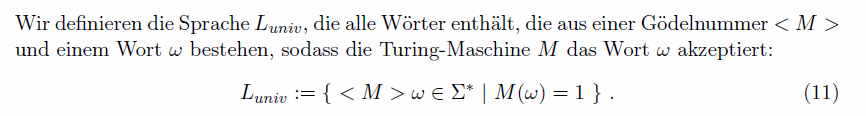
Beispiele nicht rekursiver Sprachen

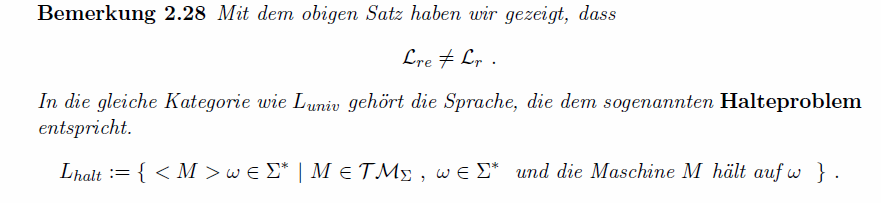




Ldiag ist dabei nicht rekursiv. Auch das Komplement ist nicht rekursiv. Zudem darf mindestens eine der beiden Sprachen nicht rekursiv aufzählbar sein

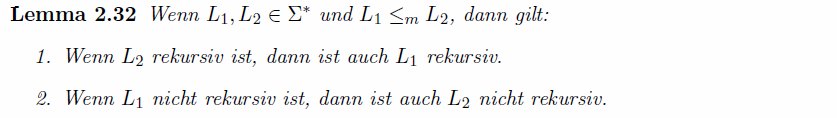
Beispiel einer rekursiv aufzählbaren, aber nicht rekursiven Sprache







Reduktion: L1 ist auf L2 reduzierbar, genau dann wenn eine total T-berechenbare Funktion existiert



Satz von Rice

