Prof. Dr. Dörte Haftendorn, Uni Lüneburg, 9. November 2002

Erweiterter Euklidischer Algorithmus

```
ggte(a,b)
Func

Local
r0_,r1_,r2_,q0_,q1_,s0_,s1_,s2_,
t0_,t1_,t2_

a→r0_:b→r1_:0→s0_:1→s1_:1→t0_:⁻i
Part(a/b)→t1_
Loop
  mod(r0_,r1_)→r2_
  If r2_=0 Then
     Return  [[r1_,s0_,t0_]]
     Exit
  EndIf
  iPart(r0_/r1_)→q0_
  iPart(r1_/r2_)→q1_
  s0_-q1_*s1_→s2_
  t0_-q1_*t1_→t2_
  r1_→r0_:r2_→r1_:q1_→q0_:
  s1_→s0_:s2_→s1_:
  t1_→t0_:t2_→t1_:
EndLoop
EndFunc
```

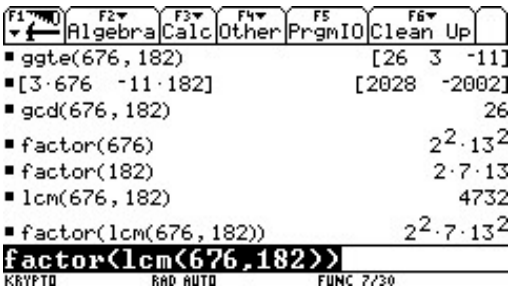
Lokale Variable

Vorbelegungen

Schleife

Ausgabe des vorigen Restes, wenn der gerade erzeugte Rest 0 ist.
Ausgabe der vorigen Linearfaktoren.
Neuberechnung

Umtaufen



ggte(13,8) [1,-3,5]
heißt 1=-3*13+3*8

MuPAD
igcdex(13,8) [1,-3,5]

gcd=greatest common divisor

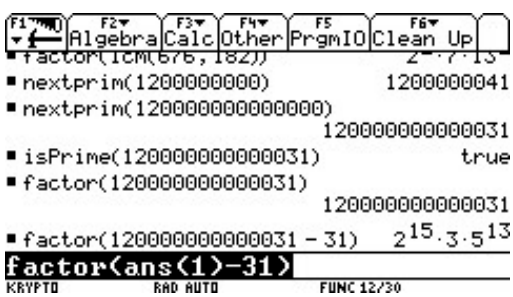
Nächste Primzahl größer gleich a

```
nextprim(a)
Func
Local  i
If mod(a,2)=0 Then
    1→i
  Else
    0→i
EndIf
  Loop
    If IsPrime(a+i) Then
      Return  a+i
      Exit
    EndIf
    i+2→i
   EndLoop
EndFunc
```

Gerade Zahlen auslassen

Schleife von a an alle Ungeraden
eingebaute Funktion **IsPrime** nutzen
wenn das wahr ist, dann gib die Zahl aus

nächste Ungerade Zahl wieder prüfen ...



120 Trillionen 31 ist die erste Primzahl hinter 120 Trillionen
Antwort in <1 Sekunden