

Mersenne-Primzahlen

lucasl.py

Eine Mersenne-Zahl ist eine natürliche Zahl der Form $M_n = 2^n - 1$. Eine Mersenne-Primzahl ist demnach eine Mersenne-Zahl, die außerdem auch eine Primzahl ist.

Wenn n keine Primzahl ist, dann ist auch M_n keine. Das folgt aus der Tatsache, dass $2^{ab} - 1 = (2^a - 1)(1 + 2^a + 2^{2a} + 2^{3a} + \dots + 2^{(b-1)a})$. Wenn n prim ist, muss M_n nicht auch prim sein. So ist zum Beispiel $M_{11} = 2^{11} - 1 = 2047 = 23 \cdot 89$.

Es gibt nun einen einfachen Test (Lucas-Lehmer-Test), der überprüft, ob für eine Primzahl $p > 2$, die Mersenne-Zahl $M_p = 2^p - 1$ eine Primzahl ist. Dafür bilden wir die Folge

$$s_i = s_{i-1}^2 - 2 \pmod{M_p} \quad \text{mit } s_0 = 4.$$

Wenn nun $s_{p-2} = 0$, dann ist M_p eine Primzahl. Das lässt sich in Python einfach implementieren.

```

1.1 1.2 1.3 *PyKurz RAD 1/10
lucasl.py
def lucasl(p):
    s=4
    m=2**p-1
    for i in range(p-2):
        s=(s*s-2)%m
    print("Mersenne-Zahl 2^",p,"-1 ist ")
    if s==0:print("Primzahl.")
    else: print("keine Primzahl.")

```

```

1.1 1.2 1.3 *PyKurz RAD 12/12
Python-Shell
>>>from lucasl import *
>>>lucasl(13)
Mersenne-Zahl 2^ 13 -1 ist
Primzahl.
>>>lucasl(127)
Mersenne-Zahl 2^ 127 -1 ist
Primzahl.
>>>lucasl(11)
Mersenne-Zahl 2^ 11 -1 ist
keine Primzahl.
>>>

```

- Wir definieren die Prozedur `lucasl(p)`, die das Argument $p > 2$ und prim erwartet.
- Wir setzen $s = 4$ und berechnen $m = M_p = 2^p - 1$.
- In einer Schleife wird $p - 2$ mal der nachfolgende s -Wert mit $s = (s*s-2)\%m$ berechnet.
- Dann wird die Mersenne-Zahl ausgegeben gemeinsam mit der Information, ob sie eine Primzahl ist ($s = 0$) oder nicht.

Mersenne-Zahlen werden sehr rasch sehr groß. Daher führt dieser einfache Test zu einer ganzen Folge von sehr großen Primzahlen. Seit 2005 ist die größte bekannte Primzahl immer eine Mersenne-Zahl gewesen.

https://de.wikipedia.org/wiki/Great_Internet_Mersenne_Prime_Search
<http://www.mathe.tu-freiberg.de/~hebisch/cafe/mersenneprim.html>