Aufgaben 12-16

November 30, 2009

Computermathematik (für Informatik)

3. Übungsblatt (Musterlösung)
25. 11. 2009
Die heutigen Übungen sollen mit dem Computeralgebrasystem **Sage** gelöst werden.
Die Lösung der Beispiele soll auf möglichst kompakte Weise erfolgen. Wenn zum Beispiel eine Funktion für mehrere Werte berechnet werden soll, soll das mittels einer geeigneten Schleifen oder Listen Operation erfolgen, und **nicht** alle Werte einzeln eingetippt werden.
Zwischenergebnisse welche in einem weiteren Berechnungsschritt benötigt werden, sollen in eine Variable gespeichert und weiterverwendet werden (**nicht** neu eintippen).

Beispiel 12

Bestimmen Sie die Anzahl der Stellen, sowie die Summe der Ziffern von $n!$ wobei n die Werte 1, 10, 100, 1000, 10000 und 100000 annimmt.

Sage code
```
[factorial(10^i).ndigits() for i in [0..5]]
```

```
[1, 7, 158, 2568, 35660, 456574]
```

Sage code
```
[sum(factorial(10^i).digits(base = 10)) for i in [0..5]]
```

```
[1, 27, 648, 10539, 149346, 1938780]
```

Sage code
```
factorials = [factorial(10^i) for i in [0..5]]
table = [[f.ndigits(), sum(f.digits(base = 10))] for f in factorials]

table = [["Number of Digits", "Sum of Digits"]] + table
html.table(table, header = True)
```

Number of Digits	Sum of Digits
1	1
7	27
158	648
2568	10539
35660	149346
456574	1938780

Beispiel 13

Untersuchen Sie, für welche $n \leq 30$ die Primfaktorzerlegung von $f(n) = n! - 1$ Primfaktoren mehrfach enthält. Geben sie eine Liste dieser Zahlen und ihrer Primfaktorzerlegungen aus.

Sage code

```
def f(n):
  return factorial(n) + 1
```

Sage code

```
fact_primes = [[n, f(n), factor(f(n))] for n in [1..30]
               if squarefree_part(f(n)) != f(n)
              ]

fact_primes = [["n", "n!", "Primfaktoren"]] + fact_primes
html.table(fact_primes, header = True)
```

n	n!	Primfaktoren
4	25	5^2
5	121	11^2
7	5041	71^2
12	479001601	$13^2 \cdot 2834329$
23	25852016738884976640001	$47^2 \cdot 79 \cdot 148139754736864591$

Sage code

```
squareList=[]

for n in [2..30]:
  fact = factor(f(n))
  for (primfactor, mult) in fact:
    if mult > 1:
      squareList.append([n, f(n), fact])
      break

html.table(squareList)
```

4	25	5^2
5	121	11^2
7	5041	71^2
12	479001601	$13^2 \cdot 2834329$
23	25852016738884976640001	$47^2 \cdot 79 \cdot 148139754736864591$

Beispiel 14

Eine Zahl $M_p = 2^p - 1$ ist höchstens dann eine Primzahl wenn p selbst prim ist. Primzahlen dieser Form nennt man Mersennesche Primzahlen.Die größten bekannten Primzahlen haben diese Form. Es ist unbekannt, ob es unendlich viele Mersennesche Primzahlen gibt.
Bestimmen Sie für $p \leq 500$ alle Mersenneschen Primzahlen. Geben Sie Ihr Ergebnis als Tabelle mit den Spalten p und M_p an.

Sage code

```
table = [[p, 2^p - 1] for p in [1..500] if is_prime(p) and is_prime(2^p - 1)]

table = [var('p, M_p')] + table
html.table(table, header = True)
```

p	M_p
2	3
3	7
5	31
7	127
13	8191
17	131071
19	524287
31	2147483647
61	2305843009213693951
89	618970019642690137449562111
107	162259276829213363391578010288127
127	170141183460469231731687303715884105727

Beispiel 15

Der älteste Algorithmus zur Berechnung von Primzahlen, ist der Sieb des Eratosthenes (Siehe: Wikipedia). Schreiben Sie eine Funktion **sieb(n)**, die eine Liste aller Primzahlen $\leq n$ mit Hilfe des Siebes von Eratosthenes berechnet.
Achtung: Die in Sage eingebauten Funktionen zur Berechnung von Primzahlen, dürfen in diesem Beispiel nicht verwendet werden.

Sage code

```
def sieb(n):
    primes = []
    liste = [2..n]
    while liste != []:
        current_prime = liste[0]
        primes.append(current_prime)
        liste = [p for p in liste if not current_prime.divides(p)]

    return primes
```

Sage code

```
prime_range(50)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]
```

Sage code

```
sieb(50)
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]
```

Beispiel 16

Bringen Sie den folgenden Ausdruck auf eine möglichst einfache Form

$$\left(\frac{(\sqrt{x^3}-\sqrt{8})(\sqrt{x}+\sqrt{2})}{x+\sqrt{2x}+2}\right)^2+\sqrt{(x^2+2)^2-8x^2}.$$

Sage code

```
var('x')
expr = ((sqrt(x^3)-sqrt(8))*(sqrt(x)+sqrt(2)) /
        (x+sqrt(2*x)+2))^2+sqrt((x^2+2)^2-8*x^2)
show(expr)
```

$$\frac{\left(\sqrt{2}+\sqrt{x}\right)^2\left(2\sqrt{2}-\sqrt{x^3}\right)^2}{\left(\sqrt{2}\sqrt{x}+x+2\right)^2}+\sqrt{\left(x^2+2\right)^2-8\,x^2}$$

Sage code

```
expr = expr.simplify_full().factor()
show(expr)
```

$$2\,(x-1)^2$$