Das Problem 48 – 24

Beim Lesen eines Rätselbuchs stieß ich auf folgendes Problem:

Erhöht man die Zahl 48 um eins, dann ergibt sich die Quadratzahl 49. Halbiert man 48 und fügt eins hinzu, dann erhält man 25, ebenfalls eine Quadratzahl.

Die Aufgabe: Suche die nächstgrößere Zahl, die bei gleicher Vorgehensweise ebenfalls zu Quadratzahlen führt.

Ich stellte mir folgende Fragen: Gibt es noch weitere Zahlen, gibt es auch eine kleinere Lösung?

Gibt es auch Lösungen, wenn man 1 abzieht? Gibt es auch eine „Formel“, um solche Zahlen zumindest aus einer Lösung abzuleiten?

Ich fand einige Löungen durch Probieren und stellte strukturelle Untersuchungen an: Differenzen und Quotienten quer und vertikal.

Dann fand ich, dass dies eigentlich den Einsatz eines Computerprogramms herausforderte, lernte ein wenig Python 3.4 und schrieb ein paar einfache Programme.

Struktur 1: Zur Zahl und zu deren Hälfte je 1 dazu geben. Wenn dann beide Quadrate sind, haben wir eine Lösung gefunden.

Betrachtet man die Quadratbasen (rechts), dann kann man folgende Beziehungen erkennen

1	Summe 2		1
4			8
5	Summe 12, Differenz 2		7
24			48
29	Summe 70 , Differenz 12		41
140			280
169	Summe 408	Differenz 70	239

Hier kann man viele nette Beziehungen sehen: Links die Zahlen zwischen den Lösungen sind die Differenzen, rechts die Summen . Zugleich sind diese Zahlen auch das Doppelte und 4-fache der oberhalb stehenden Summen. Die Summe zweier Lösungen findet sich beim nächsten Paar als dessen Differenz.

Daraus kann man eine einfache Formel ableiten, wie man von einer Zeile zur nächsten hochrechnen kann: Man addiert zur linken Zahl die die doppelte Summe, und zieht – um die rechte Zahl zu bekommen – von der 4-fachen Summe die rechte Startzahl ab.

Wir wollen jetzt von der letzten Zeile eine Zeile hochrechnen.

L5 = 169 +2 * 408 = 985 R5= 4 * 408 – 239 = 1393

Voilá! Siehe roter Block auf der nächsten Seite!

Das Ganze in ein kleines Programm gegossen, bei dem ich mir die Anzahl der Lösungen wünschen darf:

Programm 1:

```
x = 1.0
y = 1.0
print ( "Wie viele Paare?" )      Bezieht sich auch Startzahl und deren Hälfte
z = input ()
z = int (z)
for i in range (1,z):
    print(x,y,x**2 - 1,y**2-1)
    a = x+y
    u = x + 2*a
    v = 4*a - y
    x = u
    y = v
```

10 Lösungen!

1.0 1.0 0.0 0.0 Die kleinste Lösung heißt 0! 0+1 =1 , ein Quadrat!

5.0 7.0 24.0 48.0 0/2=0 . und wieder 0+1=1!!

29.0 41.0 840.0 1680.0

169.0 239.0 28560.0 57120.0

985.0 1393.0 970224.0 1940448.0

5741.0 8119.0 32959080.0 65918160.0

33461.0 47321.0 1119638520.0 2239277040.0

195025.0 275807.0 38034750624.0 76069501248.0

1136689.0 1607521.0 1292061882720.0 2584123765440.0

Setzt man in die obere Formel die Startzahlen als Symbole ein, dann ergeben sich etwas andere Formeln, um zur nächsten Ergbnisreihe zu gelangen. Siehe roter Block im nächsten Programm!

Programm 2:

```
x = 1.0
y = 1.0
```

```
print ( "Wie viele Paare?" )

z = input ()

z = int (z)

for i in range (1,z):

    print(x,y,x**2 - 1,y**2-1) Wie viele Paare?

a = 3*x + 2*

b = 4*x + 3*y

    x = a

    y = b
```

5 Lösungen:

5

1.0 1.0 0.0 0.0

5.0 7.0 24.0 48.0

29.0 41.0 840.0 1680.0

169.0 239.0 28560.0 57120.0

STRUKTUR 2

Nun soll 1 abgezogen werden! Wir starten mit der Zahl 10. 10 -1 = 9, offensichtlich ein Quadrat!

10/2 = 5; 5 – 1 = 4, ebenfalls ein Quadrat? Damit ergibt sich als Startpaar 2 und 3 als Basen obiger Quadrate.

Programm 3

```
x = 2.0

y = 3.0

print ("Wie viele Paare?")

z = input ()

z = int (z)


for i in range (1,z):

    print (x,y,x**2 + 1,       y**2 + 1)
```

a = x + y

x = 2*a + x   Dieser Formelteil ist gleich wie bei Struktur 1

y = 4*a – y   Auch diese Formel stimmt genau mit jener der Struktur 1 überein!

10 Lösungen:

Wie viele Paare?

10

2.0   5   3.0   5.0   10.0

12.0   19   17.0   145.0   290.0

70.0   169   99.0   4901.0   9802.0

408.0   985   577.0 166465.0 332930.0

2378.0 3363.0 5654885.0 11309770.0

13860.0 19601.0 192099601.0 384199202.0

80782.0 114243.0 6525731525.0 13051463050.0

470832.0 665857.0 221682772225.0 443365544450.0

2744210.0 3880899.0 7530688524101.0 15061377048202.0

Als nette Beziehung ergibt sich, dass die linken Basiszahlen, die Summen der Pärchen aus Struktur 1

sind. Die bis zur Zeile 4 nachträglich eingefügten Zahlen sind die linken Zahlen aus Struktur 1. Eine ähnliche Beziehung lässt sich in Struktur 1 in Bezug auf Struktur 2 finden.

Mein erster Versuch ließ mich allerdings direkt nach Quadraten suchen. Ein zähes Unterfangen, weil man für 5 Lösungen bereits bis zur Milliarde gehen muss. Und das dauert.

Die Geschwister von 48

for i in range (5,1000000000,2): Nur ungerade Zahlen möglich!

x = i**0.5

a = int(x)

if a < x :

continue

else:

z =(i-1)/2+1   Wenn die gefundene Zahl ein Quadrat ist, dann wird 1 abgezogen, halbiert,

```
y = z**0.5      und nachgesehen, ob dieses Ergebnis auch ein Quadrat ist.
b = int (y)
if b == y :
    print (int(y),int(x), int(z) ,i )
    continue
```

5	7	49	25
29	41	841	1681
169	239	28	57121
985	1393	970225	1940449
5741	8119	32959081	65918161

Die erste Lösung fehlt hier, weil ich mit der aus dem Rätselbuch bekannten Lösung begann!