Teilbarkeitstests in *B*-adischen Stellenwertsystemen

1 Definition benötigter Funktionen

1.1 Grundlegende Funktionen

■ MultMod

Linksmultiplikation (von x) mit a (modulo m):

```
MultMod[m_, a_][x_] := Mod[Mod[a, m] * Mod[x, m], m]
```

■ PotModList

Liste der verschiedenen Potenzen a^j mod m für alle $j \geq 1$:

```
PotModList[m_][a_] :=
 Drop[ NestWhileList[MultMod[m, a], Mod[a, m], UnsameQ, All], -1]
```

■ MinRestOrd

Berechnet zu gegebenem Testteiler d und Basis B das geordnete Paar bestehend aus dem kleinsten Rest $r = B^j$ mod d und der Ordnung von r:

```
MinRestOrd[B_, d_] := Module[{pml = PotModList[d][B], r},
  r = Min[pml];
  pos = Flatten[Position[pml, r]];
  {r, pos[[1]]}]
```

■ PosVonMinuseins

Prüft, ob der Reduktionsrest -1 auftritt, und gibt seine Position zurück (Position 0 bedeutet: Fehlanzeige):

```
PosVonMinuseins[B_, d_] := Module[{pml = PotModList[d][B]},
  pos = Flatten[Position[pml, d - 1]];
  If[pos == {}, 0, pos[[1]]]]
```

■ GesamtWert

Zu gegebener reellwertiger Bewertungsfunktion g wird die Summe aller $g(a)$ mit $a \in xlist$ berechnet:

```
GesamtWert[g_][xlist_] := Plus @@ Map[g, xlist]
```

■ Zwei Bewertungsfunktionen g_1 und g_2

g_2 erteilt einem Test der Ordnung s den Wert $\frac{1}{s}$. Der Test wird dabei als Quadrupel (B, d, ρ, s) notiert; g_1 erteilt konstant den Wert 1, d.h. realisiert via GesamtWert eine Anzahlbestimmung.

```
g2[test_] := 1 / test[[4]];
g1[test_] := 1
```

■ Stabdiagramm

Zeichnet ein hellgrau gefülltes Stabdiagramm zu *datliste* (Liste bestehend aus geordneten Paaren < Urbild, Bild >):

```
Needs["Graphics`Graphics`"]
```

```
Stabdiagramm[datliste_] :=
 BarChart[Map[Reverse, datliste], BarStyle → GrayLevel[.8]]
```

1.2 Funktionen, die Testmengen berechnen

■ TTest

Berechnet zu gegebener Basis *B* und gegebenem Testteiler *d* die Menge der Teilbarkeitstests von höchstens der Ordnung *m*:

```
TTest[m_][B_, d_] := Module[{ttestlist = {}, minrest, ord, posm1},
  {minrest, ord} = MinRestOrd[B, d];
  posm1 = PosVonMinuseins[B, d];
  If[(0 ≤ minrest ≤ 1) && (ord ≤ m), AppendTo[ttestlist, {B, d, minrest, ord}]];
  If[0 < posm1 ≤ m, AppendTo[ttestlist, {B, d, -1, posm1}]];
  ttestlist]
```

■ TTestDiv

Berechnet zu gegebener Basenmenge *BList* und gegebenem Testteiler *d* die Vereinigungsmenge TTest[*m*][*B*, *d*] für alle $B \in BList$:

```
TTestDiv[m_][BList_, d_] := Module[{ttdivlist = {}, i},
  For[i = 1, i ≤ Length[BList],
   ttdivlist = Join[ttdivlist, TTest[m][BList[[i]], d]]; i++];
  ttdlist]
```

■ TTestBas

Berechnet zu gegebener Basis *B* und gegebener Testteilermenge *dList* die Vereinigungsenge TTest[*m*][*B*, *d*] für alle $d \in dList$:

```
TTestBas[m_][B_, dList_] := Module[{ttbaslist = {}, i},
  For[i = 1, i ≤ Length[dList],
   ttbaslist = Join[ttbaslist, TTest[m][B, dList[[i]]]]; i++];
  ttbaslist]
```

■ TTestAll

Berechnet zu gegebener Basenmenge *BList* und gegebener Testteilermenge *dList* die Vereinigungsmenge der TTest[*m*][*B*, *d*] für alle $B \in BList$ und alle $d \in dList$:

```
TTestAll[m_][BList_, dList_] := Module[{ttallist = {}, i},
  For[i = 1, i ≤ Length[BList],
   ttallist = Join[ttallist, TTestBas[m][BList[[i]], dList]]; i++];
  ttallist]
```

■ TTestAuswahl

Berechnet aus einer Testmenge *ttList* die Menge aller Tests aus *ttList*, die den Reduktionsrest ρ haben:

```
TTestAuswahl[ttList_, rho_] := Select[ttList, Part[#, 3] == rho &]
```

1.3 Funktionen, die Testmengen bewerten bzw. kennzeichnen

■ TestSignatur

Berechnet zu gegebener Bewertungsfunktion *g* und gegebener Testmenge *ttList* das Tripel bestehend aus dem Gesamtwert (bzgl. *g*) jeweils von Endstellentests, von alternierenden Quersummentests und von Quersummentests:

```
TestSignatur[g_][ttList_] :=
 {GesamtWert[g][TTestAuswahl[ttList, 0]], GesamtWert[g][
   TTestAuswahl[ttList, -1]], GesamtWert[g][TTestAuswahl[ttList, 1]]}
```

■ TestAnzahl

Berechnet zu gegebener Basenmenge *BList* und gegebener Testteilermenge *dList* die Anzahl vorhandener brauchbarer (Ordnung ≤ 3) Tests:

```
TestAnzahl[BList_, dList_] := GesamtWert[g1][TTestAll[3][BList, dList]]
```

■ TestWert

Berechnet zu gegebener Basenmenge *BList* und gegebener Testteilermenge *dList* den Gesamtwert (bzgl. g_2) vorhandener brauchbarer (Ordnung ≤ 3) Tests:

```
TestWert[BList_, dList_] := GesamtWert[g2][TTestAll[3][BList, dList]]
```

■ TSignaturBas

Berechnet zu gegebener Basis *B* die drei relativen Wertanteile am Gesamtwert von Endstellentests, alternierenden Quersummentests und Quersummentests (als Tripel zurückgegeben):

```
TSignaturBas[B_] := Module[{tsg = TestSignatur[g2][TTestBas[3][B, Div2]]},
  N[tsg / Plus @@ tsg]]
```

■ PrefRestBas

Gibt den Reduktionsrest einer Basis *B* zurück, bei dessen zugehörigem Test im *B*-adischen Stellenwertsystem mit ein- und zweistelligen Testteilern der beste Effekt (d.i. der größte Wert) erzielt wird. Dabei bedeutet: 1 = E = Endstellentest, 2 = A = alternierender Quersummentest, 3 = Q = Quersummentest.

```
PrefRestBas[B_] := Module[{ts = TestSignatur[g2][TTestBas[3][B, Div2]]},
  Flatten[Position[ts, Max[ts]]]]
```

2 Untersuchung von Testmengen

2.1 Konstante Mengen

■ Basenmengen und Testteilermengen

```
(* gebräuchliche Stellenwertbasen von 2 bis 16 *)
Basm = Table[B, {B, 2, 16}];
 (* alle höchstens zweistelligen Testteiler *)
Div2 = Table[d, {d, 2, 99}];
(* alle höchstens zweistelligen Primteiler *)
Div2p = Select[Div2, PrimeQ];
```

■ Diverse Testmengen (Mengen von brauchbaren Teilbarkeitstests)

```
(* alle Tests *)
tmAlle = TTestAll[3][Basm, Div2];
(* alle Tests zu Primteilern *)
tmPrimAlle = TTestAll[3][Basm, Div2p];
(* alle Tests im Dezimalsystem *)
tmDezAlle = TTestBas[3][10, Div2];
```

■ Wie viele (brauchbare) Tests gibt es …

insgesamt?

```
Length[tmAlle]
```

```
327
```

zu Primteilern?

```
Length[tmPrimAlle]
```

```
114
```

im Dezimalsystem mit allen Testteilern?

```
Length[tmDezAlle]
```

```
21
```

Übersicht:

```
TTestBas[3][10, Div2] // TableForm
```

```
10      2       0       1
10      3       1       1
10      4       0       2
10      5       0       1
10      7       -1      3
10      8       0       3
10      9       1       1
10      10      0       1
10      11      1       2
10      11      -1      1
10      13      -1      3
10      20      0       2
10      25      0       2
10      27      1       3
10      33      1       2
10      37      1       3
10      40      0       3
10      50      0       2
10      77      -1      3
10      91      -1      3
10      99      1       2
```

2.2 Bewertung von Testmengen zu Testteilern

■ 2.2.1 Zu welchen Testteilern gibt es im Dezimalsystem (k)einen Test?

tab221: Anzahlen der Tests zu allen Testteilern im Dezimalsystem

```
tab221 = Table[{d, TestAnzahl[{10}, {d}]}, {d, 2, 99}];
```

```
Flatten[Drop[Select[tab221, Part[#, 2] == 0 &], {}, -1]]
Length[%]
```

```
{6, 12, 14, 15, 16, 17, 18, 19, 21, 22, 23, 24, 26, 28, 29,
 30, 31, 32, 34, 35, 36, 38, 39, 41, 42, 43, 44, 45, 46, 47,
 48, 49, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64,
 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 78, 79, 80, 81,
 82, 83, 84, 85, 86, 87, 88, 89, 90, 92, 93, 94, 95, 96, 97, 98}
```

```
78
```

Folgende 20 ein- bis zweistellige Testteiler haben einen (dezimalen) Test:

```
Drop[Select[tab221, Part[#, 2] > 0 &], {}, -1] // Flatten
```

```
{2, 3, 4, 5, 7, 8, 9, 10, 11, 13, 20, 25, 27, 33, 37, 40, 50, 77, 91, 99}
```

Bem.: 11 hat als einziger Testteiler 2 Tests.

■ 2.2.2 Zu welchen Testteilern gibt es in keinem Stellenwertsystem einen Test?

tab222: Anzahlen der Tests zu allen Testteilern in allen Stellenwertsystemen

```
tab222 = Table[{d, TestAnzahl[Basm, {d}]}, {d, 2, 99}];
```

Die kleinsten fünf (von 26) ohne irgendeinen Test: 22, 23, 44, 46, 47

```
Flatten[Drop[Select[tab222, Part[#, 2] == 0 &], {}, -1]]
Length[%]
```

```
{22, 23, 44, 46, 47, 53, 55, 58, 59, 66, 67, 68,
 69, 71, 76, 78, 79, 83, 87, 88, 89, 90, 92, 93, 94, 97}
```

```
26
```

Nur 1 Test haben:

```
Flatten[Drop[Select[tab222, Part[#, 2] == 1 &], {}, -1]]
Length[%]
```

```
{29, 30, 33, 34, 41, 51, 52, 60, 62, 70,
 74, 75, 77, 80, 81, 82, 84, 86, 95, 96, 98, 99}
```

```
22
```

■ Welche Testteiler haben die meisten Tests?

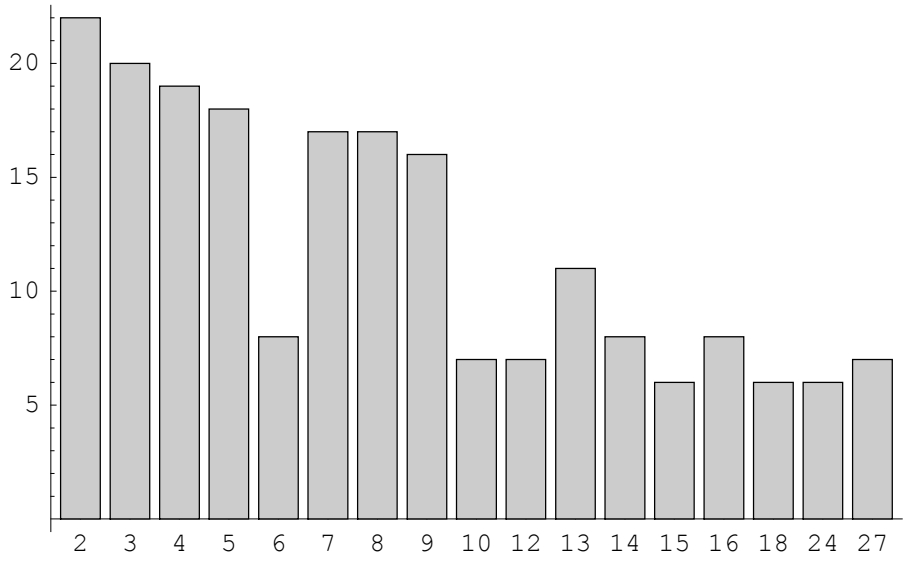
17 Spitzenreiter auf 10 Rangplätzen!

```
top10 = Select[tab222, Part[#, 2] ≥ 6 &];
Length[top10]
top10 // TableForm
```

```
17
```

```
2       22
3       20
4       19
5       18
6       8
7       17
8       17
9       16
10      7
12      7
13      11
14      8
15      6
16      8
18      6
24      6
27      7
```

```
Stabdiagramm[top10];
```



■ Welche Bewertung vereinigen die Top10-Teiler auf sich?

```
top10dliste = Flatten[Drop[Sort[top10, #2[[2]] < #1[[2]] &], {}, -1]]
```

```
{2, 3, 4, 5, 8, 7, 9, 13, 16, 14, 6, 27, 12, 10, 24, 18, 15}
```

```
TestAnzahl[Basm, top10dliste]
```

```
203
```

Anzahlmäßiger Anteil an allen Tests (ca. 62 %):

```
N[TestAnzahl[Basm, top10dliste] / Length[tmAlle]]
```

```
0.620795
```

Bewertungsmäßiger Anteil (ca. 72,4 %):

```
N[TestWert[Basm, top10dliste] / TestWert[Basm, Div2]]
```

```
0.72402
```

Bewertung der 10 auf den vordersten Rängen plazierten Teiler:

Anzahlmäßiger Anteil (ca. 47,7 %)

```
N[TestAnzahl[Basm, Take[top10dliste, 10]] / Length[tmAlle]]
```

```
0.477064
```

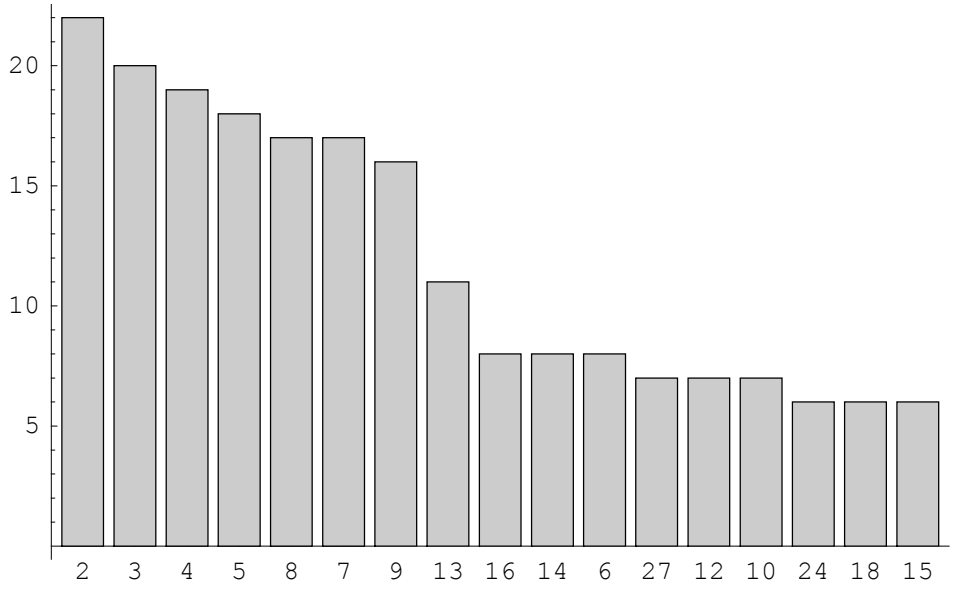
Bewertungsmäßiger Anteil (ca. 57,5 %):

```
N[TestWert[Basm, Take[top10dliste, 10]] / TestWert[Basm, Div2]]
```

```
0.574957
```

Schaubild zur Rangfolge der Testteiler:

```
bild1 = Stabdiagramm[Sort[top10, #2[[2]] < #1[[2]] &]];
```



2.3 Bewertung von Testmengen zu Basen

■ 2.3.1 Vorbereitende Definitionen

Berechnung der Testmenge:

```
TmBas[B_] := TTestBas[3][B, Div2]
```

Bewertungsfunktion:

```
TBewBas[g_][B_] := GesamtWert[g][TmBas[B]]
```

■ 2.3.2 Bewertung aller Tests zu gegebener Basis

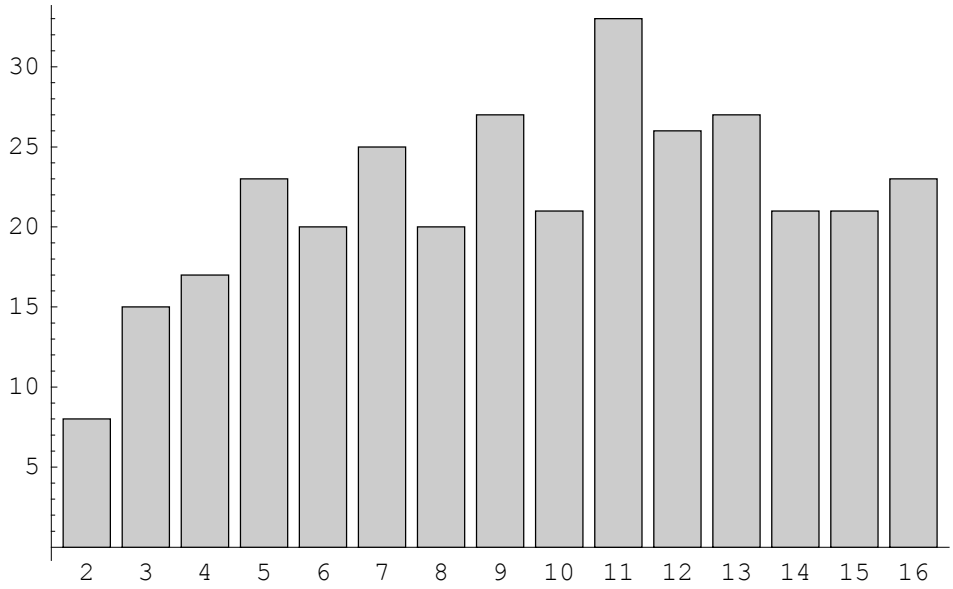
der Anzahl nach:

```
tab232a = Table[{B, TBewBas[g1][B]}, {B, 2, 16}];
```

```
tab232a
```

```
{{2, 8}, {3, 15}, {4, 17}, {5, 23}, {6, 20}, {7, 25}, {8, 20}, {9, 27},
 {10, 21}, {11, 33}, {12, 26}, {13, 27}, {14, 21}, {15, 21}, {16, 23}}
```

```
Stabdiagramm[tab232a];
```



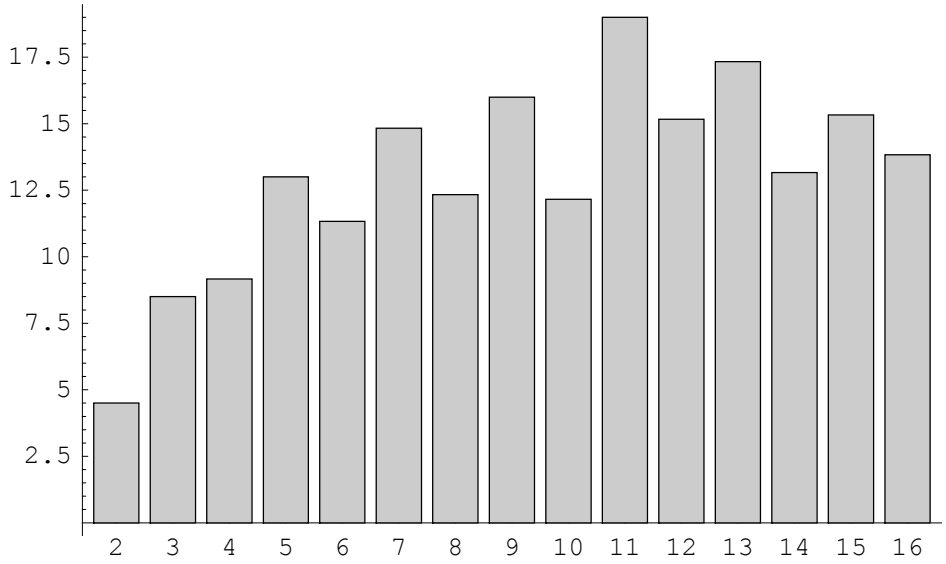
der Bewertungszahl nach:

```
tab232b = Table[{B, N[TBewBas[g2][B]]}, {B, 2, 16}];
```

```
tab232b // TableForm
```

```
2       4.5
3       8.5
4       9.16667
5       13.
6       11.3333
7       14.8333
8       12.3333
9       16.
10      12.1667
11      19.
12      15.1667
13      17.3333
14      13.1667
15      15.3333
16      13.8333
```

```
Stabdiagramm[tab232b];
```



■ 2.3.3 Statistische Vergleichszahlen

Mittlere Anzahl:

```
mw232a = Mean[Flatten[Drop[tab232a, {}, 1]]] // N
```

```
21.8
```

Es gibt 8 unterdurchschnittliche Basen, zu denen 10 gehört (auf gleicher Höhe mit 14, 15):

```
Select[tab232a, Part[#, 2] < mw232a &]
```

```
{{2, 8}, {3, 15}, {4, 17}, {6, 20}, {8, 20}, {10, 21}, {14, 21}, {15, 21}}
```

Median/Zentralwert der Anzahl:

```
med232a = Median[Flatten[Drop[tab232a, {}, 1]]]
```

```
21
```

In der unteren Hälfte gibt es 8 Basen, darunter 10, sowie 12 (exakt der Median):

```
Select[tab232a, Part[#, 2] ≤ med232a &]
```

```
{{2, 8}, {3, 15}, {4, 17}, {6, 20}, {8, 20}, {10, 21}, {14, 21}, {15, 21}}
```

Standarabweichung (Anzahl):

```
StandardDeviation[Flatten[Drop[tab232a, {}, 1]]] // N
```

```
5.84563
```

Mittlere Bewertungszahl:

```
mw232b = Mean[Flatten[Drop[tab232b, {}, 1]]] // N
```

```
13.0444
```

Es gibt 7 unterdurchschnittliche Basen, zu denen wiederum 10 gehört:

```
Select[tab232b, Part[#, 2] < mw232b &]
```

```
{{2, 4.5}, {3, 8.5}, {4, 9.16667},
 {5, 13.}, {6, 11.3333}, {8, 12.3333}, {10, 12.1667}}
```

Median der Bewertungszahl:

```
med232b = Median[Flatten[Drop[tab232b, {}, 1]]]
```

```
13.1667
```

In der unteren Hälfte gibt es 8 Basen, darunter wieder 10, sowie 14; $B = 12$ ist nicht mehr dabei:

```
Select[tab232b, Part[#, 2] ≤ med232b &] // TableForm
```

```
2       4.5
3       8.5
4       9.16667
5       13.
6       11.3333
8       12.3333
10      12.1667
14      13.1667
```

Standardabweichung (Bewertungszahl):

```
StandardDeviation[Flatten[Drop[tab232b, {}, 1]]] // N
```

```
3.67503
```

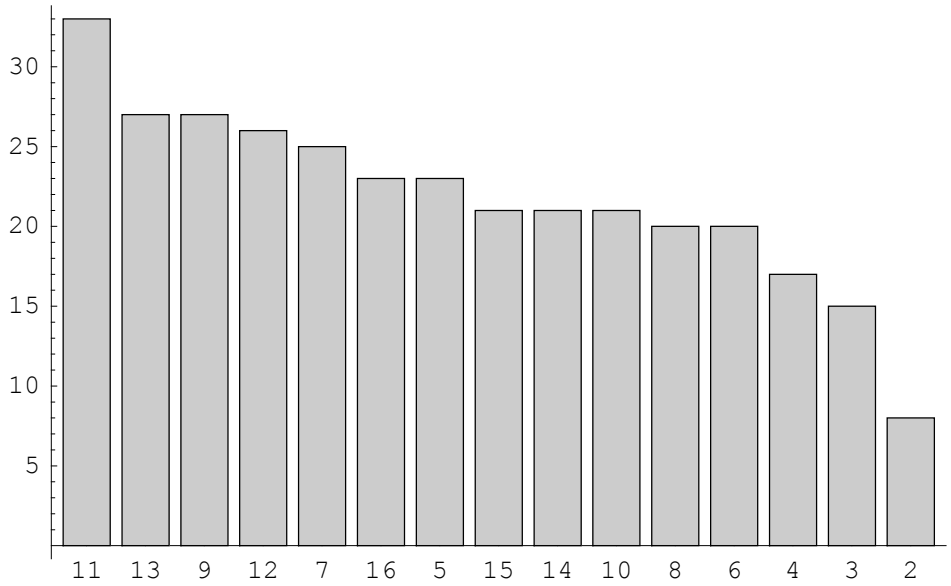
■ 2.3.4 Tabellen und Schaubilder zur Rangfolge

nach Anzahl:

```
Sort[tab232a, #2[[2]] < #1[[2]] &]
```

```
{{11, 33}, {13, 27}, {9, 27}, {12, 26}, {7, 25}, {16, 23}, {5, 23}, {15, 21},
 {14, 21}, {10, 21}, {8, 20}, {6, 20}, {4, 17}, {3, 15}, {2, 8}}
```

```
Stabdiagramm[Sort[tab232a, #2[[2]] < #1[[2]] &]];
```

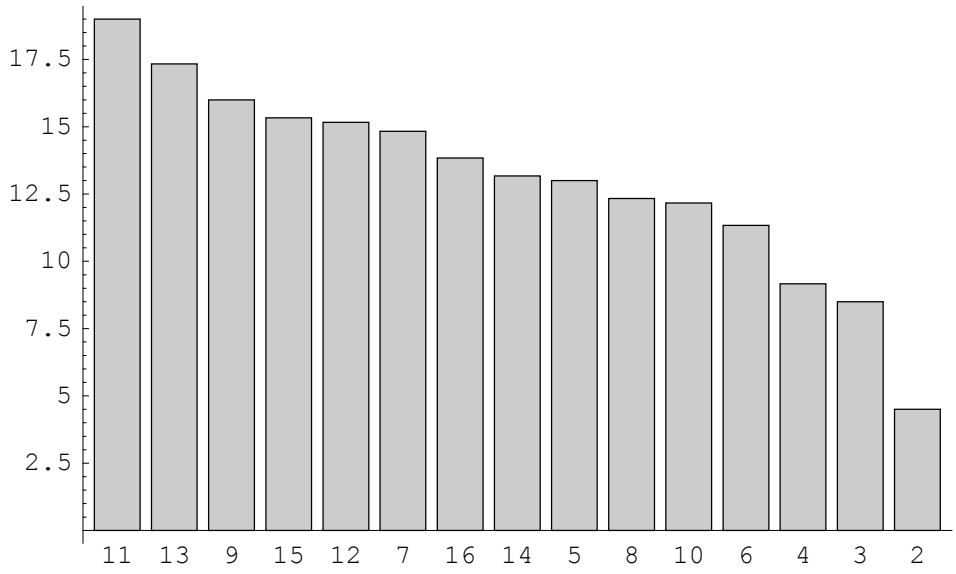


nach Bewertungszahl:

```
Sort[tab232b, #2[[2]] < #1[[2]] &]
```

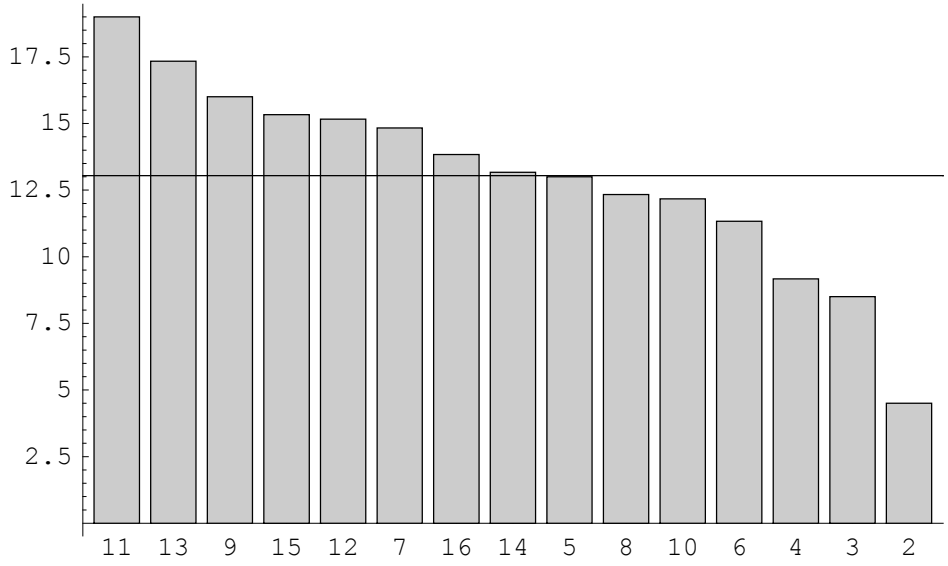
```
{{11, 19.}, {13, 17.3333}, {9, 16.}, {15, 15.3333}, {12, 15.1667},
 {7, 14.8333}, {16, 13.8333}, {14, 13.1667}, {5, 13.}, {8, 12.3333},
 {10, 12.1667}, {6, 11.3333}, {4, 9.16667}, {3, 8.5}, {2, 4.5}}
```

```
tdiag = Stabdiagramm[Sort[tab232b, #2[[2]] < #1[[2]] &]];
```



Rangschaubild mit eingezeichnetem Mittelwert (13,0444):

```
bild2 = DisplayTogether[{tdiag, Plot[mw232b, {x, 0, 17}]}];
```



■ 2.3.5 Bevorzugte Reduktionsreste

```
Table[{B, List[PrefRestBas[B]]}, {B, 2, 16}] // TableForm
```

```
2       1  2
3       2
4       1
5       2
6       1
7       3
8       1  2
9       3
10      1
11      3
12      1
13      3
14      1
15      3
16      3
```

```
TestSignatur[g2][TTestBas[3][6, Div2]] // N
```

```
{7.16667, 1.83333, 2.33333}
```

2.4 Übersicht über alle Teilbarkeitstests

■ 2.4.1 Alle Endstellentests in der Folge ihrer Ordnung

```
tmEndAlle = Sort[Select[tmAlle, Part[#, 3] == 0 &], Part[#1, 4] < Part[#2, 4] &];
```

Alle Paare (*B*, *d*) mit Test-Ordnung 1:

```
Select[tmEndAlle, Part[#, 4] == 1 &]
Length[%]
```

```
{{16, 16, 0, 1}, {16, 8, 0, 1}, {16, 4, 0, 1}, {16, 2, 0, 1}, {15, 15, 0, 1},
 {15, 5, 0, 1}, {15, 3, 0, 1}, {14, 14, 0, 1}, {14, 7, 0, 1}, {14, 2, 0, 1},
 {13, 13, 0, 1}, {12, 12, 0, 1}, {12, 6, 0, 1}, {12, 4, 0, 1}, {12, 3, 0, 1},
 {12, 2, 0, 1}, {11, 11, 0, 1}, {10, 10, 0, 1}, {10, 5, 0, 1},
 {10, 2, 0, 1}, {9, 9, 0, 1}, {9, 3, 0, 1}, {8, 8, 0, 1}, {8, 4, 0, 1},
 {8, 2, 0, 1}, {7, 7, 0, 1}, {6, 6, 0, 1}, {6, 3, 0, 1}, {6, 2, 0, 1},
 {5, 5, 0, 1}, {4, 4, 0, 1}, {4, 2, 0, 1}, {3, 3, 0, 1}, {2, 2, 0, 1}}
```

```
34
```

Trivial ist: Alle Testteiler sind Teiler der Basis. Daher kann die Menge "verkleinert" werden: Zu jeder Basis B gibt es einen Endstellentest der Ordnung 1 für alle Testteiler $d \mid B$.

```
Union[Flatten[Drop[Select[tmEndAlle, Part[#, 4] == 1 &], {}, -3]]]
```

```
{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16}
```

Alle Paare (B, d) mit Test-Ordnung 2:

```
Select[tmEndAlle, Part[#, 4] == 2 &]
Length[%]
```

```
{{16, 64, 0, 2}, {16, 32, 0, 2}, {15, 75, 0, 2}, {15, 45, 0, 2},
 {15, 25, 0, 2}, {15, 9, 0, 2}, {14, 98, 0, 2}, {14, 49, 0, 2},
 {14, 28, 0, 2}, {14, 4, 0, 2}, {12, 72, 0, 2}, {12, 48, 0, 2},
 {12, 36, 0, 2}, {12, 24, 0, 2}, {12, 18, 0, 2}, {12, 16, 0, 2},
 {12, 9, 0, 2}, {12, 8, 0, 2}, {10, 50, 0, 2}, {10, 25, 0, 2},
 {10, 20, 0, 2}, {10, 4, 0, 2}, {9, 81, 0, 2}, {9, 27, 0, 2},
 {8, 64, 0, 2}, {8, 32, 0, 2}, {8, 16, 0, 2}, {7, 49, 0, 2},
 {6, 36, 0, 2}, {6, 18, 0, 2}, {6, 12, 0, 2}, {6, 9, 0, 2}, {6, 4, 0, 2},
 {5, 25, 0, 2}, {4, 16, 0, 2}, {4, 8, 0, 2}, {3, 9, 0, 2}, {2, 4, 0, 2}}
```

```
38
```

```
Drop[Select[tmEndAlle, Part[#, 4] == 2 &], {}, -2]
```

```
{{16, 64}, {16, 32}, {15, 75}, {15, 45}, {15, 25}, {15, 9}, {14, 98},
 {14, 49}, {14, 28}, {14, 4}, {12, 72}, {12, 48}, {12, 36}, {12, 24},
 {12, 18}, {12, 16}, {12, 9}, {12, 8}, {10, 50}, {10, 25}, {10, 20},
 {10, 4}, {9, 81}, {9, 27}, {8, 64}, {8, 32}, {8, 16}, {7, 49}, {6, 36},
 {6, 18}, {6, 12}, {6, 9}, {6, 4}, {5, 25}, {4, 16}, {4, 8}, {3, 9}, {2, 4}}
```

Alle Paare (B, d) mit Test-Ordnung 3:

```
Select[tmEndAlle, Part[#, 4] == 3 &]
Length[%]
```

```
{{15, 27, 0, 3}, {14, 56, 0, 3}, {14, 8, 0, 3},
 {12, 96, 0, 3}, {12, 64, 0, 3}, {12, 54, 0, 3}, {12, 32, 0, 3},
 {12, 27, 0, 3}, {10, 40, 0, 3}, {10, 8, 0, 3}, {6, 72, 0, 3},
 {6, 54, 0, 3}, {6, 27, 0, 3}, {6, 24, 0, 3}, {6, 8, 0, 3},
 {4, 64, 0, 3}, {4, 32, 0, 3}, {3, 27, 0, 3}, {2, 8, 0, 3}}
```

```
19
```

```
Drop[Select[tmEndAlle, Part[#, 4] == 3 &], {}, -2]
```

```
{{15, 27}, {14, 56}, {14, 8}, {12, 96}, {12, 64}, {12, 54},
 {12, 32}, {12, 27}, {10, 40}, {10, 8}, {6, 72}, {6, 54},
 {6, 27}, {6, 24}, {6, 8}, {4, 64}, {4, 32}, {3, 27}, {2, 8}}
```

■ 2.4.2 Alle alternierenden Quersummentests in der Folge ihrer Ordnung

```
tmAQsAlle = Sort[Select[tmAlle, Part[#, 3] == -1 &], Part[#1, 4] < Part[#2, 4] &];
```

Alle Paare (B, d) mit Test-Ordnung 1:

```
Select[tmAQsAlle, Part[#, 4] == 1 &]
Length[%]
```

```
{{16, 17, -1, 1}, {15, 16, -1, 1}, {15, 8, -1, 1}, {15, 4, -1, 1},
 {15, 2, -1, 1}, {14, 15, -1, 1}, {14, 5, -1, 1}, {14, 3, -1, 1},
 {13, 14, -1, 1}, {13, 7, -1, 1}, {13, 2, -1, 1}, {12, 13, -1, 1},
 {11, 12, -1, 1}, {11, 6, -1, 1}, {11, 4, -1, 1}, {11, 3, -1, 1},
 {11, 2, -1, 1}, {10, 11, -1, 1}, {9, 10, -1, 1}, {9, 5, -1, 1},
 {9, 2, -1, 1}, {8, 9, -1, 1}, {8, 3, -1, 1}, {7, 8, -1, 1}, {7, 4, -1, 1},
 {7, 2, -1, 1}, {6, 7, -1, 1}, {5, 6, -1, 1}, {5, 3, -1, 1}, {5, 2, -1, 1},
 {4, 5, -1, 1}, {3, 4, -1, 1}, {3, 2, -1, 1}, {2, 3, -1, 1}}
```

```
34
```

Zu jeder Basis B gibt es einen alternierenden Quersummentest der Ordnung 1.

```
Union[Flatten[Drop[Select[tmAQsAlle, Part[#, 4] == 1 &], {}, -3]]]
```

```
{2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16}
```

Alle Paare (*B*, *d*) mit Test-Ordnung 2:

```
Select[tmAQsAlle, Part[#, 4] == 2 &]
Length[%]
```

```
{{13, 85, -1, 2}, {13, 34, -1, 2}, {13, 17, -1, 2}, {13, 10, -1, 2},
 {13, 5, -1, 2}, {12, 29, -1, 2}, {12, 5, -1, 2}, {11, 61, -1, 2},
 {9, 82, -1, 2}, {9, 41, -1, 2}, {8, 65, -1, 2}, {8, 13, -1, 2},
 {8, 5, -1, 2}, {7, 50, -1, 2}, {7, 25, -1, 2}, {7, 10, -1, 2},
 {7, 5, -1, 2}, {6, 37, -1, 2}, {5, 26, -1, 2}, {5, 13, -1, 2},
 {4, 17, -1, 2}, {3, 10, -1, 2}, {3, 5, -1, 2}, {2, 5, -1, 2}}
```

```
24
```

```
Drop[Select[tmAQsAlle, Part[#, 4] == 2 &], {}, -2]
```

```
{{13, 85}, {13, 34}, {13, 17}, {13, 10}, {13, 5},
 {12, 29}, {12, 5}, {11, 61}, {9, 82}, {9, 41}, {8, 65},
 {8, 13}, {8, 5}, {7, 50}, {7, 25}, {7, 10}, {7, 5},
 {6, 37}, {5, 26}, {5, 13}, {4, 17}, {3, 10}, {3, 5}, {2, 5}}
```

Alle Paare (*B*, *d*) mit Test-Ordnung 3:

```
Select[tmAQsAlle, Part[#, 4] == 3 &]
Length[%]
```

```
{{14, 61, -1, 3}, {14, 45, -1, 3}, {14, 9, -1, 3},
 {12, 91, -1, 3}, {12, 19, -1, 3}, {12, 7, -1, 3}, {11, 74, -1, 3},
 {11, 37, -1, 3}, {11, 36, -1, 3}, {11, 18, -1, 3}, {11, 9, -1, 3},
 {10, 91, -1, 3}, {10, 77, -1, 3}, {10, 13, -1, 3}, {10, 7, -1, 3},
 {9, 73, -1, 3}, {8, 57, -1, 3}, {8, 27, -1, 3}, {8, 19, -1, 3},
 {7, 86, -1, 3}, {7, 43, -1, 3}, {6, 31, -1, 3}, {5, 63, -1, 3},
 {5, 42, -1, 3}, {5, 21, -1, 3}, {5, 18, -1, 3}, {5, 14, -1, 3},
 {5, 9, -1, 3}, {5, 7, -1, 3}, {4, 65, -1, 3}, {4, 13, -1, 3},
 {3, 28, -1, 3}, {3, 14, -1, 3}, {3, 7, -1, 3}, {2, 9, -1, 3}}
```

```
35
```

```
Drop[Select[tmAQsAlle, Part[#, 4] == 3 &], {}, -2]
```

```
{{14, 61}, {14, 45}, {14, 9}, {12, 91}, {12, 19}, {12, 7},
 {11, 74}, {11, 37}, {11, 36}, {11, 18}, {11, 9}, {10, 91}, {10, 77},
 {10, 13}, {10, 7}, {9, 73}, {8, 57}, {8, 27}, {8, 19}, {7, 86},
 {7, 43}, {6, 31}, {5, 63}, {5, 42}, {5, 21}, {5, 18}, {5, 14},
 {5, 9}, {5, 7}, {4, 65}, {4, 13}, {3, 28}, {3, 14}, {3, 7}, {2, 9}}
```

■ 2.4.3 Alle Quersummentests in der Folge ihrer Ordnung

```
tmQsAlle = Sort[Select[tmAlle, Part[#, 3] == 1 &], Part[#1, 4] < Part[#2, 4] &];
```

Alle Paare (B, d) mit Test-Ordnung 1:

```
Select[tmQsAlle, Part[#, 4] == 1 &]
Length[%]
```

```
{{16, 15, 1, 1}, {16, 5, 1, 1}, {16, 3, 1, 1}, {15, 14, 1, 1}, {15, 7, 1, 1},
 {15, 2, 1, 1}, {14, 13, 1, 1}, {13, 12, 1, 1}, {13, 6, 1, 1}, {13, 4, 1, 1},
 {13, 3, 1, 1}, {13, 2, 1, 1}, {12, 11, 1, 1}, {11, 10, 1, 1}, {11, 5, 1, 1},
 {11, 2, 1, 1}, {10, 9, 1, 1}, {10, 3, 1, 1}, {9, 8, 1, 1}, {9, 4, 1, 1},
 {9, 2, 1, 1}, {8, 7, 1, 1}, {7, 6, 1, 1}, {7, 3, 1, 1}, {7, 2, 1, 1},
 {6, 5, 1, 1}, {5, 4, 1, 1}, {5, 2, 1, 1}, {4, 3, 1, 1}, {3, 2, 1, 1}}
```

```
30
```

Zu jeder Basis B gibt es einen Quersummentest der Ordnung 1.

```
Union[Flatten[Drop[Select[tmQsAlle, Part[#, 4] == 1 &], {}, -3]]]
```

```
{3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16}
```

Alle Paare (B, d) mit Test-Ordnung 2:

```
Select[tmQsAlle, Part[#, 4] == 2 &]
Length[%]
```

```
{{16, 85, 1, 2}, {16, 51, 1, 2}, {16, 17, 1, 2}, {15, 56, 1, 2},
 {15, 32, 1, 2}, {15, 28, 1, 2}, {15, 16, 1, 2}, {15, 8, 1, 2},
 {15, 4, 1, 2}, {14, 65, 1, 2}, {14, 39, 1, 2}, {14, 15, 1, 2},
 {14, 5, 1, 2}, {14, 3, 1, 2}, {13, 84, 1, 2}, {13, 56, 1, 2},
 {13, 42, 1, 2}, {13, 28, 1, 2}, {13, 24, 1, 2}, {13, 21, 1, 2},
 {13, 14, 1, 2}, {13, 8, 1, 2}, {13, 7, 1, 2}, {12, 13, 1, 2},
 {11, 60, 1, 2}, {11, 40, 1, 2}, {11, 30, 1, 2}, {11, 24, 1, 2},
 {11, 20, 1, 2}, {11, 15, 1, 2}, {11, 12, 1, 2}, {11, 8, 1, 2},
 {11, 6, 1, 2}, {11, 4, 1, 2}, {11, 3, 1, 2}, {10, 99, 1, 2}, {10, 33, 1, 2},
 {10, 11, 1, 2}, {9, 80, 1, 2}, {9, 40, 1, 2}, {9, 20, 1, 2},
 {9, 16, 1, 2}, {9, 10, 1, 2}, {9, 5, 1, 2}, {8, 63, 1, 2}, {8, 21, 1, 2},
 {8, 9, 1, 2}, {8, 3, 1, 2}, {7, 48, 1, 2}, {7, 24, 1, 2}, {7, 16, 1, 2},
 {7, 12, 1, 2}, {7, 8, 1, 2}, {7, 4, 1, 2}, {6, 35, 1, 2}, {6, 7, 1, 2},
 {5, 24, 1, 2}, {5, 12, 1, 2}, {5, 8, 1, 2}, {5, 6, 1, 2}, {5, 3, 1, 2},
 {4, 15, 1, 2}, {4, 5, 1, 2}, {3, 8, 1, 2}, {3, 4, 1, 2}, {2, 3, 1, 2}}
```

```
66
```

```
Drop[Select[tmQsAlle, Part[#, 4] == 2 &], {}, -2]
```

```
{{16, 85}, {16, 51}, {16, 17}, {15, 56}, {15, 32}, {15, 28}, {15, 16},
 {15, 8}, {15, 4}, {14, 65}, {14, 39}, {14, 15}, {14, 5}, {14, 3},
 {13, 84}, {13, 56}, {13, 42}, {13, 28}, {13, 24}, {13, 21}, {13, 14},
 {13, 8}, {13, 7}, {12, 13}, {11, 60}, {11, 40}, {11, 30}, {11, 24},
 {11, 20}, {11, 15}, {11, 12}, {11, 8}, {11, 6}, {11, 4}, {11, 3},
 {10, 99}, {10, 33}, {10, 11}, {9, 80}, {9, 40}, {9, 20}, {9, 16},
 {9, 10}, {9, 5}, {8, 63}, {8, 21}, {8, 9}, {8, 3}, {7, 48}, {7, 24},
 {7, 16}, {7, 12}, {7, 8}, {7, 4}, {6, 35}, {6, 7}, {5, 24}, {5, 12},
 {5, 8}, {5, 6}, {5, 3}, {4, 15}, {4, 5}, {3, 8}, {3, 4}, {2, 3}}
```

Alle Paare (B, d) mit Test-Ordnung 3:

```
Select[tmAQsAlle, Part[#, 4] == 3 &]
Length[%]
```

```
{{14, 61, -1, 3}, {14, 45, -1, 3}, {14, 9, -1, 3},
 {12, 91, -1, 3}, {12, 19, -1, 3}, {12, 7, -1, 3}, {11, 74, -1, 3},
 {11, 37, -1, 3}, {11, 36, -1, 3}, {11, 18, -1, 3}, {11, 9, -1, 3},
 {10, 91, -1, 3}, {10, 77, -1, 3}, {10, 13, -1, 3}, {10, 7, -1, 3},
 {9, 73, -1, 3}, {8, 57, -1, 3}, {8, 27, -1, 3}, {8, 19, -1, 3},
 {7, 86, -1, 3}, {7, 43, -1, 3}, {6, 31, -1, 3}, {5, 63, -1, 3},
 {5, 42, -1, 3}, {5, 21, -1, 3}, {5, 18, -1, 3}, {5, 14, -1, 3},
 {5, 9, -1, 3}, {5, 7, -1, 3}, {4, 65, -1, 3}, {4, 13, -1, 3},
 {3, 28, -1, 3}, {3, 14, -1, 3}, {3, 7, -1, 3}, {2, 9, -1, 3}}
```

```
35
```

```
Drop[Select[tmQsAlle, Part[#, 4] == 3 &], {}, -2]
```

```
{{16, 91}, {16, 65}, {16, 63}, {16, 45}, {16, 39}, {16, 35},
 {16, 21}, {16, 13}, {16, 9}, {16, 7}, {13, 61}, {13, 36}, {13, 18},
 {13, 9}, {11, 95}, {11, 70}, {11, 38}, {11, 35}, {11, 19},
 {11, 14}, {11, 7}, {10, 37}, {10, 27}, {9, 91}, {9, 56}, {9, 52},
 {9, 28}, {9, 26}, {9, 14}, {9, 13}, {9, 7}, {8, 73}, {7, 57},
 {7, 38}, {7, 19}, {7, 18}, {7, 9}, {6, 43}, {5, 62}, {5, 31},
 {4, 63}, {4, 21}, {4, 9}, {4, 7}, {3, 26}, {3, 13}, {2, 7}}
```