8. You're doing some stress-testing on various models of glass jars to determine the height from which they can be dropped and still not break. The setup for this experiment, on a particular type of jar, is as follows. You have a ladder with n rungs, and you want to find the highest rung from which you can drop a copy of the jar and not have it break. We call this the *highest safe rung*.

It might be natural to try binary search: drop a jar from the middle rung, see if it breaks, and then recursively try from rung $n/4$ or $3n/4$ depending on the outcome. But this has the drawback that you could break a lot of jars in finding the answer.

If your primary goal were to conserve jars, on the other hand, you could try the following strategy. Start by dropping a jar from the first rung, then the second rung, and so forth, climbing one higher each time until the jar breaks. In this way, you only need a single jar—at the moment it breaks, you have the correct answer—but you may have to drop it n times (rather than $\log n$ as in the binary search solution).

So here is the trade-off: it seems you can perform fewer drops if you're willing to break more jars. To understand better how this trade-off works at a quantitative level, let's consider how to run this experiment given a fixed "budget" of $k \geq 1$ jars. In other words, you have to determine the correct answer—the highest safe rung—and can use at most k jars in doing so.

(a) Suppose you are given a budget of $k = 2$ jars. Describe a strategy for finding the highest safe rung that requires you to drop a jar at most $f(n)$ times, for some function $f(n)$ that grows slower than linearly. (In other words, it should be the case that $\lim_{n\to\infty} f(n)/n = 0$.)

(b) Now suppose you have a budget of $k > 2$ jars, for some given k. Describe a strategy for finding the highest safe rung using at most k jars. If $f_k(n)$ denotes the number of times you need to drop a jar according to your strategy, then the functions $f_1, f_2, f_3, \ldots$ should have the property that each grows asymptotically slower than the previous one: $\lim_{n\to\infty} f_k(n)/f_{k-1}(n) = 0$ for each k.

100 קומות

צנצנת 1 :
1
2
3
4
...
100

מקסימום
זריקות -
100

2 צנצנות :
14
27
39
50
60
69
77
84
90
95
99

מקסימום
זריקות -
14

הבעיה :
קשה
להכללה

2 צנצנות :
10
20
30
40
50
60
70
80
90
100

מקסימום
זריקות –
2 * N^1/2

הבעיה :
איך
מכלילים ?

3 צנצנות :
100
200
300
400
500
600
700
800
900
1000

מקסימום
זריקות –
3 * N^1/3

K צנצנות :
צנצנת ראשונה -
N^((k-1)/k)
צנצנת שנייה -
N^((k-2)/k)
צנצנת שלישית -
N^((k-3)/k)

צנצנתK-
N^((k-k)/k)

מקסימום
זריקות –
K * N^(1/k)