Issue Date: 12.05.2014
Version: 2014-05-19 15:04

Exercise Sheet 4 for Algorithm Engineering, SS 14

Hand In: *Until Monday, 19.05.2014, 10:00 am, email to* `wild@cs...` *or in lecture.*

Problem 8

$2+1+1+2+3$ points

Use generating functions to count the following sets of objects.

Hint: You may use the Mathematica function `SeriesCoefficient` (or equivalent functions of other computer algebra systems) for this task. A simple version is available on our website: `http://wwwagak.cs.uni-kl.de/mathe-tools.html`
(currently only in German; use button „Koeffizient" in section „Potenzreihenentwicklung")

a) *Partitions* of $n = 41$, i. e., representations of n as the sum of non-zero natural numbers, where the order of summands is ignored.

For example, $n = 4$ has 5 different partitions, namely

$$4, \qquad 3+1, \qquad 2+2$$
$$2+1+1, \qquad 1+1+1+1.$$

b) *Compositions* of $n = 41$, i. e., representations of n as the sum of non-zero natural numbers, where the order of summands is important.

For example, $n = 4$ has 8 different compositions, namely

$$4, \qquad 3+1, \qquad 2+2, \qquad 2+1+1$$
$$1+3, \qquad 1+2+1, \qquad 1+1+2, \qquad 1+1+1+1.$$

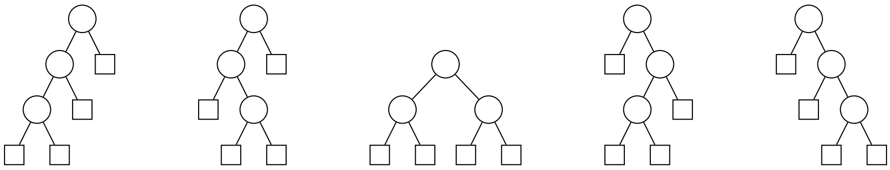
c) Partitions of $n = 41$ with distinct parts, i. e., representations of n as the sum of pairwise different, non-zero natural numbers, where the order of summands is ignored.

$n = 4$ has 2 partitions with distinct parts, namely

$$4 \qquad \text{and} \qquad 3+1.$$

d) (Extended) binary trees with 13 inner nodes.

For example, there are the following 5 extended binary trees with 3 inner nodes.



e) RNA secondary structures of length 21, where we model RNA secondary structures as words over the alphabet {(, •,)}, satisfying the following conditions:

(1) The number of opening and closing parentheses is identical.

(2) No prefix of the word contains more closing parentheses than opening ones.

(3) The string () does not occur as a substring.

(A string satisfying (1) and (2) is called *correctly parenthesized.*)

For example, there are the 8 structures of length 5:

•••••, (•••), (••)•, (•)••,
•(••), •(•)•, ••(•), ((•)).

Problem 9

4 points

Give an efficient algorithm in pseudocode that inserts a single element into a jumplist. Argue why your algorithm keeps the property that all list structures are equally likely.