266 (partitions) A list of positive integers is called a partition of natural number n if the sum of its items is n. Write a program to find

(a) a list of all partitions of a given natural n. For example, if n=3 then an acceptable answer is [[3]; [1; 2]; [2; 1]; [1; 1; 1]] .

(b) a list of all sorted partitions of a given natural n. For example, if n=3 then an acceptable answer is [[3]; [1; 2]; [1; 1; 1]] .

(c) the sorted list of all partitions of a given natural n. For example, if n=3 then the answer is [[1; 1; 1]; [1; 2]; [2; 1]; [3]] .

(d) the sorted list of all sorted partitions of a given natural n. For example, if n=3 then the answer is [[1; 1; 1]; [1; 2]; [3]] .

After trying the question, scroll down to the solution.

(a) a list of all partitions of a given natural n . For example, if $n=3$ then an acceptable answer is [[3]; [1; 2]; [2; 1]; [1; 1; 1]] .

§ Part (a) is subsumed by part (c).

(b) a list of all sorted partitions of a given natural n . For example, if $n=3$ then an acceptable answer is [[3]; [1; 2]; [1; 1; 1]] .

§ Part (b) is subsumed by part (d).

(c) the sorted list of all partitions of a given natural n . For example, if $n=3$ then the answer is [[1; 1; 1]; [1; 2]; [2; 1]; [3]] .

§ Given a partition, to get the next partition: cut off the final item; increase the new final item by 1 ; join 1s as necessary to make up the right sum (easily determined from the item that was cut off). Let L: [*[*(nat+1)]] be a list-of-partitions variable whose final value is what we want. Then the problem is R , defined as

$$\begin{aligned} R \;&=\; (\,L' \text{ is the sorted list of all partitions of } n\,) \\ &=\; \quad (\forall i,j\colon 0,..\#L'\cdot\ i<j \Rightarrow L'i<L'j) \\ &\quad\wedge\ (\forall i\colon 0,..\#L'\cdot(\Sigma\, L'i) = n) \\ &\quad\wedge\ (\forall Q\colon [*(nat+1)]\cdot(\Sigma Q)=n \Rightarrow Q\colon L'(0,..\#L')) \end{aligned}$$

Introduce partition variable P: [*(nat+1)] and define

$$\begin{aligned} A \;&=\; (\,P \text{ is a partition}) \wedge (\,L \text{ is the sorted list of all partitions of } n \text{ that precede } P\,) \\ &=\; \quad (\Sigma P)=n \\ &\quad\wedge\ (\forall i,j\colon 0,..\#L\cdot\ i<j \Rightarrow L\,i < L\,j < P) \\ &\quad\wedge\ (\forall i\colon 0,..\#L\cdot(\Sigma\, L\,i) = n) \\ &\quad\wedge\ (\forall Q\colon [*(nat+1)]\cdot(\Sigma Q)=n \wedge Q<P \;\Rightarrow\; Q\colon L(0,..\#L)) \end{aligned}$$

Now the refinements.

$$\begin{aligned} &R \;\Leftarrow\; L:= [nil].\ \ P:= [n*1].\ \ A\Rightarrow R \\ &A\Rightarrow R \;\Leftarrow\; L:= L;;[P]. \\ &\qquad\qquad \textbf{if } \#P < 2 \textbf{ then } ok \\ &\qquad\qquad \textbf{else } P:= P[0;..\#P-2]\ ;;\ [P(\#P-2)+1\ ;\ (P(\#P-1)-1)*1].\ \ A\Rightarrow R \textbf{ fi} \end{aligned}$$

Here is a program using loops (Chapter 5); instead of gathering the partitions into a list, I print them (let's say $!x$ prints the value of x and $?x$ reads into variable x).

```
var n, m: int·          n is the length of P and m is a temporary
!"n=". ?n.
var P: [n*int]·
for i:= 0,..n do P i:= 1 od.
do    for i:= 0,..n do !P i, " " od. !newline.
      exit when n<2.
      P(n–2):= P(n–2)+1.
      m:= n–1. n:= m + P m – 1.
      for i:= m,..n do P i:= 1 od od
```

The exact execution time is obtained by putting $t:= t+1$ in front of the recursive call, and

replace R by $t' = t + 2^{n-1}-1$

replace $A\Rightarrow R$ by $t' = t + \Sigma i\colon 1,..\#P\cdot\ 2^{(\Sigma P[i;..\#P])-1}$

(d) the sorted list of all sorted partitions of a given natural n . For example, if $n=3$ then the answer is [[1; 1; 1]; [1; 2]; [3]] .

§ Given a sorted partition, to get the next sorted partition: cut off the final item; increase the new final item by 1 and call this f ; join as many fs as possible without making the sum too big; increase the final item to get the right sum. This solution is very similar to part (c), but the assignment

$$P:= P[0;..\#P-2]\ ;;\ [P(\#P-2)+1\ ;\ (P(\#P-1)-1)*1]$$

has to be replaced by

$d := P(\#P-2) + P(\#P-1)$. $f := P(\#P-2)+1$.
$P := P[0;..\#P-2]$;; $[(div\ d\ f - 1)*f\ ;\ f + mod\ d\ f]$

and we have to modify R and A.