(See https://cs.stanford.edu/~knuth/programs.html for date.)

1. Intro. Given n, generate BDDL to compute a representation of all restricted growth sequences $a_1 \ldots a_n$ (and thus of all set partitions of $\{1, \ldots, n\}$).

```
#define maxn 500
#include <stdio.h>
#include <stdlib.h>
  int n;
  int subscr[maxn + 1][maxn];      /* allocation of variable subscripts */
  main(int argc, char *argv[])
  {
    register int i, j, k;
    if (argc ≠ 2 ∨ sscanf(argv[1], "%d", &n) ≠ 1 ∨ n ≤ 0) {
      fprintf(stderr, "Usage:␣%s␣n\n", argv[0]);
      exit(−1);
    }
    printf("#␣beginning␣the␣output␣of␣BDDL-RGROWTH␣%d\n", n);
    for (i = 0, k = n; k; k−−)
      for (j = 0; j < k; j++) subscr[k][j] = i++;
    for (j = 1; j ≤ n; j++) printf("f%d=c1\n", j);
    for (k = n; k; k−−)
      for (j = 1; j < k; j++) {
        printf("f0=x%d?f%d:c0\n", subscr[k][0], j);
        printf("f%d=x%d?c0:f%d\n", maxn, subscr[k][0], j + 1);
        printf("f%d=x%d?c0:f%d\n", maxn + 1, subscr[k][0], j);
        for (i = 1; i < j; i++) {
          printf("f0=x%d?f%d:f0\n", subscr[k][i], maxn + 1);
          printf("f%d=x%d?c0:f%d\n", maxn, subscr[k][i], maxn);
          printf("f%d=x%d?c0:f%d\n", maxn + 1, subscr[k][i], maxn + 1);
        }
        printf("f0=x%d?f%d:f0\n", subscr[k][j], maxn);
        for (i++; i < k; i++) printf("f0=x%d?c0:f0\n", subscr[k][i], j);
        printf("f%d=f0\n", j);
      }
    printf("f1=x%d?f1:c0\n", subscr[1][0]);
    printf("!␣f1␣represents␣restricted␣growth␣sequences␣of␣length␣%d\n", n);
  }
```

2. Index.

BDDL-RGROWTH