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

1. Introduction. This is a quick-and-dirty program related to exercise 3.6–14. I'm finding how many terms appear in the representation of z^n with respect to bases of the form $z^0, \ldots, z^{t-1}, z^{n-r+t}, \ldots, z^{n-1}$, modulo $z^r + z^{r-s} + 1$ and mod 2, where $1 \le t \le r$.

```
#define r 100    /* the longer lag */
#define s 37    /* the shorter lag */
#define n 400    /* the number of elements generated simultaneously by ran_array */
#include <stdio.h>
  ⟨Global variables 2⟩ main()
  {
    register int i, j, k, m, t;
    ⟨Initialize for the case t = r 3⟩;
    while (t) {
      ⟨Gather statistics for case t 5⟩;
      t--;
      ⟨Change the basis to eliminate z^t 4⟩;
    }
    ⟨Print the statistics 8⟩;
  }
```

2. The representation of $z^k = a_{k0}z^{b_0} + \cdots + a_{k(r-1)}z^{b_{r-1}}$ appears in arrays a and b. The largest power of z less than z^n that is not in the basis is z^m.

```
⟨Global variables 2⟩ ≡
  int a[n + 1][r];    /* I could make this char, but int aids debugging */
  int b[r];    /* identifies the basis */
  int c[r], d[n + 2];    /* for working storage */
  int p[n];    /* is this power of z in the basis? */
```

See also section 6.

This code is used in section 1.

3. ⟨Initialize for the case $t = r$ 3⟩ ≡

```
  for (k = 0; k < r; k++) {
    a[k][k] = 1;
    b[k] = k;
    p[k] = 1;
  }
  for ( ; k ≤ n; k++) {
    for (j = 1; j < r; j++) a[k][j] = a[k − 1][j − 1];    /* z^k = z · z^{k−1} */
    if (a[k − 1][r − 1]) {
      a[k][0] = 1;
      a[k][r − s] ⊕= 1;
    }
  }
  m = n − 1;
  t = r;
```

This code is used in section 1.

4. ⟨ Change the basis to eliminate z^t 4 ⟩ ≡

```
for (k = m; a[k][t] ≡ 0; k−−) ;
b[t] = k;
for (j = 0; j < r; j++) c[j] = a[k][j];
c[t] = 0;
p[t] = 0;
p[k] = 1;
for ( ; k ≥ t; k−−)
  if (a[k][t])
    for (j = 0; j < r; j++) a[k][j] ⊕= c[j];
if (a[n][t])
  for (j = 0; j < r; j++) a[n][j] ⊕= c[j];
while (p[m] ≡ 1) m−−;
```

This code is used in section 1.

5. We are interested in the number of nonzero coefficients in the representation of z^n. However, if this representation depends on any of the "forbidden" powers z^t, ..., $z^{n-r+t-1}$, we want rather to exhibit the representation of z^m.

⟨ Gather statistics for case t 5 ⟩ ≡

```
{
  register int forbidden = 0;
  for (j = 0, i = 0; j < r; j++)
    if (a[n][j]) {
      if (b[j] < n − r + t ∧ b[j] ≥ t) forbidden = 1;
      else i++;
    }
  if (forbidden) ⟨ Print out an interesting linear dependency 7 ⟩
  else stat[i]++;
}
```

This code is used in section 1.

6. ⟨ Global variables 2 ⟩ +≡

```
int stat[r + 1];    /* the number of cases with a given number of nonzero terms */
```

7. ⟨Print out an interesting linear dependency 7⟩ ≡

```
{
  for (i = 0; i < n; i++) d[i] = 0;
  for (j = 0; j < r; j++)
    if (a[m][j]) d[b[j]] = 1;
  d[m] = 1;
  d[n] = 1;
  printf("%d:", t);
  for (i = 0; ; ) {
    while (d[i] ≡ 0) i++;
    if (i ≡ n) break;
    printf("␣%d", i);
    while (d[i] ≡ 1) i++;
    if (i > n) i = n;
    printf("..%d", i − 1);
  }
  printf("\n");
}
```

This code is used in section 5.

8. ⟨Print the statistics 8⟩ ≡

```
for (j = 0; j ≤ r; j++) printf("␣%3d:␣%d\n", j, stat[j]);
```

This code is used in section 1.

9. Index.

LAGFIB