1* Intro. Generate SAT instances for Erdős discrepancy patterns: The sequences \((x_d, x_{2d}, \ldots, x_{\lfloor n/d \rfloor d})\) are supposed to be strongly balanced, for \(1 \leq d \leq n\), where a sequence \((y_1, \ldots, y_t)\) is “strongly balanced” if the corresponding sequence of \(\pm 1\)s defined by \(z_j = 2y_j - 1\) has all partial sums satisfying \(-2 \leq z_1 + \cdots + z_k \leq 2\).

It’s easy to see that the latter property needs to be checked only for odd values of \(k\) with \(3 \leq k \leq t\).

```c
#include <stdio.h>
#include <stdlib.h>
int n;

main(int argc, char *argv[])
{    register int d;
    (Subroutine 3*)

    if (argc != 2 || sscanf(argv[1], "%d", &n) != 1) {
        fprintf(stderr, "Usage: %s \n
        exit(-1);
    }

This code is used in section 1*.
```
3* Our task is to generate clauses that characterize a strongly balanced sequence, and it turns out that there’s a very interesting way to do this. The subroutine \texttt{generate}(d, n) makes clauses for the sequence with \( y_j = x_j d \).

Sinze’s cardinality clauses (see TAOCP Section 7.2.2.2) have the property that \( y_1 + \cdots + y_{j+k-1} \geq k \) implies \( S_j^k \); hence we want \( S_j^{k+2} = 0 \) for \( j < n/2 \). The dual clauses have the property that \( y_1 + \cdots + y_{j+k-1} \geq k \) implies \( S_j^k \); we can rewrite this to say that \( S_j^k \) implies \( y_1 + \cdots + y_{j+k-1} \geq j \). Hence we also want \( S_{k+2}^k = 1 \) for \( k < n/2 \). It follows that we need only deal with auxiliary variables \( S_j^k \) when \( |j-k| \leq 1 \). The variables \( S_{k-1}^k, S_k^k \), and \( S_{k+1}^k \) will be denoted respectively by \( dA_k, dB_k \), and \( dC_k \).

The clauses
\[
(S_i^t \lor S_{i+1}^t) \land (S_i^{t+1} \lor S_{i+1}^{t+1}) \land (S_i^t \lor S_{i+1}^{t+1}) \land (S_i^{t+1} \lor S_{i+1}^{t+1})
\]
are needed when \( n \geq 2t + 3 \). The clauses
\[
(y_{2t-2} \lor S_i^{t-1}) \land (y_{2t-1} \lor S_i^{t-1} \lor S_i^t) \land (y_{2t-1} \lor S_i^t \lor S_i^{t+1}) \land (y_{2t+1} \lor S_i^{t+1})
\]
and their duals
\[
(y_{2t-2} \lor S_i^{t-1}) \land (y_{2t-1} \lor S_i^{t-1} \lor S_i^t) \land (y_{2t-1} \lor S_i^t \lor S_i^{t+1}) \land (y_{2t+1} \lor S_i^{t+1})
\]
are needed when \( n \geq 2t + 1 \). (And we simplify these clauses for small \( t \) by using the facts that \( S_0^0 = 1 \) and \( S_1^1 = 0 \).)

Furthermore, we simplify yet again by using resolution to eliminate the \( A \) and \( C \) variables, as well as

(Subroutine 3*) \( \equiv \)
\[
\text{void \texttt{generate(int \ d, int \ n)}}
\]
\[
\{\text{register int \ i, \ j, \ k, \ t;}
\text{for \ (t = 1; \ 2 * t + 3 \leq \ n; \ t++) \ { \text{Generate the first clauses 4*};}
\text{for \ (t = 1; \ 2 * t + 1 \leq \ n; \ t++) \ { \text{Generate the second clauses 5*};}}
\}
\]
This code is used in section 1*.

4* \{ Generate the first clauses 4* \} \( \equiv \)
\[
\text{if \ (t > 1) \{}
\text{printf("\%d\%d\%d\%d\%d\n", \ d * (t + t + 1), \ d, \ d, \ d, \ t + 1);}
\text{printf("\%d\%d\%d\%d\%d\n", \ d * (t + t), \ d, \ d, \ d, \ t + 1);}
\text{printf("\%d\%d\%d\%d\%d\n", \ d * (t + t + 1), \ d, \ d, \ d, \ t + 1);}
\text{printf("\%d\%d\%d\%d\%d\n", \ d * (t + t), \ d, \ d, \ d, \ t + 1);}
\}
\text{else \{}
\text{printf("\%d\%d\%d\%d\%d\n", \ d * 3, \ d, \ d, \ d, \ t + 1);}
\text{printf("\%d\%d\%d\%d\%d\n", \ d * 2, \ d, \ d, \ d, \ t + 1);}
\text{printf("\%d\%d\%d\%d\%d\n", \ d * 3, \ d, \ d, \ d, \ t + 1);}
\text{printf("\%d\%d\%d\%d\%d\n", \ d * 2, \ d, \ d, \ d, \ t + 1);}
\}
\]
This code is used in section 3*.
Generate the second clauses \( 5^* \) \( \equiv \)

\[
\begin{aligned}
\text{if } (t > 1) \{ \\
\quad \text{printf ("X%d\ X%d\ %dB\ \n", } d*(t+t), d*(t+t+1), d, t); \\
\quad \text{if } (2\ t+3 \leq n) \text{ printf ("X%d\ X%d\ %dB\ \n", } d*(t+t), d*(t+t+1), d+1); \\
\quad \text{printf ("X%d\ X%d\ %dB\ \n", } d*(t+t), d*(t+t+1), d, t); \\
\quad \text{if } (2\ t+3 \leq n) \text{ printf ("X%d\ X%d\ %dB\ \n", } d*(t+t), d*(t+t+1), d+1); \\
\quad \text{else } \{ \\
\qquad \text{printf ("X%d\ X%d\ X%d\ \n", } d*(t+t), d*(t+t+1), d); \\
\qquad \text{if } (5 \leq n) \text{ printf ("X%d\ X%d\ %dB\ \n", } d*(t+t), d*(t+t+1), d, 2); \\
\qquad \text{printf ("X%d\ X%d\ %dB\ \n", } d*(t+t), d*(t+t+1), d); \\
\qquad \text{if } (5 \leq n) \text{ printf ("X%d\ X%d\ %dB\ \n", } d*(t+t), d*(t+t+1), d, 2); \\
\quad \}
\end{aligned}
\]

This code is used in section 3\( ^* \).
**Index.**

The following sections were changed by the change file: 1, 3, 4, 5, 6.

- `argc`: 1* 2.
- `argv`: 1* 2.
- `d`: 1* 3*.
- `exit`: 2.
- `fprintf`: 2.
- `generate`: 1* 3*.
- `i`: 3*.
- `j`: 3*.
- `k`: 3*.
- `main`: 1*.
- `n`: 1* 3*.
- `printf`: 1* 4* 5*.
- `sscanf`: 2.
- `stderr`: 2.
- `t`: 3*.
(Generate the first clauses 4*) Used in section 3*.
(Generate the second clauses 5*) Used in section 3*.
(Process the command line 2) Used in section 1*.
(Subroutine 3*) Used in section 1*.