1. În teorie, predicatul ar fi trebuit să calculeze produsul tuturor nr. din listă care sunt strict mai mari decât o valoare dată. Totuşi, rezultatul final va fi întotdeauna 0, deoarece produsul nu este inițializat cu 1, ci cu 0.

R17

=> rezultatul interogării este:

P = 0.

3. Să se genereze lista permutărilor cu elem. $N, N+1, \ldots, 2*N-1$, având prop. că modulul dif. dintre 2 valori consecutive este ≤ 2.

Modele matematice:

candidat (n_1, n_2) =

1. n_1, $n_1 < n_2$

2. candidat (n_1+1, n_2), $n_1 < n_2$

$$\text{exista}(x, e_1 \ldots e_n) = \begin{cases} \text{false}, & n = 0 \\ \text{true}, & e_1 = x \\ \text{exista}(x, e_2 \ldots e_n), & \text{altfel} \end{cases}$$

$$\text{perm}(n, lg, col) = \begin{cases} col, & n = lg \\ \text{perm}(n, lg+1, e \oplus col), \\ \text{dacă: } |e - col_1| \leq 2 \ (col = col_1 \ldots col_m) \\ \quad \text{exista}(e, col) = \text{fals} \end{cases}$$

unde: $e = \text{candidat}(n, 2*n)$

$\text{permutare}(n) = \text{perm}(n, 1, (e))$, unde

$e = \text{candidat}(n, 2*n)$

main(n) = ∪ permutare(n)

```
candidat(N1, N2, N1):- N1 < N2.
candidat(N1, N2, R):-
        N1 < N2,
        N11 is N1+1,
        candidat(N11, N2, R).
```

model de flux: (i, i, o)
nedeterminist

```
exista(H, [H|_]):- !.
exista(E, [_|T]):-
      exista(E, T).
```

altfel
era nedeterminist

model de flux: (i, i)
determinist

```
perm(N, N, Col, Col).
perm(N, Lg, [H|T], R):-
        N2 is N*2,
        candidat(N, N2, E),
        not(exista(E, [H|T])),
        abs(E-H) =< 2,

        Lg1 is Lg+1,
        perm(N, Lg1, [E|[H|T]], R).
```

model de flux: (i, i, i, o)
nedeterminist

```
permutare(N, R):-
        N2 is N*2,
        candidat(N, N2, E),
        perm(N, 1, [E], R).
```

model de flux: (i, o)
nedeterminist

```
main(N, R) :-
    findall(R1, permutare(N, R1), R).
```

model de flux: (i, o)
determinist

5. Nr. de subliste pt. care atomul numeric maxim de pe nivelurile impare este par. (nivelul superficial e 1)

$maxim(l_1 \dots l_n, m, niv) =$

$$= \begin{cases} m, & n = 0 \\ maxim(l_2 \dots l_n, l_1, niv), & l_1 \text{ e nr., } niv \text{ e impar și } l_1 > m \\ maxim(l_2 \dots l_n, m, niv), & l_1 \text{ e nr, } niv \text{ e impar și } l_1 \leq m \\ maxim(l_2 \dots l_n, m, niv), & l_1 \text{ e nr, } niv \text{ e par} \\ maxim(l_2 \dots l_n, m, niv), & l_1 \text{ e atom nenumeric} \\ maxim(l_2 \dots l_n, X, niv), & \text{altfel} \end{cases}$$

unde $X = maxim(l_1, m, niv+1)$

$$subliste(l) = \begin{cases} 0, & l \text{ atom} \\ 1 + \sum_{i=1}^{n} subliste(l_i), & maxim(l, -\infty, 0) \text{ e nr. și e par} \\ \sum_{i=1}^{n} subliste(l_i), & \text{altfel} \end{cases}$$

```
(defun maxim (l m niv)
    (cond
        ((null l) m)
        ((and (numberp (car l)) (oddp niv) (> (car l) m))
            (maxim (cdr l) (car l) niv))
        ((and (numberp (car l)) (oddp niv) (<= (car l) m))
```

```
(maxim (cdr e) m niv))

((and (numberp (car e))) (evenp niv))
 (maxim (cdr e) m niv))

((atom (car e))) (maxim (cdr e) m niv)

(+ (maxim (cdr e) (maxim (car e) m (+ niv 1)))
 niv))
)
)

(defun subliste (e)
 (cond
 ((atom e) 0)
 ((and (not (equal (maxim e most-negative-fixnum 0)
 most-negative-fixnum))
 (evenp (maxim e most-negative-fixnum 0)))
 (+1 (apply #'+ (mapcar #'subliste e))))
 )
 (+ (apply #'+ (mapcar #'subliste e)))
 )
)
```