3

procedure bubble (T[1..n])

for i ← n to 2 do

for j ← 1 to i-1

if T[j] > T[j+1]

p ← T[j]

T[j] ← T[j+1]

T[j+1] ← p

4

$21 \cdot 54$ $\qquad$ $21_{(10)} = 10101_{(2)}$

$54_{(10)} = 110110_{(2)}$

a:	21	10	5	2	1
b:	<u>54</u>	108	<u>216</u>	432	<u>864</u>

$= 1134$

$a_1 = a, \quad a_k = 1, \quad a_{i+1} = \left\lfloor \frac{a_i}{2} \right\rfloor \quad (\text{dla } i = 1, 2, \ldots, k-1)$

$b_1 = b, \quad b_{i+1} = 2b_i \quad (\text{dla } i = 1, 2, \ldots, k-1) \qquad b_i = b \cdot 2^{i-1}$

$$w = \sum_{\substack{i=1 \\ i\text{-nieparzyste}}}^{k} b_i$$

Ciąg a_i odpowiada zapisowi dwójkowemu a gdzie nieparzyste a_i odpowiada 1 a parzyste 0

$$w = b \sum_{\substack{i=1 \\ a_i\text{-nieparzyste}}}^{k} 2^{i-1}$$

$$\sum_{\substack{i=1 \\ a_i \text{ nieparzyste}}}^{k} 2^{i-1}$$ $\leftarrow$ ~~a~~ konwersja a z systemu binarnego na dziesiętny

Pisemne mnożenie liczb binarnych

Kryterium jednorodne:

zł. cz.: $O(\log_2 a)$

zł. p.: $O(1)$

Kryterium logarytmiczne

zł. cz.: $O(\log_2 a \cdot \log_2 b)$

zł. p.: $O(\log_2 a + \log_2 b)$

5

$$\begin{cases} a_1 = c_1 \\ a_2 = c_2 \\ \vdots \\ a_k = c_k \\ a_{k+1} = a_k \cdot b_k + a_{k-1} \cdot b_{k-1} + \dots + a_1 b_1 \end{cases}$$

$$\underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & 0 & 0 & & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \\ b_1 & b_2 & b_3 & b_4 & & b_k \end{bmatrix}}_{M} \cdot \begin{bmatrix} a_i \\ a_{i+1} \\ a_{i+2} \\ \vdots \\ \vdots \\ a_{i+k-1} \end{bmatrix} = \begin{bmatrix} a_{i+1} \\ a_{i+2} \\ a_{i+3} \\ \vdots \\ \vdots \\ a_{i+k} \end{bmatrix}$$

$$M^{n-1} \cdot \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \end{bmatrix} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_k \end{bmatrix} \qquad \boxed{w_1 = a_n}$$

$$a_{k+1} = a_k \cdot b_k + a_{k-1} \cdot b_{k-1} + \ldots + a_1 b_1 + w(n)$$

$$w(n) = w_0 + w_1 n^1 + w_2 n^2 + \ldots + w_l n^l$$

$$\begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 & 0 & & 0 \\ b_1 & b_2 & b_3 & \cdots & b_k & w_0 & w_1 & \cdots & w_l \\ 0 & 0 & \cdots & & 0 & \binom{0}{0} & \binom{0}{1} & \cdots & \binom{0}{l} \\ 0 & 0 & & & 0 & \binom{1}{0} & \binom{1}{1} & & \binom{1}{l} \\ \vdots & & 0 & & \vdots & \binom{2}{0} & & \ddots & \vdots \\ \vdots & & & \ddots & 0 & \vdots & & & \vdots \\ 0 & \cdots & \cdots & 0 & 0 & \binom{l}{0} & \binom{l}{1} & & \binom{l}{l} \end{bmatrix} \begin{bmatrix} a_n \\ a_{n+1} \\ \vdots \\ a_{n+k-2} \\ a_{n+k-1} \\ 1 \\ n \\ n^2 \\ \vdots \\ n^l \end{bmatrix} = \begin{bmatrix} a_{n+1} \\ a_{n+2} \\ \vdots \\ a_{n+k-1} \\ a_{n+k} \\ 1 \\ n+1 \\ (n+1)^2 \\ \vdots \\ (n+1)^l \end{bmatrix}$$

6 Wynikiem algorytmu jest najmłodszy bit ostatniego elementu ~~ciągu~~ zbioru A

$(a - b) \bmod 2 = (a \bmod 2 - b \bmod 2) \bmod 2$

$((a \bmod 2) - (b \bmod 2)) \bmod 2$		$a \bmod 2$	$b \bmod 2$
0	$\leftarrow$	0	0
1		0	1
1		1	0
0		1	1

XOR jest łączny

res $\leftarrow a_1 \% 2$
dla każdego a_i ze zbioru A
    res $\leftarrow$ res XOR $(a_i \% 2)$
return res

7

Z wejściowej listy krawędzi tworzymy tablicę parent [1... n]

parent [u] = v   v jest rodzicem u

korzeń: parent [1] = 1

```
Procedure is_ancestor (u, v)
    if u = 1 or parent [v] = u
        return true
    return is_ancestor (u, parent [v])
```