Z7)

a) Zauważmy, że w macierzy Teoplitza wartości na każdej przekątnej są takie same. Co więcej, wartości te wyznaczone są jednoznacznie przez pierwszą kolumnę oraz pierwszy wiersz.

Możemy zatem trzymać takie macierze jako wektor o $2n-1$ elementach. Przykład:

$$\begin{bmatrix} a_{00} & a_{10} & a_{20} & \cdots & a_{n0} \\ a_{01} & \ddots & & & \vdots \\ a_{02} & & \ddots & & \vdots \\ \vdots & & & \ddots & \vdots \\ a_{0n} & \cdots & \cdots & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} a_{n0} \\ a_{n-1\,0} \\ \vdots \\ a_{00} \\ a_{01} \\ \vdots \\ a_{0n} \end{bmatrix}$$

Wtedy dodawanie takich macierzy sprowadza się do dodania dwóch wektorów. Czyli $O(n)$.

b)

Na potrzeby tego algorytmu rozszerzmy naszą macierz do wymiarów $n' \times n'$, gdzie $n' = 2^{\lceil \log n \rceil}$ uzupełniając nowo powstałe miejsca zerami. Czyli otrzymamy coś w tym stylu:

$A \to A'$

$$A' = \begin{bmatrix} A & \begin{matrix} 0 & \cdots & 0 \\ \vdots & & \vdots \end{matrix} \\ \begin{matrix} 0 & \cdots & \cdots \\ \vdots & & \\ 0 & \cdots & \cdots \end{matrix} & \begin{matrix} \vdots & \ddots & \vdots \\ \cdots & & 0 \end{matrix} \end{bmatrix}$$

, analogicznie zrobimy z wektorem przez, który chcemy pomnożyć macierz:

$v \to v'$

$$v' = \begin{bmatrix} v \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Zauważmy, że nasza macierz A' ma następującą postać:

$$A' = \begin{bmatrix} X & Y \\ Z & X \end{bmatrix}$$, gdzie X, Y, Z to kwadratowe macierze o rozmiarach $\frac{n'}{2}$

a wektor $$v' = \begin{bmatrix} P \\ R \end{bmatrix}$$ gdzie P, R to wektory o rozmiarze $\frac{n'}{2}$.

Wtedy $$A'v' = \begin{bmatrix} X \cdot P + Y \cdot R \\ Z \cdot P + X \cdot R \end{bmatrix}$$. Jednak na razie takie rozbicie nic nam nie daje: mamy 4 mnożenia macierzy 4 razy mniejszych.

Przekształćmy jednak te sumy dodając 0:

$$A'v' = \begin{bmatrix} X \cdot P + Y \cdot R \\ Z \cdot P + X \cdot R \end{bmatrix} = \begin{bmatrix} X \cdot P + Y \cdot R + (X \cdot R - X \cdot R) \\ Z \cdot P + X \cdot R + (X \cdot P - X \cdot P) \end{bmatrix} = \begin{bmatrix} R(X-Y) - \underline{X(R-P)} \\ P(X+Z) + \underline{X(R-P)} \end{bmatrix}$$

Teraz zamiast 4 mnożeń macierz × wektor, mamy:

- $A = R(X-Y)$ ← 1 dodawanie macierzy, 1 mnożenie macierz × wektor
- $B = X(R-P)$ ← 1 dodawanie wektorów, 1 mnożenie macierz × wektor
- $C = P(X+Z)$ ← 1 dodawanie macierzy, 1 mnożenie macierz × wektor
- $A - B$ ← 1 dodawanie wektorów
- $C + B$ ← 1 dodawanie wektorów

- dodawanie macierzy to $2 \cdot \frac{n'}{2} - 1 = n' - 1$ operacji

dodawanie wektorów to $\frac{n'}{2}$ operacji

a mnożenie macierzy × wektor będziemy wykonywać rekurencyjnie, czyli $T(\frac{n'}{2})$

Sumarycznie: $T(n') = \frac{7n'-4}{2} + 3 \cdot T(\frac{n'}{2})$, z tw. z wykładu: $T(n') = \Theta(n'^{\log 3})$

a skoro $n' \leq 2 \cdot n$ to $T(n) = \Theta(n^{\log 3})$.

Z8)

Zauważmy, że znalezienie mediany jest równoważne ze znalezieniem pierwszej takiej liczby, która jest większa równa od $\frac{n}{2}$ liczb.

Algorytm:

- Dla każdej z list:
  - $p = 0$, $k = n$, ~~kandydat $= \frac{k+p}{2}$~~
  - (*) kandydat $= \frac{k+p}{2}$, (jeśli $k==p$ break)
  - W pozostałych listach bin-searchem znajdujemy liczbę liczb $\leq$ kandydat. Jeśli suma $= \frac{n}{2}$ return kandydat.

    Jeśli suma $> \frac{n}{2}$:

    ~~kandydat~~ $k =$ kandydat

    else if suma $< \frac{n}{2}$:

    $p =$ kandydat
  - go to (*)

Dla każdej z list w najgorszym przypadku rozważymy $O(\log n)$ kandydatów. I dla każdego kandydata wyliczamy jego pozycję w $O(k \cdot \log(n))$. Czyli złożoność całego algorytmu to $O(k^2 \cdot \log^2(n))$