Lista 4

Zadanie 1

Rozważając jakikolwiek algorytm szukający elementu x mamy do czynienia z pewną liczbą porównań pomiędzy elementem szukanym x oraz elementami tablicy wejściowej A.

W tym przypadku wiemy, że A jest już posortowana oraz jesteśmy w stanie stwierdzić w stałym czasie czy dana liczba $z \leqslant A[i]$, czy też $z \geqslant A[i]$ dla wszystkich $i \in \{1..n\}$. Korzystając tylko z takich operacji porównań chcemy sie dowiedzieć czy dany element x należy do tablicy A. Dzielimy więc tablicę A na dwie połowy $A\left[1 \ldots \left\lfloor \frac{n}{2} \right\rfloor\right]$ oraz $A\left[\left(\left\lfloor \frac{n}{2} \right\rfloor + 1\right) \ldots n\right]$ i wybieramy lewą bądź prawą stronę na podstawie wyniku nierówności $x \leqslant A[i]$ i powtarzamy rekurencyjnie ten krok $\lg n$ razy. Po zredukowaniu tablicy A do tablicy jednoelementowej wynik równości $A[0] == x$ jest rozwiązaniem zadanego problemu.

Żądany wynik osiągniemy w czasie $\Omega(\lg n)$ ponieważ dzielimy daną połówkę tablicy (bądź początkową tablicę) na kolejne połówki i robimy tak do momentu kiedy nie mamy jednego elementu czyli $\lg n$ razy.

Nie biorąc pod uwagę sposobów o podobnej idei do algorytmu BogoSort nie ma lepszego sposobu na znalezienie danego elementu w posortowanej tablicy, dlatego minimalny czas na znalezienie elementu x w posortowanej tablicy A zajmie przynajmniej $\Omega(\lg n)$.