Lekcja: Algorytm sortowania binarnego

Wyobraź sobie, że masz książkę z alfabetycznie uporządkowanymi nazwiskami i chcesz znaleźć konkretną osobę, na przykład "Kowalski". Zamiast przeglądać każdą stronę po kolei, intuicyjnie otworzyłbyś książkę mniej więcej w połowie i sprawdził, czy "Kowalski" znajduje się w lewej czy prawej części książki. Jeśli nazwiska na środkowej stronie są alfabetycznie przed "Kowalski", to wiesz, że musisz szukać w prawej połowie książki. W przeciwnym razie, szukasz w lewej połowie. Następnie proces ten powtarzasz dla wybranej połowy książki, za każdym razem dzieląc ją na pół, aż znajdziesz "Kowalskiego". To właśnie jest algorytm wyszukiwania binarnego.
W każdej iteracje ignorujemy niemal połowę zbioru danych – z tego powodu algorytm ten oferuje logarytmiczną złożoność obliczeniową (O(logn)).

Kluczowe punkty algorytmu wyszukiwania binarnego:

Dane Muszą Być Posortowane: Aby algorytm działał, dane, w których szukamy (np. lista liczb, nazwiska w książce), muszą być uporządkowane. To jak posiadanie alfabetycznie uporządkowanej książki telefonicznej.

Podział na Pół: W każdym kroku algorytm dzieli dane na dwie równe części i sprawdza, czy szukany element znajduje się w lewej czy prawej połowie.

Powtarzanie Procesu: Proces dzielenia jest powtarzany na coraz mniejszej liczbie elementów, aż znajdziemy szukany element lub stwierdzimy, że go nie ma w danych.

Efektywność: Algorytm wyszukiwania binarnego jest bardzo efektywny, szczególnie w przypadku dużych zbiorów danych, ponieważ z każdym krokiem znacząco redukuje liczbę elementów, które musi sprawdzić.

Opis algorytmu

Wejście:

i p – indeks pierwszego elementu w tablicy Z, i p ∈ C.
i k – indeks ostatniego elementu w tablicy Z, i k ∈ C.
Z – tablica do wyszukania elementu. Indeksy od i p do i k.
k – wartość poszukiwanego elementu – tzw. klucz.

Wejście:

i p – indeks pierwszego elementu w tablicy Z, i p ∈ C.
i k – indeks ostatniego elementu w tablicy Z, i k ∈ C.
Z – tablica do wyszukania elementu. Indeksy od i p do i k.
k – wartość poszukiwanego elementu – tzw. klucz.

Zmienne pomocnicze:

i sr – indeks elementu środkowego. i sr ∈ C.

Lista kroków:

K01: p ← -1 zakładamy, iż k nie występuje w zbiorze
K02: **Dopóki** i p ≤ i k,
wykonuj kroki K02...K10 w pętli poszukujemy elementu o wartości k
K03: $i_{sr} = \left[\frac{i_p + i_k}{2} \right]$; wyznaczamy element środkowy
K04: **Jeśli** k ≠ Z [i sr],
to idź do kroku K07
K05: p ← i sr element znaleziony, kończymy
K06: **Idź do kroku** K11
K07: **Jeśli** k < Z [i sr],
to idź do kroku K10 wybieramy odpowiednią połówkę zbioru na dalsze poszukiwania
K08: i p ← i sr + 1 k > Z [i sr] → druga połówka
K09: **Następny obieg pętli** K02
K10: i k ← i sr - 1 k < Z [i sr] → pierwsza połówka
K11: **Zakończ z wynikiem** p

Zadanie

Zaimplementuj algorytm wyszukiwania binarnego w języku Java. Przetestuj algorytm z różnymi przykładami danych wejściowych.