Mochammad Rizky Septian
1302220121

Muhammad Fauzan Majid
1302220144

Muhammad Nur Shodiq
1302223054

Memecahkan masalah pengurutan data jumlah difabel di seluruh dunia melalui pendekatan rekursif & iterative dari algoritma bubble sort

Deskripsi masalah

Proses managerial dalam mengolah data jumlah difabel di seluruh dunia membutuhkan algoritma yang dapat mempermudah proses pengurutan input array dalam jumlah besar, maka dari itu, algoritma dari **Bubble Sort** dan **Insertion Sort** dapat menjadi solusi dari pengurutan nilai mahasiswa dengan input yang banyak

Algoritma Bubble Sort

Bubble sort adalah algoritma yang bekerja dengan membandingkan elemen-elemen dalam array atau daftar data secara berurutan, dari kiri ke kanan. Jika elemen di kiri lebih besar dari elemen di kanan, maka elemen-elemen tersebut ditukar posisinya. Proses ini diulang hingga tidak ada lagi elemen yang perlu ditukar.

Algoritma Rekursif Bubble Sort

Algoritma Bubble Sort rekursif bekerja dengan cara membagi masalah menjadi masalah yang lebih kecil dan memanggil dirinya sendiri untuk menyelesaikan masalah tersebut.

Algoritma Iteratif Bubble Sort

Algoritma Bubble Sort iteratif bekerja dengan cara mengulang proses pengurutan elemen sampai tidak ada lagi elemen yang perlu ditukar.

Analisis Bubble Sort

Rekursif

```
void bubbleSortRecursive(int arr[], int n) {
    if (n <= 1) {
        return;
    }else{
        for (int i = 0; i < n - 1; i++) {
            if (arr[i] > arr[i + 1]) {
                int temp = arr[i];
                arr[i] = arr[i + 1];
                arr[i + 1] = temp;
            }
        }
        bubbleSortRecursive(arr, n - 1);
    }
}
```

Ukuran input:
n (ukuran array arr[])s
Operasi dasar:
Perbandingan arr[i] > arr[j] dan penukaran nilai arr[i] dan arr[j]
Solusi umum:
C(n) = Σ(i=1 to n-1) Σ(j=i+1 to n-1) (perbandingan + penukaran)
= Σ(i=1 to n-1) (n-i)
= n + (n-1) + (n-2) + ... + 1
= n(n+1)/2 ∈ O(n^2)
Jadi, kompleksitas waktu algoritma bubble sort adalah O(n^2). Hal ini sesuai karena bubble sort termasuk algoritma sorting dengan kompleksitas quadratic. Dengan kata lain, waktu eksekusi bubble sort berbanding lurus dengan kuadrat dari ukuran input.

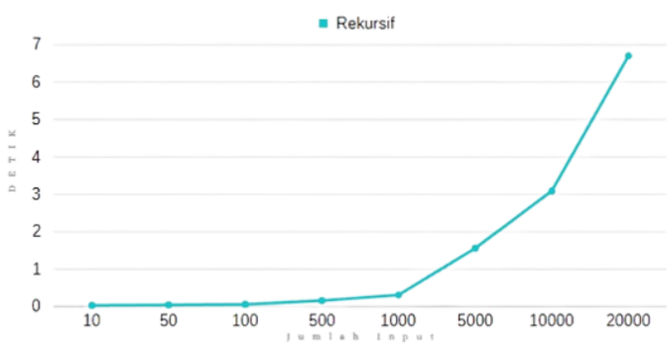
Iteratif

```
void bubble_sort(int arr[], int n) {
  bool swapped = true;

  for (int i = 0; i < n - 1; i++) {
    swapped = false;

    for (int j = i + 1; j < n; j++) {
      if (arr[i] > arr[j]) {
        int temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;

        swapped = true;
      }
    }

    if (!swapped) {
      break;
    }
  }
}
```

Ukuran input:
n (ukuran array arr[])
Operasi dasar:
Perbandingan arr[i] > arr[j]
Solusi umum dengan cara substitusi:
T(n) = T(n - 1) + O(n)
T(n - 1) = T(n - 2) + O(n - 1)
T(n - 2) = T(n - 3) + O(n - 2)
...
T(1) = O(1)
T(n) = O(1) + O(2) + O(3) + ... + O(n)
T(n) = O(1) * (1 + 2 + 3 + ... + n)
T(n) = O(1) * n(n + 1) / 2
T(n) = O(n^2)
Jadi, kompleksitas algoritma Bubble Sort Rekursif adalah O(n^2). Hal ini sesuai karena bubble sort termasuk algoritma sorting dengan kompleksitas quadratic. Dengan kata lain, waktu eksekusi bubble sort berbanding lurus dengan kuadrat dari ukuran input.

Running time Bubble sort

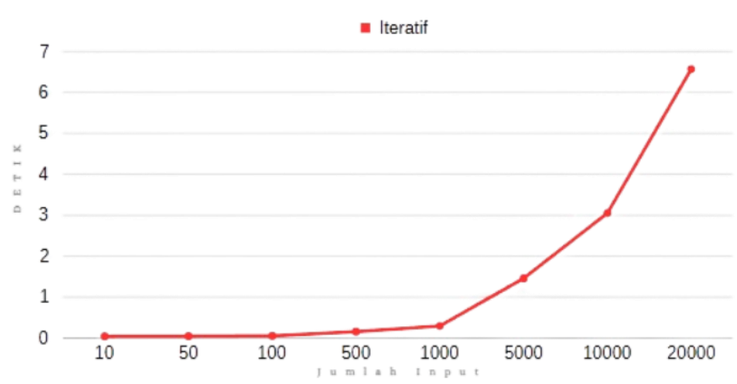
Rekursif



Input (n) = 10, 50, 100, 500, 1000, 5000

Running time = 6.704 detik

Iteratif



Input (n) = 10, 50, 100, 500, 1000, 5000

Running time = 6.569 detik

Referensi

1. Chat GPT
2. Bard.ai
3. Materi AKA