Program Studi Teknologi Informasi, Fakultas Teknik
Universitas Muhammadiyah Yogyakarta

Algoritma dan Struktur Data

Dosen: Muhammad Abdul Haq, S.Tr.T., M.Eng.

Data Structures and Algorithm
Alasan

- Ilmu komputer adalah bidang studi yang menangani penyelesaian berbagai masalah dengan menggunakan komputer.
  - Untuk memecahkan suatu masalah tertentu dengan menggunakan komputer, Anda perlu merancang algoritma untuk masalah tersebut.

- Beberapa algoritma dapat dirancang untuk memecahkan masalah tertentu.

- Suatu algoritma yang memberikan efisiensi maksimum harus digunakan untuk memecahkan masalah tersebut.
  - Efisiensi suatu algoritma dapat ditingkatkan dengan menggunakan struktur data yang tepat.

- Struktur data membantu dalam membuat program yang sederhana, dapat digunakan kembali, dan mudah dipelihara.

Data Structures and Algorithm
Tujuan

- Dalam sesi ini, Anda akan belajar untuk:
  - Menjelaskan peran struktur data dan algoritma dalam pemecahan masalah melalui computer
  - Mengidentifikasi teknik untuk merancang algoritma dan mengukur efisiensinya

Data Structures and Algorithm
Peran Algoritma dan Struktur Data dalam Pemecahan Masalah

- Pemecahan masalah merupakan bagian penting dari setiap disiplin ilmu.
- Komputer digunakan secara luas untuk memecahkan masalah yang berkaitan dengan berbagai domain, seperti perbankan, perdagangan, kedokteran, manufaktur, dan transportasi.
- Untuk memecahkan suatu masalah tertentu dengan menggunakan komputer, Anda perlu menulis program untuk masalah tersebut.
- Suatu program terdiri dari dua komponen, algoritma dan struktur data.

Data Structures and Algorithm
Peran Algoritma

- Kata algoritma berasal dari nama matematikawan Persia yaitu Al Khawarizmi.
- Suatu algoritma dapat didefinisikan sebagai step-by-step untuk memecahkan suatu masalah.
- Suatu algoritma membantu pengguna memperoleh hasil yang benar dalam sejumlah langkah yang terbatas.

Data Structures and Algorithm
Peran Algoritma

- Suatu algoritma memiliki lima poin penting:
  - Keterbatasan
  - Kepastian
  - Masukan
  - Keluaran
  - Efektivitas

Data Structures and Algorithm
Peran Algoritma

- Algoritma memberikan manfaat berikut:
  - Bantuan dalam penulisan program yang sesuai
  - Membantu membagi masalah-masalah yang sulit menjadi serangkaian masalah kecil yang dapat dipecahkan
  - Menjadikan pengambilan keputusan sebagai proses yang lebih rasional
  - Membuat proses menjadi konsisten dan handal

Data Structures and Algorithm
Peran Struktur Data

- Algoritma yang berbeda dapat digunakan untuk memecahkan masalah yang sama.
- Beberapa algoritma mungkin dapat memecahkan masalah lebih efisien daripada yang lain.
- Suatu algoritma yang memberikan efisiensi maksimum harus digunakan untuk memecahkan suatu masalah.
- Salah satu teknik dasar untuk meningkatkan efisiensi algoritma adalah dengan menggunakan struktur data yang tepat.
- Struktur data didefinisikan sebagai cara untuk mengatur berbagai elemen data dalam memori terhadap satu sama lain.

Data Structures and Algorithm
Peran Struktur Data

- Data dapat diatur dengan berbagai cara. Oleh karena itu, Anda dapat membuat struktur data sebanyak yang Anda inginkan.

- Beberapa struktur data yang terbukti berguna selama bertahun-tahun adalah:
  - Arrays
  - Linked Lists
  - Stacks
  - Queues
  - Trees
  - Graphs

Data Structures and Algorithm
Mengidentifikasi Teknik untuk Mendesain Algoritma

- Dua teknik yang umum digunakan untuk merancang algoritma adalah:
  - Divide and conquer approach
  - Greedy approach

Data Structures and Algorithm
Mengidentifikasi Teknik untuk Mendesain Algoritma

- Divide and conquer merupakan pendekatan yang cukup powerfull untuk memecahkan masalah yang sulit secara konseptual.
- Pendekatan Divide and conquer mengharuskan Anda menemukan cara untuk:
  - Memecah masalah menjadi sub-masalah
  - Memecahkan kasus-kasus mudah
  - Menggabungkan solusi dari sub masalah untuk menyelesaikan masalah sebenarnya

Data Structures and Algorithm
Mengidentifikasi Teknik untuk Mendesain Algoritma

- Algoritma berdasarkan pendekatan greedy digunakan untuk memecahkan masalah optimasi, di mana Anda perlu memaksimalkan keuntungan atau meminimalkan biaya dalam serangkaian kondisi tertentu.

- Contoh masalah optimasi adalah:
  - Menemukan jarak terpendek dari kota asal ke sekumpulan kota tujuan, dengan mempertimbangkan jarak antara pasangan kota tersebut. Misalnya aplikasi maps.

Data Structures and Algorithm
Merancang Algoritma Menggunakan Rekursi

Data Structures and Algorithm
Merancang Algoritma Menggunakan Rekursi

```cpp
#include <iostream>
using namespace std;

int findIndex(int t) {
    int a = 0, b = 1, i = 1;
    while (b < t) {
        int temp = b;
        b = a + b;
        a = temp;
        i++;
    }
    return (b == t) ? i : -1;
}

int main() {
    int x = 3;
    cout << findIndex(x);
}
```

Data Structures and Algorithm
Merancang Algoritma Menggunakan Rekursi

```cpp
#include <iostream>
using namespace std;

int findIndex(int t, int a = 0, int b = 1, int i = 1) {
    return (b == t) ? i : (b > t) ? -1 : findIndex(t, b, a + b, i + 1);
}

int main() {
    int x = 3;
    cout << findIndex(x);
}
```

Data Structures and Algorithm
Menentukan Efisiensi Suatu Algoritma

- Efisiensi suatu algoritma dapat dihitung dengan menentukan jumlah sumber daya yang dikonsumsinya.

- Sumber daya utama yang dikonsumsi oleh suatu algoritma adalah:
  - Time : Waktu CPU yang dibutuhkan untuk mengeksekusi algoritma.
  - Space : Jumlah memori yang digunakan oleh algoritma untuk eksekusinya.

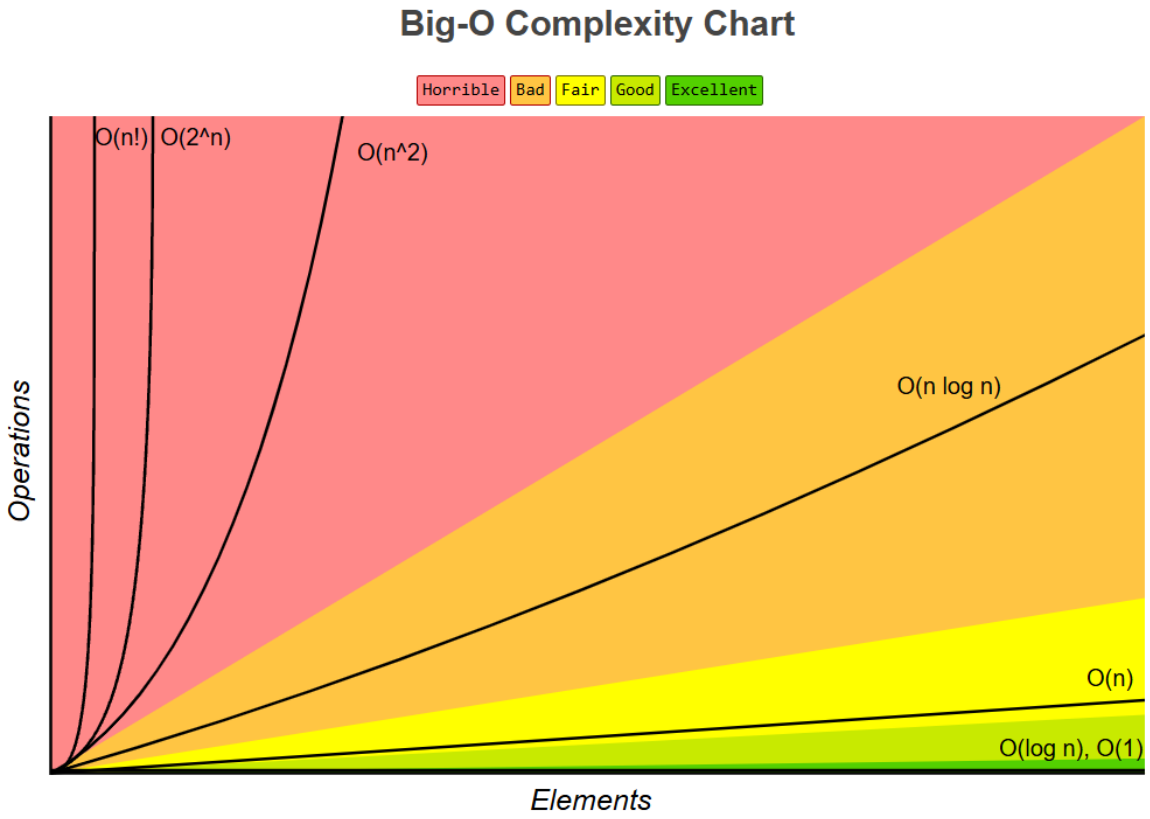
- Semakin sedikit sumber daya yang dikonsumsi suatu algoritma, semakin efisien algoritma tersebut.

Data Structures and Algorithm
Menentukan Efisiensi Suatu Algoritma

- Time/Space Tradeoff:
  - Situasi di mana Anda dapat mengurangi penggunaan memori dengan mengorbankan eksekusi program yang lebih lambat.
  - Contohnya adalah penyimpanan data dalam bentuk terkompresi/tidak terkompresi.

- Memori dapat diperluas, tetapi waktu tidak. Oleh karena itu, pertimbangan waktu umumnya mengesampingkan pertimbangan memori. Contoh:
  - Fibonacci dengan Rekursi (hemat memori, tetapi lambat karena banyak perhitungan ulang).
  - Fibonacci dengan Iterasi atau Dynamic Programming (menggunakan lebih banyak memori untuk menyimpan hasil sebelumnya, tetapi jauh lebih cepat).

Data Structures and Algorithm
Big-O



https://www.bigocheatsheet.com/

Any Question ?