

# Modul 1

## Abstract Data Types

### Tujuan

1. Praktikan dapat memahami konsep abstract data types (ADT)
2. Praktikan dapat menerapkan konsep abstract data types (ADT) dalam pembuatan program

### Tipe Data

Tipe data adalah klasifikasi data berdasarkan bagaimana data tersebut akan digunakan dalam sebuah program. Terdapat 2 hal penting dalam tipe data :

1. Tipe data memiliki domain yaitu himpunan nilai yang terdapat dalam suatu tipe data
2. Setiap tipe data memiliki operasi untuk manipulasi tipe data

Contohnya adalah tipe data `int`

- Domain : bilangan bulat
- Operasi : penambahan, pengurangan, perkalian, pembagian, dan lain sebagainya sesuai bahasa pemrograman yang digunakan

### Tipe Data Dasar


Tipe data dasar adalah tipe data yang biasanya sudah disediakan dari bahasa pemrograman yang digunakan.

Tipe Data	Keyword	Domain	Contoh
Integer	<code>int</code>	Bilangan bulat	-1, 0, 2, 14, 2020
Decimal	<code>float</code>	Bilangan Real	- 2.5, 0.0, 3.75
Character	<code>char</code>	Karakter	'a', '6', '#'
Boolean	<code>bool</code>	Logika	true, false

## Tipe Data Bentuk (User Defined Data Types)

Tipe data bentuk atau seringkali disebut juga *user defined data type* adalah tipe data yang didefinisikan sendiri oleh *user* dan merupakan turunan dari tipe data dasar ataupun tipe data bentuk lain. Operasi pada tipe data bentuk harus didefinisikan atau dibuat sendiri oleh user.

Contohnya adalah menggunakan `typedef` untuk membuat tipe data bentuk dengan



```
typedef struct{
    string nama, npm;
} Mahasiswa;
```

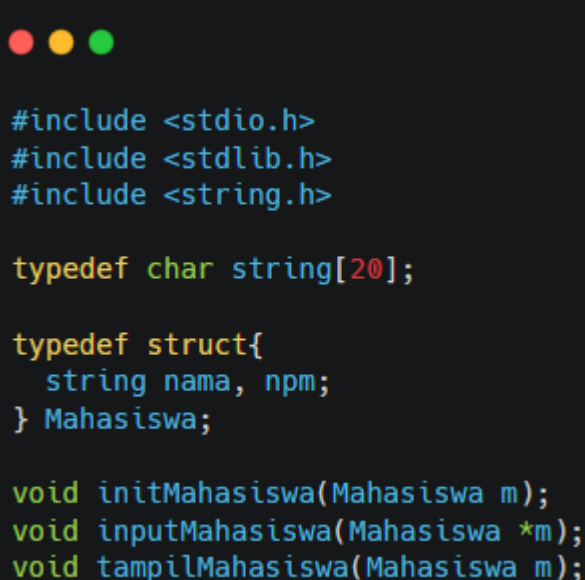
struct

## Abstract Data Types

ADT mirip dengan tipe data bentuk. Pada ADT operasi biasanya didefinisikan menggunakan prosedur atau fungsi namun detail implementasinya disembunyikan. ADT menerapkan konsep abstraksi yaitu proses memodelkan permasalahan dari dunia nyata menjadi model yang akan diselesaikan dengan solusi program.

Contohnya kita ingin membuat sebuah sistem yang dapat menyimpan dan menampilkan nama dan NPM mahasiswa.

header.h



```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef char string[20];

typedef struct{
    string nama, npm;
} Mahasiswa;

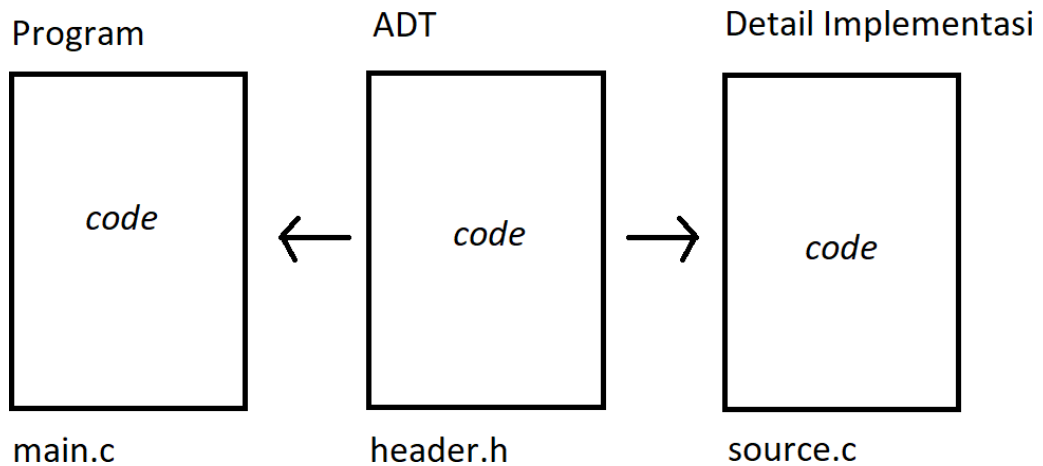
void initMahasiswa(Mahasiswa m);
void inputMahasiswa(Mahasiswa *m);
void tampilMahasiswa(Mahasiswa m);
```

Tipe data `Mahasiswa` menyimpan `npm` dan `nama mahasiswa`

Operasi yang bisa dikenakan pada tipe data `Mahasiswa` :

- `initMahasiswa()` - inialisasi data mahasiswa
- `inputMahasiswa()` - memasukkan data mahasiswa untuk disimpan
- `tampilMahasiswa()` - menampilkan data mahasiswa
- dan lain sebagainya sesuai keperluan *user*

Operasi yang ada pada ADT hanyalah *header* saja (hanya deklarasi fungsi dan prosedur saja) tanpa adanya badan fungsi maka agar operasi tersebut dapat digunakan oleh program maka detail implementasinya perlu ditulis terlebih dahulu atau dengan kata lain kita perlu menulis badan fungsinya.



Berikut merupakan penjelasan singkat dari kegunaan masing masing file :

- `main.c` : berisi code yang kemudian akan dijalankan dan ditampilkan kepada user sebagai antarmuka
- `header.h` : berisi library, deklarasi prosedur dan fungsi yang akan dipakai
- `source.c` : penulisan badan atau isi prosedur dan fungsi yang sudah dideklarasikan pada file `header.h`

Dengan menggunakan ADT apabila jika ke depannya akan ada perubahan pada detail implementasi program maka kita hanya perlu mengganti *code* yang ada di detail implementasi tanpa perlu mengubah code pada program utama yang menggunakan ADT tersebut.

***“First, solve the problem. Then, write the code”***

- John Johnson