LAPORAN PRAKTIKUM
STRUKTUR DATA

MODUL 3
ABSTRACT DATA TYPE
(ADT)

Disusun Oleh :

NAMA : Muhammad Omar Nadiv

NIM : 103112430063

Dosen

Fahrudin Mukti Wibowo, S.Kom., M.Eng.

PROGRAM STUDI STRUKTUR DATA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

A. Dasar Teori

Tipe data abstrak (TDA) atau lebih dikenal dalam bahasa Inggris sebagai Abstract data type (ADT) merupakan model matematika yang merujuk pada sejumlah bentuk struktur data yang memiliki kegunaan atau perilaku yang serupa; atau suatu tipe data dari suatu bahasa pemrograman yang memiliki sematik yang serupa.

Tipe data abstrak umumnya didefinisikan tidak secara langsung, melainkan hanya melalui operasi matematis tertentu sehingga membutuhkan penggunaan tipe data tersebut meski dengan risiko kompleksitas yang lebih tinggi atas operasi tersebut.

Sedangkan Structure atau struct adalah kumpulan dari beberapa variabel dengan beragam tipe data yang dibungkus dalam satu variabel. Struct yang menggunakan typedef akan dianggap sebagai tipe data dan kita bisa buat variabel instance tanpa harus menggunakan struct.

B. Guided (berisi screenshot source code & output program disertai penjelasannya)

Guided 1 (file mahasiswa.cpp)

```
#include "mahasiswa.h"
#include <iostream>
using namespace std;

void inputMhs(mahasiswa &m)
{
    cout << "input nama = ";
    cin >> (m).nim;
    cout << "input nilai = ";
    cin >> (m).nilai1;
    cout << "input nilai2 = ";
    cin >> (m).nilai2;
}

float rata2(mahasiswa m)
{
    return float(m.nilai1 + m.nilai2) / 2;
}
```

Guided 1 (file mahasiswa.h)

```
#ifndef MAHASISWA_H_INCLUDED
#define MAHASISWA_H_INCLUDED
struct mahasiswa
{
   char nim[10];
   int nilai1, nilai2;
};
void inputMhs(mahasiswa &m);
float rata2(mahasiswa m);
#endif
```

Guided 1 (file main.cpp)

```
#include <iostream>
#include "mahasiswa.h"
using namespace std;

int main()
{
   mahasiswa mhs;
   inputMhs(mhs);
   cout << "rata - rata = " << rata2(mhs) << endl;
   return 0;
}
```

Screenshots Output

```
nadiv@omarnadip Guided % g++ main.cpp mahasiswa.cpp -o main && ./main
input nama = nadiv
input nilai = 90
input nilai2 = 95
rata - rata = 92.5
nadiv@omarnadip Guided %
```

Deskripsi:

Program ini terdiri dari tiga file, yaitu main.cpp, mahasiswa.cpp, dan mahasiswa.h, yang bersama-sama digunakan untuk mengelola data mahasiswa secara modular. File mahasiswa.h berisi definisi struct mahasiswa yang menyimpan data NIM serta dua nilai ujian, dan juga deklarasi fungsi inputMhs dan rata2. File mahasiswa.cpp berisi implementasi dari fungsi-fungsi tersebut inputMhs untuk menerima input

NIM dan dua nilai dari pengguna, serta rata2 untuk menghitung rata-rata dari dua nilai yang dimasukkan.

Sementara itu, main.cpp berfungsi sebagai program utama yang membuat variabel bertipe mahasiswa, memanggil fungsi input untuk mengisi datanya, kemudian memanggil fungsi rata2 untuk menghitung dan menampilkan rata-rata nilai mahasiswa tersebut.

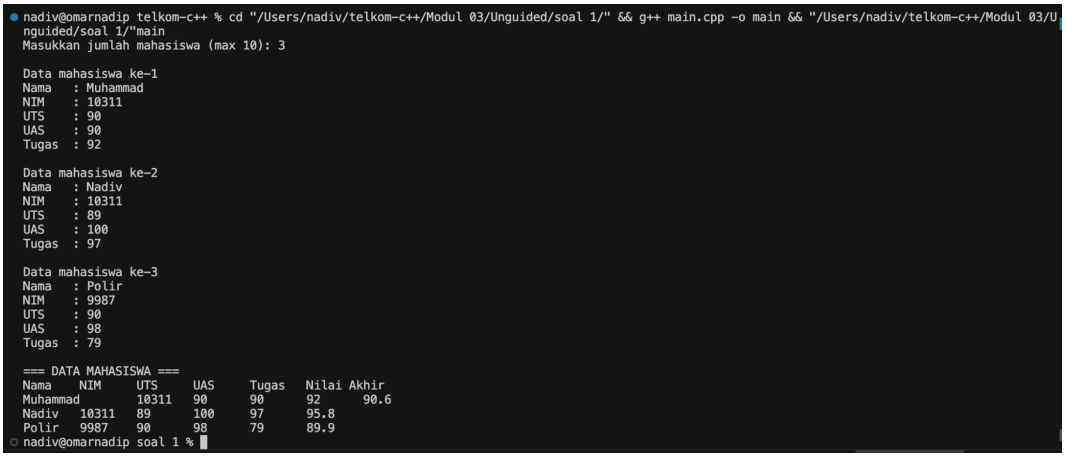
C. Unguided/Tugas (berisi screenshot source code & output program disertai penjelasannya)

Unguided 1

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

struct Mahasiswa
{
   string nama;
   string nim;
   float uts;
   float uas;
   float tugas;
   float nilaiAkhir;
};

float hitungNilaiAkhir(float uts, float uas, float tugas)
{
   return (0.3 * uts) + (0.4 * uas) + (0.3 * tugas);
}

int main()
{
   Mahasiswa mhs[10];
   int n;

   cout << "Masukkan jumlah mahasiswa (max 10): ";
   cin >> n;

   for (int i = 0; i < n; i++)
   {
       cout << "\nData mahasiswa ke-" << i + 1 << endl;
```

```
        cin.ignore();
        cout << "Nama   : ";
        getline(cin, mhs[i].nama);
        cout << "NIM    : ";
        getline(cin, mhs[i].nim);
        cout << "UTS    : ";
        cin >> mhs[i].uts;
        cout << "UAS    : ";
        cin >> mhs[i].uas;
        cout << "Tugas  : ";
        cin >> mhs[i].tugas;

        mhs[i].nilaiAkhir = hitungNilaiAkhir(mhs[i].uts, mhs[i].uas,
mhs[i].tugas);
    }

    cout << "\n=== DATA MAHASISWA ===\n";
    cout << "Nama\tNIM\tUTS\tUAS\tTugas\tNilai Akhir\n";
    for (int i = 0; i < n; i++)
    {
        cout << mhs[i].nama << "\t"
             << mhs[i].nim << "\t"
             << mhs[i].uts << "\t"
             << mhs[i].uas << "\t"
             << mhs[i].tugas << "\t"
             << mhs[i].nilaiAkhir << endl;
    }

    return 0;
}
```

Screenshots Output



Deskripsi:

Program ini digunakan untuk menyimpan dan menghitung nilai akhir mahasiswa dengan menggunakan struct dan array. Struct Mahasiswa menyimpan data berupa nama, NIM, nilai UTS, UAS, tugas, dan nilai akhir. Program meminta pengguna memasukkan jumlah mahasiswa, kemudian menginput data setiap mahasiswa. Nilai akhir dihitung menggunakan fungsi hitungNilaiAkhir dengan rumus $0.3\times UTS+0.4\times UAS+0.3\times Tugas$. Setelah semua data dimasukkan dan dihitung, program menampilkan seluruh data mahasiswa beserta nilai akhirnya dalam bentuk tabel sederhana.

Unguided 2 (file pelajaran.h)

```
#ifndef PELAJARAN_H
#define PELAJARAN_H

#include <string>
using namespace std;

struct Pelajaran
{
   string namaMapel;
   string kodeMapel;
};

Pelajaran create_pelajaran(string namapel, string kodepel);
void tampil_pelajaran(Pelajaran pel);

#endif
```

Unguided 2 (file pelajaran.cpp)

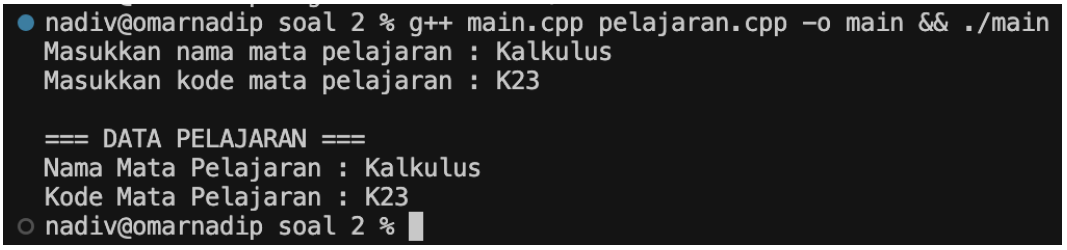
```
#include <iostream>
#include "pelajaran.h"
using namespace std;

Pelajaran create_pelajaran(string namapel, string kodepel)
{
   Pelajaran pel;
   pel.namaMapel = namapel;
   pel.kodeMapel = kodepel;
   return pel;
}

void tampil_pelajaran(Pelajaran pel)
{
   cout << "Nama Mata Pelajaran : " << pel.namaMapel << endl;
   cout << "Kode Mata Pelajaran : " << pel.kodeMapel << endl;
}
```

Unguided 2 (file main.cpp)

```
#include <iostream>
#include "pelajaran.h"
```

```cpp
using namespace std;

int main() {
    string nama, kode;
    cout << "Masukkan nama mata pelajaran : ";
    getline(cin, nama);
    cout << "Masukkan kode mata pelajaran : ";
    getline(cin, kode);

    Pelajaran pel = create_pelajaran(nama, kode);
    cout << "\n=== DATA PELAJARAN ===" << endl;
    tampil_pelajaran(pel);

    return 0;
}
```

Screenshots Output



Deskripsi:

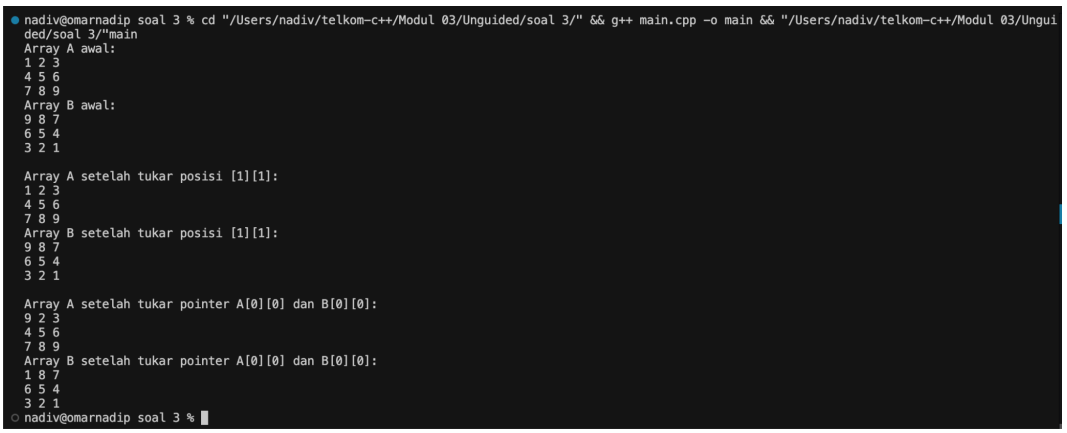
Program ini menggunakan konsep ADT (Abstract Data Type) untuk mengelola data mata pelajaran secara terstruktur dan modular. File pelajaran.h berisi deklarasi struct Pelajaran yang menyimpan nama dan kode mata pelajaran, serta deklarasi fungsi create_pelajaran dan tampil_pelajaran. File pelajaran.cpp berisi implementasi fungsi create_pelajaran untuk membuat dan mengembalikan objek Pelajaran berdasarkan input user, dan tampil_pelajaran untuk menampilkan data yang telah dimasukkan. Pada main.cpp, program meminta user untuk menginput nama dan kode mata pelajaran, lalu memanggil create_pelajaran untuk menyimpannya dalam sebuah objek pel. Setelah itu, fungsi tampil_pelajaran dipanggil untuk menampilkan hasilnya.

Unguided 3

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

const int BARIS = 3;
const int KOLOM = 3;

void tampilArray(int arr[BARIS][KOLOM]) {
   for (int i = 0; i < BARIS; i++) {
       for (int j = 0; j < KOLOM; j++) {
           cout << arr[i][j] << " ";
       }
       cout << endl;
   }
}

void tukarPosisi(int arr1[BARIS][KOLOM], int arr2[BARIS][KOLOM], int baris,
int kolom) {
   int temp = arr1[baris][kolom];
   arr1[baris][kolom] = arr2[baris][kolom];
   arr2[baris][kolom] = temp;
}

void tukarPointer(int *p1, int *p2) {
   int temp = *p1;
   *p1 = *p2;
   *p2 = temp;
}

int main() {
   int A[BARIS][KOLOM] = {
       {1, 2, 3},
       {4, 5, 6},
       {7, 8, 9}
   };

   int B[BARIS][KOLOM] = {
       {9, 8, 7},
       {6, 5, 4},
       {3, 2, 1}
   };
```

```
    int *ptr1 = &A[0][0];
    int *ptr2 = &B[0][0];

    cout << "Array A awal:\n";
    tampilArray(A);
    cout << "Array B awal:\n";
    tampilArray(B);

    tukarPosisi(A, B, 1, 1);

    cout << "\nArray A setelah tukar posisi [1][1]:\n";
    tampilArray(A);
    cout << "Array B setelah tukar posisi [1][1]:\n";
    tampilArray(B);

    tukarPointer(ptr1, ptr2);

    cout << "\nArray A setelah tukar pointer A[0][0] dan B[0][0]:\n";
    tampilArray(A);
    cout << "Array B setelah tukar pointer A[0][0] dan B[0][0]:\n";
    tampilArray(B);

    return 0;
}
```

Screenshots Output



Deskripsi:

Program ini menunjukkan cara penggunaan array 2 dimensi dan pointer dalam C++. Program mendefinisikan dua buah array 3×3 (A dan B) yang masing-masing berisi nilai berbeda. Fungsi tampilArray digunakan untuk menampilkan isi array ke layar dalam bentuk matriks. Fungsi tukarPosisi berfungsi untuk menukar elemen pada posisi tertentu antara dua array 2 dimensi, dengan menggunakan indeks baris dan kolom sebagai penentu posisi. Selanjutnya, fungsi tukarPointer digunakan untuk menukar nilai dari dua elemen array yang ditunjuk oleh pointer ptr1 dan ptr2.

Pada main, program menampilkan isi awal kedua array, lalu melakukan pertukaran elemen pada posisi [1][1] di kedua array. Setelah itu, pointer ptr1 dan ptr2 (yang menunjuk ke elemen pertama dari array A dan B) juga ditukar nilainya. Hasilnya, nilai pada elemen pertama dari kedua array ikut tertukar.

D. Kesimpulan

Pada modul 3 ini, saya belajar bagaimana menggunakan struktur data dasar seperti struct, fungsi, array, dan pointer dalam pemrograman C++. Saya memahami cara menyimpan dan mengolah data menggunakan struct, menghitung nilai menggunakan fungsi, serta membuat ADT (Abstract Data Type) untuk memisahkan logika program. Selain itu, saya juga belajar cara menukar data antar array 2 dimensi dan menggunakan pointer untuk memanipulasi nilai secara langsung. Pembelajaran ini membantu saya memahami logika program yang lebih efisien dan terorganisir.

E. Referensi

https://www.petanikode.com/cpp-struct/

https://id.wikipedia.org/wiki/Tipe_data_abstrak