LAPORAN PRAKTIKUM
STRUKTUR DATA

MODUL III
ABSTRACT DATA TYPE (ADT)

Disusun Oleh :

NAMA : Balawan Satria Lhaksana Putra M.

NIM : 103112430004

Dosen

FAHRUDIN MUKTI WIBOWO

PROGRAM STUDI STRUKTUR DATA

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2025

A. Dasar Teori

Abstract Data Type (ADT) atau Tipe Data Abstrak merupakan konsep penting dalam pemrograman yang mendefinisikan struktur data berdasarkan perilaku dan operasinya, bukan pada cara implementasinya. Dalam bahasa C++, ADT dapat diwujudkan menggunakan struct, class, serta kombinasi antara header file (.h) dan source file (.cpp) untuk memisahkan deklarasi dan implementasi. Pendekatan ini membantu menjaga modularitas, kemudahan pemeliharaan, serta keterbacaan kode.

Selain itu, konsep fungsi dan prosedur digunakan untuk membangun logika terpisah, sedangkan pointer memungkinkan manipulasi data secara langsung melalui alamat memori. Penggunaan array 2D, fungsi untuk pertukaran nilai, serta penerapan ADT pelajaran dan mahasiswa pada laporan ini menunjukkan bagaimana ADT digunakan untuk mengelola data yang kompleks secara terstruktur dan efisien.

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

Guided 1 (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 (mahasiswa.cpp)

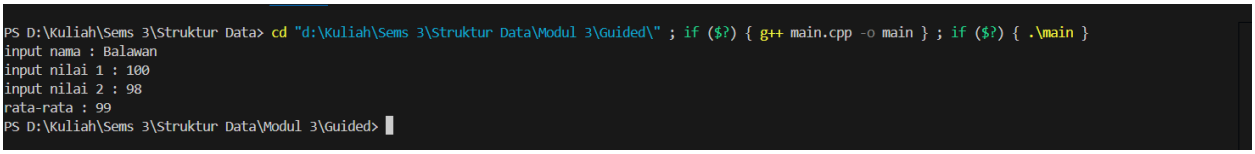
```
#include "mahasiswa.h"
#include <iostream>

using namespace std;

void inputMhs(mahasiswa &m)
{
    cout << "input nama : ";
    cin >> m.nim;
    cout << "input nilai 1 : ";
    cin >> m.nilai1;
    cout << "input nilai 2 : ";
    cin >> m.nilai2;
```

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

Guided 1 (main.cpp)

```
#include <iostream>
#include "mahasiswa.h"
#include "mahasiswa.cpp"

using namespace std;

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

Screenshots Output



Deskripsi:

Program ini menggunakan header file (mahasiswa.h) dan file implementasi (mahasiswa.cpp) untuk memisahkan deklarasi dan definisi fungsi, sehingga struktur program menjadi lebih rapi dan modular.

Di dalam header, didefinisikan struct mahasiswa yang berisi atribut nim, nilai1, dan nilai2, serta deklarasi dua fungsi yaitu inputMhs() untuk menerima input data mahasiswa dan rata2() untuk menghitung rata-rata dari dua nilai.

Pada file mahasiswa.cpp, fungsi inputMhs() digunakan untuk meminta pengguna menginput NIM serta dua nilai, sedangkan fungsi rata2() menghitung rata-rata dari kedua nilai tersebut dan mengembalikannya dalam bentuk float.

Kemudian pada fungsi main(), objek mahasiswa mhs dibuat, lalu program memanggil fungsi inputMhs() untuk mengisi data mahasiswa dan menampilkan hasil perhitungan rata-ratanya menggunakan fungsi rata2().

Tujuan dari program ini adalah untuk mendemonstrasikan penggunaan struct, fungsi dengan parameter referensi, serta konsep pemisahan kode menggunakan header dan source file di C++.

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

Unguided 1

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

// Struktur data mahasiswa
struct Mahasiswa {
    string nama;
    string nim;
    float uts, uas, tugas, nilaiAkhir;
};

// Fungsi untuk menghitung nilai akhir
float hitungNilaiAkhir(float uts, float uas, float tugas) {
    return (0.3 * uts) + (0.4 * uas) + (0.3 * tugas);
}

int main() {
    Mahasiswa data[10];
    int jumlah;

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

    if (jumlah > 10) {
        cout << "Jumlah melebihi batas maksimum (10)!" << endl;
        return 0;
    }

    // Input data mahasiswa
    for (int i = 0; i < jumlah; i++) {
        cout << "\nData Mahasiswa ke-" << i + 1 << endl;
        cout << "Nama   : ";
        cin.ignore();
        getline(cin, data[i].nama);
        cout << "NIM    : ";
        cin >> data[i].nim;
        cout << "Nilai UTS  : ";
        cin >> data[i].uts;
        cout << "Nilai UAS  : ";
        cin >> data[i].uas;
        cout << "Nilai Tugas: ";
        cin >> data[i].tugas;

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

```
    // Tampilkan data mahasiswa
    cout << "\n=== Data Mahasiswa ===\n";
    for (int i = 0; i < jumlah; i++) {
        cout << "\nNama   : " << data[i].nama;
        cout << "\nNIM    : " << data[i].nim;
        cout << "\nUTS    : " << data[i].uts;
        cout << "\nUAS    : " << data[i].uas;
        cout << "\nTugas  : " << data[i].tugas;
        cout << "\nNilai Akhir : " << data[i].nilaiAkhir << endl;
    }

    return 0;
}
```

Screenshots Output

Deskripsi:

Program ini digunakan untuk menyimpan dan menampilkan data maksimal 10 mahasiswa, yang mencakup nama, NIM, nilai UTS, UAS, tugas, serta nilai akhir. Program menggunakan struktur Mahasiswa untuk menyimpan data tiap mahasiswa dalam array. Nilai akhir dihitung melalui fungsi hitungNilaiAkhir() dengan rumus 0.3 * uts + 0.4 * uas + 0.3 * tugas. Setelah pengguna memasukkan data mahasiswa, program akan menghitung nilai akhir masing-masing dan menampilkannya kembali secara berurutan di layar.

Unguided 2 (pelajaran.h)

```
#ifndef PELAJARAN_H_INCLUDED
#define PELAJARAN_H_INCLUDED
#include <string>
using namespace std;

struct pelajaran {
    string namaMapel;
    string kodeMapel;
};

pelajaran create_pelajaran(string namaMapel, string kodeMapel);

void tampil_pelajaran(pelajaran pel);
```

```
#endif
```

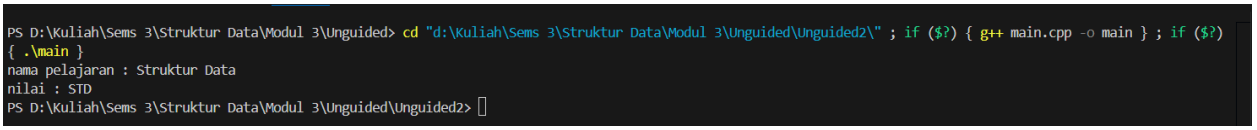
Unguided 2 (pelajaran.cpp)

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

pelajaran create_pelajaran(string namaMapel, string kodeMapel) {
    pelajaran p;
    p.namaMapel = namaMapel;
    p.kodeMapel = kodeMapel;
    return p;
}

void tampil_pelajaran(pelajaran pel) {
    cout << "nama pelajaran : " << pel.namaMapel << endl;
    cout << "nilai : " << pel.kodeMapel << endl;
}
```

Unguided 2 (main.cpp)

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

int main() {
    string namapel = "Struktur Data";
    string kodepel = "STD";

    pelajaran pel = create_pelajaran(namapel, kodepel);
    tampil_pelajaran(pel);

    return 0;
}
```

Screenshots Output



Deskripsi:

Program ini menggunakan Abstract Data Type (ADT) bernama pelajaran yang disimpan dalam tiga file, yaitu pelajaran.h, pelajaran.cpp, dan main.cpp. Pada file pelajaran.h, didefinisikan struktur pelajaran yang memiliki dua atribut yaitu namaMapel dan kodeMapel, serta deklarasi fungsi create_pelajaran() dan prosedur tampil_pelajaran().

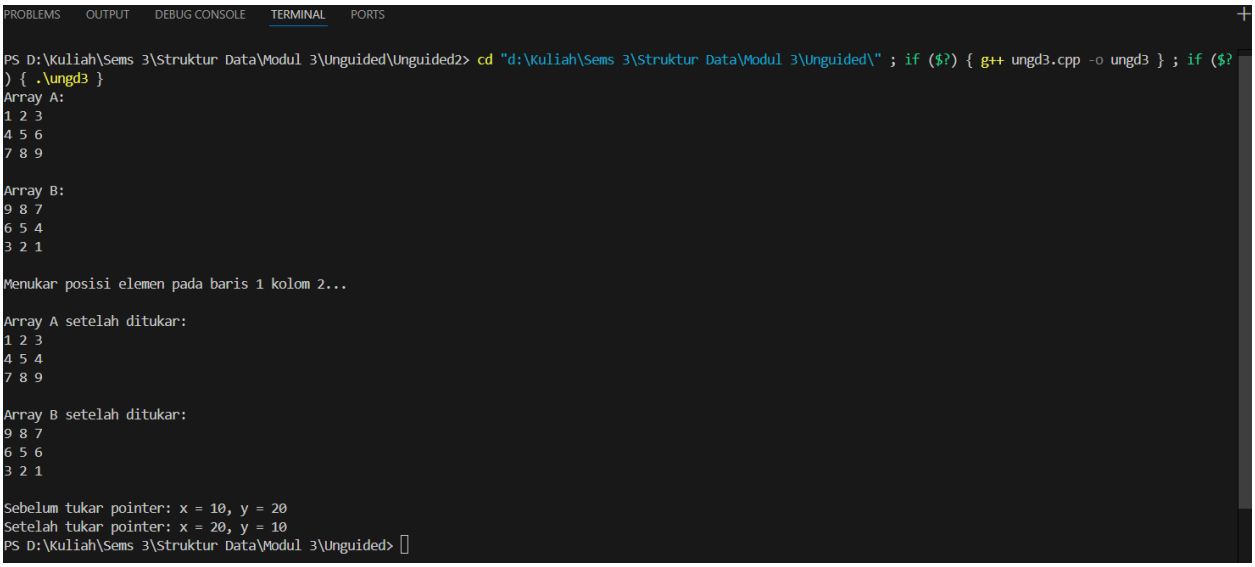
Pada file pelajaran.cpp, fungsi create_pelajaran() digunakan untuk membuat dan mengembalikan data pelajaran berdasarkan input nama dan kode, sedangkan prosedur tampil_pelajaran() menampilkan informasi pelajaran ke layar. File main.cpp berfungsi sebagai program utama yang memanggil fungsi dan prosedur tersebut untuk menampilkan hasil berupa nama pelajaran dan kodenya.

Unguided 3

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

// Fungsi untuk menampilkan isi array 2D
void tampilArray(int arr[3][3]) {
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }
}

// Fungsi untuk menukarkan isi dua array 2D pada posisi tertentu
void tukarPosisi(int arr1[3][3], int arr2[3][3], int baris, int kolom) {
    int temp = arr1[baris][kolom];
    arr1[baris][kolom] = arr2[baris][kolom];
    arr2[baris][kolom] = temp;
}

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

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

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

    int *ptr1, *ptr2;
```

```
    int x = 10, y = 20;
    ptr1 = &x;
    ptr2 = &y;

    cout << "Array A:" << endl;
    tampilArray(A);
    cout << "\nArray B:" << endl;
    tampilArray(B);


    cout << "\nMenukar posisi elemen pada baris 1 kolom 2..." << endl;
    tukarPosisi(A, B, 1, 2);

    cout << "\nArray A setelah ditukar:" << endl;
    tampilArray(A);
    cout << "\nArray B setelah ditukar:" << endl;
    tampilArray(B);


    cout << "\nSebelum tukar pointer: x = " << x << ", y = " << y <<
endl;
    tukarPointer(ptr1, ptr2);
    cout << "Setelah tukar pointer: x = " << x << ", y = " << y << endl;

    return 0;
}
```

Screenshots Output



Deskripsi:

Program ini menggunakan dua array 2 dimensi berukuran 3×3 dan dua pointer integer untuk mendemonstrasikan konsep pertukaran nilai. Fungsi tampilArray() digunakan untuk menampilkan isi array 2D, sedangkan fungsi tukarPosisi() menukar elemen antara dua

array pada posisi baris dan kolom tertentu. Selain itu, terdapat fungsi tukarPointer() yang menukar nilai dua variabel melalui pointer. Pada bagian utama program (main), array A dan B ditampilkan sebelum dan sesudah pertukaran, serta variabel x dan y juga ditukar menggunakan pointer untuk menunjukkan hasil perubahannya.

D. Kesimpulan

Dari hasil praktikum, dapat disimpulkan bahwa penerapan Abstract Data Type (ADT) dalam C++ membantu mengorganisasi program menjadi lebih modular, efisien, dan mudah dipahami. Dengan memisahkan deklarasi dan implementasi menggunakan file header dan source, pemrograman menjadi lebih terstruktur. Mahasiswa juga memahami cara menggunakan struct, fungsi, array 2D, serta pointer untuk melakukan operasi pertukaran data dan manipulasi nilai secara langsung. Secara keseluruhan, praktikum ini memperkuat pemahaman konsep dasar ADT dan penerapannya dalam membangun program yang terarah dan fleksibel.

E. Referensi

- https://id.scribd.com/document/411113387/Abstract-Data-Type-ADT-1
- https://brilliant.org/wiki/abstract-data-types/
- https://www.scholarhat.com/tutorial/datastructures/abstract-data-type