LAPORAN PRAKTIKUM

MODUL 3

“ABSTRACT DATA TYPE (ADT) ”

Disusun Oleh :

Farhan Kurniawan (2311104073)

Kelas:

SE-07-B

Dosen :

Wahyu Andi Saputra, S.Pd, M.Eng,

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

I. TUJUAN.

1. Memahami konsep Abstract Data Type (ADT) dan penggunaannya dalam pemrograman

II. LANDASAN TEORI

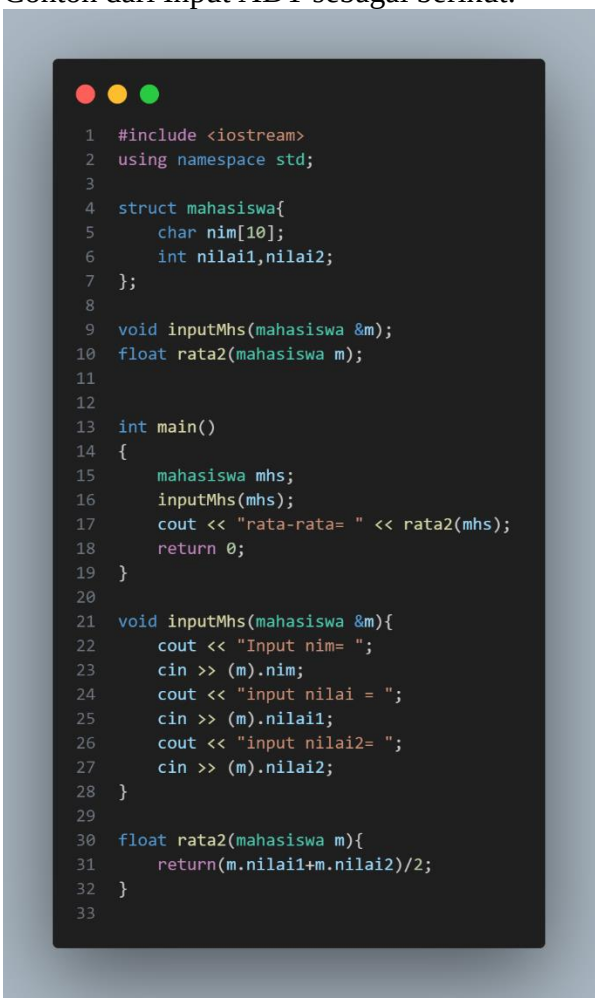
Abstract Data Type (ADT) adalah konsep fundamental dalam pemrograman yang mengacu pada jenis data yang didefinisikan melalui perilaku atau operasi-operasi yang dapat dilakukan terhadap data tersebut, tanpa memperhatikan bagaimana data tersebut diimplementasikan secara konkret. Dalam konteks bahasa pemrograman C++, ADT merupakan kumpulan nilai dan operasi yang terkait, di mana detail implementasinya disembunyikan dari pengguna. Contoh ADT yang sering digunakan termasuk **List**, **Stack**, **Queue**, dan **Tree**. ADT memungkinkan pemisahan antara spesifikasi dan implementasi, sehingga memudahkan dalam merancang program yang modular, efisien, dan mudah dipelihara. Penggunaan ADT dalam C++ biasanya diwujudkan melalui class, di mana class tersebut mendefinisikan atribut (data) dan metode (operasi) yang dapat diakses oleh pengguna tanpa mengetahui cara kerja internalnya. Prinsip ini mendukung konsep **encapsulation** dalam pemrograman berorientasi objek, yang melindungi data dan memastikan bahwa interaksi dengan data dilakukan melalui interface yang telah ditentukan.

III. GUIDE

3.1. Abstract Data Type (ADT)

ADT (Abstract Data Type) adalah tipe data abstrak yang terdiri dari sekumpulan operasi dasar atau primitif yang dapat dilakukan terhadap tipe tersebut. Dalam ADT yang lengkap, disertakan juga invarian tipe dan aksioma yang berlaku, sehingga ADT merupakan definisi statis. Definisi tipe dalam sebuah ADT dapat mencakup ADT lain, seperti ADT waktu yang terdiri dari ADT jam dan ADT tanggal, atau segi empat yang dibentuk oleh pasangan dua buah titik (Top, Left) dan (Bottom, Right). Tipe dalam ADT biasanya diterjemahkan ke tipe terdefinisi dalam bahasa pemrograman terkait. Misalnya, dalam C, tipe diterjemahkan sebagai **struct** dan operasi primitif sebagai fungsi atau prosedur. Primitif dalam ADT meliputi beberapa kategori, seperti konstruktor (pembentuk nilai tipe), selector (akses komponen tipe), prosedur pengubah nilai komponen, validator (untuk memeriksa validitas tipe), destruktor (untuk menghapus nilai dan memori), operasi input/output, operator relasional (seperti lebih besar atau sama dengan), operasi aritmatika, serta konversi antar tipe. ADT biasanya diimplementasikan dalam dua modul utama, yaitu definisi spesifikasi tipe dan primitif (header file .h) dan realisasi primitif (body file .cpp). Implementasi fungsi dan prosedur umumnya memanfaatkan konstruktor dan selector untuk mempermudah pengelolaan data.

Contoh dari Input ADT sebagai berikut:



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

struct mahasiswa{
    char nim[10];
    int nilai1,nilai2;
};

void inputMhs(mahasiswa &m);
float rata2(mahasiswa m);


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

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

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

```

Dari inputan diatas akan menghasilkan output sebagai berikut:

```
PS C:\Users\Farhan Kurniawan\Desktop\pemograman\c++> cd
) { .\pertemuan4 }
Input nim= 2311104073
input nilai = 20
input nilai2= 30
rata-rata= 25
PS C:\Users\Farhan Kurniawan\Desktop\pemograman\c++>
```

Program ini bertujuan untuk menghitung rata-rata dua nilai yang dimiliki seorang mahasiswa. Program menggunakan tipe data struct untuk mendefinisikan data mahasiswa yang terdiri dari NIM, nilai pertama, dan nilai kedua. Input data dilakukan melalui fungsi inputMhs(), sementara perhitungan rata-rata dilakukan oleh fungsi rata2(). Program ini menunjukkan penggunaan dasar struct, fungsi, dan referensi dalam C++.

IV. UNGUIDED

1. Buat program yang dapat menyimpan data mahasiswa (max. 10) ke dalam sebuah *array* dengan field nama, nim, uts, uas, tugas, dan nilai akhir. Nilai akhir diperoleh dari FUNGSI dengan rumus 0.3*uts+0.4*uas+0.3*tugas.
Jawabannya adalah:

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

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


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


void inputMahasiswa(Mahasiswa &mhs) {
    cout << "Masukkan Nama: ";
    cin.ignore();
    getline(cin, mhs.nama);
    cout << "Masukkan NIM: ";
    cin >> mhs.nim;
    cout << "Masukkan Nilai UTS: ";
    cin >> mhs.uts;
    cout << "Masukkan Nilai UAS: ";
    cin >> mhs.uas;
    cout << "Masukkan Nilai Tugas: ";
    cin >> mhs.tugas;


    mhs.nilai_akhir = hitungNilaiAkhir(mhs.uts, mhs.uas, mhs.tugas);
}


void tampilkanMahasiswa(const Mahasiswa &mhs) {
    cout << "Nama: " << mhs.nama << endl;
    cout << "NIM: " << mhs.nim << endl;
    cout << "Nilai UTS: " << mhs.uts << endl;
    cout << "Nilai UAS: " << mhs.uas << endl;
    cout << "Nilai Tugas: " << mhs.tugas << endl;
    cout << "Nilai Akhir: " << mhs.nilai_akhir << endl;
}

int main() {
    const int jumlah_mahasiswa = 10;
    Mahasiswa data_mahasiswa[jumlah_mahasiswa];
    int n;

    cout << "Berapa jumlah mahasiswa yang akan diinput (max 10)? ";
    cin >> n;

    if(n > jumlah_mahasiswa) {
        cout << "Jumlah mahasiswa melebihi batas maksimal (10)." << endl;
        return 1;
    }


    for(int i = 0; i < n; i++) {
        cout << "\nMasukkan data mahasiswa ke-" << i+1 << endl;
        inputMahasiswa(data_mahasiswa[i]);
    }


    cout << "\nData Mahasiswa:" << endl;
    for(int i = 0; i < n; i++) {
        cout << "\nMahasiswa ke-" << i+1 << endl;
        tampilkanMahasiswa(data_mahasiswa[i]);
    }

    return 0;
}

```

Maka akan menghasilkan output sebagai berikut:

```
PS C:\Users\Farhan Kurniawan\Desktop\pemograman\c++> cd
 Unguided3 } ; if ($?) { .\Unguided3 }
Berapa jumlah mahasiswa yang akan diinput (max 10)? 1

Masukkan data mahasiswa ke-1
Masukkan Nama: Farhan Kurniawan
Masukkan NIM: 2311104073
Masukkan Nilai UTS: 85
Masukkan Nilai UAS: 90
Masukkan Nilai Tugas: 95
```

```
Data Mahasiswa:

Mahasiswa ke-1
Nama: Farhan Kurniawan
NIM: 2311104073
Nilai UTS: 85
Nilai UAS: 90
Nilai Tugas: 95
Nilai Akhir: 90
PS C:\Users\Farhan Kurniawan\Desktop\pemograman\c++>
```

2. Buatlah ADT pelajaran sebagai berikut di dalam *file* “pelajaran.h”:

```
tipe pelajaran <
        namaMapel : string
        kodeMapel : string
>
fungsi create_pelajaran( namapel : string, kodepel : string ) 
pelajaran
prosedur tampil_pelajaran( pel : pelajaran )
```

Buatlah implementasi ADT pelajaran pada *file* “pelajaran.cpp” Cobalah hasil implementasi ADT pada *file* “main.cpp”

```
using namespace std;
int main(){
 string namapel = "Struktur Data";  string
kodepel = "STD";
 pelajaran pel = create_pelajaran(namapel,kodepel);  tampil_pelajaran(pel);

 return 0;
}
```

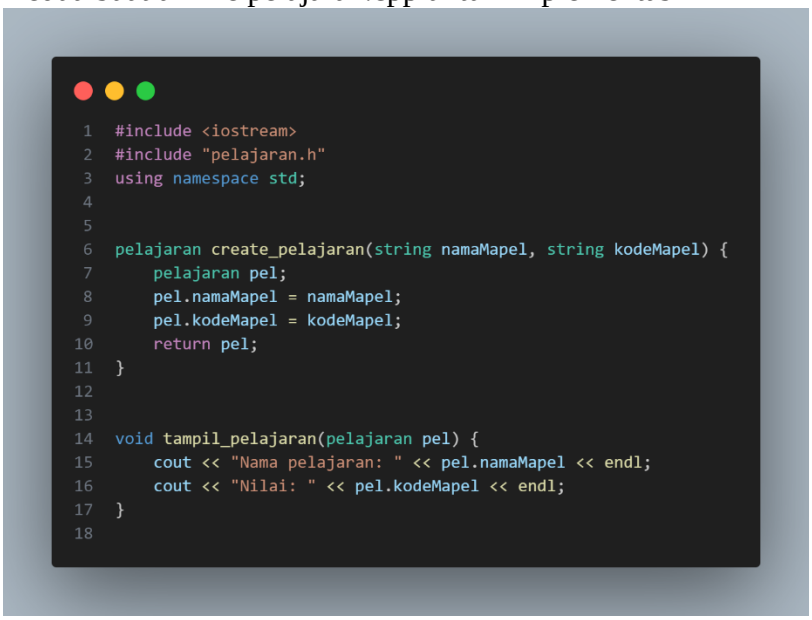
output hasil:

```
nama pelajaran : Struktur Data
nilai : STD
```

Jawabannya adalah:
Pertama buatlah file pelajaran.h

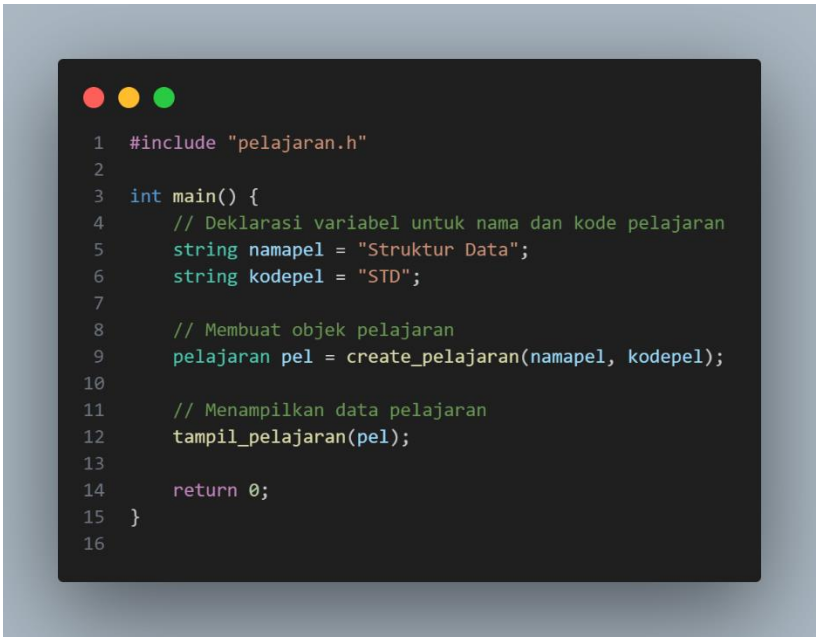
```h
#ifndef PELAJARAN_H
#define PELAJARAN_H

#include <string>
using namespace std;


struct pelajaran {
    string namaMapel;
    string kodeMapel;
};


pelajaran create_pelajaran(string namaMapel, string kodeMapel);


void tampil_pelajaran(pelajaran pel);

#endif

```

Kedua buatlah file pelajaran.cpp untuk implementasi ADT



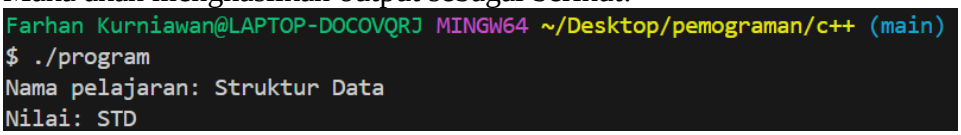
```cpp
#include <iostream>
#include "pelajaran.h"
using namespace std;


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


void tampil_pelajaran(pelajaran pel) {
    cout << "Nama pelajaran: " << pel.namaMapel << endl;
    cout << "Nilai: " << pel.kodeMapel << endl;
}

```

Ketiga buatlah file main.cpp untuk membuat program utama agar bisa di eksekusi.



```cpp
#include "pelajaran.h"

int main() {
    // Deklarasi variabel untuk nama dan kode pelajaran
    string namapel = "Struktur Data";
    string kodepel = "STD";

    // Membuat objek pelajaran
    pelajaran pel = create_pelajaran(namapel, kodepel);

    // Menampilkan data pelajaran
    tampil_pelajaran(pel);

    return 0;
}

```

Maka akan menghasilkan output sebagai berikut:



```
Farhan Kurniawan@LAPTOP-DOCOVQRJ MINGW64 ~/Desktop/pemograman/c++ (main)
$ ./program
Nama pelajaran: Struktur Data
Nilai: STD
```

V. KESIMPULAN

Abstract Data Type (ADT) adalah konsep penting dalam pemrograman yang memisahkan spesifikasi logis dari implementasi teknis suatu tipe data. Dengan ADT, programmer dapat mendefinisikan tipe data dan operasi-operasi yang dapat dilakukan terhadapnya tanpa perlu mengungkapkan bagaimana data tersebut disimpan atau bagaimana operasinya diimplementasikan. Ini memungkinkan pengembangan program yang lebih modular, mudah dipahami, dan mudah dikelola. ADT juga mendukung prinsip encapsulation, di mana detail implementasi tersembunyi dari pengguna, sehingga meningkatkan keamanan dan fleksibilitas kode. Dengan memahami dan menggunakan ADT, programmer dapat menciptakan struktur data yang lebih efisien dan mudah digunakan dalam berbagai konteks pemrograman.