LAPORAN PRAKTIKUM
Modul 3
“Abstract Data Type (ADT)”

Disusun Oleh:

Muhammad Shafiq Rasuna - 2311104043

Kelas :

S1SE-07-02

Dosen :

Wahyu Andi Saputra, S.Pd, M.Eng

PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
PURWOKERTO
2024

1. Tujuan

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

2. Landasan Teori

2.1. Abstract Data Type

Abstract Data Type (ADT) adalah model matematika yang mendefinisikan tipe data dengan mengaitkannya pada operasi-operasi yang dapat dilakukan terhadap data tersebut. Konsep ini berfungsi untuk memisahkan struktur penyimpanan dari perilaku tipe data, sehingga pemrogram tidak perlu mengetahui rincian implementasi internal dari tipe data tersebut. ADT memiliki beberapa tujuan dan manfaat, seperti modularitas, yang memungkinkan pengembangan sistem secara terpisah tanpa saling mengganggu, serta pengkapsulan, yang menyembunyikan informasi internal dari pengguna. Dengan demikian, perubahan pada implementasi tidak akan mempengaruhi program yang menggunakan ADT tersebut. Selain itu, ADT juga memberikan abstraksi, sehingga pemrogram dapat fokus pada perilaku objek tanpa harus memperhatikan detail implementasi.

Dalam struktur ADT terdapat dua komponen utama: definisi tipe, yang menyediakan spesifikasi tentang bagaimana tipe data diorganisir, dan operasi dasar, yaitu kumpulan fungsi atau prosedur yang dapat digunakan untuk memanipulasi data dalam tipe tersebut. ADT dapat dibedakan menjadi beberapa jenis berdasarkan sifat datanya, seperti ADT homogen yang hanya menampung variabel dengan tipe data yang sama, dan ADT heterogen yang dapat menampung variabel dengan berbagai tipe data. Sebagai contoh implementasi, dalam bahasa C terdapat struktur Jam yang mendefinisikan waktu dan dilengkapi dengan fungsi MakeJam serta TulisJam untuk membuat dan menampilkan waktu. Secara keseluruhan, ADT merupakan konsep fundamental dalam pemrograman yang membantu dalam pengembangan perangkat lunak yang lebih terstruktur dan mudah dikelola, memungkinkan pemrogram menciptakan solusi yang lebih efisien dan modular.

3. Guided

3.1. ADT

Kode program ini digunakan untuk menghitung rata – rata nilai dua mata pelajaran dari seorang mahasiswa. Program ini terdiri dari struct mahasiswa yang memiliki atribut NIM, nilai 1, dan nilai 2. Fungsi inputMahasiswa adalah untuk menginputkan data mahasiswa, sedangkan fungsi rata2 menghitung rata – rata nilai. Pada fungsi main, program memanggil kedua fungsi tersebut dan menampilkan hasil rata – rata untuk nilai mahasiswa.

Kode program :

```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 nilai = ";
    cin>>(m).nilai2;
}
float rata2(mahasiswa m){
    return(m.nilai1+m.nilai2)/2;
}
```

Output program :

```
Input nim = 2311104043
Input nilai = 69
Input nilai = 87
Rata - rata = 78
```

4. Unguided

4.1. Kode program :

v

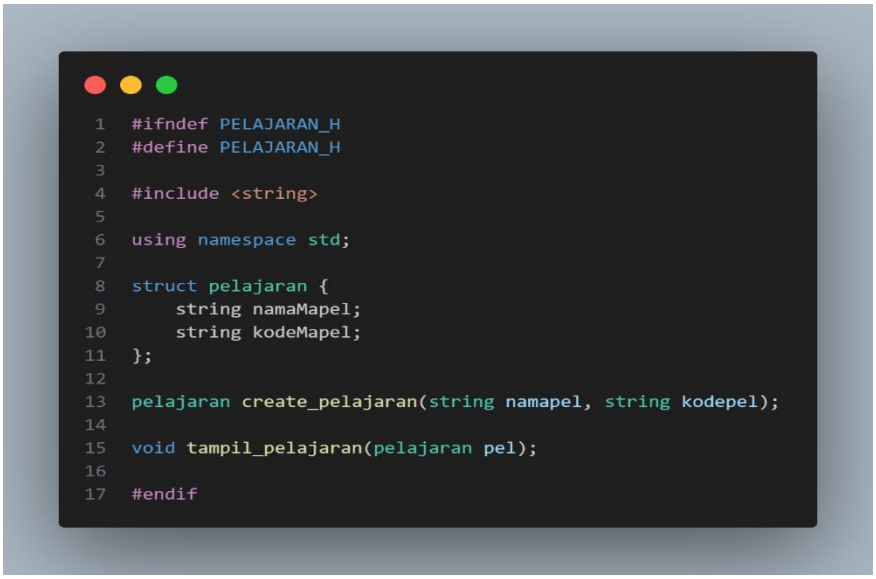
```cpp
#include <iostream>
#include <string>

using namespace std;

const int MAX_MAHASISWA = 10;

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

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

int main() {
    Mahasiswa mahasiswa[MAX_MAHASISWA];
    int jumlahMahasiswa;

    cout << "Masukkan jumlah mahasiswa (maksimal " << MAX_MAHASISWA << "): ";
    cin >> jumlahMahasiswa;

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

        // Hitung nilai akhir
        mahasiswa[i].nilaiAkhir = hitungNilaiAkhir(mahasiswa[i].uts, mahasiswa[i].uas, mahasiswa[i].tugas);
    }

    cout << "\nData Mahasiswa:" << endl;
    for (int i = 0; i < jumlahMahasiswa; i++) {
        cout << "Nama: " << mahasiswa[i].nama << endl;
        cout << "NIM: " << mahasiswa[i].nim << endl;
        cout << "Nilai Akhir: " << mahasiswa[i].nilaiAkhir << endl << endl;
    }

    return 0;
}
```

Output dari kode program :

```
Masukkan jumlah mahasiswa (maksimal 10): 3

Data Mahasiswa ke-1
Nama: memet
NIM: 123456789
Nilai UTS: 75
Nilai UAS: 85
Nilai Tugas: 95

Data Mahasiswa ke-2
Nama: ujang
NIM: 987654321
Nilai UTS: 90
Nilai UAS: 80
Nilai Tugas: 70

Data Mahasiswa ke-3
Nama: sachi
NIM: 2311104043
Nilai UTS: 99
Nilai UAS: 98
Nilai Tugas: 97

Data Mahasiswa:
Nama: memet
NIM: 123456789
Nilai Akhir: 85

Nama: ujang
NIM: 987654321
Nilai Akhir: 80

Nama: sachi
NIM: 2311104043
Nilai Akhir: 98
```

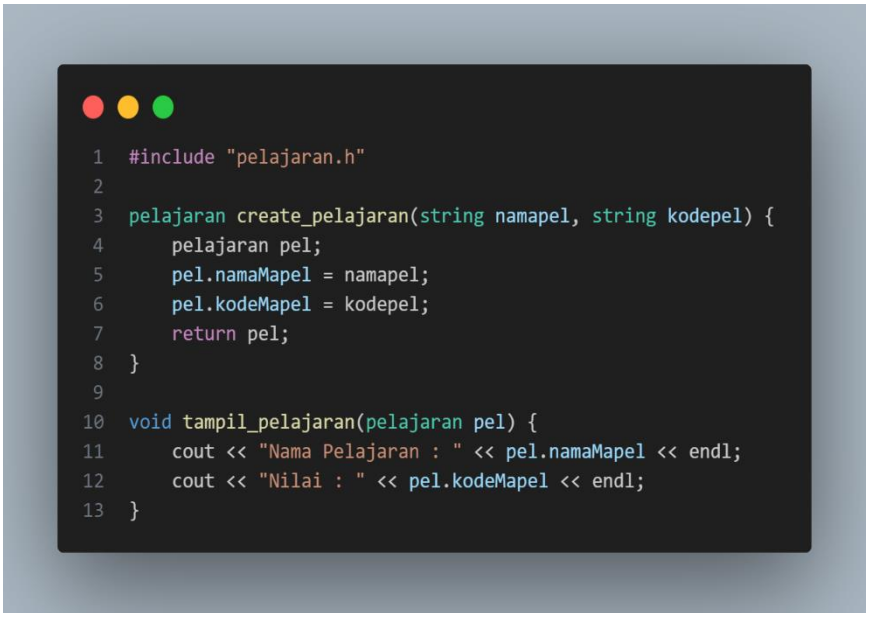
4.2.Kode program :

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
```

pelajaran.cpp



```
#include "pelajaran.h"

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

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

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;
}
```

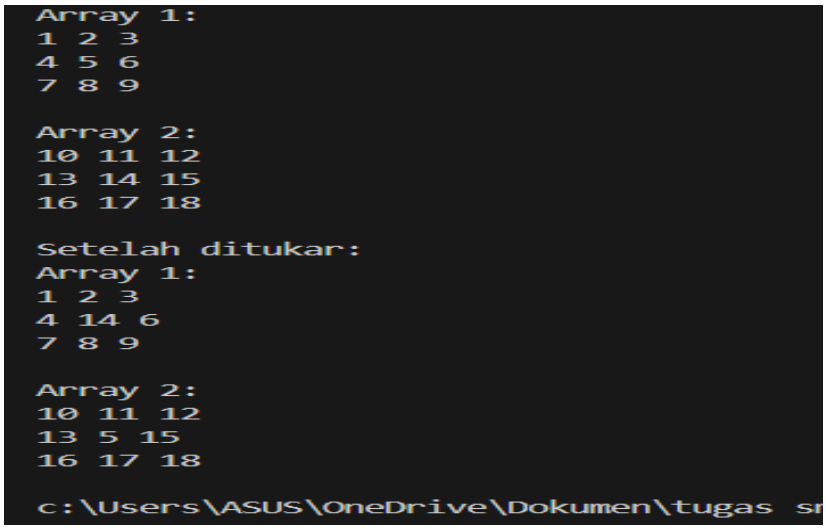
Output dari kode program main.cpp :

```
Nama Pelajaran : Struktur Data
Nilai : STD
```

4.3. Kode program :

```cpp
#include <iostream>

using namespace std;

void cetakArray(int arr[][3], int baris, int kolom) {
    for (int i = 0; i < baris; i++) {
        for (int j = 0; j < kolom; j++) {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }
}

void tukarElemen(int *ptr1, int *ptr2) {
    int temp = *ptr1;
    *ptr1 = *ptr2;
    *ptr2 = temp;
}

int main() {
    int arr1[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
    int arr2[3][3] = {{10, 11, 12}, {13, 14, 15}, {16, 17, 18}};

    cout << "Array 1:\n";
    cetakArray(arr1, 3, 3);

    cout << "\nArray 2:\n";
    cetakArray(arr2, 3, 3);

    // Tukar elemen pada posisi (1,1) dari kedua array
    int *ptr1 = &arr1[1][1]; // Menunjuk ke elemen pada baris 1, kolom 1 di arr1
    int *ptr2 = &arr2[1][1]; // Menunjuk ke elemen pada baris 1, kolom 1 di arr2
    tukarElemen(ptr1, ptr2);

    cout << "\nSetelah ditukar:\n";
    cout << "Array 1:\n";
    cetakArray(arr1, 3, 3);

    cout << "\nArray 2:\n";
    cetakArray(arr2, 3, 3);

    return 0;
}
```

Output dari kode program :



5. Kesimpulan

Dari laporan di atas, dapat disimpulkan bahwa Abstract Data Type (ADT) adalah konsep penting dalam pemrograman yang membantu pemrogram untuk mendefinisikan tipe data berdasarkan operasi yang dapat dilakukan terhadapnya, tanpa perlu memikirkan detail implementasi. ADT memungkinkan modularitas dan pengkapsulan, sehingga memudahkan pengembangan perangkat lunak secara terstruktur dan efisien.

Melalui contoh program yang menghitung rata-rata nilai dua mata pelajaran, kita dapat melihat penerapan ADT dalam struktur data seperti struct dalam bahasa C. Program tersebut menunjukkan bagaimana data dapat diorganisir dan dioperasikan menggunakan fungsi-fungsi yang terpisah, yang mendemonstrasikan prinsip-prinsip dasar dari ADT. Dengan menggunakan ADT, pemrogram dapat fokus pada logika program tanpa terganggu oleh rincian teknis dari implementasi, yang mengarah pada pengembangan perangkat lunak yang lebih baik dan lebih mudah dikelola.