LAPORAN PRAKTIKUM

Modul 03

“ABSTRACT DATA TYPE (ADT)”

Disusun Oleh:

Marvel Sanjaya Setiawan (2311104053)

SE-07-02

Dosen :

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

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY

PURWOKERTO

2024

1. **Tujuan**
    - Memahami konsep dasar ADT.
    - Menerapkan ADT dalam program C++.
2. **Landasan Teori**

    Absract Data Type (ADT)

    ADT (Abstract Data Type) adalah sebuah konsep dalam pemrograman yang mendefinisikan tipe data baru beserta operasinya. Bayangkan seperti membuat cetakan kue: ADT adalah cetakannya, sedangkan nilai-nilai yang dihasilkan dari cetakan tersebut adalah kue-kue yang berbeda.

    Komponen Utama ADT:

    1. Tipe Data: Jenis data yang ingin didefinisikan (misal: waktu, titik, segiempat).
    2. Operasi (Primitif): Fungsi-fungsi yang bisa dilakukan pada tipe data tersebut (misal: membuat waktu baru, menghitung jarak antara dua titik).
    3. Invarian: Aturan yang harus selalu dipenuhi oleh tipe data tersebut.
    4. Aksioma: Sifat-sifat matematis dari operasi-operasi yang ada.

    Jenis-jenis Operasi pada ADT:

    1. Konstruktor: Membuat objek baru dari tipe data tersebut.
    2. Selector: Mengambil nilai komponen dari objek.
    3. Prosedur: Mengubah nilai komponen dari objek.
    4. Validator: Memeriksa apakah nilai yang diberikan valid untuk tipe data tersebut.
    5. Destruktor: Menghapus objek dari memori.
    6. Operator: Operasi-operasi seperti perbandingan, aritmatika, konversi.

    Tujuan ADT:

    1. Abstraksi: Memisahkan antara interface (cara penggunaan) dengan implementasi (cara kerja internal).
    2. Modularitas: Membagi program menjadi bagian-bagian yang lebih kecil dan mudah dikelola.
    3. Reusabilitas: Kode ADT dapat digunakan kembali dalam berbagai bagian program.

3. **Guided**

```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;
}
```

```
Input nim = 2311104053
Input nilai = 95
Input nilai = 95
rata-rata = 95
```

Alur Program:

1. Deklarasi Struktur: Dibuat struktur mahasiswa untuk menampung data mahasiswa.
2. Input Data: Fungsi inputMhs meminta pengguna memasukkan data mahasiswa dan menyimpannya ke dalam variabel mhs.
3. Perhitungan Rata-rata: Fungsi rata2 menghitung rata-rata nilai dari data mahasiswa.
4. Output: Hasil perhitungan rata-rata ditampilkan ke layar.menggunakan float.

Penjelasan Tiap Bagian:

1. struct mahasiswa: Membuat cetak biru untuk data mahasiswa.
2. inputMhs: Fungsi untuk mengisi data mahasiswa.
3. rata2: Fungsi untuk menghitung rata-rata.
4. main: Fungsi utama yang menjalankan program.

Telkom University Purwokerto

4. Unguided

1. Program Data Mahasiswa

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

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

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

void tampilkanHeaderTabel() {
    cout << left << setw(15) << "Nama"
         << setw(10) << "NIM"
         << setw(6) << "UTS"
         << setw(6) << "UAS"
         << setw(6) << "Tugas"
         << setw(10) << "Nilai Akhir" << endl;
    cout << string(55, '-') << endl;
}

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

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

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

    for (int i = 0; i < n; i++) {
        cout << "\nMahasiswa ke-" << i + 1 << ":\n";
        cout << "Nama: ";
        cin >> mhs[i].nama;
        cout << "NIM: ";
        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 << "\nData Mahasiswa:\n";
    tampilkanHeaderTabel();
    for (int i = 0; i < n; i++) {
        cout << left << setw(15) << mhs[i].nama
             << setw(10) << mhs[i].nim
             << setw(6) << mhs[i].uts
             << setw(6) << mhs[i].uas
             << setw(6) << mhs[i].tugas
             << setw(10) << mhs[i].nilaiAkhir << endl;
    }

    return 0;
}
```

Telkom University Purwokerto

```
Mahasiswa ke-1:
Nama: Marvel
NIM: 2311104053
UTS: 90
UAS: 95
Tugas: 100

Mahasiswa ke-2:
Nama: Annafi
NIM: 2311104052
UTS: 85
UAS: 90
Tugas: 95

Mahasiswa ke-3:
Nama: Felix
NIM: 2311104050
UTS: 95
UAS: 95
Tugas: 100

Mahasiswa ke-4:
Nama: Dinand
NIM: 2311104051
UTS: 95
UAS: 95
Tugas: 95

Mahasiswa ke-5:
Nama: Richard
NIM: 2311104054
UTS: 90
UAS: 95
Tugas: 95

Mahasiswa ke-6:
Nama: Wahyu
NIM: 2311104055
UTS: 85
UAS: 85
Tugas: 90
```

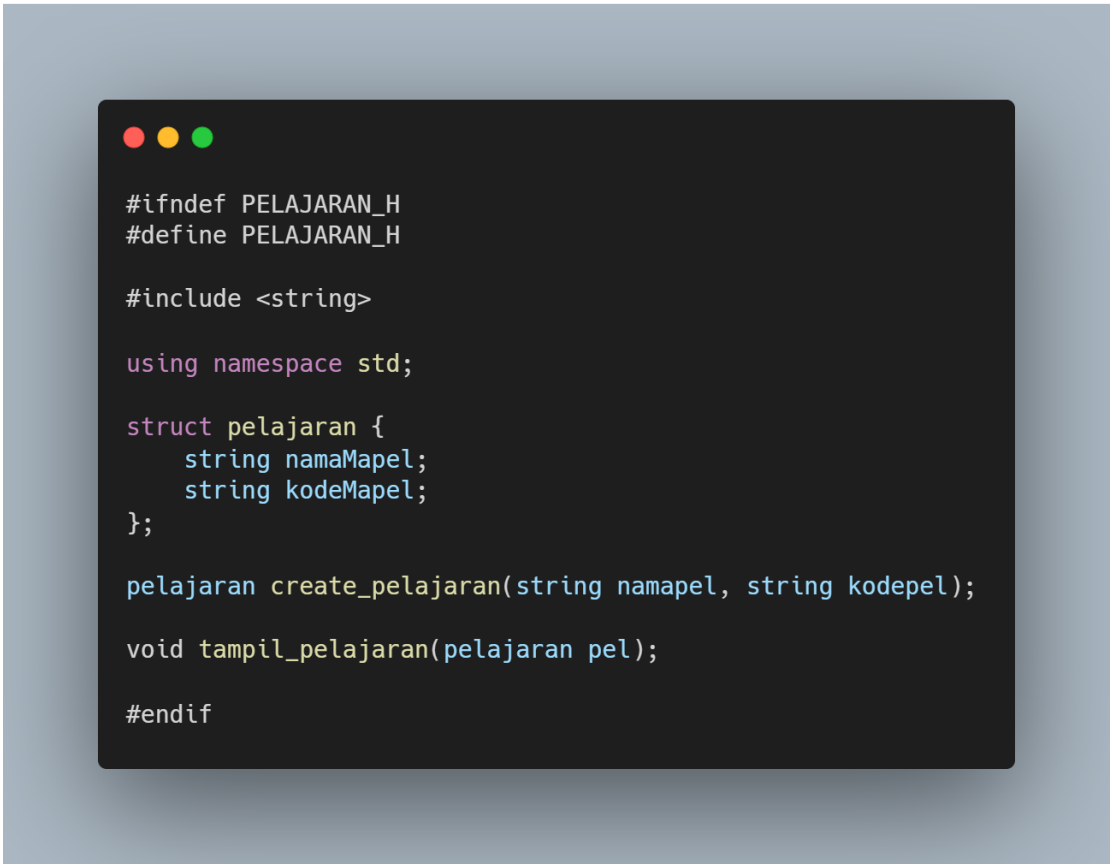
```
Mahasiswa ke-7:
Nama: Husen
NIM: 2311104056
UTS: 85
UAS: 90
Tugas: 95

Mahasiswa ke-8:
Nama: Sapta
NIM: 2311104057
UTS: 90
UAS: 85
Tugas: 95

Mahasiswa ke-9:
Nama: Kresna
NIM: 2311104058
UTS: 95
UAS: 95
Tugas: 90

Mahasiswa ke-10:
Nama: Egi
NIM: 2311104059
UTS: 90
UAS: 95
Tugas: 85
```

```
Data Mahasiswa:
Nama            NIM         UTS   UAS   Tugas Nilai Akhir
-------------------------------------------------------
Marvel          231110405390    95    100   95
Annafi          231110405285    90    95    90
Felix           231110405095    95    100   96.5
Dinand          231110405195    95    95    95
Richard         231110405490    95    95    93.5
Wahyu           231110405585    85    90    86.5
Husen           231110405685    90    95    90
Sapta           231110405790    85    95    89.5
Kresna          231110405895    95    90    93.5
Egi             231110405990    95    85    90.5
```

Penjelasan:

1. Deklarasi Struktur:
    - Dibuat struktur Mahasiswa untuk menyimpan data setiap mahasiswa.
    - Struktur ini memiliki anggota: nama, nim, uts, uas, tugas, dan nilaiAkhir.
2. Fungsi hitungNilaiAkhir:
    - Fungsi ini menghitung nilai akhir mahasiswa berdasarkan bobot UTS, UAS, dan tugas.
3. Fungsi tampilkanHeaderTabel:
    - Fungsi ini mencetak header tabel untuk menampilkan data mahasiswa secara terstruktur.
4. Fungsi main:
    - Deklarasi Array: Membuat array mhs berukuran 10 untuk menyimpan data maksimal 10 mahasiswa.
    - Input Data: Meminta pengguna memasukkan jumlah mahasiswa dan data masing-masing mahasiswa (nama, NIM, nilai).
    - Perhitungan Nilai Akhir: Menghitung nilai akhir setiap mahasiswa menggunakan fungsi hitungNilaiAkhir.
    - Tampilan Data: Menampilkan data semua mahasiswa dalam bentuk tabel yang rapi menggunakan fungsi tampilkanHeaderTabel.

2. **Program ADT dengan bentuk code program yang dipisah.**



```
#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
```

File “pelajaran.h”

File pelajaran.h ini berfungsi sebagai header file untuk mendefinisikan sebuah struktur data yang merepresentasikan sebuah pelajaran.

Struktur data pelajaran memiliki dua anggota:

a. namaMapel: Menyimpan nama mata pelajaran (misalnya, "Matematika", "Bahasa Indonesia").
b. kodeMapel: Menyimpan kode unik untuk mata pelajaran (misalnya, "MTK", "BIN").

Selain itu, file ini juga mendeklarasikan dua fungsi:

a. create_pelajaran: Fungsi ini digunakan untuk membuat objek pelajaran baru dengan memberikan nama dan kode mata pelajaran.
b. tampil_pelajaran: Fungsi ini digunakan untuk menampilkan informasi mengenai suatu objek pelajaran, seperti nama dan kode mata pelajaran.

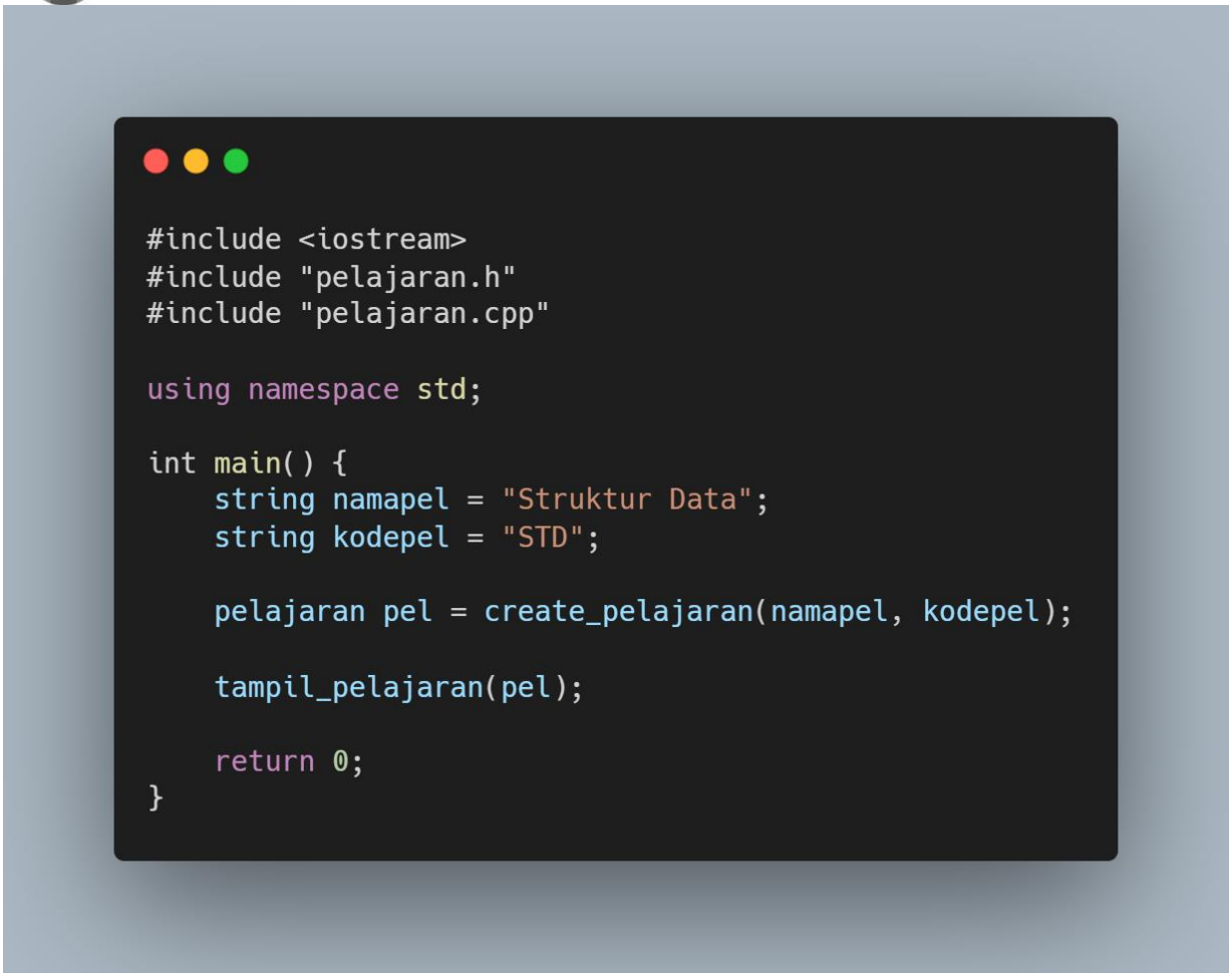
```cpp
#include "pelajaran.h"

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;
}
```

File “pelajaran.cpp”

File pelajaran.cpp ini berisi implementasi dari fungsi-fungsi yang dideklarasikan di pelajaran.h.

Fungsi-fungsi tersebut adalah:

a. create_pelajaran: Fungsi ini berfungsi untuk membuat objek pelajaran baru. Ketika dipanggil, fungsi ini akan membuat sebuah objek pelajaran baru dengan nilai namaMapel dan kodeMapel yang diberikan sebagai parameter.
b. tampil_pelajaran: Fungsi ini berfungsi untuk menampilkan informasi dari sebuah objek pelajaran ke layar. Informasi yang ditampilkan adalah nama mata pelajaran dan kode mata pelajaran.



```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;
}
```

File “main.cpp”

File main.cpp adalah titik awal eksekusi program.

Di sini, program melakukan hal-hal berikut:

a. Menginisialisasi data: Membuat dua variabel string, namapel dan kodepel, untuk menyimpan nama dan kode mata pelajaran.
b. Membuat objek pelajaran: Memanggil fungsi create_pelajaran untuk membuat objek pelajaran baru dengan nilai namapel dan kodepel yang telah diinisialisasi.
c. Menampilkan data: Memanggil fungsi tampil_pelajaran untuk menampilkan informasi dari objek pelajaran yang baru dibuat.

Output :

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

3. **Program array 2D integer berukuran 3x3 dan 2 buah pointer integer.**

```cpp
#include <iostream>

using namespace std;

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

void tukarElemenArray(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;
}

void tukarNilaiPointer(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 sebelum ditukar:\n";
    tampilkanArray(arr1);

    cout << "\nArray 2 sebelum ditukar:\n";
    tampilkanArray(arr2);

    tukarElemenArray(arr1, arr2, 1, 1);

    cout << "\nArray 1 setelah ditukar:\n";
    tampilkanArray(arr1);

    cout << "\nArray 2 setelah ditukar:\n";
    tampilkanArray(arr2);

    int *ptr1 = &arr1[0][0];
    int *ptr2 = &arr2[0][0];

    tukarNilaiPointer(ptr1, ptr2);

    cout << "\nNilai yang ditunjuk oleh ptr1 setelah ditukar: " << *ptr1 << endl;
    cout << "Nilai yang ditunjuk oleh ptr2 setelah ditukar: " << *ptr2 << endl;

    return 0;
}
```

```
Array 1 sebelum ditukar:
1 2 3
4 5 6
7 8 9

Array 2 sebelum ditukar:
10 11 12
13 14 15
16 17 18

Array 1 setelah ditukar:
1 2 3
4 14 6
7 8 9

Array 2 setelah ditukar:
10 11 12
13 5 15
16 17 18

Nilai yang ditunjuk oleh ptr1 setelah ditukar: 10
Nilai yang ditunjuk oleh ptr2 setelah ditukar: 1
```

Penjelasan:

1. Mendefinisikan fungsi: Program mendefinisikan beberapa fungsi untuk menampilkan array 2D, menukar elemen dalam dua array 2D, dan menukar nilai yang ditunjuk oleh dua pointer.
2. Membuat array: Di dalam fungsi main, program membuat dua array 2D berukuran 3x3.
3. Menampilkan array: Program memanggil fungsi tampilkanArray untuk menampilkan isi dari kedua array sebelum ditukar.
4. Menukar elemen: Program memanggil fungsi tukarElemenArray untuk menukar elemen pada posisi tertentu dari kedua array.
5. Menampilkan array setelah ditukar: Program kembali memanggil tampilkanArray untuk menunjukkan perubahan setelah penukaran.

6. Menggunakan pointer: Program menggunakan pointer untuk menunjuk ke elemen pertama dari masing-masing array dan kemudian menukar nilai yang ditunjuk oleh kedua pointer menggunakan fungsi tukarNilaiPointer.
7. Menampilkan nilai pointer: Program menampilkan nilai yang ditunjuk oleh kedua pointer setelah penukaran.

5. Kesimpulan

Praktikum ini memberikan pemahaman dasar tentang ADT dan cara mengimplementasikannya dalam bahasa C++. Melalui contoh-contoh yang diberikan, agar pembaca dapat memahami konsep ADT dan penerapannya dalam berbagai situasi pemrograman.