LAPORAN PRAKTIKUM

MODUL 3

ABSTRACT DATA TYPE (ADT0

Disusun Oleh:

Dhiemas Tulus Ikhsan 2311104046

SE-07-02

Dosen :

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

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY

PURWOKERTO

2024

I. TUJUAN

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

II. LANDASAN TEORI

1. *Abstract Data Type*

ADT (Abstract Data Type) adalah tipe data yang mendefinisikan jenis data dan sekumpulan operasi dasar yang dapat dilakukan terhadapnya. ADT mencakup definisi statis dari tipe data, termasuk invarian (aturan yang harus dipenuhi) dan aksioma yang berlaku. Sebuah ADT dapat terdiri dari ADT lainnya, misalnya ADT waktu yang terdiri dari ADT JAM dan DATE, atau ADT garis yang terdiri dari dua buah ADT POINT. Dalam implementasi, ADT diterjemahkan menjadi tipe data yang sesuai dengan bahasa pemrograman, seperti `struct` dalam bahasa C. Operasi dasar atau primitif dalam ADT mencakup beberapa jenis, seperti konstruktor untuk membentuk objek, selector untuk mengakses komponen, prosedur pengubah nilai komponen, validator untuk memeriksa kesesuaian nilai, destruktor untuk menghapus objek, serta fungsi baca/tulis untuk menghubungkan dengan perangkat input/output. Selain itu, ADT juga dapat memiliki operator relasional untuk perbandingan, operasi aritmatika, dan konversi tipe. Implementasi ADT biasanya melibatkan dua modul utama: satu modul untuk spesifikasi tipe dan operasi dasar (dalam file `.h`), dan modul lain untuk realisasi atau implementasi dari operasi tersebut (dalam file `.c`). Spesifikasi ini menggambarkan deskripsi tipe dan operasi, sementara realisasi adalah kode program yang memanfaatkan selector dan konstruktor.

III. 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 nama = ";
    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

```
Input nama = 2311104046
input nilai = 70
input nilai = 98
rata-rata = 84
Process returned 0 (0x0)   execution time : 10.653 s
Press any key to continue.
```

Program di atas menggunakan bahasa C++ untuk mengelola data seorang mahasiswa, dengan struktur data `mahasiswa` yang memiliki tiga anggota: `nim` (Nomor Induk Mahasiswa), `nilai1`, dan `nilai2`.

- Struct `mahasiswa' : Mendefinisikan tipe data yang menyimpan `nim` sebagai karakter array (`char nim[10]`) dan dua nilai integer (`nilai1` dan `nilai2`).
- Fungsi `inputMhs` : Menerima referensi dari objek `mahasiswa` sebagai parameter dan meminta input dari pengguna untuk `nim`, `nilai1`, dan `nilai2`.
- Fungsi `rata2`: Menghitung rata-rata dari `nilai1` dan `nilai2` dari objek `mahasiswa` yang diterima sebagai parameter, dan mengembalikannya dalam bentuk `float`.

Di dalam fungsi `main`, program membuat objek `mhs` dari tipe `mahasiswa`, lalu memanggil `inputMhs` untuk memasukkan data. Setelah itu, program menghitung dan menampilkan rata-rata nilai dengan memanggil fungsi `rata2` dan mencetak hasilnya ke layar.

IV. UNGUIDED

1. Task 1

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

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

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

// Fungsi untuk menginput data mahasiswa
void inputMahasiswa(Mahasiswa& m) {
    cout << "Nama: ";
    cin.ignore();
    getline(cin, m.nama);
    cout << "NIM: ";
    cin >> m.nim;
    cout << "Nilai UTS: ";
    cin >> m.uts;
    cout << "Nilai UAS: ";
    cin >> m.uas;
    cout << "Nilai Tugas: ";
    cin >> m.tugas;
    m.nilaiAkhir = hitungNilaiAkhir(m.uts, m.uas, m.tugas);
}
```

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

int main() {
    const int maxMahasiswa = 10;
    Mahasiswa daftarMahasiswa[maxMahasiswa];
    int jumlahMahasiswa;

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

    if (jumlahMahasiswa > maxMahasiswa || jumlahMahasiswa <= 0) {
        cout << "Jumlah mahasiswa harus antara 1 hingga 10." << endl;
        return 1;
    }

    // Input data mahasiswa
    for (int i = 0; i < jumlahMahasiswa; ++i) {
        cout << "\nInput data mahasiswa ke-" << (i + 1) << endl;
        inputMahasiswa(daftarMahasiswa[i]);
    }

    // Tampilkan data mahasiswa
    cout << "\nData Mahasiswa:\n";
    for (int i = 0; i < jumlahMahasiswa; ++i) {
        tampilkanMahasiswa(daftarMahasiswa[i]);
    }
```

Penjelasan :

- **Struct Mahasiswa**: Menyimpan informasi nama, nim, uts, uas, tugas, dan nilaiAkhir.
- **Fungsi hitungNilaiAkhir**: Menghitung nilai akhir berdasarkan rumus 0.3 * uts + 0.4 * uas + 0.3 * tugas.
- **Fungsi inputMahasiswa**: Meminta input pengguna untuk mengisi data mahasiswa, lalu menghitung nilaiAkhir.
- **Fungsi tampilkanMahasiswa**: Menampilkan data dari satu mahasiswa.
- **main Function**:
    - Meminta input jumlah mahasiswa (maksimal 10).
    - Memasukkan data mahasiswa menggunakan fungsi inputMahasiswa.
    - Menampilkan seluruh data mahasiswa menggunakan tampilkanMahasiswa.

Output :

```
Masukkan jumlah mahasiswa (max 10): 4

Input data mahasiswa ke-1
Nama: tulus
NIM: 2311104046
Nilai UTS: 87
Nilai UAS: 78
Nilai Tugas: 77

Input data mahasiswa ke-2
Nama: tiurma
NIM: 2311104042
Nilai UTS: 90
Nilai UAS: 89
Nilai Tugas: 70

Input data mahasiswa ke-3
Nama: ren
NIM: 2311104065
Nilai UTS: 79
Nilai UAS: 84
Nilai Tugas: 80

Input data mahasiswa ke-4
Nama: naur
NIM: 2311104078
Nilai UTS: 70
Nilai UAS: 80
Nilai Tugas: 85
```

```
Data Mahasiswa:
Nama: tulus
NIM: 2311104046
Nilai UTS: 87
Nilai UAS: 78
Nilai Tugas: 77
Nilai Akhir: 80.4
-----------------------------
Nama: tiurma
NIM: 2311104042
Nilai UTS: 90
Nilai UAS: 89
Nilai Tugas: 70
Nilai Akhir: 83.6
-----------------------------
Nama: ren
NIM: 2311104065
Nilai UTS: 79
Nilai UAS: 84
Nilai Tugas: 80
Nilai Akhir: 81.3
-----------------------------
Nama: naur
NIM: 2311104078
Nilai UTS: 70
Nilai UAS: 80
Nilai Tugas: 85
Nilai Akhir: 78.5
-----------------------------
```

2. Task 2

```h
#ifndef PELAJARAN_H
#define PELAJARAN_H

#include <string>
using namespace std;

struct pelajaran {
    string namaMapel;
    string kodeMapel;
};

// Fungsi untuk membuat pelajaran baru
pelajaran create_pelajaran(string namapel, string kodepel);

// Prosedur untuk menampilkan data pelajaran
void tampil_pelajaran(pelajaran pel);

#endif
```

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

// Implementasi fungsi untuk membuat pelajaran
pelajaran create_pelajaran(string namapel, string kodepel) {
    pelajaran pel;
    pel.namaMapel = namapel;
    pel.kodeMapel = kodepel;
    return pel;
}

// Implementasi prosedur untuk menampilkan data pelajaran
void tampil_pelajaran(pelajaran pel) {
    cout << "Nama Pelajaran: " << pel.namaMapel << endl;
    cout << "Kode Pelajaran: " << pel.kodeMapel << endl;
}
```

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

int main() {
    string namapel = "Struktur Data";
    string kodepel = "STD";
    pelajaran pel = create_pelajaran(namapel, kodepel);
    tampil_pelajaran(pel);
    return 0;
}
```

Penjelasan Program:

- pelajaran.h:
  - Mendefinisikan tipe data pelajaran yang terdiri dari namaMapel dan kodeMapel (keduanya bertipe string).
  - Deklarasi fungsi create_pelajaran untuk membuat objek pelajaran.
  - Deklarasi prosedur tampil_pelajaran untuk menampilkan data pelajaran.
- pelajaran.cpp:

- Mengimplementasikan fungsi create_pelajaran, yang menerima nama pelajaran dan kode pelajaran sebagai parameter, lalu mengembalikan objek pelajaran yang berisi data tersebut.
- Mengimplementasikan prosedur tampil_pelajaran untuk menampilkan nama dan kode pelajaran ke layar.

- main.cpp:
  - Membuat objek pelajaran dengan nama "Struktur Data" dan kode "STD" menggunakan fungsi create_pelajaran.
  - Menampilkan informasi pelajaran tersebut dengan memanggil prosedur tampil_pelajaran.

Output Program :

```
Nama Pelajaran: Struktur Data
Kode Pelajaran: STD
```

3. Task 3

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

// Fungsi untuk menampilkan isi array 2D
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;
    }
}

// Fungsi untuk menukar elemen pada posisi tertentu antara dua array 2D
void tukarElemenArray(int arr1[3][3], int arr2[3][3], int row, int col) {
    int temp = arr1[row][col];
    arr1[row][col] = arr2[row][col];
    arr2[row][col] = temp;
}

// Fungsi untuk menukar isi dari variabel yang ditunjuk oleh dua pointer
void tukarIsiPointer(int* ptr1, int* ptr2) {
    int temp = *ptr1;
    *ptr1 = *ptr2;
    *ptr2 = temp;
}

int main() {
    // Inisialisasi dua array 2D 3x3
    int array1[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
    int array2[3][3] = {{9, 8, 7}, {6, 5, 4}, {3, 2, 1}};

    // Inisialisasi dua pointer integer
    int a = 10;
    int b = 20;
    int* ptr1 = &a;
    int* ptr2 = &b;

    cout << "Isi Array 1 sebelum ditukar:" << endl;
    tampilkanArray(array1);
    cout << "Isi Array 2 sebelum ditukar:" << endl;
    tampilkanArray(array2);

    // Menukar elemen pada posisi (1, 1) (baris 1, kolom 1)
    tukarElemenArray(array1, array2, 1, 1);
```

```
cout << "\nIsi Array 1 setelah menukar elemen di posisi (1,1):" << endl;
tampilkanArray(array1);
cout << "Isi Array 2 setelah menukar elemen di posisi (1,1):" << endl;
tampilkanArray(array2);

// Menampilkan nilai sebelum penukaran pointer
cout << "\nNilai sebelum penukaran pointer:" << endl;
cout << "a = " << a << ", b = " << b << endl;

// Menukar isi dari variabel yang ditunjuk oleh ptr1 dan ptr2
tukarIsiPointer(ptr1, ptr2);

// Menampilkan nilai setelah penukaran pointer
cout << "\nNilai setelah penukaran pointer:" << endl;
cout << "a = " << a << ", b = " << b << endl;

return 0;
```

Penjelasan Program:

1. Array 2D: array1 dan array2 masing-masing berukuran 3x3, diinisialisasi dengan nilai-nilai berbeda.
2. Pointer: ptr1 menunjuk ke a (bernilai 10), dan ptr2 menunjuk ke b (bernilai 20).
3. Fungsi tampilkanArray: Mencetak isi dari array 2D yang diberikan.
4. Fungsi tukarElemenArray: Menukar elemen di posisi tertentu (row, col) antara array1 dan array2.
5. Fungsi tukarIsiPointer: Menukar nilai dari dua variabel yang ditunjuk oleh ptr1 dan ptr2.
6. main Function:
   - Menampilkan isi array1 dan array2 sebelum dan sesudah penukaran elemen pada posisi (1, 1).
   - Menampilkan nilai a dan b sebelum dan sesudah penukaran menggunakan pointer.

V. KESIMPULAN

Pada praktikum ini, kami mengimplementasikan beberapa konsep ADT (Abstract Data Type) dalam program C++:

1. Penggunaan Struktur Data sebagai ADT:

   Kami menggunakan struktur Mahasiswa untuk menyimpan data (nama, NIM, nilai UTS, UAS, dan tugas), serta menghitung nilai akhir dengan fungsi. Ini menunjukkan bagaimana ADT dapat mengelola data secara terstruktur.
2. Manipulasi Array 2D:

   Kami menggunakan dua array 2D untuk menyimpan data dalam bentuk matriks dan menukarkan elemen di posisi tertentu. Ini menggambarkan bagaimana ADT bekerja dengan data yang lebih kompleks seperti matriks.
3. Operasi dengan Pointer:

   Dengan dua pointer integer, kami menukar nilai dari dua variabel. Ini menekankan fleksibilitas pointer dalam manipulasi data dinamis, penting dalam pengelolaan memori. Secara keseluruhan, praktikum ini mengajarkan pentingnya ADT untuk membuat program lebih modular dan mudah dikelola, dengan memisahkan data dan operasi yang dapat dilakukan terhadap data tersebut.