LAPORAN PRAKTIKUM MODUL 3
ABSTRACT DATA TYPE
(ADT)

Disusun Oleh:
Ade Fatkhul Anam 2211104051

Asisten Praktikum :
Aldi Putra
Andini Nur Hidayah

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

PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
PURWOKERTO
2024

MODUL 3
ABSTRAK DATA TYPE

1. Tujuan

a. Mahasiswa diharapkan memahami konsep dan definisi dasar Abstract Data Type (ADT) dalam pemrograman, termasuk pemisahan antara cara data direpresentasikan dan cara operasinya dijalankan.

b. Mahasiswa mampu mengimplementasikan ADT dalam program dengan menggunakan bahasa pemrograman, serta memisahkan deklarasi dan implementasinya melalui file header dan file implementasi.

c. Mahasiswa dapat membangun program yang modular dengan memanfaatkan ADT, sehingga struktur program menjadi lebih rapi, mudah dikelola, dan dapat dikembangkan lebih lanjut.

d. Mahasiswa mampu meningkatkan tingkat abstraksi dalam program dengan memisahkan penggunaan data dari detail teknis implementasi struktur data.

e. Mahasiswa memahami pentingnya prinsip enkapsulasi dalam ADT, yang membatasi akses langsung ke data dan memungkinkan interaksi hanya melalui fungsi atau prosedur yang telah ditentukan.

2. Landasan Teori

a. Abstract Data Type

Abstract Data Type (ADT) dalam C++ adalah konsep yang memungkinkan pengembang untuk mendefinisikan struktur data dan operasinya secara terpisah dari detail implementasinya. Dengan ADT, programmer dapat merancang tipe data yang kompleks, seperti stack, queue, atau linked list, yang memungkinkan pengolahan data secara efisien dan terorganisir. ADT menyediakan antarmuka yang jelas melalui fungsi atau metode, sambil menyembunyikan detail implementasi, mendukung prinsip enkapsulasi dalam pemrograman berorientasi objek. Konsep ini meningkatkan modularitas dan mempermudah pemeliharaan kode, karena perubahan pada implementasi tidak akan berdampak pada bagian lain dari program yang menggunakan ADT tersebut.

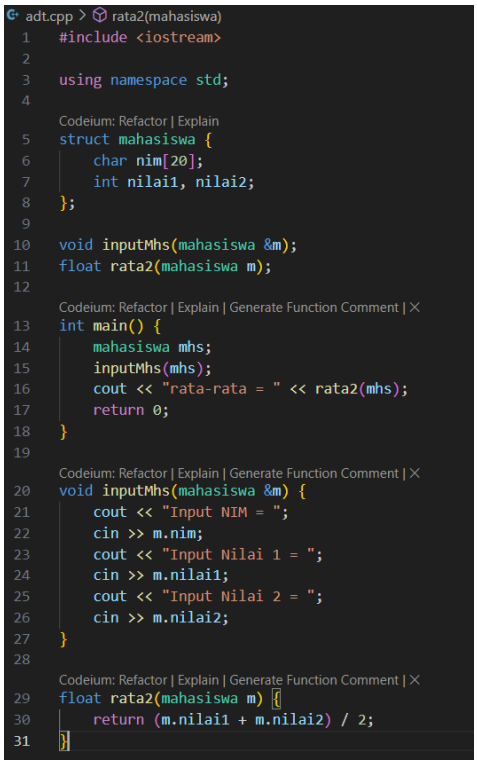
3. Guided

a) Abstract Data Type

Abstract Data Type (ADT) adalah konsep pemrograman yang memisahkan definisi tipe data dari implementasinya. ADT menyediakan antarmuka yang jelas untuk operasi-operasi pada struktur data tanpa mengungkapkan detail pengelolaannya. Contoh ADT meliputi stack, queue, dan list. Konsep ini mendukung prinsip pemrograman berorientasi objek seperti enkapsulasi dan modularitas, yang mempermudah pengembangan dan pemeliharaan kode.

Source Code:

```
#include <iostream>

using namespace std;

struct mahasiswa {
    char nim[20];
    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
   1 = ";
    cin >> m.nilai1;
    cout << "Input Nilai
   2 = ";
    cin >> m.nilai2;
```



```
    }


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

Output:

```
PS D:\PRAKTIKUM DATA STRUCTURE\LAPRAK\03_Abstrak_Data_Type> cd 'd:\PRAKTIKUM DATA STRUCTURE
PS D:\PRAKTIKUM DATA STRUCTURE\LAPRAK\03_Abstrak_Data_Type\output> & .\'adt.exe'
Input NIM = 2211104051
Input Nilai 1 = 90
Input Nilai 2 = 98
```

4. Unguided

a. 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.

Source code:

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

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

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

int main() {
    const int MAKS_MAHASISWA = 10;
    Mahasiswa mahasiswa[MAKS_MAHASISWA];
    int jumlahMahasiswa;

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

    if (jumlahMahasiswa > MAKS_MAHASISWA) {
        cout << "Jumlah mahasiswa melebihi batas maksimal (10). Program dihentikan." << endl;
        return 1;
    }

    for (int i = 0; i < jumlahMahasiswa; i++) {
        cout << "\nData mahasiswa ke-" << (i + 1) << endl;
        cout << "Nama      : ";
        cin.ignore();
        getline(cin, mahasiswa[i].nama);
        cout << "NIM       : ";
        getline(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;

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

    cout << "\nData Mahasiswa: \n";
    cout << "=====================================================================\n";
    cout << left << setw(5) << "No" << setw(20) << "Nama" << setw(15) << "NIM" << setw(10) << "UTS" <<
setw(10) << "UAS" << setw(10) << "Tugas" << setw(10) << "Nilai Akhir" << endl;
    cout << "=====================================================================\n";
    for (int i = 0; i < jumlahMahasiswa; i++) {
        cout << left << setw(5) << (i + 1)
             << setw(20) << mahasiswa[i].nama
             << setw(15) << mahasiswa[i].nim
             << setw(10) << mahasiswa[i].uts
             << setw(10) << mahasiswa[i].uas
             << setw(10) << mahasiswa[i].tugas
             << setw(10) << fixed << setprecision(2) << mahasiswa[i].nilaiAkhir << endl;
    }

    return 0;
}
```

Output:

```
Nilai UAS : 89
Nilai Tugas: 70

Data Mahasiswa:
=====================================================================
No   Nama                NIM          UTS       UAS       Tugas     Nilai Akhir
=====================================================================
1    Ade                 221110451    90        98        100       96.20
2    Dimas               221110460    90.00     89.00     90.00     89.60
3    Ariq                221110431    90.00     80.00     98.00     88.40
4    Rio                 22111030     90.00     80.00     90.00     86.00
5    Rafli               2211104050   9.00      90.00     98.00     68.10
6    Salman              2211104040   90.00     89.00     90.00     89.60
7    Cahyo               2211104044   90.00     89.00     90.00     89.60
8    Aji                 2211104049   90.00     89.00     70.00     83.60
9    Yogi                2211104045   70.00     80.00     90.00     80.00
10   Candra              2211104046   90.00     89.00     70.00     83.60
```

b. Buatlah ADT pelajaran sebagai berikut di dalam file “pelajaran.h”: Buatlah implementasi ADT pelajaran pada file “pelajaran.cpp” Cobalah hasil implementasi ADT pada file “main.cpp”!

Source code:

Main.cpp

```cpp
#include <iostream>
#include "soal-02-pelajaran.h"
#include "soal-02-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;
}
```

Soal-02-pelajaran.cpp

```cpp
#include "soal-02-pelajaran.h"
#include <iostream>
using namespace std;

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;
}
```

Soal-02-pelajaran.h

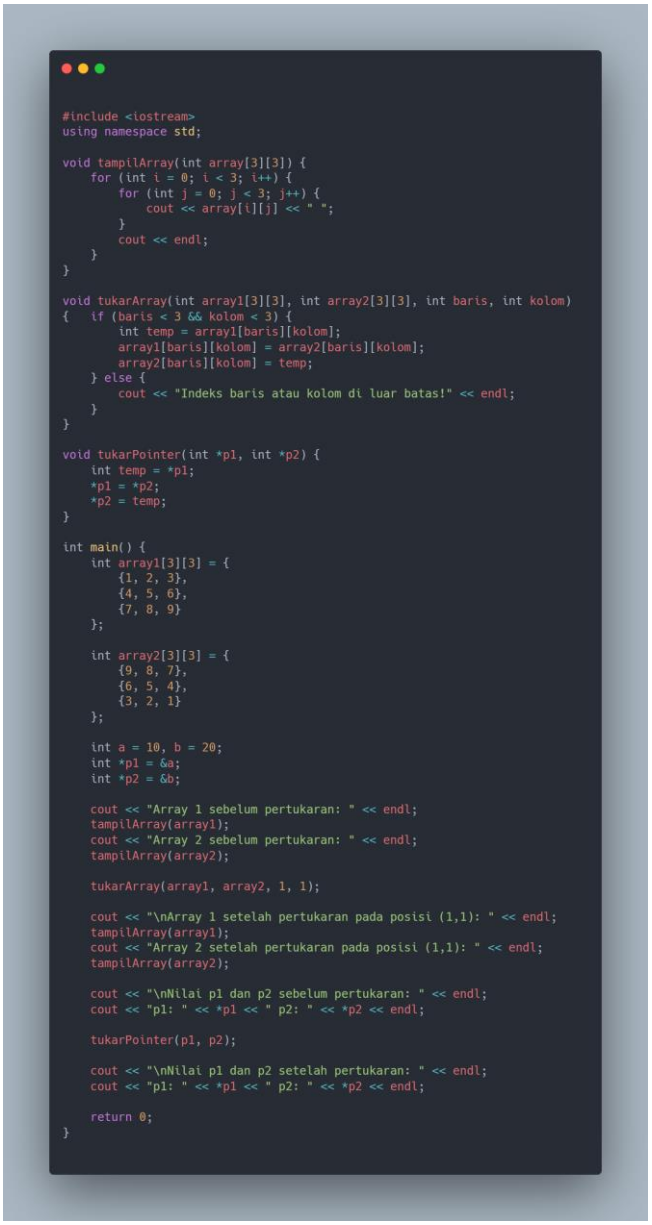
```h
#ifndef PELAJARAN_H
#define PELAJARAN_H

#include <string>
using namespace std;

struct pelajaran {
    string namaMapel;
    string kodeMapel;
};

pelajaran create_pelajaran(string namapel, string kodepel);

void tampil_pelajaran(pelajaran pel);

#endif
```

Output:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  COMMENTS

PS D:\PRAKTIKUM DATA STRUCTURE\LAPRAK\03_Abstrak_Data_Type\soal 2\output> cd 'd:\PRAKTIKUM
PS D:\PRAKTIKUM DATA STRUCTURE\LAPRAK\03_Abstrak_Data_Type\soal 2\output> & .\'main.exe'
nama pelajaran : Struktur Data
nilai : STD
PS D:\PRAKTIKUM DATA STRUCTURE\LAPRAK\03_Abstrak_Data_Type\soal 2\output>
```

3. Buatlah program dengan ketentuan :
   - 2 buah array 2D integer berukuran 3x3 dan 2 buah pointer integer
   - fungsi/prosedur yang menampilkan isi sebuah array integer 2D
   - fungsi/prosedur yang akan menukarkan isi dari 2 array integer 2D pada posisi tertentu
   - fungsi/prosedur yang akan menukarkan isi dari variabel yang ditunjuk oleh 2 buah pointer

Source code:



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

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

void tukarArray(int array1[3][3], int array2[3][3], int baris, int kolom)
{   if (baris < 3 && kolom < 3) {
        int temp = array1[baris][kolom];
        array1[baris][kolom] = array2[baris][kolom];
        array2[baris][kolom] = temp;
    } else {
        cout << "Indeks baris atau kolom di luar batas!" << endl;
    }
}

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

int main() {
    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}
    };

    int a = 10, b = 20;
    int *p1 = &a;
    int *p2 = &b;

    cout << "Array 1 sebelum pertukaran: " << endl;
    tampilArray(array1);
    cout << "Array 2 sebelum pertukaran: " << endl;
    tampilArray(array2);

    tukarArray(array1, array2, 1, 1);

    cout << "\nArray 1 setelah pertukaran pada posisi (1,1): " << endl;
    tampilArray(array1);
    cout << "Array 2 setelah pertukaran pada posisi (1,1): " << endl;
    tampilArray(array2);

    cout << "\nNilai p1 dan p2 sebelum pertukaran: " << endl;
    cout << "p1: " << *p1 << " p2: " << *p2 << endl;

    tukarPointer(p1, p2);

    cout << "\nNilai p1 dan p2 setelah pertukaran: " << endl;
    cout << "p1: " << *p1 << " p2: " << *p2 << endl;

    return 0;
}
```

Output:

```
6 5 4
PS D:\PRAKTIKUM DATA STRUCTURE\LAPRAK\03_Abstrak_Data_Type\output> cd 'd:\PRAKTIKUM DATA S
PS D:\PRAKTIKUM DATA STRUCTURE\LAPRAK\03_Abstrak_Data_Type\output> & .\'soal3.exe'
Array 1 sebelum pertukaran:
1 2 3
4 5 6
7 8 9
Array 2 sebelum pertukaran:
9 8 7
6 5 4
3 2 1

Array 1 setelah pertukaran pada posisi (1,1):
1 2 3
4 5 6
7 8 9
Array 2 setelah pertukaran pada posisi (1,1):
9 8 7
6 5 4
3 2 1

Nilai p1 dan p2 sebelum pertukaran:
p1: 10 p2: 20

Nilai p1 dan p2 setelah pertukaran:
p1: 20 p2: 10
PS D:\PRAKTIKUM DATA STRUCTURE\LAPRAK\03_Abstrak_Data_Type\output>
```

5. Kesimpulan

Kesimpulan dari laporan ini adalah bahwa mahasiswa diharapkan mampu memahami dan mengimplementasikan konsep Abstract Data Type (ADT) dalam pemrograman. ADT memisahkan definisi tipe data dari detail implementasinya, menyediakan antarmuka yang jelas untuk operasi-operasi pada struktur data seperti stack, queue, dan list. Dengan mendukung prinsip enkapsulasi dan modularitas, ADT mempermudah pengembangan dan pemeliharaan kode. Selain itu, melalui praktik dan implementasi, mahasiswa belajar membangun program yang modular, terstruktur, dan efisien, serta memanfaatkan fungsi-fungsi untuk memisahkan logika dan pengolahan data.