

**LAPORAN PRAKTIKUM  
STRUKTUR DATA**

**MODUL III  
ABSTRACT DATA TYPE (ADT)**



**Disusun Oleh :**

**NAMA : HILMI HAKIM RAMADANI**

**NIM : 103112430016**

**Dosen**

**FAHRUDIN MUKTI WIBOWO**

**PROGRAM STUDI STRUKTUR DATA  
FAKULTAS INFORMATIKA  
TELKOM UNIVERSITY PURWOKERTO  
2025**

## A. Dasar Teori

C++ adalah bahasa pemrograman yang dikembangkan oleh Bjarne Stroustrup di Bell Labs pada awal 1980-an sebagai pengembangan dari bahasa C. Awalnya dirancang untuk sistem Unix, C++ kemudian menjadi bahasa pemrograman tujuan umum (general-purpose programming language).

Bahasa ini mendukung pemrograman tingkat rendah sekaligus menambahkan fitur baru seperti class, inheritance, overloading, serta konsep pemrograman berorientasi objek (OOP) yang membedakannya dari bahasa C.

## B. Guided (berisi screenshot source code & output program disertai penjelasannya)

### Guided 1

#### #Main.cpp

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

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

#### #Mahasiswa.cpp

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

void inputMhs(mahasiswa &m)
{
    cout << "input nama: ";
    cin >> (m).nim;
    cout << "input nilai 1: ";
    cin >> (m).nilai1;
    cout << "input nilai 2: ";
    cin >> (m).nilai2;
}

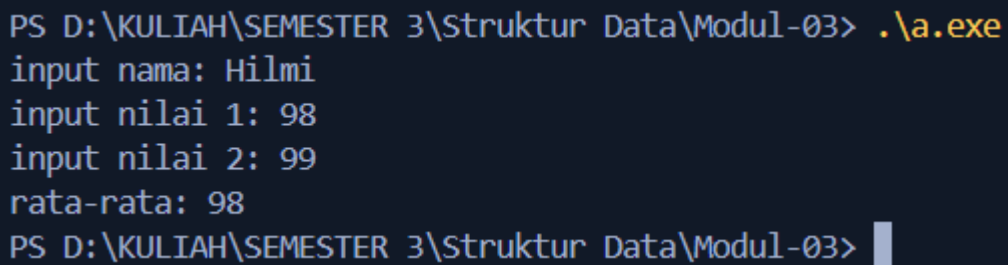
float rata2(mahasiswa m)
{
    return (m.nilai1 + m.nilai2) / 2;
}
```

#Mahasiswa.h

```
#ifndef MAHASISWA_H_INCLUDED
#define MAHASISWA_H_INCLUDED
struct mahasiswa
{
    char nim[10];
    int nilai1, nilai2;
};

void inputMhs(mahasiswa &m);
float rata2(mahasiswa m);
#endif
```

Screenshots Output



```
PS D:\KULIAH\SEMESTER 3\Struktur Data\Modul-03> .\a.exe
input nama: Hilmi
input nilai 1: 98
input nilai 2: 99
rata-rata: 98
PS D:\KULIAH\SEMESTER 3\Struktur Data\Modul-03> █
```

Deskripsi:

Program di atas merupakan contoh program modular C++ yang terdiri dari tiga file: Main.cpp, Mahasiswa.cpp, dan Mahasiswa.h. File Mahasiswa.h berisi deklarasi struct mahasiswa yang menyimpan NIM dan dua nilai, serta deklarasi dua fungsi; inputMhs() untuk menginput data, dan rata2() untuk menghitung rata-rata nilai. Implementasi fungsi tersebut ada di Mahasiswa.cpp; inputMhs() meminta pengguna mengisi NIM dan dua nilai melalui cin, sedangkan rata2() menghitung rata-rata dari dua nilai tersebut. Dalam Main.cpp, program utama membuat variabel mhs bertipe mahasiswa, memanggil inputMhs(mhs) untuk mengisi data, lalu mencetak hasil rata-rata dengan cout.

C. Unguided/Tugas (berisi screenshot source code & output program disertai penjelasannya)

Unguided1

#main1.cpp

```
#include <iostream>
#include "unguided1.h"
#include "unguided1.cpp"
using namespace std;

struct Mahasiswa
{
```

```

string nama;
string nim;
float uts, uas, tugas, nilaiAkhir;
};
int main()
{
    Mahasiswa mhs[10];
    int n;
    cout << "Masukkan jumlah mahasiswa (maks 10): ";
    cin >> n;
    if (n > 10)
        n = 10;
    for (int i = 0; i < n; i++)
    {
        cout << "\nData mahasiswa ke-" << i + 1 << endl;
        cout << "Nama : ";
        cin.ignore();
        getline(cin, mhs[i].nama);
        cout << "NIM : ";
        cin >> mhs[i].nim;
        cout << "Nilai UTS : ";
        cin >> mhs[i].uts;
        cout << "Nilai UAS : ";
        cin >> mhs[i].uas;
        cout << "Nilai Tugas : ";
        cin >> mhs[i].tugas;
        mhs[i].nilaiAkhir = hitungNilaiAkhir(mhs[i].uts, mhs[i].uas, mhs[i].tugas);
    }
    cout << "\nData Mahasiswa:\n";
    for (int i = 0; i < n; i++)
    {
        cout << "\nNama : " << mhs[i].nama;
        cout << "\nNIM : " << mhs[i].nim;
        cout << "\nUTS : " << mhs[i].uts;
        cout << "\nUAS : " << mhs[i].uas;
        cout << "\nTugas : " << mhs[i].tugas;
        cout << "\nNilai Akhir : " << mhs[i].nilaiAkhir << endl;
    }
    return 0;
}

```

#Unguided1.cpp

```

#include "unguided1.h"
float hitungNilaiAkhir(float uts, float uas, float tugas)
{
    return (0.3 * uts) + (0.4 * uas) + (0.3 * tugas);
}

```

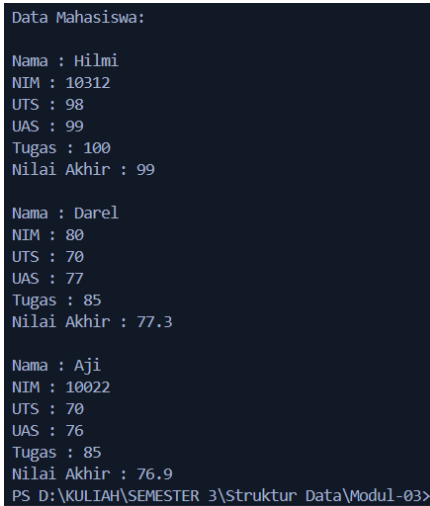
#Unguided1.h

```
#ifndef FUNGSI_H
#define FUNGSI_H

float hitungNilaiAkhir(float uts, float uas, float tugas);

#endif
```

## Screenshots Output



```
Data Mahasiswa:
Nama : Hilmi
NIM : 10312
UTS : 98
UAS : 99
Tugas : 100
Nilai Akhir : 99

Nama : Darel
NIM : 80
UTS : 70
UAS : 77
Tugas : 85
Nilai Akhir : 77.3

Nama : Aji
NIM : 10022
UTS : 70
UAS : 76
Tugas : 85
Nilai Akhir : 76.9
PS D:\KULIAH\SEMESTER 3\Struktur Data\Modul-03>
```

## Deskripsi:

Program di atas adalah contoh program C++ modular untuk menghitung nilai akhir beberapa mahasiswa berdasarkan nilai UTS, UAS, dan tugas. Struktur Mahasiswa digunakan untuk menyimpan data tiap mahasiswa seperti nama, NIM, nilai UTS, UAS, tugas, dan nilai akhir. Pada file main.cpp, program meminta jumlah mahasiswa, lalu melakukan input data untuk masing-masing mahasiswa. Nilai akhir dihitung dengan memanggil fungsi `hitungNilaiAkhir()` yang didefinisikan di `unguided1.cpp` dan dideklarasikan di `unguided1.h`. Fungsi ini menggunakan rumus:  $30\% \text{ UTS} + 40\% \text{ UAS} + 30\% \text{ tugas}$ . Setelah semua data dimasukkan, program menampilkan kembali informasi setiap mahasiswa beserta nilai akhirnya.

## Unguided 2

#Main2.cpp

```
#include <iostream>
#include <string>
#include "unguided2.h"
#include "unguided2.cpp"
```

```

using namespace std;

int main()
{
    string namapel = "Struktur Data";
    string kodepel = "STD";

    pelajaran pel = create_pelajaran(namapel, kodepel);
    tampil_pelajaran(pel);

    return 0;
}

```

#unguided2.cpp

```

#include "unguided2.h"
#include <iostream>
using namespace std;

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

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

```

#unguided2.h

```

#ifndef PELAJARAN_H_INCLUDED
#define PELAJARAN_H_INCLUDED

#include <string>
using namespace std;

struct pelajaran
{
    string namaMapel;
    string kodeMapel;
};

pelajaran create_pelajaran(string namaMapel, string kodeMapel);

void tampil_pelajaran(pelajaran pel);

```

```
#endif
```

### Screenshots Output

```
PS D:\KULIAH\SEMESTER 3\Struktur Data\Modul-03>
ndowsDebugLauncher.exe' '--stdin=Microsoft-MIEng
ine-Error-bnucydcy.vv4' '--pid=Microsoft-MIEngin
nama pelajaran : Struktur Data
nilai : STD
PS D:\KULIAH\SEMESTER 3\Struktur Data\Modul-03>
```

### Deskripsi:

Pada file `unguided2.h`, didefinisikan struct pelajaran yang memiliki dua atribut: `namaMapel` dan `kodeMapel`, serta deklarasi dua fungsi: `create_pelajaran()` untuk membuat objek pelajaran dan `tampil_pelajaran()` untuk menampilkannya. Implementasi kedua fungsi tersebut terdapat pada `unguided2.cpp`. Fungsi `create_pelajaran()` mengisi atribut struct berdasarkan parameter yang diberikan dan mengembalikannya sebagai objek. Fungsi `tampil_pelajaran()` mencetak data pelajaran ke layar. Dalam `main2.cpp`, dibuat variabel `pel` dengan mengisi nama pelajaran dan kode, lalu dipanggil fungsi `tampil_pelajaran()` untuk menampilkan hasilnya.

### Unguided 3

#### #main3.cpp

```
#include <iostream>
#include "unguided3.h"
#include "unguided3.cpp"
using namespace std;

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

    cout << "Array A sebelum ditukar:\n";
    tampilArray(A);
    cout << "Array B sebelum ditukar:\n";
    tampilArray(B);

    tukarArray(A, B, 1, 1);

    cout << "\nArray A setelah ditukar pada [1][1]:\n";
    tampilArray(A);
```

```

    cout << "Array B setelah ditukar pada [1][1]:\n";
    tampilArray(B);

    int x = 10, y = 20;
    int *ptr1 = &x;
    int *ptr2 = &y;

    cout << "\nSebelum tukar pointer:\n";
    cout << "x = " << x << ", y = " << y << endl;

    tukarPointer(ptr1, ptr2);

    cout << "Setelah tukar pointer:\n";
    cout << "x = " << x << ", y = " << y << endl;

    return 0;
}

```

#unguided3.cpp

```

#include <iostream>
#include "unguided3.h"
using namespace std;

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

```



#unguided3.h

```
#ifndef FUNGSI_H
#define FUNGSI_H

void tampilArray(int arr[3][3]);
void tukarArray(int arr1[3][3], int arr2[3][3], int baris, int kolom);
void tukarPointer(int *p1, int *p2);

#endif
```

### Screenshots Output

```
Array A sebelum ditukar:
1 2 3
4 5 6
7 8 9
Array B sebelum ditukar:
10 11 12
13 14 15
16 17 18

Array A setelah ditukar pada [1][1]:
1 2 3
4 14 6
7 8 9
Array B setelah ditukar pada [1][1]:
10 11 12
13 5 15
16 17 18

Sebelum tukar pointer:
x = 10, y = 20
Setelah tukar pointer:
x = 20, y = 10
PS D:\KULIAH\SEMESTER 3\Struktur Data\Modul-03>
```

### Deskripsi:

Program ini menggunakan tiga file agar lebih terstruktur: `unguided3.h` sebagai header yang berisi deklarasi fungsi, `unguided3.cpp` sebagai implementasi fungsi, dan `main3.cpp` sebagai program utama. Dua array 2D berukuran  $3 \times 3$  digunakan untuk menyimpan data, lalu ditampilkan menggunakan fungsi `tampilArray()`. Fungsi `tukarArray()` menukar nilai elemen antara dua array pada posisi tertentu, sedangkan `tukarPointer()` menukar nilai dua variabel melalui pointer. Dalam `main3.cpp`, program menampilkan isi array sebelum dan sesudah penukaran, serta mendemonstrasikan pertukaran nilai variabel `x` dan `y`.

### D. Kesimpulan

Dari praktikum ini dapat disimpulkan bahwa penggunaan Abstract Data Type (ADT) dalam bahasa C++ membantu membuat program lebih modular, terstruktur, dan mudah dipelihara. Dengan memisahkan kode ke dalam tiga file yaitu header untuk deklarasi, cpp implementasi untuk logika fungsi, dan main untuk program utama yaitu proses pengembangan menjadi lebih rapi dan fleksibel. Penggunaan struct memudahkan

pengelolaan data kompleks seperti data mahasiswa dan pelajaran, sedangkan penggunaan fungsi dan pointer memungkinkan manipulasi data yang lebih efisien, seperti perhitungan nilai rata-rata dan pertukaran nilai dalam array maupun variabel. Secara keseluruhan, penerapan konsep ADT ini melatih pemahaman tentang pengorganisasian program dan prinsip modularity dalam pemrograman C++.

#### E. Referensi

Dewi, L. J. E. (2010). Media Pembelajaran Bahasa Pemrograman C++. Jurnal Pendidikan Teknologi dan Kejuruan, 7(1).

Wikipedia. 2025, 29 September. C++. Diakses pada 1 Oktober. Dari <https://en.wikipedia.org/wiki/C%2B%2B>