

**LAPORAN PRAKTIKUM
STRUKTUR DATA**

**MODUL III
ABSTRACT DATA TYPE**



Disusun Oleh :

NAMA : Muhamad Naufal Ammar

NIM : 103112430036

Dosen

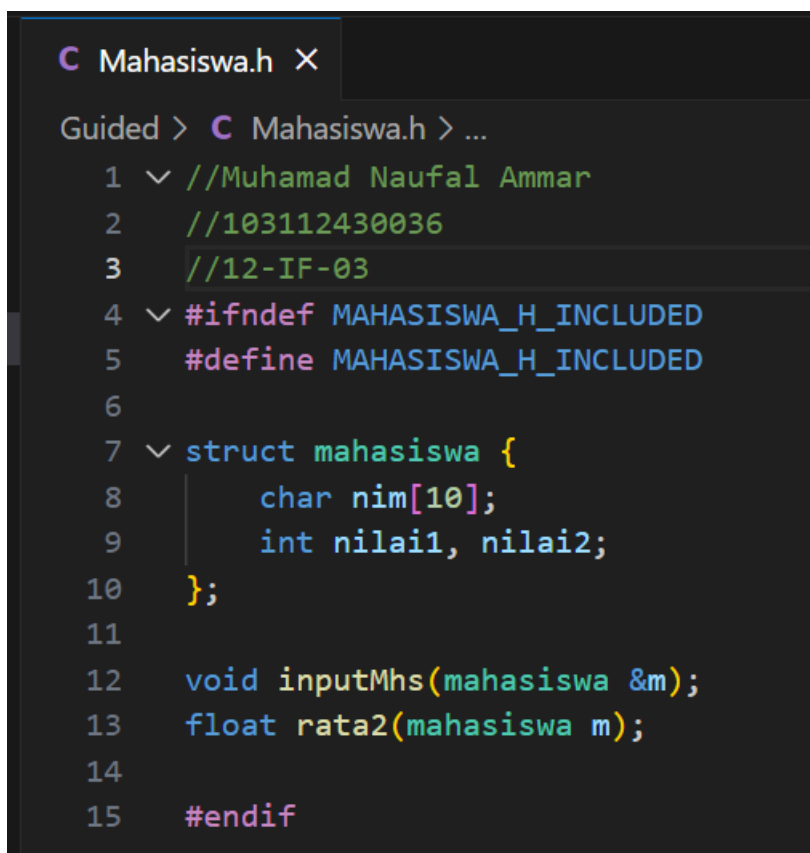
WAHYU ANDI SAPUTRA

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

A. Dasar Teori

Dasar teori di balik kode-kode seperti penggunaan ADT (Abstract Data Type), array dua dimensi, dan pointer dalam C++ sangat berkaitan dengan konsep abstraksi data dan modularitas yang sering dibahas dalam literatur pendidikan komputer. Abstraksi data memungkinkan penyembunyian detail implementasi dan mengelompokkan data serta fungsi yang berhubungan ke dalam satu unit (seperti struct + fungsi/prosedur), sehingga bagian-bagian kode bisa digunakan ulang dan lebih mudah dipelihara. Misalnya, bagian input, logika perhitungan (seperti rata-rata atau pertukaran data), dan bagian output bisa dipisah agar tanggung jawab masing-masing jelas, sesuai prinsip modular programming (Data Abstraction and Modularity). Selain itu, pointer dan array multidimensi mendukung pengajaran tentang bagaimana memori diakses dan bagaimana variabel, alamat, dan data berinteraksi di level rendah ini penting agar mahasiswa tidak hanya mengerti sintaks, tetapi juga cara kerja internal program. Penelitian seperti *Classroom practice for understanding pointers using learning support system for visualizing memory image and target domain world* menekankan bahwa visualisasi dan penggunaan pointer konkret sangat membantu mahasiswa memahami topik yang abstrak seperti alamat memori dan dereferensiasi variabel. Guided (berisi screenshot source code & output program disertai penjelasannya)

Guided 1



```
C Mahasiswa.h X
Guided > C Mahasiswa.h > ...
1  //Muhamad Naufal Ammar
2  //103112430036
3  //12-IF-03
4  #ifndef MAHASISWA_H_INCLUDED
5  #define MAHASISWA_H_INCLUDED
6
7  struct mahasiswa {
8      char nim[10];
9      int nilai1, nilai2;
10 };
11
12 void inputMhs(mahasiswa &m);
13 float rata2(mahasiswa m);
14
15 #endif
```

C Mahasiswa.h

➤ Mahasiswa.cpp ✕

Guided > ➤ Mahasiswa.cpp > ...

```
1 //Muhamad Naufal Ammar
2 //103112430036
3 //12-IF-03
4 #include <iostream>
5 #include "mahasiswa.h"
6 using namespace std;
7
8 void inputMhs(mahasiswa &m) {
9     cout << "NIM: " ;
10    cin >> m.nim;
11    cout << "Nilai 1: ";
12    cin >> m.nilai1;
13    cout << "Nilai 2: ";
14    cin >> m.nilai2;
15 }
16
17 float rata2(mahasiswa m) {
18     return float(m.nilai1 + m.nilai2) / 2.0;
19 }
```

```
C Mahasiswa.h  Mahasiswa.cpp  main.cpp
Guided > main.cpp > ...
1 //Muhamad Naufal Ammar
2 //103112430036
3 //12-IF-03
4 #include <iostream>
5 #include "mahasiswa.h"
6 #include "mahasiswa.cpp"
7 using namespace std;
8
9 int main(){
10     mahasiswa mhs;
11     inputMhs(mhs);
12     cout << "Rata-rata = " << rata2(mhs);
13     return 0;
14 }
```

Screenshots Output

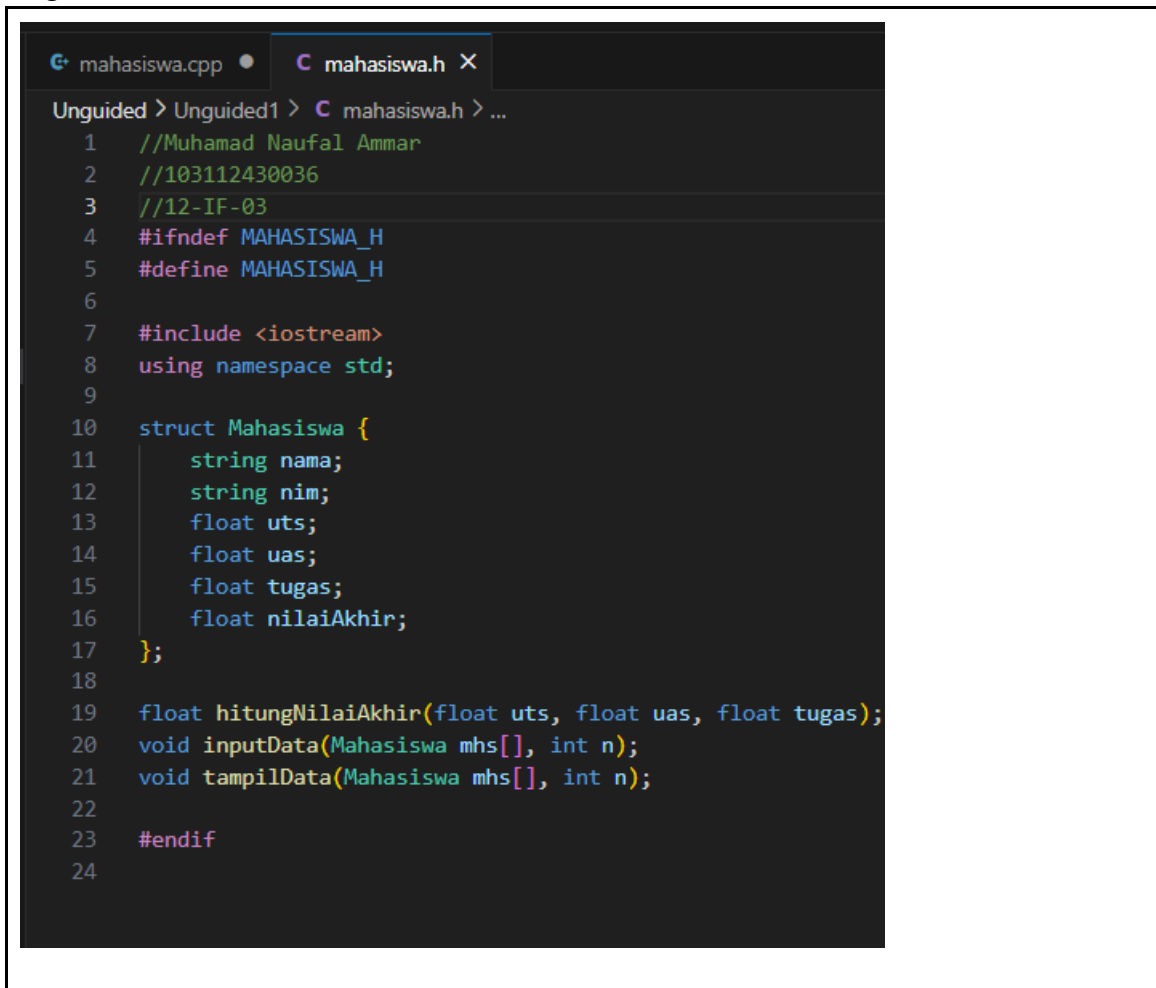
```
PS C:\alpro\SMT 3\modul3\Guided> cd "c:\alpro\SMT 3\modul3\Guided\" ; if ($?) { g++ main.cpp -o main } ; if ($?) { .\main }
NIM: 103112430099
Nilai 1: 90
Nilai 2: 98
Rata-rata = 94
```

Deskripsi:

Program ini mengimplementasikan konsep ADT (Abstract Data Type) untuk mempermudah pengelolaan data mahasiswa. Dalam file Mahasiswa.h, dideklarasikan struktur data yang berisi atribut *nim*, *nilai1*, dan *nilai2*, serta deklarasi fungsi-fungsi pendukung. Implementasi dari fungsi tersebut terdapat pada file Mahasiswa.cpp, yang mencakup fungsi inputMhs() untuk menerima data dari pengguna dan rata2() untuk menghitung nilai rata-rata dari dua nilai yang dimasukkan. File main.cpp berperan sebagai program utama yang memanggil kedua fungsi tersebut. Pengguna diminta mengisi data mahasiswa, dan program kemudian menampilkan hasil perhitungan rata-ratanya. Melalui pemisahan file ini, program menjadi lebih terstruktur, mudah dibaca, serta efisien dalam pengelolaan kode..

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

Unguided 1



The screenshot shows a code editor with two tabs: 'mahasiswa.cpp' and 'mahasiswa.h'. The 'mahasiswa.h' tab is active, displaying the following C++ code:

```
Unguided > Unguided1 > C mahasiswa.h > ...
1  //Muhamad Naufal Ammar
2  //103112430036
3  //12-IF-03
4  #ifndef MAHASISWA_H
5  #define MAHASISWA_H
6
7  #include <iostream>
8  using namespace std;
9
10 struct Mahasiswa {
11     string nama;
12     string nim;
13     float uts;
14     float uas;
15     float tugas;
16     float nilaiAkhir;
17 };
18
19 float hitungNilaiAkhir(float uts, float uas, float tugas);
20 void inputData(Mahasiswa mhs[], int n);
21 void tampilData(Mahasiswa mhs[], int n);
22
23 #endif
24
```

mahasiswa.cpp

Unguided > Unguided1 > mahasiswa.cpp > ...

```
1 //Muhamad Naufal Ammar
2 //103112430036
3 //12-IF-03
4 #include "mahasiswa.h"
5
6 float hitungNilaiAkhir(float uts, float uas, float tugas) {
7     return (0.3 * uts) + (0.4 * uas) + (0.3 * tugas);
8 }
9
10 void inputData(Mahasiswa mhs[], int n) {
11     for (int i = 0; i < n; i++) {
12         cout << "\nData mahasiswa ke-" << (i + 1) << endl;
13         cout << "Nama : ";
14         cin >> mhs[i].nama;
15         cout << "NIM : ";
16         cin >> mhs[i].nim;
17         cout << "Nilai UTS : ";
18         cin >> mhs[i].uts;
19         cout << "Nilai UAS : ";
20         cin >> mhs[i].uas;
21         cout << "Nilai Tugas : ";
22         cin >> mhs[i].tugas;
23
24         mhs[i].nilaiAkhir = hitungNilaiAkhir(mhs[i].uts, mhs[i].uas, mhs[i].tugas);
25     }
26 }
27
28 void tampilData(Mahasiswa mhs[], int n) {
29     cout << "\n=== DATA NILAI MAHASISWA ===" << endl;
30     cout << "-----" << endl;
31     cout << "Nama\tNIM\tUTS\tUAS\tTugas\tNilai Akhir" << endl;
32     cout << "-----" << endl;
33
34     for (int i = 0; i < n; i++) {
35         cout << mhs[i].nama << "\t"
36             << mhs[i].nim << "\t"
37             << mhs[i].uts << "\t"
38             << mhs[i].uas << "\t"
39             << mhs[i].tugas << "\t"
40             << mhs[i].nilaiAkhir << endl;
41     }
42 }
43
```

```

mahasiswa.cpp  mahasiswa.h  main.cpp X
Unguided > Unguided1 > main.cpp > ...
1  //Muhamad Naufal Ammar
2  //103112430036
3  //12-IF-03
4  #include "mahasiswa.h"
5  #include "mahasiswa.cpp"
6  #include <iostream>
7  using namespace std;
8
9  int main() {
10     Mahasiswa mhs[10];
11     int n;
12
13     cout << "Masukkan jumlah mahasiswa (maks 10): ";
14     cin >> n;
15
16     if (n > 10) {
17         cout << "Jumlah mahasiswa melebihi batas maksimal (10)!" << endl;
18         return 0;
19     }
20
21     inputData(mhs, n);
22     tampilData(mhs, n);
23
24     return 0;
25 }
```

Screenshots Output

```

PS C:\alpro\SMT 3\modul-amer\103112430036_MuhamadNaufalAmmar_12-IF-03_Modul3\Unguided\Unguided1> cd "c:\alpro\SMT 3\modul-amer\103112430036_MuhamadNaufalAmmar_12-IF-03_Modul3\Unguided\Unguided1\" ; if ($?) { g++ main.cpp -o main } ; if ($?) { .\main }
Masukkan jumlah mahasiswa (maks 10): 2

Data mahasiswa ke-1
Nama : Ammar
NIM : 103112430036
Nilai UTS : 90
Nilai UAS : 98
Nilai Tugas : 99

Data mahasiswa ke-2
Nama : Naufal
NIM : 90
Nilai UTS : 87
Nilai UAS : 88
Nilai Tugas : 90

=== DATA NILAI MAHASISWA ===
-----
Nama      NIM      UTS      UAS      Tugas      Nilai Akhir
-----
Ammar     103112430036  90      98      99      95.9
Naufal    90          87      88      90      88.3
```

Deskripsi:

Program ini dirancang untuk mempermudah pengguna dalam menyimpan, menghitung, serta menampilkan data nilai mahasiswa secara efisien dan terstruktur. Selain itu, program ini juga berfungsi sebagai penerapan praktis dari konsep struktur data, fungsi, dan prosedur dalam proses pengembangan perangkat lunak menggunakan bahasa pemrograman C++, sehingga membantu pengguna memahami penerapan modularitas dalam pembuatan program.

Unguided 2

C pelajaran.h X

Unguided > Unguided2 > C pelajaran.h > ...

```
1  //Muhamad Naufal Ammar
2  //103112430036
3  //12-IF-03
4  #ifndef PELAJARAN_H_INCLUDED
5  #define PELAJARAN_H_INCLUDED
6
7  #include <string>
8  using namespace std;
9
10 struct pelajaran {
11     string namaMapel;
12     string kodeMapel;
13 };
14 pelajaran create_pelajaran(string namapel, string kodepel);
15 void tampil_pelajaran(pelajaran p);
16
17 #endif
18
```


C pelajaran.h

G+ pelajaran.cpp •



Unguided > Unguided2 > G+ pelajaran.cpp > ...

```
1  ∨ //Muhamad Naufal Ammar
2  //103112430036
3  //12-IF-03
4  ∨ #include "pelajaran.h"
5  #include <iostream>
6  #include <string>
7  using namespace std;
8
9  ∨ pelajaran create_pelajaran(string namapel, string kodepel) {
10     pelajaran p;
11     p.namaMapel = namapel;
12     p.kodeMapel = kodepel;
13     return p;
14 }
15
16 ∨ void tampil_pelajaran(pelajaran p) {
17     cout << "nama pelajaran : " << p.namaMapel << endl;
18     cout << "nilai : " << p.kodeMapel << endl;
19 }
20
```

```

C pelajaran.h X  G+ pelajaran.cpp  G+ main.cpp X
Unguided > Unguided2 > G+ main.cpp > ...
1  //Muhamad Naufal Ammar
2  //103112430036
3  //12-IF-03
4  #include <iostream>
5  #include "pelajaran.h"
6  #include "pelajaran.cpp"
7  using namespace std;
8
9  int main() {
10     string namapel = "Struktur Data";
11     string kodepel = "STD";
12
13     pelajaran pel = create_pelajaran(namapel, kodepel);
14     tampil_pelajaran(pel);
15
16     return 0;
17 }
18
```

Screenshots Output

```

PS C:\alpro\SMT 3\modul-amer\103112430036_MuhamadNaufalAmmar_12-IF-03_Modul3\Unguided\Unguided1> cd "c:\alpro\SMT 3\modul-amer\103112430036_MuhamadNaufalAmmar_12-IF-03_Modul3\Unguided\Unguided2\" ; if ($?) { g++ main.cpp -o main } ; if ($?) { .\main }
nama pelajaran : Struktur Data
nilai : STD
```

Deskripsi:

Kode ini berfungsi untuk mengelola serta menampilkan data mata pelajaran dengan menerapkan konsep Abstract Data Type (ADT) dalam bahasa pemrograman C++. Struktur data, fungsi pembuat objek, dan prosedur penampil data dipisahkan ke dalam file berbeda agar program lebih rapi dan mudah dikelola. Melalui implementasi ini, pengguna dapat membuat objek pelajaran baru dengan memasukkan nama dan kode mata pelajaran, lalu menampilkan hasilnya ke layar. Secara keseluruhan, kode ini bertujuan untuk memberikan pemahaman mengenai penerapan pemrograman modular berbasis ADT, di mana setiap komponen memiliki peran dan tanggung jawab yang terdefinisi dengan baik.

Unguided 3

C arraypointer.h X

Unguided > Unguided3 > C arraypointer.h > ...

```
1 //Muhamad Naufal Ammar
2 //103112430036
3 //12-IF-03
4 #ifndef ARRAYPOINTER_H_INCLUDED
5 #define ARRAYPOINTER_H_INCLUDED
6
7 #include <iostream>
8 using namespace std;
9
10 void tampilArray(int arr[3][3]);
11 void tukarArray(int arr1[3][3], int arr2[3][3], int baris, int kolom);
12 void tukarPointer(int *ptr1, int *ptr2);
13
14 #endif
15
```

C arraypointer.h

C arraypointer.cpp ●

Unguided > Unguided3 > C arraypointer.cpp > ...

```
1 //Muhamad Naufal Ammar
2 //103112430036
3 //12-IF-03
4 #include "arrayPointer.h"
5
6 void tampilArray(int arr[3][3]) {
7     for (int i = 0; i < 3; i++) {
8         for (int j = 0; j < 3; j++) {
9             cout << arr[i][j] << " ";
10        }
11        cout << endl;
12    }
13 }
14
15 void tukarArray(int arr1[3][3], int arr2[3][3], int baris, int kolom) {
16     int temp = arr1[baris][kolom];
17     arr1[baris][kolom] = arr2[baris][kolom];
18     arr2[baris][kolom] = temp;
19 }
20
21 void tukarPointer(int *ptr1, int *ptr2) {
22     int temp = *ptr1;
23     *ptr1 = *ptr2;
24     *ptr2 = temp;
25 }
26
```

```

arraypointer.h  arraypointer.cpp  main.cpp
Unguided > Unguided3 > main.cpp > main()
1  //Muhamad Naufal Ammar
2  //103112430036
3  //12-IF-03
4  #include <iostream>
5  #include "arrayPointer.h"
6  #include "arrayPointer.cpp"
7  using namespace std;
8
9  int main() {
10     int arr1[3][3] = {
11         {1, 2, 3},
12         {4, 5, 6},
13         {7, 8, 9}
14     };
15     int arr2[3][3] = {
16         {9, 8, 7},
17         {6, 5, 4},
18         {3, 2, 1}
19     };
20
21     int a = 10, b = 20;
22     int *ptr1 = &a;
23     int *ptr2 = &b;
24
25     cout << "Array 1 awal:\n";
26     tampilArray(arr1);
27
28     cout << "\nArray 2 awal:\n";
29     tampilArray(arr2);
30
31     tukarArray(arr1, arr2, 1, 1);
32     void tampilArray(int (*arr)[3]) {
33         for (int i = 0; i < 3; i++) {
34             for (int j = 0; j < 3; j++) {
35                 cout << arr[i][j] << " ";
36                 if (j == 2) cout << "\n";
37             }
38         }
39     }
40     cout << "\nSebelum pertukaran pointer:\n";
41     cout << "a = " << a << ", b = " << b << endl;
42     tukarPointer(ptr1, ptr2);
43     cout << "Setelah pertukaran pointer:\n";
44     cout << "a = " << a << ", b = " << b << endl;
45     return 0;
46 }
47

```

Screenshots Output

```
PS C:\alpro\SMT 3\modul-amer\103112430036_MuhamadNaufalAmmar_12-IF-03_Modul3\Unguided\Unguided3> cd "c:\alpro\SMT 3\modul-amer\103112430036_MuhamadNaufalAmmar_12-IF-03_Modul3\Unguided\Unguided3\" ; if ($?) { g++ main.cpp -o main } ; if ($?) { .\main }
Array 1 awal:
1 2 3
4 5 6
7 8 9

Array 2 awal:
9 8 7
6 5 4
3 2 1

Setelah pertukaran elemen [1][1]:
Array 1:
1 2 3
4 5 6
7 8 9
Array 2:
9 8 7
6 5 4
3 2 1

Sebelum pertukaran pointer:
a = 10, b = 20
Setelah pertukaran pointer:
a = 20, b = 10
```

Deskripsi:

Program ini digunakan untuk memahami dan mengimplementasikan konsep array dua dimensi serta pointer dalam bahasa C++. Melalui program ini, pengguna dapat melihat bagaimana dua buah array 2D berukuran 3x3 dapat diproses dan dimodifikasi dengan menggunakan fungsi atau prosedur terpisah. Program ini juga memperlihatkan cara menukar nilai antar elemen dari dua array pada posisi tertentu, serta menukar isi dari dua variabel menggunakan pointer.

C. Kesimpulan

Berdasarkan dasar teori yang telah dijelaskan, dapat disimpulkan bahwa penerapan konsep ADT, array, pointer, dan modularitas dalam C++ memiliki peran penting dalam membangun pemahaman mahasiswa terhadap struktur program yang efisien dan terorganisasi. Melalui pendekatan ini, mahasiswa tidak hanya belajar menulis kode, tetapi juga memahami bagaimana data diolah, disimpan, dan dipanggil secara sistematis. Penggunaan ADT membantu memisahkan tanggung jawab antarbagian kode agar lebih mudah dikelola, sementara pointer dan array mengajarkan cara kerja memori secara langsung. Dengan demikian, penerapan konsep-konsep ini tidak hanya meningkatkan kemampuan teknis dalam pemrograman, tetapi juga menumbuhkan pola pikir logis dan analitis dalam merancang perangkat lunak yang modular, efisien, dan mudah dikembangkan.

D. Referensi

Classroom practice for understanding pointers using learning support system for visualizing memory image and target domain world. (2017). *Research and Practice in Technology Enhanced Learning*, 12, Article 17.

Mitchell, J. C. (2012). *Data Abstraction and Modularity*. Dalam *Concepts in Programming Languages* (Bab 9). Cambridge University Press

