

LAPORAN PRAKTIKUM
Modul III
“”



Disusun Oleh:
Dewi Atika Muthi -2211104042
SE-07-02

Dosen:
Wahyu Andi Saputra

PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
PURWOKERTO
202

1. Tujuan

Tujuan praktikum:

- Memahami konsep Abstract Data Type (ADT) dan penggunaannya dalam pemrograman
- Memahami penerapan konsep ADT pada program untuk menyimpan data mahasiswa, mengelola pelajaran, dan memanipulasi array 2D menggunakan pointer.

2. Landasan Teori

Abstract Data Type (ADT) adalah sebuah konsep dalam ilmu komputer yang mendefinisikan tipe data beserta operasi-operasi yang dapat dilakukan terhadapnya tanpa memperhatikan implementasi detail dari operasi tersebut. ADT memberikan struktur data yang abstrak, yang berarti penggunaannya hanya perlu memahami operasinya, tanpa mengetahui bagaimana data tersebut dikelola di dalam memori.

Komponen ADT meliputi:

- **Konstruktor/Kreator:** Membuat nilai dari tipe data.
- **Selector:** Mengakses komponen dari tipe data.
- **Prosedur Pengubah Nilai:** Mengubah komponen dalam tipe data.
- **Validator:** Mengecek apakah nilai memenuhi batasan tipe.
- **Destruktor:** Menghapus nilai dari tipe data.
- **Input/Output:** Berinteraksi dengan perangkat input/output.
- **Operator Relasional:** Untuk perbandingan antar nilai.
- **Aritmatika:** Operasi matematika terhadap tipe data tersebut.

Pada ADT, deklarasi dan definisi tipe data serta operasi dibagi menjadi dua bagian: file header (.h) untuk deklarasi, dan file implementasi (.cpp) untuk realisasi.

3. Guided

```
1  #include <iostream>
2
3  using namespace std;
4
5  struct mahasiswa{
6      char nim[10];
7      int nilai1, nilai2;
8  };
9
10 void inputMhs(mahasiswa &m);
11 float rata2(mahasiswa m);
12
13 int main()
14 {
15     mahasiswa mhs;
16     inputMhs(mhs);
17     cout << "rata-rata = " << rata2(mhs);
18     return 0;
19 }
20
21 void inputMhs(mahasiswa &m){
22     cout << "Input NIM = ";
23     cin >> (m).nim;
24     cout << "Input Nilai = ";
25     cin >> (m).nilai1;
26     cout << "Input Nilai = ";
27     cin >> (m).nilai2;
28 }
29
30 float rata2(mahasiswa m){
31     return (m.nilai1+m.nilai2)/2;
32 }
33
34
```

Untuk menerapkan konsep ADT, kita harus memisah deklarasi tipe, variabel, dan fungsi dari program ke dalam sebuah file.h dan memisah definisi fungsi dari program ke sebuah file.cpp. Sehingga jika kita menerapkan konsep ADT berdasarkan contoh program di atas, bentuk code program akan dipisah menjadi seperti berikut:

mahasiswa.h:

```
struct mahasiswa{  
    char nim[10];  
    int nilai1, nilai2;  
};  
  
void inputMhs(mahasiswa &m);  
float rata2(mahasiswa m);
```

mahasiswa.cpp:

```
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;  
}
```

main.cpp:

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

4. Unguided

- 1) Membuat program untuk menyimpan data mahasiswa (maksimal 10) dengan field nama, nim, uts, uas, tugas, dan nilai akhir menggunakan array. Nilai akhir dihitung menggunakan fungsi dengan rumus:
$$\text{Nilai Akhir} = 0.3 \times \text{UTS} + 0.4 \times \text{UAS} + 0.3 \times \text{Tugas}$$

mahasiswa.h:

```
1  #ifndef MAHASISWA_H
2  #define MAHASISWA_H
3
4  #include <string>
5
6  struct Mahasiswa {
7      std::string nama;
8      std::string nim;
9      float uts, uas, tugas;
10     float nilai_akhir;
11 };
12
13 void inputMahasiswa(Mahasiswa &m);
14 float hitungNilaiAkhir(const Mahasiswa &m);
15 void tampilkanMahasiswa(const Mahasiswa &m);
16 void inputArrayMahasiswa(Mahasiswa arr[], int &n, int max);
17 void tampilkanArrayMahasiswa(const Mahasiswa arr[], int n);
18
19 #endif
```

Penjelasan:

- **Struct Mahasiswa:**

Struktur Mahasiswa digunakan untuk menyimpan informasi terkait seorang mahasiswa, yaitu:

- nama: Nama mahasiswa (tipe data std::string).
- nim: Nomor Induk Mahasiswa (NIM) (tipe data std::string).
- uts, uas, tugas: Nilai UTS, UAS, dan tugas (tipe data float).
- nilai_akhir: Nilai akhir mahasiswa, hasil perhitungan dari UTS, UAS, dan tugas.

- **Function Prototypes:**

- **inputMahasiswa(Mahasiswa &m):** Fungsi untuk menginput data seorang mahasiswa, termasuk nilai UTS, UAS, tugas, dan menghitung nilai akhirnya.
- **hitungNilaiAkhir(const Mahasiswa &m):** Fungsi untuk menghitung nilai akhir mahasiswa berdasarkan bobot dari nilai UTS (30%), UAS (40%), dan tugas (30%).
- **tampilkanMahasiswa(const Mahasiswa &m):** Fungsi untuk menampilkan data seorang mahasiswa, termasuk nama, NIM, dan nilai akhirnya.

- **inputArrayMahasiswa(Mahasiswa arr[], int &n, int max):**
Fungsi untuk menginput data beberapa mahasiswa hingga jumlah maksimum (max), disimpan dalam array arr.
- **tampilkanArrayMahasiswa(const Mahasiswa arr[], int n):**
Fungsi untuk menampilkan data seluruh mahasiswa dalam array arr hingga jumlah n.

mahasiswa.cpp:

```

1  #include "mahasiswa.h"
2  #include <iostream>
3  #include <iomanip>
4
5
6  float hitungNilaiAkhir(const Mahasiswa &m) {
7      return 0.3 * m.uts + 0.4 * m.uas + 0.3 * m.tugas;
8  }
9
10 void inputMahasiswa(Mahasiswa &m) {
11     std::cout << "Nama: ";
12     std::getline(std::cin, m.nama);
13     std::cout << "NIM: ";
14     std::getline(std::cin, m.nim);
15     std::cout << "Nilai UTS: ";
16     std::cin >> m.uts;
17     std::cout << "Nilai UAS: ";
18     std::cin >> m.uas;
19     std::cout << "Nilai Tugas: ";
20     std::cin >> m.tugas;
21
22     std::cin.ignore();
23
24     m.nilai_akhir = hitungNilaiAkhir(m);
25     std::cout << "Nilai Akhir: " << std::fixed << std::setprecision(2) << m.nilai_akhir << std::endl;
26 }
27
28 void tampilkanMahasiswa(const Mahasiswa &m) {
29     std::cout << "Nama: " << m.nama << ", NIM: " << m.nim
30     << ", Nilai Akhir: " << m.nilai_akhir << std::endl;
31 }
32
33 void inputArrayMahasiswa(Mahasiswa arr[], int &n, int max) {
34     n = 0;
35     char lanjut;
36     do {
37         if (n < max) {
38             inputMahasiswa(arr[n]);
39             n++;
40             if (n < max) {
41                 std::cout << "Input lagi? (y/n): ";
42                 std::cin >> lanjut;
43                 std::cin.ignore();
44             }
45         } while (n < max && (lanjut == 'y' || lanjut == 'Y'));
46     }
47
48 void tampilkanArrayMahasiswa(const Mahasiswa arr[], int n) {
49     for (int i = 0; i < n; i++) {
50         tampilkanMahasiswa(arr[i]);
51     }
52 }
53

```

Penjelasan:

- **hitungNilaiAkhir:** Menghitung nilai akhir mahasiswa berdasarkan bobot UTS (30%), UAS (40%), dan tugas (30%).

- **inputMahasiswa:** Menginput data seorang mahasiswa (nama, NIM, nilai UTS, UAS, tugas), lalu menghitung nilai akhirnya.
- **tampilkanMahasiswa:** Menampilkan data seorang mahasiswa (nama, NIM, dan nilai akhir).
- **inputArrayMahasiswa:** Menginput beberapa mahasiswa ke dalam array hingga jumlah maksimum (`max`), dengan opsi untuk melanjutkan input.
- **tampilkanArrayMahasiswa:** Menampilkan data semua mahasiswa dalam array.

main.cpp:

```

1  #include "mahasiswa.h"
2  #include <iostream>
3
4  int main() {
5      const int MAX_MAHASISWA = 10;
6      Mahasiswa daftarMahasiswa[MAX_MAHASISWA];
7      int jumlahMahasiswa = 0;
8
9      inputArrayMahasiswa(daftarMahasiswa, jumlahMahasiswa, MAX_MAHASISWA);
10     std::cout << "\nDaftar Mahasiswa:" << std::endl;
11     tampilkanArrayMahasiswa(daftarMahasiswa, jumlahMahasiswa);
12
13     return 0;
14 }
15

```

Penjelasan:

- **Inisialisasi:**
 - `MAX_MAHASISWA` diset sebagai batas maksimum 10 mahasiswa.
 - Array `daftarMahasiswa` digunakan untuk menyimpan data mahasiswa.
 - `jumlahMahasiswa` menyimpan jumlah mahasiswa yang dimasukkan.
- **Proses Input:**
 - Fungsi `inputArrayMahasiswa` digunakan untuk menginput data beberapa mahasiswa ke dalam array.
- **Proses Output:**
 - Setelah input selesai, fungsi `tampilkanArrayMahasiswa` akan menampilkan daftar mahasiswa beserta nilai akhirnya.

OUTPUT:

```

Nama: Dewi Atika
NIM: 2211104042
Nilai UTS: 98
Nilai UAS: 97
Nilai Tugas: 100
Nilai Akhir: 98.20
Input lagi? (y/n): y
Nama: Chipa
NIM: 2100112
Nilai UTS: 87
Nilai UAS: 90
Nilai Tugas: 94
Nilai Akhir: 90.30
Input lagi? (y/n): n

Daftar Mahasiswa:
Nama: Dewi Atika, NIM: 2211104042, Nilai Akhir: 98.20
Nama: Chipa, NIM: 2100112, Nilai Akhir: 90.30

Process returned 0 (0x0)   execution time : 177.434 s
Press any key to continue.

```

2) Implementasi ADT Pelajaran

Buat ADT pelajaran dengan dua atribut: `namaMapel` dan `kodeMapel`. Gunakan fungsi untuk membuat pelajaran dan prosedur untuk menampilkannya.

pelajaran.h:

```

1  #ifndef PELAJARAN_H
2  #define PELAJARAN_H
3
4  #include <string>
5
6  struct Pelajaran {
7      std::string namaMapel;
8      std::string kodeMapel;
9  };
10
11 Pelajaran createPelajaran(const std::string& nama, const std::string& kode);
12 void tampilPelajaran(const Pelajaran& pel);
13
14 #endif

```

pelajaran.cpp:

```

1  #include "pelajaran.h"
2  #include <iostream>
3
4  Pelajaran createPelajaran(const std::string& nama, const std::string& kode) {
5      Pelajaran pel;
6      pel.namaMapel = nama;
7      pel.kodeMapel = kode;
8      return pel;
9  }
10
11 void tampilPelajaran(const Pelajaran& pel) {
12     std::cout << "Nama Pelajaran: " << pel.namaMapel << std::endl;
13     std::cout << "Kode Pelajaran: " << pel.kodeMapel << std::endl;
14 }

```

main.cpp:

```
1  #include "pelajaran.h"
2  #include <iostream>
3
4  int main() {
5      Pelajaran pel = createPelajaran("Struktur Data", "STD");
6      tampilPelajaran(pel);
7      return 0;
8  }
9
```

OUTPUT:

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

Process returned 0 (0x0)   execution time :
Press any key to continue.
```

3) Manipulasi Array 2D dengan Pointer

Buat dua array 2D berukuran 3x3 dan dua pointer integer. Implementasikan prosedur untuk menampilkan isi array, menukarkan isi array pada posisi tertentu, serta menukarkan nilai pada variabel yang ditunjuk oleh dua pointer

array2d.cpp:

```
1  #include <iostream>
2
3  const int SIZE = 3;
4
5  void displayArray(int arr[SIZE][SIZE]) {
6      for (int i = 0; i < SIZE; i++) {
7          for (int j = 0; j < SIZE; j++) {
8              std::cout << arr[i][j] << " ";
9          }
10         std::cout << std::endl;
11     }
12     std::cout << std::endl;
13 }
14
15 void swapArrayElements(int arr[SIZE][SIZE], int row1, int col1, int row2, int col2) {
16     int temp = arr[row1][col1];
17     arr[row1][col1] = arr[row2][col2];
18     arr[row2][col2] = temp;
19 }
20
21 void swapPointerValues(int* ptr1, int* ptr2) {
22     int temp = *ptr1;
23     *ptr1 = *ptr2;
24     *ptr2 = temp;
25 }
26
```

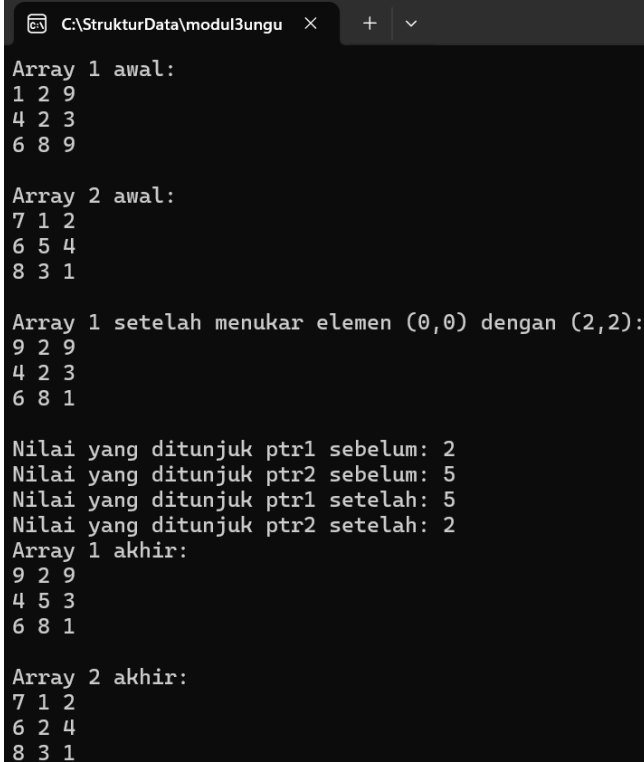


```

26
27 int main() {
28     int array1[SIZE][SIZE] = {{1, 2, 9}, {4, 2, 3}, {6, 8, 9}};
29     int array2[SIZE][SIZE] = {{7, 1, 2}, {6, 5, 4}, {8, 3, 1}};
30
31     std::cout << "Array 1 awal:" << std::endl;
32     displayArray(array1);
33
34     std::cout << "Array 2 awal:" << std::endl;
35     displayArray(array2);
36
37     // Menukar elemen pada posisi tertentu
38     swapArrayElements(array1, 0, 0, 2, 2);
39     std::cout << "Array 1 setelah menukar elemen (0,0) dengan (2,2):" << std::endl;
40     displayArray(array1);
41
42     // Menukar nilai menggunakan pointer
43     int* ptr1 = &array1[1][1];
44     int* ptr2 = &array2[1][1];
45
46     std::cout << "Nilai yang ditunjuk ptr1 sebelum: " << *ptr1 << std::endl;
47     std::cout << "Nilai yang ditunjuk ptr2 sebelum: " << *ptr2 << std::endl;
48
49     swapPointerValues(ptr1, ptr2);
50
51     std::cout << "Nilai yang ditunjuk ptr1 setelah: " << *ptr1 << std::endl;
52     std::cout << "Nilai yang ditunjuk ptr2 setelah: " << *ptr2 << std::endl;
53
54     std::cout << "Array 1 akhir:" << std::endl;
55     displayArray(array1);
56
57     std::cout << "Array 2 akhir:" << std::endl;
58     displayArray(array2);
59
60     return 0;
61 }
62

```

OUTPUT:



```

C:\StrukturData\modul3ungu
Array 1 awal:
1 2 9
4 2 3
6 8 9

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

Array 1 setelah menukar elemen (0,0) dengan (2,2):
9 2 9
4 2 3
6 8 1

Nilai yang ditunjuk ptr1 sebelum: 2
Nilai yang ditunjuk ptr2 sebelum: 5
Nilai yang ditunjuk ptr1 setelah: 5
Nilai yang ditunjuk ptr2 setelah: 2
Array 1 akhir:
9 2 9
4 5 3
6 8 1

Array 2 akhir:
7 1 2
6 2 4
8 3 1

```

5. Kesimpulan

Melalui praktikum ini, konsep Abstract Data Type (ADT) berhasil diterapkan dalam pemrograman, mulai dari penyimpanan data mahasiswa hingga manipulasi array 2D. ADT memungkinkan modularitas dalam program, dimana deklarasi dan implementasi dipisahkan untuk memudahkan pengelolaan dan pemeliharaan program.