

LAPORAN PRAKTIKUM
Modul 3
ABSTRACT DATA TYPE (ADT)



Disusun Oleh:

Rifqi M. Ramdani

2311104044

SE07-02

Dosen :

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

PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
PURWOKERTO
2024

1. Tujuan

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

2. Landasan Teori

Abstract Data Type (ADT)

ADT adalah TYPE dan sekumpulan PRIMITIF (operasi dasar) terhadap TYPE tersebut. Selain itu, dalam sebuah ADT yang lengkap, disertakan pula definisi invarian dari TYPE dan aksioma yang berlaku. ADT merupakan definisi STATIK.

3. Guided

Definisi type dari sebuah ADT dapat mengandung sebuah definisi ADT lain. Misalnya :

3. ADT waktu yang terdiri dari ADT JAM dan ADT DATE

4. Garis terdiri dari duah buah ADT POINT

SEGI4 yang terdiri dari pasangan dua buah POINT (Top,Left) dan (Bottom,Right)

TYPE diterjemahkan menjadi type terdefinisi dalam bahasa yang bersangkutan. Jika dalam bahasa C menggunakan struct PRIMITIF, dalam konteks prosedural, diterjemahkan menjadi fungsi atau prosedur. PRIMITIF dikelompokkan menjadi:

1. Konstruktor/Kreator, pemebentuk nilai type.Semua objek (variabel) bertipe tersebut harus melalui konstruktor. Biasanya namanya diawali Make.
2. Selector, untuk mengakses tipe komponen(biasanya namanya diawali Get).
3. Prosedur pengubah nilai komponen (biasanya namanya diawali Get).
4. Tipe validator komponen, yang dipakai untuk mentest apakah dapat membentuk tipe sesuai dengan batasan.
5. Destruktor/Dealokator yaitu untuk “menghancurkan” nilai objek/variabel (sekaligus memori penyimpanannya).
6. Baca/Tulis, untuk interface dengan input/output device.
7. Operator relasional, terhadap tipe tersebut untuk mendefinisikan lebih besar, lebih kecil, sama dengan dan sebagainya.
8. Aritmatika terhadap tipe tersebut, karena biasanya aritmatika dalam bahasa C hanya terdefinisi untuk bilangan numerik.
9. Konversi dari tipe tersebut ke tipe dasar dan sebaliknya

ADT biasanya diimplementasikan menjadi dua buah modul utama dan 1 modul interface program utama (driver). Dua modul tersebut adalah sebagai berikut:

1. Definisi/Spesifikasi Type dan Primitif/Header fungsi (.h)

- Spesifikasi type sesuai dengan kaidah bahasa yang dipakai
- Spesifikasi dari primitif sesuai dengan kaidah dalam konteks prosedural, yaitu:
 - Fungsi : nama, domain, range, dan prekondisi jika ada
 - Prosedur : Initial state, Final state, dan proses yang dilakukan

2. Body/realisasi dari primitif (.c) Berupa kode program dalam bahasa yang bersangkutan (dalam praktikum ini berarti dengan bahasa C++). Realisasi fungsi dan prosedur harus sedapat mungkin memanfaatkan selector dan konstruktor. Untuk memahami lebih jelas mengenai konsep ADT, perhatikan ilustrasi berikut

Algoritma	C++
Program coba_ADT Type mahasiswa < nim : char[10] nilai1, nilai2 : integer Kamus mhs : mahasiswa procedure inputMhs(input/output m : mahasiswa) function rata2(input: m : mahasiswa) : real Algoritma inputMhs(mhs) output(rata2(mhs)) procedure inputMhs(input/output m : mahasiswa) kamus algoritma input(m.nim, m.nilai1, m.nilai2) function rata2(input: m : mahasiswa) : real kamus algoritma → (m.nilai1 + m.nilai2) / 2	<pre> #include <iostream> #include <conio.h> #include <stdlib.h> 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 nilai2 = "; cin >> (m).nilai2; } float rata2(mahasiswa m){ return (m.nilai1+m.nilai2)/2; } </pre> <div style="position: absolute; top: 380px; left: 810px; border: 1px solid black; padding: 5px;"> Definisi/ Spesifikasi Type dan Primitif / Header fungsi (.h) </div> <div style="position: absolute; top: 600px; left: 850px; border: 1px solid black; padding: 5px;"> Body/ relisasi dari primitif (.c) </div>

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.

Algoritma	C++
Program coba_ADT Type mahasiswa < nim : char[10] nilai1, nilai2 : integer Kamus mhs : mahasiswa procedure inputMhs(i/o m : mahasiswa) function rata2(input: m : mahasiswa) : real Algoritma inputMhs(mhs) output(rata2(mhs)) procedure inputMhs(input/output m : mahasiswa) kamus algoritma input(m.nim, m.nilai1, m.nilai2)	mahasiswa.h
	<pre> #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 // MAHASISWA_H_INCLUDED </pre>
	mahasiswa.cpp
	<pre> void inputMhs(mahasiswa &m){ cout << "input nama = "; cin >> (m).nim; cout << "input nilai = "; cin >> (m).nilai1; cout << "input nilai2 = "; cin >> (m).nilai2; } </pre>
<pre> function rata2(input: m : mahasiswa) : real kamus algoritma → (m.nilai1 + m.nilai2) / 2 </pre>	<pre> float rata2(mahasiswa m){ return (m.nilai1+m.nilai2)/2; } </pre>
	main.cpp
	<pre> #include <iostream> #include <conio.h> #include <stdlib.h> #include "mahasiswa.cpp" using namespace std; int main() { mahasiswa mhs; inputMhs(mhs); cout << "rata-rata = " << rata2(mhs); return 0; } </pre>

Latihan praktikum

```
main.cpp X main.cpp X main.cpp X pelajaran.cpp X pelajaran.h X main.cpp X
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
```

Maka akan menghasilkan output

```
"D:\TUGAS SEMESTER 3\guide" X + v
input nim = 2311104044
input nilai = 90
input nilai = 86
rata-rata = 88
Process returned 0 (0x0)   execution time : 23.143 s
Press any key to continue.
|
```

4. Unguided

1. 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 $0.3*uts+0.4*uas+0.3*tugas$

JAWAB

```

*main.cpp x
1  #include <iostream>
2  #include <string>
3
4  using namespace std;
5
6  struct Mahasiswa {
7      string nama;
8      string nim;
9      float uts;
10     float uas;
11     float tugas;
12     float nilaiAkhir;
13 };
14
15 float hitungNilaiAkhir(float uts, float uas, float tugas) {
16     return (0.3 * uts) + (0.4 * uas) + (0.3 * tugas);
17 }
18
19 void tambahMahasiswa(Mahasiswa mahasiswa[], int &jumlahMahasiswa) {
20     if (jumlahMahasiswa < 10) {
21         cout << "Masukkan nama mahasiswa: ";
22         cin >> mahasiswa[jumlahMahasiswa].nama;
23         cout << "Masukkan NIM mahasiswa: ";
24         cin >> mahasiswa[jumlahMahasiswa].nim;
25         cout << "Masukkan nilai UTS: ";
26         cin >> mahasiswa[jumlahMahasiswa].uts;
27         cout << "Masukkan nilai UAS: ";
28         cin >> mahasiswa[jumlahMahasiswa].uas;
29         cout << "Masukkan nilai Tugas: ";
30         cin >> mahasiswa[jumlahMahasiswa].tugas;
31
32         mahasiswa[jumlahMahasiswa].nilaiAkhir = hitungNilaiAkhir(
33             mahasiswa[jumlahMahasiswa].uts,
34             mahasiswa[jumlahMahasiswa].uas,
35             mahasiswa[jumlahMahasiswa].tugas
36         );
37
38         cout << "Data mahasiswa berhasil ditambahkan!\n";
39         jumlahMahasiswa++;

```

Maka akan menghasilkan output

```
"D:\TUGAS SEMESTER 3\UNGI" × + v
1. Tambah Data Mahasiswa
2. Tampilkan Data Mahasiswa
3. Keluar
Pilih menu: 1
Masukkan nama mahasiswa: RAMDAN
Masukkan NIM mahasiswa: 2311104044
Masukkan nilai UTS: 90
Masukkan nilai UAS: 85
Masukkan nilai Tugas: 90
Data mahasiswa berhasil ditambahkan!

Menu:
1. Tambah Data Mahasiswa
2. Tampilkan Data Mahasiswa
3. Keluar
Pilih menu: 2
Mahasiswa 1:
Nama      : RAMDAN
NIM       : 2311104044
UTS       : 90
UAS       : 85
Tugas     : 90
Nilai Akhir : 88
-----

Menu:
1. Tambah Data Mahasiswa
2. Tampilkan Data Mahasiswa
3. Keluar
Pilih menu:
```


2.

2. Buatlah ADT pelajaran sebagai berikut di dalam file "pelajaran.h":

```
type pelajaran <
    namaMapel : string
    kodeMapel : string
>
fungsi create_pelajaran( namapel : string, kodepel : string ) →
    pelajaran
prosedur tampil_pelajaran( pel : pelajaran )
```

Buatlah implementasi ADT pelajaran pada file "pelajaran.cpp"

Cobalah hasil implementasi ADT pada file "main.cpp"

```
using namespace std;
int main() {
    string namapel = "Struktur Data";
    string kodepel = "STD";
    pelajaran pel = create_pelajaran(namapel, kodepel);
    tampil_pelajaran(pel);

    return 0;
}
```

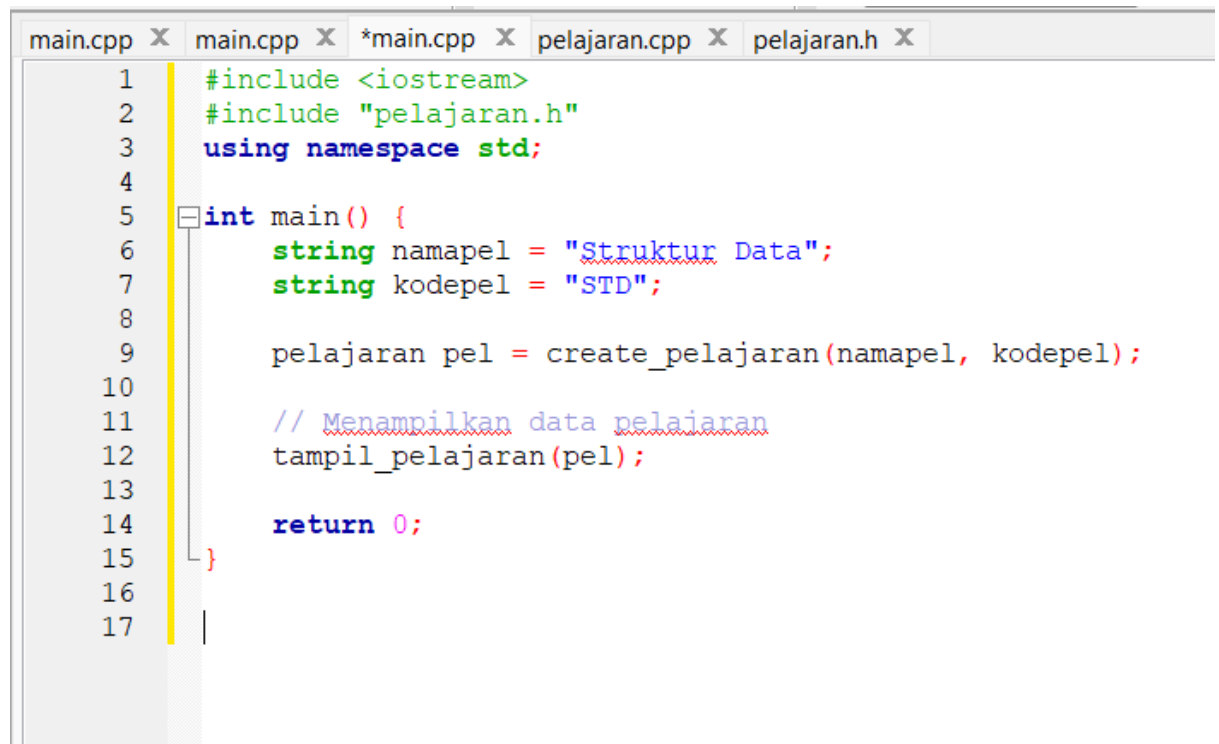
Gambar 3-1 Main.cpp pelajaran

Contoh output hasil:

```
nama pelajaran : Struktur Data
nilai : STD
```

JAWAB

Main.cpp



```
main.cpp X main.cpp X *main.cpp X pelajaran.cpp X pelajaran.h X
1  #include <iostream>
2  #include "pelajaran.h"
3  using namespace std;
4
5  int main() {
6      string namapel = "Struktur Data";
7      string kodepel = "STD";
8
9      pelajaran pel = create_pelajaran(namapel, kodepel);
10
11     // Menampilkan data pelajaran
12     tampil_pelajaran(pel);
13
14     return 0;
15 }
16
17
```

Pelajaran.cpp

```
main.cpp X main.cpp X *main.cpp X pelajaran.cpp X pelajaran.h X
1  #include "pelajaran.h"
2  #include <iostream>
3  using namespace std;
4
5  // Implementasi fungsi create pelajaran
6  pelajaran create_pelajaran(string namapel, string kodepel) {
7      pelajaran pel;
8      pel.namaMapel = namapel;
9      pel.kodeMapel = kodepel;
10     return pel;
11 }
12
13 // Implementasi prosedur tampil pelajaran
14 void tampil_pelajaran(pelajaran pel) {
15     cout << "Nama pelajaran: " << pel.namaMapel << endl;
16     cout << "Kode pelajaran: " << pel.kodeMapel << endl;
17 }
18
```

Pelajaran.h

```
main.cpp X main.cpp X *main.cpp X pelajaran.cpp X pelajaran.h X
1  #ifndef PELAJARAN_H_INCLUDED
2  #define PELAJARAN_H_INCLUDED
3
4  #include <string>
5  using namespace std;
6
7  // Definisi tipe pelajaran
8  struct pelajaran {
9      string namaMapel;
10     string kodeMapel;
11 };
12
13 // Fungsi untuk membuat pelajaran
14 pelajaran create_pelajaran(string namapel, string kodepel);
15
16 // Prosedur untuk menampilkan pelajaran
17 void tampil_pelajaran(pelajaran pel);
18
19 #endif
20
```

Maka akan menghasilkan output

```
D:\TUGAS SEMESTER 3\UNGUIDEDNO2.1SDP3>program  
Nama pelajaran: Struktur Data  
Kode pelajaran: STD
```

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
- STRUKTUR DATA 46
- fungsi/prosedur yang akan menukarkan isi dari variabel yang ditunjuk oleh 2 buah pointer

JAWAB

```
main.cpp X *main.cpp X
1  #include <iostream>
2
3  using namespace std;
4
5  void tampilkanArray(int arr[3][3]) {
6      for (int i = 0; i < 3; i++) {
7          for (int j = 0; j < 3; j++) {
8              cout << arr[i][j] << " ";
9          }
10         cout << endl;
11     }
12 }
13
14 void tukarArray(int arr1[3][3], int arr2[3][3], int row, int col) {
15     int temp = arr1[row][col];
16     arr1[row][col] = arr2[row][col];
17     arr2[row][col] = temp;
18 }
19
20 void tukarPointer(int* ptr1, int* ptr2) {
21     int temp = *ptr1;
22     *ptr1 = *ptr2;
23     *ptr2 = temp;
24 }
25
26 int main() {
27
28     int array1[3][3] = {
29         {1, 2, 3},
30         {4, 5, 6},
31         {7, 8, 9}
32     };
33
34     int array2[3][3] = {
35         {9, 8, 7},
36         {6, 5, 4},
37         {3, 2, 1}
38     };
39 }
```

Maka akan menghasilkan output

```
"D:\TUGAS SEMESTER 3\UNGI" x + v
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 posisi [1][1]:
1 2 3
4 5 6
7 8 9

Array 2 setelah pertukaran posisi [1][1]:
9 8 7
6 5 4
3 2 1

Sebelum pertukaran pointer:
Pointer 1 menunjuk nilai: 100
Pointer 2 menunjuk nilai: 200

Setelah pertukaran pointer:
Pointer 1 menunjuk nilai: 200
Pointer 2 menunjuk nilai: 100

Process returned 0 (0x0)   execution time : 0.197 s
Press any key to continue.
|
```

5. Kesimpulan

Dalam praktikum ini, kami telah mempelajari dan mengimplementasikan konsep Abstract Data Type (ADT) dalam pemrograman C++. Kami berhasil membuat program yang menyimpan data mahasiswa, serta menghitung nilai akhir berdasarkan rumus tertentu. Selain itu, kami juga melakukan implementasi ADT dengan mendefinisikan tipe data dan operasinya dalam program. Melalui praktikum ini, kami memahami pentingnya penggunaan ADT dalam menyusun program yang terstruktur dan efisien, serta bagaimana ADT dapat mempermudah pengelolaan data dalam aplikasi yang kompleks.