

**LAPORAN PRAKTIKUM**  
**PERTEMUAN 3**  
**ABSTRACT DATA TYPE**



**Nama :**

Tsaqif Hisyam Saputra (2311104024)

**Dosen :**

Yudha Islami Sulistiya

**PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK**  
**FAKULTAS INFORMATIKA**  
**TELKOM UNIVERSITY PURWOKERTO**  
**2024**

Tugas Pendahuluan Modul 3  
STRUKTUR DATA - Ganjil 2024/2025  
"Abstract Data Type"

A. Ketentuan Tugas Pendahuluan

1. Tugas Pendahuluan dikerjakan secara **Individu**.
2. TP ini bersifat **WAJIB**, tidak mengerjakan = **PENGURANGAN POIN JURNAL / TES ASESMEN**.
3. Hanya **MENGUMPULKAN** tetapi **TIDAK MENGERJAKAN** = **PENGURANGAN POIN JURNAL / TES ASESMEN**.
4. Deadline pengumpulan TP Modul 2 adalah Senin, 23 September 2024 pukul 06.00 WIB.
5. **TIDAK ADA TOLERANSI KETERLAMBATAN, TERLAMBAT ATAU TIDAK MENGUMPULKAN TP MAKA DIANGGAP TIDAK MENGERJAKAN**.
6. **DILARANG PLAGIAT (PLAGIAT = E)**.
7. Kerjakan TP dengan jelas agar dapat dimengerti.
8. File diupload di LMS menggunakan format **PDF** dengan ketentuan: **TP\_MOD\_[XX]\_NIM\_NAMA.pdf**

**CP (WA):**

- Andini (082243700965)
- Imelda (082135374187)

**SELAMAT MENGERJAKAN^^**

## 1. Tugas Soal Latihan Modul 3

```
1 #include <iostream>
2 #include <string>
3 using namespace std;
4
5 // Struktur untuk menyimpan data mahasiswa
6 struct Mahasiswa {
7     string nama;
8     string nim;
9     float uts;
10    float uas;
11    float tugas;
12    float nilaiAkhir;
13 };
14
15 // Fungsi untuk menghitung nilai akhir
16 float hitungNilaiAkhir(float uts, float uas, float tugas) {
17     return (0.3 * uts) + (0.4 * uas) + (0.3 * tugas);
18 }
19
20 int main() {
21     const int MAX_MAHASISWA = 10;
22     Mahasiswa mahasiswa[MAX_MAHASISWA];
23     int jumlahMahasiswa;
24
25     cout << "Masukkan jumlah mahasiswa (maks 10): ";
26     cin >> jumlahMahasiswa;
27
28     // Validasi jumlah mahasiswa
29     if (jumlahMahasiswa > MAX_MAHASISWA || jumlahMahasiswa <= 0) {
30         cout << "Jumlah mahasiswa tidak valid!" << endl;
31         return 1;
32     }
33
34     // Input data mahasiswa
35     for (int i = 0; i < jumlahMahasiswa; i++) {
36         cout << "\nMahasiswa ke-" << (i + 1) << endl;
37         cout << "Nama: ";
38         cin.ignore();
39         getline(cin, mahasiswa[i].nama);
40         cout << "NIM: ";
41         cin >> mahasiswa[i].nim;
42         cout << "Nilai UTS: ";
43         cin >> mahasiswa[i].uts;
44         cout << "Nilai UAS: ";
45         cin >> mahasiswa[i].uas;
46         cout << "Nilai Tugas: ";
47         cin >> mahasiswa[i].tugas;
48
49         // Hitung nilai akhir
50         mahasiswa[i].nilaiAkhir = hitungNilaiAkhir(mahasiswa[i].uts, mahasiswa[i].uas, mahasiswa[i].tugas);
51     }
52
53     // Tampilkan data mahasiswa
54     cout << "\nData Mahasiswa:\n";
55     for (int i = 0; i < jumlahMahasiswa; i++) {
56         cout << "Mahasiswa ke-" << (i + 1) << endl;
57         cout << "Nama      : " << mahasiswa[i].nama << endl;
58         cout << "NIM       : " << mahasiswa[i].nim << endl;
59         cout << "Nilai UTS  : " << mahasiswa[i].uts << endl;
60         cout << "Nilai UAS  : " << mahasiswa[i].uas << endl;
61         cout << "Nilai Tugas : " << mahasiswa[i].tugas << endl;
62         cout << "Nilai Akhir : " << mahasiswa[i].nilaiAkhir << endl;
63     }
64
65     return 0;
66 }
67
```

Kodingan tersebut dapat digunakan untuk menyimpan data mahasiswa dan menghitung nilai akhir dengan memanfaatkan struktur (struct) atau kelas (class) untuk merepresentasikan setiap mahasiswa, yang mencakup atribut seperti nama, NIM, nilai UTS, UAS, dan tugas. Program ini dapat memiliki fungsi yang menghitung nilai akhir berdasarkan formula tertentu, contoh:  $\text{nilai\_akhir} = 0.3 * \text{uts} + 0.4 * \text{uas} + 0.3 * \text{tugas}$ . Setiap data mahasiswa disimpan dalam array atau vector untuk menampung banyak mahasiswa, kemudian fungsi digunakan untuk menghitung dan menampilkan nilai akhir setiap mahasiswa berdasarkan data yang ada. Dengan demikian, program ini dapat mengelola data mahasiswa secara efisien dan memberikan hasil berupa nilai akhir dari setiap mahasiswa yang sudah tersimpan.

Dan berikut adalah output dari program tersebut:

Mahasiswa ke-1

Nama: Joe

NIM: 2311104001

Nilai UTS: 80

Nilai UAS: 85

Nilai Tugas: 90

Mahasiswa ke-2

Nama: Anna

NIM: 2311104002

Nilai UTS: 85

Nilai UAS: 90

Nilai Tugas: 95

Data Mahasiswa:

Mahasiswa ke-1

Nama : Joe

NIM : 2311104001

Nilai UTS : 80

Nilai UAS : 85

Nilai Tugas : 90

Nilai Akhir : 85

Mahasiswa ke-2

Nama : Anna

NIM : 2311104002

Nilai UTS : 85

Nilai UAS : 90

Nilai Tugas : 95

Nilai Akhir : 90

## main.cpp

```
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     tampil_pelajaran(pel);
11
12     return 0;
13 }
```

## pelajaran.cpp

```
main.cpp x pelajaran.cpp x pelajaran.h x
1  #include "pelajaran.h"
2  #include <iostream>
3  using namespace std;
4
5  // Implementasi fungsi untuk membuat data pelajaran
6  pelajaran create_pelajaran(string namaMapel, string kodeMapel) {
7      pelajaran pel;
8      pel.namaMapel = namaMapel;
9      pel.kodeMapel = kodeMapel;
10     return pel;
11 }
12
13 // Implementasi prosedur untuk menampilkan data pelajaran
14 void tampil_pelajaran(pelajaran pel) {
15     cout << "nama pelajaran : " << pel.namaMapel << endl;
16     cout << "nilai : " << pel.kodeMapel << endl;
17 }
```

## pelajaran.h

```
main.cpp x pelajaran.cpp x pelajaran.h x
1  #ifndef PELAJARAN_H
2  #define PELAJARAN_H
3
4  #include <string>
5  using namespace std;
6
7  struct pelajaran {
8      string namaMapel;
9      string kodeMapel;
10 };
11
12 // Deklarasi fungsi dan prosedur
13 pelajaran create_pelajaran(string namaMapel, string kodeMapel);
14 void tampil_pelajaran(pelajaran pel);
15
16 #endif
```

**pelajaran.h** berisi deklarasi struct pelajaran, fungsi create\_pelajaran untuk membuat data pelajaran, dan prosedur tampil\_pelajaran untuk menampilkan data.

**pelajaran.cpp** berisi implementasi fungsi create\_pelajaran yang mengisi namaMapel dan kodeMapel, dan prosedur tampil\_pelajaran yang menampilkan data pelajaran ke layar.

**main.cpp** adalah program utama yang membuat objek pelajaran, mengisi data, lalu memanggil fungsi untuk menampilkan data.

```

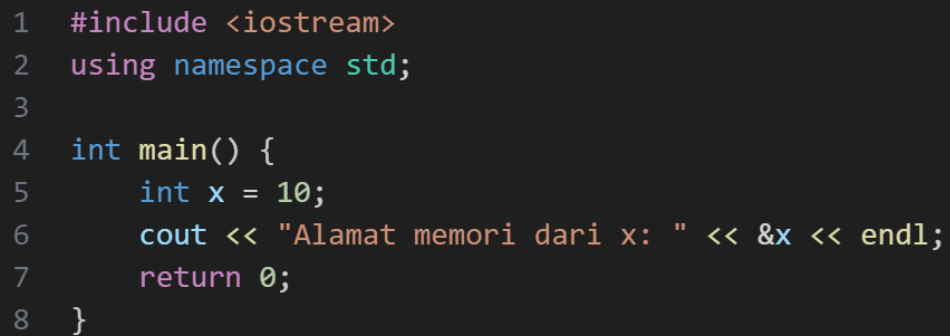
1  #include <iostream>
2  using namespace std;
3
4  // Deklarasi ukuran array
5  const int SIZE = 3;
6
7  // Fungsi untuk menampilkan isi array 2D
8  void tampilArray(int arr[SIZE][SIZE]) {
9      for (int i = 0; i < SIZE; i++) {
10         for (int j = 0; j < SIZE; j++) {
11             cout << arr[i][j] << " ";
12         }
13         cout << endl;
14     }
15 }
16
17 // Fungsi untuk menukarkan nilai pada posisi tertentu antara dua array 2D
18 void tukarArray(int arr1[SIZE][SIZE], int arr2[SIZE][SIZE], int row, int col) {
19     int temp = arr1[row][col];
20     arr1[row][col] = arr2[row][col];
21     arr2[row][col] = temp;
22 }
23
24 // Fungsi untuk menukarkan nilai yang ditunjuk oleh dua pointer integer
25 void tukarPointer(int* ptr1, int* ptr2) {
26     int temp = *ptr1;
27     *ptr1 = *ptr2;
28     *ptr2 = temp;
29 }
30
31 int main() {
32     // Deklarasi dua array 2D berukuran 3x3
33     int array1[SIZE][SIZE] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
34     int array2[SIZE][SIZE] = {{9, 8, 7}, {6, 5, 4}, {3, 2, 1}};
35
36     // Deklarasi dua pointer integer
37     int a = 10, b = 20;
38     int* ptr1 = &a;
39     int* ptr2 = &b;
40
41     // Tampilkan array sebelum ditukar
42     cout << "Array 1 sebelum ditukar:\n";
43     tampilArray(array1);
44     cout << "\nArray 2 sebelum ditukar:\n";
45     tampilArray(array2);
46
47     // Tukar nilai di posisi tertentu pada kedua array
48     tukarArray(array1, array2, 1, 1); // Contoh: menukar elemen pada posisi (1,1)
49
50     // Tampilkan array setelah ditukar
51     cout << "\nArray 1 setelah ditukar:\n";
52     tampilArray(array1);
53     cout << "\nArray 2 setelah ditukar:\n";
54     tampilArray(array2);
55
56     // Tampilkan nilai yang ditunjuk oleh pointer sebelum ditukar
57     cout << "\nNilai sebelum ditukar:\n";
58     cout << "a = " << a << ", b = " << b << endl;
59
60     // Tukar nilai dari variabel yang ditunjuk oleh pointer
61     tukarPointer(ptr1, ptr2);
62
63     // Tampilkan nilai yang ditunjuk oleh pointer setelah ditukar
64     cout << "\nNilai setelah ditukar:\n";
65     cout << "a = " << a << ", b = " << b << endl;
66
67     return 0;
68 }

```

Kodingan tersebut menggunakan array 2D integer berukuran 3x3 ini melibatkan beberapa fungsi untuk memanipulasi data di dalam array tersebut. Pertama, terdapat fungsi `tampilArray` yang digunakan untuk menampilkan semua elemen di dalam array 2D dalam format matriks. Kemudian, ada fungsi `tukarArray` yang memungkinkan pertukaran nilai di posisi tertentu antara dua array 2D yang berbeda, di mana nilai dari posisi yang sama dalam kedua array akan bertukar. Selain itu, fungsi `tukarPointer` yang bertujuan untuk menukar nilai dari dua variabel integer menggunakan pointer, dengan menyimpan nilai sementara dalam variabel sementara. Di dalam fungsi `main`, program mendeklarasikan dua array 2D dan 2 variabel integer yang ditunjuk oleh pointer, lalu menampilkan hasil sebelum dan sesudah melakukan pertukaran data untuk memperlihatkan cara fungsi-fungsi tersebut bekerja dalam mengubah dan menampilkan isi array.

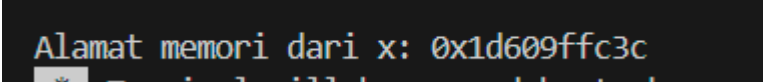
## 2. Jawaban Soal Tugas Pendahuluan Modul 3

1. Pointer adalah sebuah variabel di C++ yang menyimpan alamat memori dari variabel lain. Alih-alih menyimpan nilai data, pointer menyimpan alamat di mana data tersebut berada dalam memori.
2. Untuk menampilkan alamat memori dari suatu variabel, kita bisa menggunakan operator & (address-of) diikuti dengan nama variabelnya. Berikut adalah contoh programnya:



```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int x = 10;
6      cout << "Alamat memori dari x: " << &x << endl;
7      return 0;
8  }
```

Output:



```
Alamat memori dari x: 0x1d609ffc3c
```



3. Untuk menggunakan pointer, pertama-tama kita harus mendeklarasikan pointer dan menginisialisasinya dengan alamat variabel. Setelah itu, kita bisa mengakses nilai dari alamat yang ditunjuk pointer dengan menggunakan operator \*.

Berikut contohnya:

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int x = 20;
6      int* ptr = &x; // pointer yang menyimpan alamat dari x
7
8      // Menampilkan alamat yang disimpan dalam pointer
9      cout << "Alamat dari x: " << ptr << endl;
10
11     // Menampilkan nilai yang ditunjuk oleh pointer
12     cout << "Nilai yang ditunjuk oleh ptr: " << *ptr << endl;
13     return 0;
14 }
```

Outputnya:

```
Alamat dari x: 0xff1c9ffe54
Nilai yang ditunjuk oleh ptr: 20
```

4. Abstract Data Type adalah model atau konsep data yang mendefinisikan sebuah tipe data berdasarkan perilaku (operasi) yang dapat dilakukan terhadapnya, tanpa mendetailkan implementasi internalnya. ADT lebih fokus pada fungsi atau operasi yang disediakan, bukan pada bagaimana fungsi tersebut diimplementasikan. Contoh ADT adalah Stack, Queue, dan List.
5. Contoh ADT di dunia nyata adalah lemari arsip. Lemari arsip memiliki operasi untuk menyimpan berkas, mengambil berkas, dan menghapus berkas. Anda tidak perlu tahu bagaimana lemari itu bekerja di dalamnya (mungkin menggunakan laci, rak, atau sistem lainnya). Anda hanya perlu mengetahui fungsi-fungsi yang disediakan oleh lemari tersebut.

6.

```
1  #include <iostream>
2  #include <cmath>
3  using namespace std;
4
5  // Definisi ADT untuk Kerucut
6  struct Kerucut {
7      double radius;
8      double tinggi;
9  };
10
11 // Fungsi untuk menghitung luas permukaan kerucut
12 double hitungLuasPermukaan(const Kerucut& k) {
13     double s = sqrt((k.radius * k.radius) + (k.tinggi * k.tinggi)); // garis pelukis
14     return M_PI * k.radius * (k.radius + s);
15 }
16
17 // Fungsi untuk menghitung volume kerucut
18 double hitungVolume(const Kerucut& k) {
19     return (1.0 / 3.0) * M_PI * k.radius * k.radius * k.tinggi;
20 }
21
22 int main() {
23     Kerucut k;
24
25     // Input data kerucut
26     cout << "Masukkan radius kerucut: ";
27     cin >> k.radius;
28     cout << "Masukkan tinggi kerucut: ";
29     cin >> k.tinggi;
30
31     // Menghitung dan menampilkan luas permukaan dan volume
32     double luasPermukaan = hitungLuasPermukaan(k);
33     double volume = hitungVolume(k);
34
35     cout << "Luas Permukaan Kerucut: " << luasPermukaan << endl;
36     cout << "Volume Kerucut: " << volume << endl;
37
38     return 0;
39 }
```

```
Masukkan radius kerucut: 7
Masukkan tinggi kerucut: 9
Luas Permukaan Kerucut: 404.676
Volume Kerucut: 461.814
```