Telkom University Purwokerto

Aturan Praktikum Struktur Data

1. **Akun GitHub:** Setiap praktikan wajib memiliki akun GitHubyang aktif dan digunakan selama praktikum berlangsung.
2. **Invite Collaborator:** Setiap praktikan diwajibkan untuk menambahkan collaborator di setiap repository
    a. Asisten Praktikum: AndiniNH
    b. AsistenPraktikum:4ldiputra
3. **Repository Praktikum:** Setiap praktikan diwajibkan untuk membuat satu repository di GitHub yang akan digunakan untuk seluruh tugas dan laporan praktikum. Repository ini harus diatur dengan rapi dan sesuai dengan instruksi yang akan diberikan di lampiran.
4. **Penamaan Folder:** Penamaan folderdalam repositoryakan dibahas secara rinci di lampiran. Praktikan wajib mengikuti aturan penamaan yang telah ditentukan.

Nomor	Pertemuan	Penamaan
1	Penganalan Bahasa C++ Bagian Pertama	01_Pengenalan_CPP_Bagian_1
2	Pengenalan Bahasa C++ Bagian Kedua	02_Pengenalan_CPP_Bagian_2
3	Abstract Data Type	03_Abstract_Data_Type
4	Single Linked List Bagian Pertama	04_Single_Linked_List_Bagian_1
5	Single Linked List Bagian Kedua	05_Single_Linked_List_Bagian_2
6	Double Linked List Bagian Pertama	06_Double_Linked_List_Bagian_1
7	Stack	07_Stack
8	Queue	08_Queue
9	Assessment Bagian Pertama	09_Assessment_Bagian_1
10	Tree Bagian Pertama	10_Tree_Bagian_1
11	Tree Bagian Kedua	11_Tree_Bagian_2
12	Asistensi Tugas Besar	12_Asistensi_Tugas_Besar
13	Multi Linked List	13_Multi_Linked_List
14	Graph	14_Graph
15	Assessment Bagian Kedua	15_Assessment_Bagian_2
16	Tugas Besar	16_Tugas_Besar

5. **Jam Praktikum:**
    - Jam masuk praktikum adalah **1 jam lebih lambat** dari jadwal yang tercantum. Sebagai contoh, jika jadwalpraktikumadalah pukul 06.30 - 09.30, maka aturan praktikum akan diatur sebagai berikut:
        - **06.30 - 07.30**: Waktuini digunakan untuk **Tugas Praktikum dan Laporan Praktikum** yang dilakukan di luar laboratorium.
        - **07.30 - 0G.30**: Sesi ini mencakup **tutorial, diskusi, dan kasus problem- solving**. Kegiatan ini berlangsung di dalam laboratorium dengan alokasi waktu sebagai berikut:
            - **60 menit pertama**: Tugas terbimbing.
            - **60 menit kedua**: Tugas mandiri.
6. **Pengumpulan Tugasn Pendahuluan:** Tugas Pendahuluan (TP) wajib dikumpulkan melalui GitHub sesuai dengan format berikut:

    nama_repo/nama_pertemuan/TP_Pertemuan_Ke.md

    Sebagai contoh:

    STD_Yudha_Islalmi_Sulistya_XXXXXXXX/01_Running_Modul/TP_01.md

7. **Pengecekan Tugas Pendahuluan:** Pengumpulan laporan praktikum akan diperiksa **1 hari sebelum praktikum selanjutnya** dimulai. Pastikan tugas telah diunggah tepat waktu untuk menghindari sanksi.

LAPORAN PRAKTIKUM
MODUL 3
ABSTRACT DATA TYPE (ADT)

Disusun Oleh :

Zaenarif Putra ‘Ainurdin – 2311104049

Kelas :

SE-07-02

Dosen :

Wahyu Andi Saputra, S.pd,M.Eng

PROGRAM STUDI SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
PURWOKERTO

2024

I. TUJUAN

1. Memahami konsep Abstract Data Type (ADT) dan penggunaannya dalam program

II. LANDASAN TEORI

1. 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. Definisi type dari sebuah ADT dapat mengandung sebuah definisi ADT lain. Misalnya : ADT waktu yang terdiri dari ADT JAM dan ADT DATE Garis terdiri dari duah buah ADT POINT.

III. GUIDE

1. Abstract Data Type

   a. Syntax

```cpp
#include <iostream>

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 << "masukkan nim : ";
    cin >> (m).nim;
    cout << "masukkan nilai : ";
    cin >> (m).nilai1;
    cout << "masukkan nilai : ";
    cin >> (m).nilai2;
}

float rata2(mahasiswa m) {
    return (m.nilai1 + m.nilai2) / 2.0;
}

```

   b. Penjelasan Syntax

   struct mahasiswa { char nim[10]; int nilai1, nilai2; }; Struct ini menyimpan NIM dan dua nilai mahasiswa.

   void inputMhs(mahasiswa &m) { cin >> m.nim >> m.nilai1 >> m.nilai2; } Fungsi ini meminta pengguna memasukkan NIM, nilai1, dan nilai2.

float rata2(mahasiswa m) { return (m.nilai1 + m.nilai2) / 2.0; } Fungsi ini menghitung rata-rata dua nilai mahasiswa.

int main() { mahasiswa mhs; inputMhs(mhs); cout << "rata-rata = " << rata2(mhs); } Fungsi main memanggil inputMhs untuk menerima input dan rata2 untuk menghitung rata-rata, lalu menampilkannya.

c. Output

```
masukkan nim : 2311104049
masukkan nilai : 90
masukkan nilai : 88
rata-rata = 89
PS C:\Users\LENOVO\OneDrive - Telkom University\Documents\ALL Matkul\StrukturData\pertemuan3>
```

IV. UNGUIDED

1. Task 1

Pada soal task1 berikut tugas yang diberikan yaitu mengelola data mahasiswa yang dimana dimasukkan pada sebuah *array* dengan ketentuan *field* yaitu nama, nim, uts, uas, tugas, dan nilai akhir. Yang dimana nilai akhir diperoleh dari sebuah Fungsi yang dimana rumusnya adalah 0.3 * uts + 0.4 * uas + 0.3 * tugas. Kemudian akan memunculkan nilai dari operasi yang sudah ditentukan. Berikut adalah bukti dari syntax, penjelasan, dan outputnya :

a. Syntax

```cpp
#include <iostream>
#include <string>

using namespace std;

struct Mhs {
    string nama;
    string nim;
    float uts;
    float uas;
    float tugas;
    float nilaiakhir;
};

float hitungNilaiAkhir(float uts, float uas, float tugas) {
    return 0.3 * uts + 0.4 * uas + 0.3 * tugas;
}

int main() {
    int maksMhs;
    cout << "Masukkan jumlah Mahasiswa yang ingin diinputkan (maksimal 10): ";
    cin >> maksMhs;

    if (maksMhs > 10) {
        cout << "Alert!!!, Terlalu Banyak Mahasiswa. Jumlah maksimal adalah 10." << endl;
        return 0;
    }

    Mhs mahasiswa[maksMhs];
    int jmlMhs = 0;

    while (jmlMhs < maksMhs) {
        cout << "Masukkan Nama Mahasiswa " << jmlMhs + 1 << ": ";
        cin >> mahasiswa[jmlMhs].nama;

        if (mahasiswa[jmlMhs].nama == "Selesai") {
            break;
        }

        cout << "Masukkan NIM Mahasiswa " << jmlMhs + 1 << ": ";
        cin >> mahasiswa[jmlMhs].nim;
        cout << "Masukkan nilai UTS Mahasiswa " << jmlMhs + 1 << ": ";
        cin >> mahasiswa[jmlMhs].uts;
        cout << "Masukkan nilai UAS Mahasiswa " << jmlMhs + 1 << ": ";
        cin >> mahasiswa[jmlMhs].uas;
        cout << "Masukkan nilai Tugas Mahasiswa " << jmlMhs + 1 << ": ";
        cin >> mahasiswa[jmlMhs].tugas;

        mahasiswa[jmlMhs].nilaiakhir = hitungNilaiAkhir(mahasiswa[jmlMhs].uts, mahasiswa[jmlMhs].uas, mahasiswa[jmlMhs].tugas);

        jmlMhs++;
    }

    cout << "\nData Mahasiswa :\n";
    for (int i = 0; i < jmlMhs; i++) {
        cout << "Nama : " << mahasiswa[i].nama << endl;
        cout << "NIM : " << mahasiswa[i].nim << endl;
        cout << "Nilai UTS : " << mahasiswa[i].uts << endl;
        cout << "Nilai UAS : " << mahasiswa[i].uas << endl;
        cout << "Nilai Tugas : " << mahasiswa[i].tugas << endl;
        cout << "Nilai Akhir : " << mahasiswa[i].nilaiakhir << endl;
        cout << "*************************\n";
    }

    return 0;
}
```

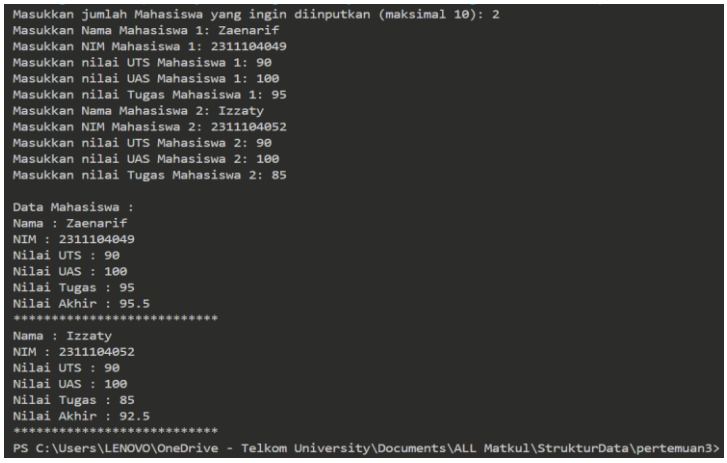
b. Penjelasan Syntax

Struktur data Mhs digunakan untuk menyimpan data mahasiswa. Data yang disimpan adalah: Nama mahasiswa, NIM mahasiswa, Nilai UTS mahasiswa, Nilai UAS mahasiswa, Nilai tugas mahasiswa, Nilai akhir mahasiswa.

Fungsi hitungNilaiAkhir digunakan untuk menghitung nilai akhir mahasiswa. Nilai akhir mahasiswa dihitung dengan menggunakan rumus yang kemudian nantinya akan muncul nilainya : 30% nilai UTS, 40% nilai UAS, 30% nilai tugas.

Fungsi main() {...} pada syntax task1 adalah fungsi utama program yang digunakan untuk menjalankan program. Fungsi ini memungkinkan pengguna untuk memasukkan data mahasiswa, menghitung nilai akhir, dan menampilkan data mahasiswa.

c. Output



2. Task 2

Pada Task2 membuat sebuah program yang dimana program tersebut bernama pelajaran dan memiliki 2 variabel yaitu *namamapel* dan *kodepel.* Yang dimana keduanya bertipe *string*, dan juga memiliki dua fungsi yaitu *create_pelajaran* dan *tampil_pelajaran.* Pada syntax task2 ini terdapat dua file yang berbeda yaitu pelajaran.cpp dan main.cpp yang dimana pelajaran.cpp berisi definisi struktur data dan fungsi, sedangkan main.cpp berisi program utama yang menggunakan struktur data dan fungsi tersebut.Berikut terdapat syntax, penjelasan dan juga outputnya :

a. Syntax

Pelajaran.cpp

```cpp
#include <iostream>
#include <string>

using namespace std;

struct pelajaran {
    string namamapel;
    string kodepel;
};

pelajaran create_pelajaran(string namapel, string kodepel) {
    pelajaran pel;
    pel.namamapel = namapel;
    pel.kodepel = kodepel;
    return pel;
}

void tampil_pelajaran(pelajaran pel) {
    cout << "Nama  Pelajaran : " << pel.namamapel << endl;
    cout << "Nilai : " << pel.kodepel << endl;
}
```

Main.cpp

```cpp
#include <iostream>
#include "task2_pelajaran.cpp"

using namespace std;

int main() {
    string namapel = "Struktur Data";
    string kodepel = "STD";
    pelajaran pel = create_pelajaran(namapel, kodepel);
    tampil_pelajaran(pel);
    return 0;
}
```

b. Penjelasan Syntax

* Pelajaran.cpp

struct pelajaran { string namamapel; string kodepel; }; Mendefinisikan sebuah data bernama pelajaran yang memiliki dua variabel anggota: namamapel dan kodepel, keduanya bertipe string.

pelajaran create_pelajaran(string namapel, string kodepel) {...} mendefinisikan sebuah fungsi bernama create_pelajaran yang mengambil dua parameter string, namapel dan kodepel, dan mengembalikan sebuah objek pelajaran. Fungsi ini digunakakan sebagai pembuatan objek pelajaran baru dan menginisialisasi variabel anggotanya dengan parameter yang diberikan.

void tampil_pelajaran(pelajaran pel) { ... } mendefinisikan sebuah fungsi bernama tampil_pelajaran yang mengambil objek pelajaran sebagai parameter dan mencetak nilai variabel anggotanya menggunakan cout. Fungsi ini akan mencetak "Nama Pelajaran" dan "Nilai" ke layar.

* Main.cpp

#include "task2_pelajaran.cpp" Merupakan importan yang menginstruksikan compiler untuk menyertakan file task2_pelajaran.cpp ke dalam program. File ini berisi definisi struktur data pelajaran dan fungsi-fungsi yang terkait.

string namapel = "Struktur Data"; dan string kodepel = "STD"; mendeklarasikan dua variabel string bernama namapel dan kodepel dan menginisialisasi mereka dengan nilai "Struktur Data" dan "STD" respectively.

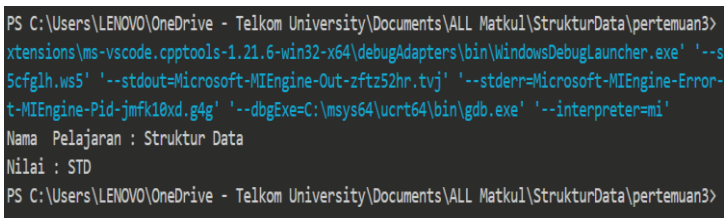
pelajaran pel = create_pelajaran(namapel, kodepel); membuat objek pelajaran baru bernama pel menggunakan fungsi create_pelajaran yang didefinisikan di file task2_pelajaran.cpp. Fungsi ini mengambil dua parameter namapel dan kodepel dan mengembalikan objek pelajaran yang baru dibuat.

tampil_pelajaran(pel); memanggil fungsi tampil_pelajaran yang didefinisikan di file task2_pelajaran.cpp dan mengirimkan objek pel sebagai parameter. Fungsi ini akan mencetak nilai variabel anggota objek pel ke layar.

return 0; mengembalikan nilai 0 ke sistem operasi, menunjukkan bahwa program telah berjalan dengan sukses.

c. Output

Output yang dijalan berada pada file main.cpp yang dimana akan mengoutputkan sebagai berikut :



3. Task 3

Pada task 3 ditugaskan membuat sebuah program dengan ketentuan yaitu :
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, dan fungsi/prosedur yang akan menukarkan isi dari variabel yang ditunjuk oleh 2 buah pointer. Berikut adalah hasil mulai dari syntax, penjelasan syntax dan outputnya.

a. Syntax

```cpp
#include <iostream>

using namespace std;

void menampilkanArray(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 menukarArray(int arr1[3][3], int arr2[3][3], int posisi1[2], int posisi2[2]) {
    int temp = arr1[posisi1[0]][posisi1[1]];
    arr1[posisi1[0]][posisi1[1]] = arr2[posisi2[0]][posisi2[1]];
    arr2[posisi2[0]][posisi2[1]] = temp;
}

void menukarPointer(int* ptr1, int* ptr2) {
    int temp = *ptr1;
    *ptr1 = *ptr2;
    *ptr2 = temp;
}

int main() {
    int arr1[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
    int arr2[3][3] = {{10, 11, 12}, {13, 14, 15}, {16, 17, 18}};

    int var1 = 20;
    int var2 = 30;
    int* ptr1 = &var1;
    int* ptr2 = &var2;

    cout << "Isi Array 1 : " << endl;
    menampilkanArray(arr1);
    cout << "Isi Array 2 : " << endl;
    menampilkanArray(arr2);
    cout << "********************" << endl;

    int posisi1[2] = {1,1};
    int posisi2[2] = {2,2};
    menukarArray(arr1, arr2, posisi1, posisi2);

    cout << "Isi Nilai Array 1 Setelah Ditukar:" << endl;
    menampilkanArray(arr1);
    cout << "Isi Nilai Array 2 Setelah Ditukar:" << endl;
    menampilkanArray(arr2);
    cout << "********************" << endl;

    menukarPointer(ptr1, ptr2);

    cout << "Isi Nilai Variabel 1 Setelah Ditukar: " << *ptr1 << endl;
    cout << "Isi Nilai Variabel 2 Detelah Ditukar: " << *ptr2 << endl;
    cout << "********************" << endl;

    return 0;
}
```

b. Penjelasan Syntax

* void menampilkanArray(int arr[3][3]) {
  for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 3; j++) {
      cout << arr[i][j] << " ";
    }
    cout << endl;
  }
}

Digunakan untuk menampilkan isi array 2D dengan ukuran 3x3. Dengan looping digunakan untuk mengakses setiap elemen pada array.

* void menukarArray(int arr1[3][3], int arr2[3][3], int posisi1[2], int posisi2[2]) {
  int temp = arr1[posisi1[0]][posisi1[1]];
  arr1[posisi1[0]][posisi1[1]] = arr2[posisi2[0]][posisi2[1]];
  arr2[posisi2[0]][posisi2[1]] = temp;
}

Digunakan untuk menukar nilai array pada posisi tertentu antara dua array. Nilai sementara digunakan untuk menyimpan nilai sebelumnya.

```
* void menukarPointer(int* ptr1, int* ptr2) {
    int temp = *ptr1;
    *ptr1 = *ptr2;
    *ptr2 = temp;
}
```

Digunakan untuk menukar nilai yang ditunjuk oleh dua pointer. Yang dimana nlai sementara digunakan untuk menyimpan nilai sebelumnya.

```
* int arr1[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
  int arr2[3][3] = {{10, 11, 12}, {13, 14, 15}, {16, 17, 18}};

  int var1 = 20;
  int var2 = 30;
  int* ptr1 = &var1;
  int* ptr2 = &var2;
```

Digunakan sebagai inisialisasi array 2D, yang dimana *arr1* dan *arr2,* dengan nilai-nilai tertentu, selain itu terdapat inisialisasi dua variabel yaitu *var1* dan *var2,* yang dimana sama dengan array 2D terdapat nilai-nilai tertentu, dan juga terdapat dua pointer yaitu *ptr1* dan *ptr2,* yang menunjuk pada *var1* dan *var2.*

```
* cout << "Isi Array 1 : " << endl;
  menampilkanArray(arr1);
  cout << "Isi Array 2 : " << endl;
  menampilkanArray(arr2);
```

Digunakan untuk menampilkan output dari *arr1* dan *arr2,* yang dimana menggunakan fungsi *menampilkanArray.*

```
* int posisi1[2] = {1, 1};
  int posisi2[2] = {2, 2};
  menukarArray(arr1, arr2, posisi1, posisi2);
```

Digunakan untuk menentukan posisi yang digunakan untuk menukar nilai pada array, yang kemudian ditukar untuk nilai pada posisi *arr1* dan *arr2.* Yang dimana menggunakan fungsi menukarArray.

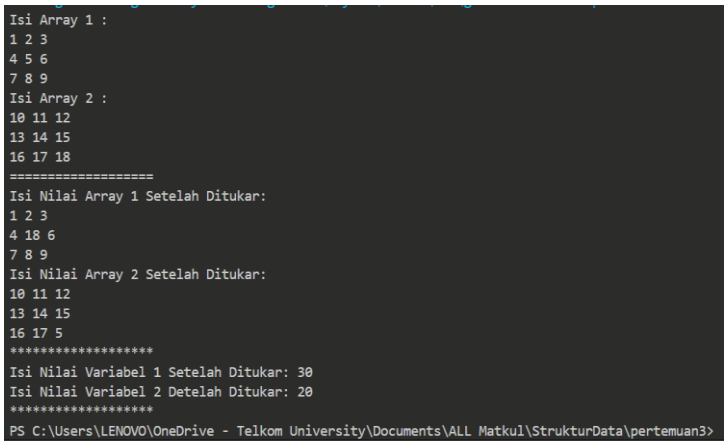
```
* cout << "Isi Nilai Array 1 Setelah Ditukar:" << endl;
  menampilkanArray(arr1);
  cout << "Isi Nilai Array 2 Setelah Ditukar:" << endl;
  menampilkanArray(arr2);
```

Digunakan untuk menampilkan kembali output dari isi *arr1* dan *arr2,* setelah nilai-nilai array ditukar.

* menukarPointer(ptr1, ptr2);
  cout << "Isi Nilai Variabel 1 Setelah Ditukar: " << *ptr1 << endl;
  cout << "Isi Nilai Variabel 2 Setelah Ditukar: " << *ptr2 << endl;

Digunakan untuk menukar nilai yang ditunjuk oleh *ptr1* dan *ptr2* yang dimana menggunakan fungsi *menukarPointer*. Yang kemudian menampilkan output dari nilai variabel yaitu *var1* dan *var2,* setelah menukar nilai-nilai tersebut.

c. Ouput



V. KESIMPULAN

Dengan melakukan praktik dan mengerjakan tugas pada file ini, saya memperoleh pemahaman yang lebih baik tentang konsep Tipe Data Abstract (ADT), yang mencakup bagaimana mendefinisikan tipe data tertentu dan operasi-operasi primitifnya dalam bahasa C++, serta bagaimana mengorganisir kode dengan lebih modular dan mengimplementasikan operasinya dalam file sumber (.cpp). Pengetahuan tentang pentingnya manajemen memori yang aman dan penghindaran risiko seperti buffer overflow diperkuat oleh praktik ini.