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

A. Ketentuan Tugas Pendahuluan

1. Tugas Pendahuluan dikerjakan secara Individu.
2. TP ini bersifat WAJIB, tidak mengerjakan = PENGURANGAN POIN JURNAL / TESASESMEN.
3. Hanya MENGUMPULKAN tetapi TIDAK MENGERJAKAN = PENGURANGAN POINJURNAL / TES ASESMEN.
4. Deadline pengumpulan TP Modul 2 adalah Senin, 30 September 2024 pukul 07.30 WIB.
5. TIDAK ADA TOLERANSI KETERLAMBATAN, TERLAMBAT ATAUTIDAK MENGUMPULKAN TP MAKA DIANGGAP TIDAK MENGERJAKAN.
6. DILARANG PLAGIAT (PLAGIAT = E).
7. Kerjakan TP dengan jelas agar dapat dimengerti.
8. Codingan diupload di Github dan upload Laporan di Labmenggunakan format PDF dengan ketentuan: TP_MOD_[XX]_NIM_NAMA.pdf

CP (WA):

- Andini (082243700965)
- Imelda (082135374187)

SELAMAT MENGERJAKAN^^

LAPORAN PRAKTIKUM
MODUL 3
“ Abstract Data Type (ADT)”

Disusun Oleh:
Shilfi Habibah - 2311104002
S1SE07-01

Dosen :
Yudha Islami Sulistya

PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2024

B. Latihan

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.

   Code:

```
#include <iostream>
#include <conio.h>
#include <stdlib.h>

using namespace std;

struct mahasiswa{
    char nama[50];
    char nim[10];
    int nilaiuts,nilaiuas,tugas;
};

void inputMhs(mahasiswa &m);
float akhir(mahasiswa m);

int main()
{
    mahasiswa mhs;
    inputMhs(mhs);
    cout <<"Nilai Akhir = " <<akhir(mhs);
    return 0;
}

void inputMhs(mahasiswa &m){
    cout << "input Nama = ";
    cin >> (m).nama;
    cout << "input Nim = ";
    cin >> (m).nim;
    cout << "input Nilai UTS = ";
    cin >> (m).nilaiuts;
    cout << "input Nilai UAS = ";
    cin >> (m).nilaiuas;
    cout << "input Nilai Tugas = ";
    cin >> (m).tugas;
}

float akhir(mahasiswa m){
    return (0.4*m.nilaiuas+0.3*m.nilaiuts+0.3*m.tugas);
}
```

- **void inputMhs(mahasiswa &m)**: fungsi untuk memasukkan data siswa, mengambil referensi ke **mahasiswa**struct sebagai argumen
- **float akhir(mahasiswa m)**: fungsi untuk menghitung skor akhir siswa, mengambil **mahasiswa**struct sebagai argumen
- Mendeklarasikan **mahasiswa**variabel struct **mhs**
- Memanggil **inputMhs**fungsi untuk memasukkan data siswa, yang dikirimkan **mhs**sebagai argumen.
- Memanggil **akhir**fungsi untuk menghitung skor akhir siswa, yang dikirimkan **mhs**sebagai argumen.
- Mencetak skor akhir ke konsol menggunakan **cout**
- Menghitung skor akhir siswa menggunakan rumus: **0.4 * nilaiuas + 0.3 * nilaiuts + 0.3 * tugas**

Output:

```
input Nama = Shilfi
input Nim = 2311104002
input Nilai UTS = 95
input Nilai UAS = 90
input Nilai Tugas = 88
Nilai Akhir = 90.9
Process returned 0 (0x0)
Press any key to continue.
```

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

```
tipe 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;
}
```

Code:

```
#include <iostream>
#include <pelajaran.h>
using namespace std;

int main(){
    string namapel =  "Struktur Data";
    string kodepel = "STD";

    pelajaran pel = create_pelajaran(namapel, kodepel);

    tampil_pelajaran (pel);
    return 0;
}
```

Main.cpp

- **<iostream>** : untuk operasi input-output
- **<pelajaran.h>**: file header khusus yang mendefinisikan **pelajaran**struct dan mendeklarasikan fungsi **create_pelajaran**dan **tampil_pelajaran**.

```
#ifndef PELAJARAN_H
#define PELAJARAN_H


#include <string>
using namespace std;

struct pelajaran {
    string namaMapel;
    string kodeMapel;
};

pelajaran create_pelajaran(string namapel, string kodepel);
void tampil_pelajaran(pelajaran pel);

#endif
```

Pelajaran.h

- **namaMapel:** a **string** untuk menyimpan nama pelajaran
- **kodeMapel**: a **string**untuk menyimpan kode pelajaran
- **pelajaran create_pelajaran(string namapel, string kodepel)**: fungsi untuk membuat **pelajaran**objek dengan nama dan kode yang diberikan
- **void tampil_pelajaran(pelajaran pel)**: fungsi untuk menampilkan informasi tentang suatu **pelajaran**objek

```
#include <iostream>
#include "pelajaran.h"

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

void tampil_pelajaran(pelajaran pel) {
    cout << "nama pelajaran : " << pel.namaMapel << endl;
    cout << "nilai : " << pel.kodeMapel << endl;
}
```

Pelajaran.cpp

- Fungsi ini **tampil_pelajaran** mengambil sebuah **pelajaran** objek sebagai argumen dan menampilkan informasi tentang pelajaran

Output:

```
nama pelajaran : Struktur Data
nilai : STD

Process returned 0 (0x0)   exec
Press any key to continue.
```

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

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

Code:

```cpp
#include <iostream>
using namespace std;

// Fungsi untuk menampilkan isi array 2D
void printArray(int arr[3][3]) {
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }
}

// Fungsi untuk menukar elemen pada posisi tertentu dalam dua array 2D
void swapElements(int arr1[3][3], int arr2[3][3], int row, int col) {
    int temp = arr1[row][col];
    arr1[row][col] = arr2[row][col];
    arr2[row][col] = temp;
}

// Fungsi untuk menukar isi dari dua variabel yang ditunjuk oleh dua pointer
void swapPointers(int* ptr1, int* ptr2) {
    int temp = *ptr1;
    *ptr1 = *ptr2;
    *ptr2 = temp;
}
```

```cpp
int main() {
    // Deklarasi dua array 2D berukuran 3x3
    int array1[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
    int array2[3][3] = {{9, 8, 7}, {6, 5, 4}, {3, 2, 1}};

    // Deklarasi dua buah pointer integer
    int a = 100, b = 200;
    int* ptr1 = &a;
    int* ptr2 = &b;

    // Menampilkan isi kedua array sebelum penukaran
    cout << "Array 1 sebelum swap:" << endl;
    printArray(array1);
    cout << "Array 2 sebelum swap:" << endl;
    printArray(array2);

    // Menukar elemen pada posisi (1,1) dari kedua array
    swapElements(array1, array2, 1, 1);

    // Menampilkan isi kedua array setelah penukaran
    cout << "\nArray 1 setelah swap:" << endl;
    printArray(array1);
    cout << "Array 2 setelah swap:" << endl;
    printArray(array2);

    // Menampilkan nilai variabel yang ditunjuk oleh pointer sebelum penukaran
    cout << "\nNilai sebelum swap pointer: ptr1 = " << *ptr1 << ", ptr2 = " << *ptr2 << endl;

    // Menukar isi variabel yang ditunjuk oleh ptr1 dan ptr2
    swapPointers(ptr1, ptr2);

    // Menampilkan nilai variabel yang ditunjuk oleh pointer setelah penukaran
    cout << "Nilai setelah swap pointer: ptr1 = " << *ptr1 << ", ptr2 = " << *ptr2 << endl;

    return 0;
}
```

- **printArray(int arr[3][3])**: Fungsi ini mengambil array bilangan 2D sebagai argumen dan mencetak isinya ke konsol. Fungsi ini menggunakan loop bersarang untuk mengulang array dan mencetak setiap elemen.

- **swapElements(int arr1[3][3], int arr2[3][3], int row, int col)** : Fungsi ini mengambil dua array 2D dan dua indeks (baris dan kolom) sebagai argumen. Fungsi ini menukar elemen pada posisi yang ditentukan di kedua array.

- **swapPointers(int* ptr1, int* ptr2)**: Fungsi ini mengambil dua penunjuk bilangan bulat sebagai argumen dan menukar nilai yang ditunjuknya.

Output:

```
Array 1 sebelum swap:
1 2 3
4 5 6
7 8 9
Array 2 sebelum swap:
9 8 7
6 5 4
3 2 1

Array 1 setelah swap:
1 2 3
4 5 6
7 8 9
Array 2 setelah swap:
9 8 7
6 5 4
3 2 1

Nilai sebelum swap pointer: ptr1 = 100, ptr2 = 200
Nilai setelah swap pointer: ptr1 = 200, ptr2 = 100

Process returned 0 (0x0)   execution time : 0.270 s
Press any key to continue.
```

4. Jelaskan apa yang dimaksud dengan pointer!

   Pointer merupakan tipe data yang berisi alamat memori dari sebuah variabel, untuk lebih mudah memahami ini, Kita akan coba membahas terlebih dahulu bagaimana bahasa pemrograman lain menyimpan nilai dari sebuah variabel. Ketika kita akan mendeklarasikan sebuah variabel (misalkan variabel angka'), bahasa pemrograman akan menyiapkan sebuah tempat di memory komputer. Tempat ini memiliki alamat, yang berfungsi untuk menandai lokasi variabel tersebut.

5. Bagaimana cara menampilkan alamat memori dari suatu variabel dalam program C++ Berikan conttoh!

   Di dalam Bahasa Pemrograman, biasanya kita membutuhkan karakter untuk memanggil alamat memori sebuah variabel, Pada bahasa Pascal menggunakan karakter '@', atau pada bahasa C/C++ menggunakan '&'.

   Contoh :

```cpp
#include <iostream>
using namespace std;

int main() {
    int num = 42;
    int* ptr = &num; // pointer ke variabel num

    // Menampilkan nilai variabel
    cout << "Nilai num: " << num << endl;

    // Menampilkan alamat memori variabel num
    cout << "Alamat memori num: " << &num << endl;

    // Menampilkan alamat memori yang ditunjuk oleh pointer
    cout << "Alamat memori yang ditunjuk oleh ptr: " << ptr << endl;

    return 0;
}
```

```
Nilai num: 42
Alamat memori num: 0x61fe14
Alamat memori yang ditunjuk oleh ptr: 0x61fe14

Process returned 0 (0x0)   execution time : 0.0
Press any key to continue.
```

6. Bagaimana cara menggunakan pointer dalam program C++. Berikan contoh cara menampilkan nilai yang tersimpan pada suata alamat melalui pointer!

   Dalam C++, pointer adalah variabel yang menyimpan alamat memori dari variabel lain. Dengan pointer, kita bisa mengakses dan memodifikasi nilai dari variabel tersebut melalui alamat memorinya. Untuk menampilkan nilai yang tersimpan di alamat memori yang ditunjuk oleh pointer, kita menggunakan operator dereference*.

   Code:

```cpp
#include <iostream>
using namespace std;

int main() {
    int num = 10;       // variabel biasa
    int* ptr = &num;    // pointer menyimpan alamat memori num

    // Menampilkan alamat memori num
    cout << "Alamat memori num: " << ptr << endl;

    // Menampilkan nilai yang tersimpan di alamat yang ditunjuk oleh ptr
    cout << "Nilai yang tersimpan di alamat memori tersebut: " << *ptr << endl;

    return 0;
}
```

   - int* ptr = # menyimpan alamat memori dari variabel num ke pointer ptr.
   - out << *ptr; menampilkan nilai yang tersimpan di alamat yang ditunjuk oleh pointer ptr, yaitu nilai dari variabel num.

   Output:

```
Alamat memori num: 0x61fe14
Nilai yang tersimpan di alamat memori tersebut: 10
```

7. Jelaskan apa yang dimaksud dengan 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.

8. Berikan contoh ilustrasi sederhana di dalam dunia nyata, tetapi di luar konteks pemrograman!

   Salah satu contoh ilustrasi sederhana di dunia nyata yang mirip dengan konsep pemrograman adalah penggunaan **mesin penjual otomatis (vending machine)**:

   1. **Input**: Pengguna memasukkan koin dan memilih minuman yang diinginkan.
   2. **Proses**: Mesin memeriksa apakah jumlah koin yang dimasukkan sesuai dengan harga minuman.
   3. **Output**: Jika sesuai, mesin akan mengeluarkan minuman yang dipilih. Jika tidak, mesin akan menampilkan pesan kesalahan atau mengembalikan uang.

   Ini mirip dengan alur program: pengguna memberikan input, mesin melakukan pemrosesan berdasarkan aturan tertentu, dan hasilnya dikembalikan kepada pengguna dalam bentuk output

9. Tuliskan ADT dari bangun ruang kerucut dalam bahasa C++!

   Code:

```cpp
#include <iostream>
#include <cmath> // untuk fungsi sqrt dan M_PI

class Kerucut {
private:
    double jari_jari; // radius
    double tinggi;    // height

public:
    // Konstruktor untuk inisialisasi jari-jari dan tinggi
    Kerucut(double r, double t) {
        jari_jari = r;
        tinggi = t;
    }

    // Fungsi untuk menghitung garis pelukis
    double hitungGarisPelukis() {
        return sqrt((jari_jari * jari_jari) + (tinggi * tinggi));
    }

    // Fungsi untuk menghitung luas permukaan kerucut
    double hitungLuasPermukaan() {
        double garis_pelukis = hitungGarisPelukis();
        return M_PI * jari_jari * (jari_jari + garis_pelukis);
    }

    // Fungsi untuk menghitung volume kerucut
    double hitungVolume() {
        return (M_PI * jari_jari * jari_jari * tinggi) / 3;
    }

    // Fungsi untuk menampilkan data kerucut
    void tampilkanData() {
        std::cout << "Jari-jari: " << jari_jari << std::endl;
        std::cout << "Tinggi: " << tinggi << std::endl;
        std::cout << "Garis pelukis: " << hitungGarisPelukis() << std::endl;
        std::cout << "Luas permukaan: " << hitungLuasPermukaan() << std::endl;
        std::cout << "Volume: " << hitungVolume() << std::endl;
    }
};

int main() {
    Kerucut kerucut(3, 5); // jari-jari = 3, tinggi = 5
    kerucut.tampilkanData();
    return 0;
}
```

- **ADT Kerucut** terdiri dari atribut jari_jari dan tinggi.
- Metode hitungGarisPelukis() menghitung panjang garis pelukis.
- Metode hitungLuasPermukaan() menghitung luas permukaan kerucut.
- Metode hitungVolume() menghitung volume kerucut.
- tampilkanData() menampilkan semua informasi dari kerucut.

Output:

```
Jari-jari: 3
Tinggi: 5
Garis pelukis: 5.83095
Luas permukaan: 83.2298
Volume: 47.1239

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