LAPORAN PRAKTIKUM

MODUL 3

“ABSTRACT DATA TYPE”

Disusun Oleh:

MARSELLA DWI JULIANTI (2311104004)

SE 07-01

Dosen :

Yudha Islami Sulistya, S.kom, M.Cs.

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO

2024

1. Jelaskan apa yang dimaksud dengan pointer!

   Jawab:

   Pointer adalah sebuah variabel dalam pemrograman yang menyimpan alamat memori dari variabel lain. Dengan kata lain, pointer itu "menunjuk" ke arah Lokasi memori di mana data tertentu disimpan. Pointer sangat berguna dalam operasi seperti pengelolaan memori dinamis, manipulasi array, dan struktur data yang kompleks seperti linked list dan tree. Pointer sendiri juga memiliki **Deklarasi**, dimana pointer ini dideklarasikan dengan menggunakan tanda bintang (*) di depan nama variabel. Contoh: 'int*ptr;', yang berarti 'ptr' adalah pointer yang dapat menyimpan alamat memori dari variabel bertipe 'int'.

   Lalu ada **Insialisasi**, dimana pointer harus diinsialisasi dengan alamat memori variabel dari variabel yang lain menggunakan operator (&). Contoh: 'ptr = &var;', dimana 'var' adalah variabel bertipe 'int'.

   Terakhir ada **Dereferensi,** dimana buat mengkses atau memodifikasi nilai yang ditunjuk oleh pointer, digunakan operator dereferensi (*). Contoh: '*ptr = 10;' akan mengubah nilai dari varibael yang ditunjuk oleh 'ptr' menjadi 10.

2. Bagaimana cara menampilkan alamat memori dari suatu variabel dalam program C++? Berikan contoh!

   Jawab:

   Untuk menampilkan alamat memori dari suatu variabel, kita bisa menggunakan operator (&), dimana operator ini akan mengembalikan alamat memori dari variabel yang ditunjuk. Berikut contoh sederhana nya:

```
#include <iostream>
using namespace std;
int main() {
   int a = 10;
   float b = 20.5;
   char c = 'Z';
```

```
    cout << "Alamat memori variabel a: " << &a << endl;
    cout << "Alamat memori variabel b: " << &b << endl;
    cout << "Alamat memori variabel c: " << static_cast<void*>(&c) <<
endl;
    return 0;
}
```

Keterangan:

- ‘&a’ mengembalikan alamat memori dari variabel ‘a’
- ‘&b’ mengembalikan alamat memori dari variabel ‘b’
- ‘&c’ mengembalikan alamat memori dari variabel ‘c’.
- ‘static_cast<void*>’ digunakan untuk memastikan alamat memori ditampilkan dengan benar untuk tipe ‘char’

3. Bagaimana cara menggunakan pointer dalam program C++? Berikan contoh cara menampilkan nilai yang tersimpan pada suatu alamat melalui pointer!

   Jawab:

   Untuk menggunakan pointer, kita perlu paham dulu beberapa konsep dasar seperti deklarasi, insialisasi, dan dereferensi pointer.

   Berikut contoh sederhana yang menunjukkan caramenggunakan pointer dan menampilkan nilai yang tersimpan pada suatu alamat melalui pointer:

```
#include <iostream>
using namespace std;
int main() {
    int var = 42;  // Deklarasi variabel biasa
    int *ptr;     // Deklarasi pointer
    ptr = &var;   // Inisialisasi pointer dengan alamat variabel var
    // Menampilkan alamat memori dari variabel var
    cout << "Alamat memori dari var: " << &var << endl;
    // Menampilkan alamat yang disimpan dalam pointer ptr
    cout << "Alamat yang disimpan dalam ptr: " << ptr << endl;
```

```
    // Menampilkan nilai yang tersimpan pada alamat yang ditunjuk oleh
pointer ptr
    cout << "Nilai yang tersimpan pada alamat yang ditunjuk oleh ptr: " <<
*ptr << endl;
    return 0;
}
```

Keterangan:

- ‘int var = 42;’ mendeklarasikan variabel ‘var’ dan menginsialisasikan dengan nilai 42.
- ‘int *ptr;’ mendeklarasikan pointer ‘ptr’ yang dapat menyimpan alamat memori dari variabel bertipe ‘int’.
- ‘ptr = &var;’ menginsialisasikan pointer ‘ptr’ dengan alamat memori variabel ‘var’
- ‘cout << &var;’ menampilkan alamat memori dari variabel ‘var’
- ‘cout’ << ptr;’ menampilkan alamat yang disimpan dalam pointer ‘ptr’
- ‘cout << *ptr;’ menampilkan nilai yang tersimpan pada alamat yang ditunjuk oleh pointer ‘ptr’

4. 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 juga menyediakan cara untuk memisahkan spesifikasi dari implementasi, memungkinkan programmer untuk fokus pada bagaimana data digunakan daripada bagaimana data disimpan atau diatur. ADT juga memiliki poin penting, yaitu ada **Abstraksi**, dimana ini mendefeinisikan data dan operasi yang dapat dilakukan tanpa mengungkapkan detail implementasi. Contoh: stack, queue, list, dan tree. Lalu ada **Operasi**, dimana ini mendefenisikan operasi yang dapat dilakukan, sperti penambahan, penghapusan, dan pencarian elemen, tanpa menunjukan bagaimana operasi tersebut diimplementasikan. Lalu terakhir ada **Keuntungan**, dengan ini, kode

menjadi lebih modular dan mudah dipelihara. Perubahan implementasi ADT tidak mempengaruhi kode yang menggunakan AADT tersebut, selama antarmuka tetap konsisten.

Contoh sederhana: dari stack, ADT yang mendukung operasi seperti ‘push’ (menambahkan elemen ke atas stack), ‘pop’ (menghapus elemen dari atas stack), dan ‘peek’ (melihat elemen di atas stack tanpa menghapusnya).

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

   Jawab:

   Bayangkan di sebuah perpustakaan, dimana memiliki struktur dan operasi tertentu yang dapat dilakukan tanpa memikirkan detail implementasinya.

   1. **Data**: buku-buku di perpustakaan
   2. **Operasi**:
      - **Menambah buku:** menambahkan buku baru ke dalam koleksi perpustakaan
      - **Menghapus buku**: mengeluarkan buku dari koleksi perpustakaan
      - **Mencari buku**: mencari buku berdasarkan judul, penulis, atau kategori
      - **Meminjam buku**: meminjam buku untuk dibawa pulang
      - **Mengembalikan buku**: mengembaikan buku yang telah dipinjam

6. Tuliskan ADT dari bangun ruang kerucut dalam Bahasa C++!

   Jawab:

```
#include <iostream>
#include <cmath>
using namespace std;
class Kerucut {
private:
   float jariJari;
```

```
    float tinggi;
public:
    // Konstruktor
    Kerucut(float r, float t) : jariJari(r), tinggi(t) {}
    // Fungsi untuk menghitung volume kerucut
    float hitungVolume() {
        return (1.0/3) * M_PI * jariJari * jariJari * tinggi;
    }
    // Fungsi untuk menghitung luas permukaan kerucut
    float hitungLuasPermukaan() {
        float garisPelukis = sqrt((jariJari * jariJari) + (tinggi * tinggi));
        return M_PI * jariJari * (jariJari + garisPelukis);
    }
    // Fungsi untuk menampilkan informasi kerucut
    void tampilkanInfo() {
        cout << "Jari-jari: " << jariJari << endl;
        cout << "Tinggi: " << tinggi << endl;
        cout << "Volume: " << hitungVolume() << endl;
        cout << "Luas Permukaan: " << hitungLuasPermukaan() << endl;
    }
};
int main() {
    float r, t;
    cout << "Masukkan jari-jari kerucut: ";
    cin >> r;
    cout << "Masukkan tinggi kerucut: ";
    cin >> t;
    Kerucut kerucut(r, t);
    kerucut.tampilkanInfo();
    return 0;
}
```

Keterangan:

- **Kelas ‘**Kerucut’: Mewakili ADT untuk bangun ruang kerucut dengan atribut ‘jariJari’ dan ‘tinggi’
- **Konstruktor**: Menginisialisasi nilai jari-jari dan tinggi kerucut.
- **Fungsi** ‘hitungVolume’: Menghitung volume kerucut menggunakan rumus (\frac{1}{3} \pi r^2 t).
- **Fungsi** ‘hitungLuasPermukaan’: Menghitung luas permukaan kerucut menggunakan rumus (\pi r (r + s)), di mana (s) adalah garis pelukis yang dihitung dengan rumus Pythagoras.
- **Fungsi** ‘tampilkanInfo’: Menampilkan informasi tentang kerucut, termasuk jari-jari, tinggi, volume, dan luas permukaan.