

LAPORAN PRAKTIKUM

MODUL 1 TIPE DATA



Disusun Oleh:

Riyon Aryono : **2211102241**

Dosen

Muhammad Afrizal Amrustian

**PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM
PURWOKERTO
2023**

BAB I

TUJUAN PRAKTIKUM

A. Tujuan Praktikum

1. Mahasiswa dapat mempelajari berbagai jenis tipe data dari tipe data primitif, abstrak dan koleksi.
2. Mahasiswa dapat memahami pengaplikasian pada tools yang digunakan
3. Mahasiswa dapat menggunakan bahasa pemrograman yang telah ditentukan

BAB II

DASAR TEORI

Tipe data adalah adalah sebuah pengklasifikasian data berdasarkan jenis data tersebut. Tipe data dibutuhkan agar kompiler dapat mengetahui bagaimana sebuah data akan digunakan. Adapun tipe data yang akan dipelajari, sebagai berikut :

1. Tipe Data Primitive
2. Tipe Data Abstrak
3. Tipe Data Koleksi

1. Tipe Data Primitive

Tipe data primitive hanya dapat menyimpan satu nilai pada satu waktu dan tidak dapat diubah menggunakan cara yang sama seperti tipe data non-primitif. Tipe data Primitif akan dianggap sama jika nilainya sama. Berikut adalah daftar tipe data primitif yang umum digunakan di bahasa pemrograman C++:

- a. int : tipe data ini digunakan untuk menyimpan bilangan bulat seperti 1, 2, 3, dan seterusnya.
- b. float : tipe data ini digunakan untuk menyimpan bilangan pecahan seperti 2.5, 3.14, dan seterusnya.
- c. char : tipe data ini digunakan untuk menyimpan karakter seperti 'a', 'b', 'c', dan seterusnya.
- d. bool : tipe data ini digunakan untuk menyimpan nilai boolean yang hanya memiliki dua nilai yaitu true dan false.

2. Tipe Data Abstrak

Abstract Data Type (ADT) adalah tipe data yang dibuat untuk menggambarkan karakter/kondisi (state) dan perilaku (behaviour) dari sebuah object. Kita dapat membuat tipe data sendiri (user defined data type), misalnya string dalam C++. Tipe data abstract String dibuat dalam C++ class. Fitur Class class adalah fitur OOP pada C++ mirip seperti Fitur Data Structures struct pada C, keduanya dapat menampung variabel sebagai anggota.

- a. Class adalah salah satu dari konsep OOP yang digunakan untuk membungkus data abstraksi procedural sebagai deskripsi

tergeneralisir atau rancangan dari sebuah object untuk mendefinisikan atau menggambarkan isi dan tingkah laku sebagai entitas dari object.

- b. Structure atau struct adalah kumpulan dari beberapa variabel dengan beragam tipe data yang dibungkus dalam satu variabel. Struct juga dikenal dengan records dalam bahasa pemrograman lain seperti Pascal.

3. Tipe Data Koleksi

Koleksi adalah tipe data yang berupa rangkaian atau kumpulan data ataupun objek yang berindeks. Dalam program C++, kita dapat menggunakan kontainer Standard Template Library (STL) secara gratis, atau jenis koleksi lain yang ditentukan pengguna. Terdapat tiga tipe dasar koleksi di C++ yaitu:

- a. Array

Tipe data array adalah tipe data yang terdiri dari kumpulan tipe data lain. Anggota atau isi dari array itu sendiri harus satu jenis tipe data, misalkan terdiri dari kumpulan angka bulat saja (integer), kumpulan karakter saja (char), maupun kumpulan angka pecahan saja (double).

- b. Map

Map terasa mirip dengan array namun dengan index yang memungkinkan untuk berupa tipe data selain integer. Pada map, indeks tersebut diberi nama “key”. Pada `std::map` digunakan Self-Balancing Tree khususnya Red-Black Tree. Beberapa fungsi dasar yang terkait dengan Map: `begin()` – Mengembalikan iterator ke elemen pertama di Map. `end()` – Mengembalikan iterator ke elemen teoretis yang mengikuti elemen terakhir di Map. `size()` – Mengembalikan jumlah elemen di map. `max_size()` – Mengembalikan jumlah maksimum elemen yang dapat ditampung map. `kosong()` – Mengembalikan apakah map kosong. `pair insert(keyvalue, mapvalue)` – Menambahkan elemen baru ke map.

`erase(iterator position)` – Menghapus elemen pada posisi yang ditunjuk oleh iterator. `erase(const g)`– Menghapus nilai kunci 'g' dari map. `clear()` – Menghapus semua elemen dari map.

c. Vector

Vector adalah Standard Template Library (STL) jika di dalam C/C++ memiliki bentuk `std::vector` . Umumnya, vector mirip seperti array yang memiliki kemampuan untuk menyimpan data dalam bentuk elemen-elemen yang alokasi memorinya dilakukan otomatis dan bersebelahan. Kemampuan vector bukan hanya pada jumlah elemen yang dinamis, vector pada C/C++ juga dilengkapi dengan fitur-fitur pelengkap seperti element access, iterators, capacity, modifi

BAB III

LATIHAN & TUGAS

1. Guided

- Demo Tipe Data Primitif & Float

Source Code

```
#include <iostream>

using namespace std;

int main(){
    char op;
    float num1, num2;
    cin >> op;
    cin >> num1 >> num2;
    switch (op)
    {
        case '+':
            cout << num1 + num2;
            break;
        case '-':
            cout << num1 - num2;
            break;
        case '*':
            cout << num1 * num2;
            break;
        case '/':
            cout << num1 / num2;
            break;
        default:
            cout << "error operator is not corrected!";
            break;
    }
}
```

Output

```
PS C:\Users\yggdrasil\Documents\Me\Kuliah\smstr2\prak algodat\code\pertemuan-
mstr2\prak algodat\code\pertemuan-1\" ; if ($?) { g++ guided-1.cpp -o guided-
+
10
20
30
```

Penjelasan

Tipe data primitive hanya dapat menyimpan satu nilai pada satu waktu dan tidak dapat diubah menggunakan cara yang sama seperti tipe data non-primitif. Tipe data Primitif akan dianggap sama jika nilainya sama.

- Demo penggunaan Struct

Source Code

```
#include <stdio.h>

struct Mahasiswa
{
    char *name;
    char *address;
    int age;
};

void main(){
    struct Mahasiswa mhs1, mhs2;
    mhs1.name = "Dian";
    mhs1.address = "Mataram";
    mhs1.age = 22;
    mhs2.name = "Bambang";
    mhs2.address = "Surabaya";
    mhs2.age = 23;

    printf("## Mahasiswa 1 ## \n");
    printf("Nama : %s\n", mhs1.name);
    printf("Alamat : %s\n", mhs1.address);
    printf("Umur : %d\n", mhs1.age);

    printf("## Mahasiswa 2 ## \n");
    printf("Nama : %s\n", mhs2.name);
    printf("Alamat : %s\n", mhs2.address);
    printf("Umur : %d\n", mhs2.age);
}
```

Output

```
## Mahasiswa 1 ##
Nama : Dian
Alamat : Mataram
Umur : 22
## Mahasiswa 2 ##
Nama : Bambang
Alamat : Surabaya
Umur : 23
```

Penjelasan

Structure atau struct adalah kumpulan dari beberapa variabel dengan beragam tipe data yang dibungkus dalam satu variabel. Struct juga dikenal dengan records dalam bahasa pemrograman lain seperti Pascal.

- Demo Array

Source Code

```
#include <iostream>

using namespace std;

int main(){
    int nilai[5];

    nilai[0] =23;
    nilai[1] =50;
    nilai[2] =34;
    nilai[3] =78;
    nilai[4] =90;

    cout << "Isi array pertama :" << nilai[0] <<endl;
    cout << "Isi array kedua :" << nilai[1] <<endl;
    cout << "Isi array ketiga :" << nilai[2] <<endl;
    cout << "Isi array keempat :" << nilai[3] <<endl;
    cout << "Isi array kelima :" << nilai[4] <<endl;
}
```

Output

```
Isi array pertama :23
Isi array kedua :50
Isi array ketiga :34
Isi array keempat :78
Isi array kelima :90
```


Penjelasan

Tipe data array adalah tipe data yang terdiri dari kumpulan tipe data lain. Anggota atau isi dari array itu sendiri harus satu jenis tipe data, misalkan terdiri dari kumpulan angka bulat saja (integer), kumpulan karakter saja (char), maupun kumpulan angka pecahan saja (double).

2. Unguided

- Buatlah program menggunakan tipe data primitif minimal dua fungsi dan bebas. Menampilkan program, jelaskan program tersebut dan ambil kesimpulan dari materi tipe data primitif!

Source Code

```
#include <iostream>

using namespace std;

int main(){
    string nama = "Riyon Aryono";
    int nim = 2211102241;

    float tinggi,berat_ideal;
    char jenis_kelamin;

    cout << "Masukan Jenis kelamin (L/P): ";
    cin >> jenis_kelamin;

    cout << "Masukan tinggi badan (CM): ";
    cin >> tinggi;

    if(jenis_kelamin == 'L' || jenis_kelamin == 'l'){
        berat_ideal = (tinggi - 100) - ((tinggi - 100) * 0.1);
    }else if(jenis_kelamin == 'P' || jenis_kelamin == 'p'){
        berat_ideal = (tinggi - 100) - ((tinggi - 100) * 0.15);
    }else{
        cout << "Jenis kelamin tidak valid"<<endl;
    }

    cout << "Berat badan ideal anda adalah " << berat_ideal
    <<" Kg." <<endl;
}
```

Output

```
PS C:\Users\yggdrasil\Documents\Me\Kuliah\smstr2\prak_algodat\code\pertemuan-1\unguided\" ; if ($?) { g
}
Masukan Jenis kelamin (L/P): L
Masukan tinggi badan (CM): 175
Berat badan ideal anda adalah 67.5 Kg.
```

Penjelasan:

Program diatas menggunakan tipe data float dan char, tipe data float digunakan untuk menampung nilai tinggi badan dan berat badan ideal sedangkan char digunakan untuk menampung nilai jenis kelamin. Program diatas menggunakan if else untuk menentukan rumus yang nanti nya akan digunakan, diakhir program akan menampilkan berat badan ideal. Kesimpulan nya penentuan tipe data sangat penting sesuai dengan kebutuhan nya karena nanti nya tipe data akan berpengaruh dalam penggunaan memory yang dipakai jika menggunakan tipe data yang tepat maka kode program akan lebih cepat dieksekusi

- Jelaskan fungsi dari class dan struktur secara detail dan berikan contoh programnya:

Source Code Class

```
#include <iostream>

using namespace std;

class Car{
public:
    string brand;
    string model;
    int year;
    string nama;
    int nim;
```

```

        void nitro(){
            cout << "Nitro Diaktiv-kan, Ngeengggg";
        }

};

int main(){

    Car carObj;
    carObj.nama = "Riyon Aryono";
    carObj.nim = 22110241;
    carObj.brand = "Toyota";
    carObj.model = "Supra";
    carObj.year = 1980;

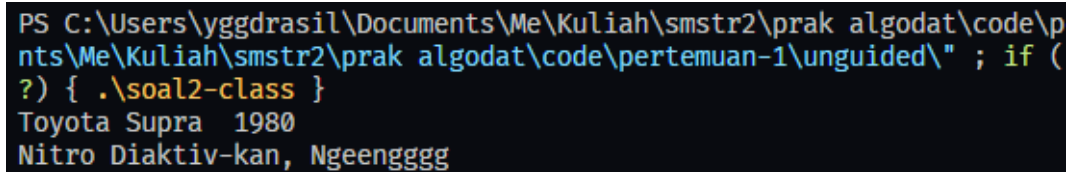
    cout << carObj.brand << " " << carObj.model << " " << " "
    << carObj.year << endl;

    carObj.nitro();

}

```

Output



```

PS C:\Users\yggdrasil\Documents\Me\Kuliah\smstr2\prak algodat\code\pnts\Me\Kuliah\smstr2\prak algodat\code\pertemuan-1\unguided\' ; if (
?) { .\soal2-class }
Toyota Supra 1980
Nitro Diaktiv-kan, Ngeengggg

```

Penjelasan

Kode program diatas menggunakan *class* dengan nama *class Car*. Didalam class terdapat *variable brand, model, dan year* dengan menggunakan property *Public* di Class *Car* juga terdapat sebuah method menggunakan prosedur dengan nama *nitro* digunakan untuk mengaktifkan nitro, Program mendeklarasi variable class ``carObj`` bertipe ``Car`` . Program memasukan data ke dalam variable class ``carObj`` dengan cara mengakses variable dalam class menggunakan operator titik (``.``)

Source Code Struct

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

using namespace std;

struct Car{
    string brand;
    string model;
    int year;
    string nama;
    int nim;
};

int main(){
    Car car1, car2;

    car1.nama = "Riyon Aryono";
    car1.nim = 2211102241;

    car1.brand = "Toyota";
    car1.model = "Supra";
    car1.year = 1999;

    car2.brand = "Toyota";
    car2.model = "Corolla DX";
    car2.year = 1980;

    cout << "Data Mobil 1: " << endl;
    cout << "Brand: " << car1.brand << endl;
    cout << "Model: " << car1.model << endl;
    cout << "Year: " << car1.year << endl << endl;

    cout << "Data Mobil 2: " << endl;
    cout << "Brand: " << car2.brand << endl;
    cout << "Model: " << car2.model << endl;
    cout << "Year: " << car2.year << endl << endl;

}
```

Output

```
($?) { .\soal2-struct }  
Data Mobil 1:  
Brand: Toyota  
Model: Supra  
Year: 1999  
  
Data Mobil 2:  
Brand: Toyota  
Model: Corolla DX  
Year: 1980
```

Penjelasan

Program diatas mendefinisikan sebuah struct bernama “Car” yang memiliki tiga variable yaitu “brand”, “model”, “year” ketiga nya memiliki tipe data masing masing. Program mendeklarasikan 2 buah variable struct “car1” dan “car2” bertipe “Car” lalu untuk memasukan data kedalam variable struct “car1” dan “car2” dengan cara mengakses variable ke dalam struct menggunakan operator titik (“.”), lalu program akan menampilkan data dari variable struct “car1” dan “car2”.

- Buat dan jelaskan program menggunakan fungsi map dan jelaskan perbedaan dari array dengan map.

Source Code

```
#include <iostream>  
#include <map>  
  
using namespace std;  
  
int main(){  
    map<string, int> mahasiswa;  
    string nama = “Riyon Aryono”;  
    int nim = 2211102241;  
  
    mahasiswa["Joko"] = 89;  
    mahasiswa["Puan"] = 76;  
    mahasiswa["Bowo"] = 65;  
  
    cout << "Daftar Nilai Mahasiswa:"<<endl;
```

```

        auto itr = mahasiswa.begin();
        while (itr != mahasiswa.end())
        {
            cout << itr->first << " : " << itr->second << endl;
            itr++;
        }
    }
}

```

Ouput

```

Daftar Nilai Mahasiswa:
Bowo : 65
Joko : 89
Puan : 76

```

Penjelasan

Kode program mendefinisikan sebuah map bernama “mahasiswa” yang memiliki key bertipe “string” dan value bertipe “int”. Untuk memasukan data kedalam map dengan menggunakan operator kurung siku (“[]”) lalu untuk menampilkan data dari variabel “mahasiswa” dengan menggunakan “while loop”. Perbedaan array dengan map yaitu array hanya bisa menyimpan satu tipe data saja sedangkan map dapat menyimpan dua tipe data yaitu pasangan key dan value yang bisa memiliki tipe data yang berbeda. Untuk pengaksesan data di array biasanya menggunakan index sedangkan map dengan menggunakan key. Ukuran array juga harus ditentukan saat awal pendeklarasian sedangkan map dapat bertambah ataupun berkurang sesuai data yang ingin dimasukan atau dihapus dengan kata lain map lebih fleksibel dibandingkan kan array.

BAB IV

KESIMPULAN

B. Kesimpulan

Tipe data digunakan untuk menampung sebuah nilai dalam variable juga untuk menentukan jenis operasi apa saja yang dapat dilakukan pada nilai tersebut. Berikut adalah beberapa tipe data yang ada

1. Tipe Data Primitive

Tipe data primitive hanya dapat menyimpan satu nilai pada satu waktu dan tidak dapat diubah menggunakan cara yang sama seperti tipe data non-primitif. Tipe data Primitif akan dianggap sama jika nilainya sama.

2. Tipe Data Abstrak

Abstract Data Type (ADT) adalah tipe data yang dibuat untuk menggambarkan karakter/kondisi (state) dan perilaku (behaviour) dari sebuah object. Kita dapat membuat tipe data sendiri (user defined data type),

3. Tipe Data Koleksi

Koleksi adalah tipe data yang berupa rangkaian atau kumpulan data ataupun objek yang berindeks. Dalam program C++, kita dapat menggunakan kontainer Standard Template Library (STL) secara gratis, atau jenis koleksi lain yang ditentukan pengguna.

Ketika menggunakan tipe data sangat penting untuk memilih tipe data yang tepat untuk tugas yang dilakukan agar memori yang digunakan dapat dihemat dan program dapat berjalan dengan lebih efisien.