LAPORAN PRAKTIKUM

MODUL I
TIPE DATA

Disusun oleh:
Anita Nurazizah Agussalim
NIM: 2311102017

Dosen Pengampu:
Wahyu Andi Saputra, S.Pd., M.Eng.

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
PURWOKERTO
2024

BAB I
TUJUAN PRAKTIKUM

1. Mahasiswa dapat mempelajari tipe data primitif, abstrak, dan kolektif.
2. Mahasiswa dapat memahami pengaplikasian pada tools yang digunakan.
3. Mahasiswa mengaplikasikan berbagai tipe data pada bahasa pemograman 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 Primitif
2. Tipe data Abstrak
3. Tipe data Koleksi

1. Tipe data Primitif

Tipe data primitif adalah tipe data yang sudah ditentukan oleh sistem, tipe data primitif ini disediakan oleh banyak bahasa pemrograman, perbedaannya terletak pada jumlah bit yang dialokasikan untuk setiap bit pada tipe data primitif tergantung pada bahasa pemrograman, compiler dan sistem operasinya. Contoh tipe data primitif adalah:

a. Int : adalah tipe data yang digunakan untuk menyimpan bilangan bulat seperti 12, 1, 4, dan sebagainya.
b. Float : tipe data yang digunakan untuk menyimpan bilangan desimal seperti 1.5, 2.1, 3.14, dan sebagainya.
c. Char : berfungsi untuk menyimpan data berupa sebuah huruf. Biasanya digunakan untuk simbol seperti A, B, C dan seterusnya
d. Boolean : tipe data ini digunakan untuk menyimpan nilai boolean yang hanya memiliki dua nilai yaitu true dan false.

2. Tipe Data Abstrak

Tipe data abstrak atau yang biasa disebut Abstrak Data Type (ADT) merupakan tipe data yang dibentuk oleh programer itu sendiri. Pada tipe data abstrak bisa berisi banyak tipe data, jadi nilainya bisa lebih dari satu dan beragam tipe data. Fitur Class adalah fitur Object Oriented Program (OPP) pada bahasa C++ yang mirip dengan fitur data structures Struct pada bahasa C. Keduanya berfungsi untuk membungkus tipe data di dalamnya sebagai anggota. menurut learn.microsoft.com perbedaan antara Struct dan Class adalah pada akses defaultnya dimana Struct bersifat public dan Class bersifat private.

3. Tipe Data Koleksi

Tipe data koleksi (Collection Data Type) adalah tipe data yang digunakan untuk mengelompokkan dan menyimpan beberapa nilai atau objek secara bersamaan. Tipe data koleksi memungkinkan Anda menyimpan, mengelola, dan mengakses sejumlah besar data dengan cara yang terstruktur. Ada beberapa tipe data koleksi yang umum digunakan dalam pemrograman, dan di antaranya adalah:

a. Array : Array adalah struktur data statis yang menyimpan elemen-elemen dengan tipe data yang sama. Elemen-elemen tersebut dapat diakses dengan menggunakan indeks. Array memiliki ukuran tetap yang ditentukan saat deklarasi.
b. 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, modifiers.
c. 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.

BAB III

GUIDED

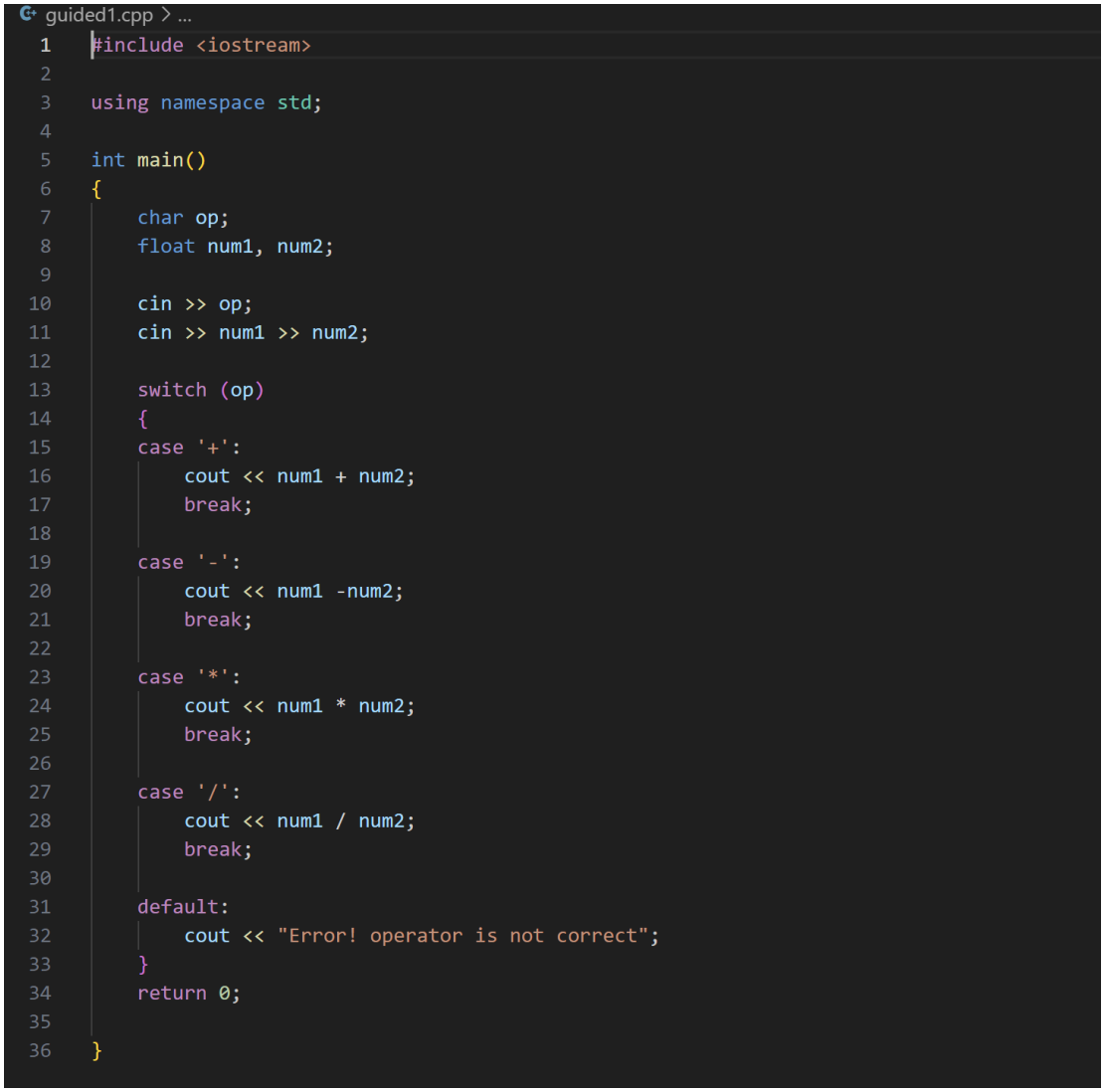
1. Guided 1

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 correct";
    }
    return 0;

}
```

Screenshoot program



Deskripsi program

Program ini adalah sebuah program kalkulator sederhana yang ditulis dalam bahasa C++. Program ini menerima operator (+, -, *, /) dan dua bilangan sebagai masukan, lalu melakukan operasi aritmatika yang sesuai berdasarkan operator yang dimasukkan.

2. Guided 2

Source code

```
#include <stdio.h>

struct mahasiswa
{
    const char *name;
    const char *addres;
    int age;
};

int main()
{
    struct mahasiswa mhs1, mhs2;
     mhs1.name = "Dian";
     mhs1.addres = "Mataram";
     mhs1.age = 22;

     mhs2.name = "Bambang";
     mhs2.addres = "Surabaya";
     mhs2.age = 23;

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

```
    printf ("## Mahasiswa 2 ##\n");
    printf ("Nama: %s\n", mhs2.name);
    printf ("Alamat: %s\n", mhs2.addres);
    printf ("Umur: %d\n", mhs2.age);
    return 0;
}
```

Screenshoot program

guided2.cpp > main()

```cpp
#include <stdio.h>

struct mahasiswa
{
    const char *name;
    const char *addres;
    int age;
};

int main()
{
    struct mahasiswa mhs1, mhs2;
    mhs1.name = "Dian";
    mhs1.addres = "Mataram";
    mhs1.age = 22;

    mhs2.name = "Bambang";
    mhs2.addres = "Surabaya";
    mhs2.age = 23;

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

    printf ("## Mahasiswa 2 ##\n");
    printf ("Nama: %s\n", mhs2.name);
    printf ("Alamat: %s\n", mhs2.addres);
    printf ("Umur: %d\n", mhs2.age);
    return 0;
}
```

Deskripsi program

Program ini adalah sebuah program untuk menampilkan informasi tentang dua mahasiswa. Pertama, sebuah struktur yang berisi informasi seperti nama, alamat, dan usia didefinisikan. Kemudian, dua variabel dari struktur tersebut untuk merepresentasikan dua mahasiswa dibuat. Setelah itu, informasi untuk setiap mahasiswa dimasukkan. Akhirnya, informasi masing-masing mahasiswa akan muncul ketika program dijalankan.

3. Guided 3

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;

    return 0;

}
```

Screenshoot program

guided3.cpp > ...

```cpp
#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;

    return 0;

}
```

Deskripsi program

Program tersebut adalah sebuah program sederhana dalam bahasa C++. Pertama, kita mendeklarasikan sebuah array bernama ‘nilai’ yang dapat menyimpan 5 angka integer. Kemudian, kita mengisi setiap elemen array dengan nilai yang berbeda. Setelah itu, kita mencetak isi masing-masing elemen array ke layar menggunakan pernyataan ‘cout’. Akhirnya, program mengembalikan nilai 0 untuk menandakan bahwa eksekusi program berjalan dengan sukses.

LATIHAN KELAS - UNGUIDED

1. Unguided 1

Source code

```
#include <iostream>

using namespace std;

int main() {

    int age;

    cout << "Masukkan umur Anda: ";
    cin >> age;

    bool CukupUmur;

    if (age >= 17) {
        CukupUmur = true;
    } else {
        CukupUmur = false;
    }

    if (CukupUmur) {
        cout << "Anda dinyatakan cukup umur" << endl;
    } else {
        cout << "Anda masih di bawah umur" << endl;
    }

    return 0;
}
```

Screenshoot program

unguided1.cpp > ...

```cpp
#include <iostream>

using namespace std;

int main() {

    int age;

    cout << "Masukkan umur Anda: ";
    cin >> age;

    bool CukupUmur;

    if (age >= 17) {
        CukupUmur = true;
    } else {
        CukupUmur = false;
    }

    if (CukupUmur) {
        cout << "Anda dinyatakan cukup umur" << endl;
    } else {
        cout << "Anda masih di bawah umur" << endl;
    }

    return 0;
}
```

Deskripsi program

Program ini meminta pengguna untuk memasukkan usia mereka dan menentukan apakah mereka termasuk cukup umur atau tidak. Program ini dimulai dengan meminta pengguna untuk memasukkan usia mereka, yang disimpan dalam variabel integer Bernama '`age`'. Kemudian, program mendeklarasikan variabel boolean '`CukupUmur`' untuk mewakili apakah pengguna telah cukup umur, menginisialisasinya menjadi `` `true` `` jika usia pengguna adalah 17 atau lebih, dan `` `false` `` jika tidak. Program sederhana ini menunjukkan penggunaan tipe data integer dan boolean, serta pernyataan kondisional, untuk membuat keputusan sederhana berdasarkan masukan pengguna.

Kesimpulan

Tipe data primitif adalah jenis data yang sudah ditentukan oleh sistem, yang tersedia dalam berbagai bahasa pemrograman dengan perbedaan dalam alokasi bit tergantung pada bahasa, compiler, dan sistem operasinya. Contoh tipe data primitif meliputi: `int` untuk bilangan bulat, `float` untuk bilangan desimal, `char` untuk karakter, dan `boolean` untuk nilai boolean dengan hanya dua nilai: `true` dan `false`. Tipe data primitif ini menyediakan dasar untuk menyimpan dan memanipulasi informasi dalam program secara efisien dan efektif.

2. Unguided 2

Class:

Source code

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

using namespace std;

class Siswa {
public:
    string name;
    int age;
    string grade;
};

int main() {
    Siswa s;

    s.name = "Ananda";
    s.age = 10;
```

```
    s.grade = "A";

    cout << "Nama: " << s.name << endl;
    cout << "Umur: " << s.age << endl;
    cout << "Kelas: " << s.grade << endl;

    return 0;
}
```

Screenshoot program

unguided2_class.cpp > ...

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

using namespace std;

class Siswa {
public:
    string name;
    int age;
    string grade;
};

int main() {
    Siswa s;

    s.name = "Ananda";
    s.age = 10;
    s.grade = "A";

    cout << "Nama: " << s.name << endl;
    cout << "Umur: " << s.age << endl;
    cout << "Kelas: " << s.grade << endl;

    return 0;
}
```

Struct:

Source code

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

using namespace std;

struct Siswa {
    string name;
    int age;
    string grade;
};

int main() {
    Siswa s;

    s.name = "Ananda";
    s.age = 10;
    s.grade = "A";

    cout << "Nama: " << s.name << endl;
    cout << "Umur: " << s.age << endl;
    cout << "Kelas: " << s.grade << endl;

    return 0;
}
```

Screenshoot program

unguided2_struct.cpp > ...

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

using namespace std;

struct Siswa {
    string name;
    int age;
    string grade;
};

int main() {
    Siswa s;

    s.name = "Ananda";
    s.age = 10;
    s.grade = "A";

    cout << "Nama: " << s.name << endl;
    cout << "Umur: " << s.age << endl;
    cout << "Kelas: " << s.grade << endl;

    return 0;
}
```

Fungsi Class dan Struct

1. Class:

- **Fungsi**: Class adalah kumpulan dari data (variabel) dan fungsi (metode) yang membentuk suatu tipe data baru.
- **Penggunaan**: Digunakan untuk mengorganisir dan mengelompokkan data dan fungsi terkait ke dalam satu kesatuan.
- **Fungsi Anggota**: Bisa memiliki fungsi anggota (metode) yang mendefinisikan perilaku atau tindakan yang dapat dilakukan oleh objek yang berbasis pada class tersebut.

- **Enkapsulasi**: Dapat menerapkan konsep enkapsulasi dengan memberikan aksesibilitas yang terbatas ke anggota-anggota class menggunakan access specifiers (public, private, protected).

2. Struct:

- **Fungsi**: Struct adalah kumpulan dari data yang dapat merepresentasikan entitas dengan atribut-atributnya.
- **Penggunaan**: Biasanya digunakan untuk merepresentasikan data sederhana yang terdiri dari beberapa atribut.
- **Fungsi Anggota**: Tidak memiliki fungsi anggota secara default, meskipun dapat memiliki jika diperlukan.
- **Enkapsulasi**: Tidak mendukung konsep enkapsulasi secara langsung, karena semua anggota struct biasanya bersifat publik secara default.

3. Unguided 3

Source code

```
#include <iostream>
#include <map>

using namespace std;

int main() {
    map<string, int> NilaiSiswa;

    NilaiSiswa["Alisa"] = 85;
    NilaiSiswa["Bobby"] = 90;
    NilaiSiswa["Caca"] = 78;
    NilaiSiswa["Diana"] = 95;

    cout << "Daftar Nilai Siswa:" << endl;
    for (auto& pair : NilaiSiswa) {
```

```
        cout << pair.first << ": " << pair.second << endl;
    }

    string CekNama = "Tyler";
    if (NilaiSiswa.find(CekNama) != NilaiSiswa.end()) {
        cout << CekNama << " memiliki nilai " << NilaiSiswa[CekNama] << endl;
    } else {
        cout << CekNama << " tidak ditemukan dalam map." << endl;
    }

    return 0;
}
```

Screenshoot program

Deskripsi program

Program ini menggunakan tipe data `map` untuk menyimpan dan mengelola nilai-nilai siswa. Pertama, sebuah `map` dengan tipe kunci string dan nilai integer dideklarasikan dengan nama `NilaiSiswa`. Kemudian, nilai-nilai untuk empat siswa ditambahkan ke dalam map dengan menggunakan operator `[]`. Program kemudian mencetak daftar nilai setiap siswa dengan melakukan iterasi melalui map menggunakan perulangan `for`. Selain itu, program juga memeriksa "Tyler" ada dalam map atau tidak. Jika "Tyler" ditemukan, nilai yang dimilikinya akan dicetak; jika tidak, pesan yang menyatakan bahwa "Tyler" tidak ditemukan dalam map akan ditampilkan.

Perbedaan Array dan Map

Array adalah struktur data yang menyimpan sekumpulan nilai dengan indeks yang berurutan dan tetap, sementara map adalah struktur data yang mengaitkan setiap kunci unik dengan nilai tertentu. Dalam array, akses ke nilai dilakukan melalui indeks numerik, sedangkan dalam map, akses dilakukan melalui kunci yang bisa berupa tipe data apa pun. Array memiliki ukuran yang tetap dan harus ditentukan pada saat deklarasi, sedangkan map memiliki ukuran dinamis dan dapat diperluas atau dikurangi saat program berjalan. Map juga menyediakan pencarian cepat berdasarkan kunci, sementara pencarian dalam array memerlukan iterasi melalui setiap elemen untuk menemukan nilai yang diinginkan.

BAB IV

KESIMPULAN

Tipe data adalah cara untuk mengklasifikasikan data berdasarkan jenisnya, yang diperlukan agar kompiler dapat memahami cara data tersebut akan digunakan. Ada tiga jenis tipe data yang umum digunakan: tipe data primitif, tipe data abstrak, dan tipe data koleksi. Tipe data primitif adalah tipe data yang sudah ditentukan oleh sistem dan mencakup jenis data dasar seperti integer, float, char, dan boolean. Tipe data abstrak, atau Abstract Data Type (ADT), adalah tipe data yang dibuat oleh pemrogram dan dapat berisi berbagai jenis data. Di C++, fitur class digunakan untuk membuat ADT, di mana class mirip dengan struct namun dengan aksesibilitas default yang berbeda. Tipe data koleksi, seperti array, vector, dan map, digunakan untuk mengelompokkan dan menyimpan beberapa nilai atau objek bersamaan. Array adalah struktur data statis dengan ukuran tetap, vector adalah struktur data dinamis dengan alokasi memorinya otomatis, dan map adalah struktur data yang menggunakan indeks yang bisa berupa tipe data selain integer. Dengan memahami berbagai tipe data ini, programer dapat mengelola data dengan lebih efisien dan terstruktur.