

LAPORAN PRAKTIKUM

MODUL I TIPE DATA



**Disusun oleh:
Tegar Bangkit Wijaya
NIM: 2311102027**

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

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

BAB I

TUJUAN PRAKTIKUM

A. TUJUAN PRAKTIKUM

1. Mahasiswa dapat mempelajari tipe data pimitif, abstrak, dan kolektif.
2. Mahasiswa dapat memahami pengaplikasian pada tools yang digunakan.
3. Mahasiswa dapat mengaplikasikan berbagai tipe data pada bahasa pemrograman yang telah ditentukan.

BAB II

DASAR TEORI

B. DASAR TEORI

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

1. Tipe data primitive
2. Tipe data abstrak
3. Tipe data koleksi

BAB III

GUIDED

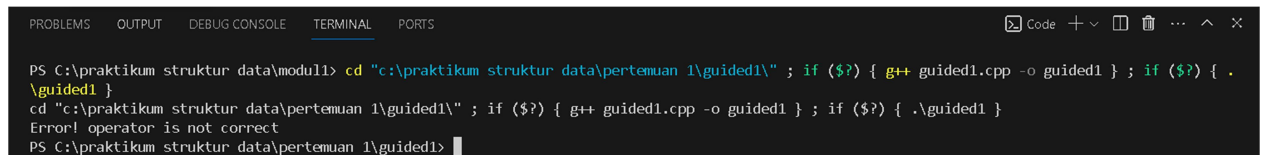
1. Guided 1

Source code

```
#include <iostream>
using namespace std;
// Main program
int main()
{
    char op;
    float num1, num2;
    // It allows user to enter operator i.e. +, -, *, /
    cin >> op;
    // It allow user to enter the operands
    cin >> num1 >> num2;
    // Switch statement begins
    switch(op)
    {
        // If user enter +
        case '+':
            cout << num1 + num2;
            break;
        // If user enter -
        case '-':
            cout << num1 - num2;
            break;
        // If user enter *
        case '*':
            cout << num1 * num2;
            break;
        // If user enter /
        case '/':
            cout << num1 / num2;
            break;
        // If the operator is other than +, -, * or /,
        // error message will display
        default:
            cout << "Error! operator is not correct";
```

```
} // switch statement ends  
return 0;  
}
```

SCREENSHOOT PROGRAM



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS  
PS C:\praktikum struktur data\modul1> cd "c:\praktikum struktur data\pertemuan 1\guided1\" ; if ($?) { g++ guided1.cpp -o guided1 } ; if ($?) { .  
.\guided1 }  
cd "c:\praktikum struktur data\pertemuan 1\guided1\" ; if ($?) { g++ guided1.cpp -o guided1 } ; if ($?) { .\guided1 }  
Error! operator is not correct  
PS C:\praktikum struktur data\pertemuan 1\guided1>
```

DESKRIPSI PROGRAM

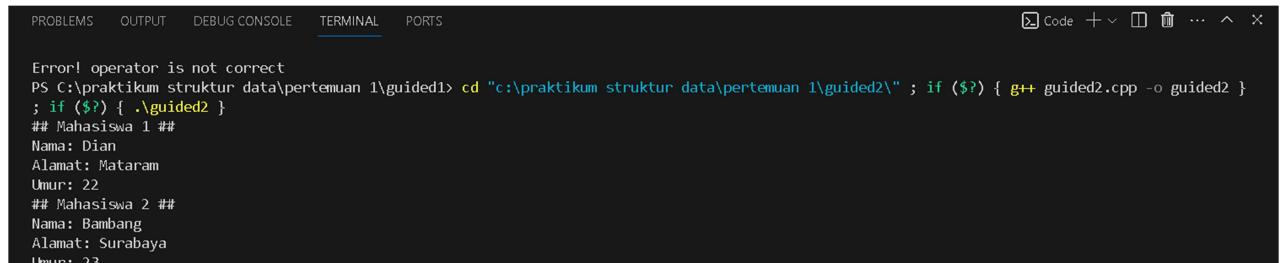
Program ini merupakan kalkulator sederhana yang meminta pengguna untuk memasukkan operator (+, -, *, /) . Setelah menerima input tersebut, program menggunakan sebuah pernyataan switch untuk mengevaluasi operator yang dimasukkan pengguna. Jika operator sesuai dengan salah satu kasus yang dijelaskan dalam switch statement (penjumlahan, pengurangan, perkalian, atau pembagian), program akan menghitung hasilnya dan menampilkannya. Namun, jika operator yang dimasukkan tidak sesuai dengan empat kasus tersebut, program akan menampilkan pesan kesalahan. Selanjutnya, program akan mengembalikan nilai 0 sebagai tanda bahwa program telah berhasil dieksekusi

2. Guided 2

Source code

```
#include <stdio.h>
//Struct
struct Mahasiswa
{
    const char *name;
    const char *address;
int age;
};
int main()
{
    // menggunakan struct
    struct Mahasiswa mhs1, mhs2;
    // mengisi nilai ke struct
    mhs1.name = "Dian";
    mhs1.address = "Mataram";
    mhs1.age = 22;
    mhs2.name = "Bambang";
    mhs2.address = "Surabaya";
    mhs2.age = 23;
    // mencetak isi struct
    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);
    return 0;
}
```

SCREENSHOT PROGRAM



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Error! operator is not correct
PS C:\praktikum struktur data\pertemuan 1\guided1> cd "c:\praktikum struktur data\pertemuan 1\guided2\" ; if ($?) { gcc guided2.cpp -o guided2 }
; if ($?) { .\guided2 }
## Mahasiswa 1 ##
Nama: Dian
Alamat: Mataram
Umur: 22
## Mahasiswa 2 ##
Nama: Bambang
Alamat: Surabaya
Umur: 23
```

DESKRIPSI PROGRAM

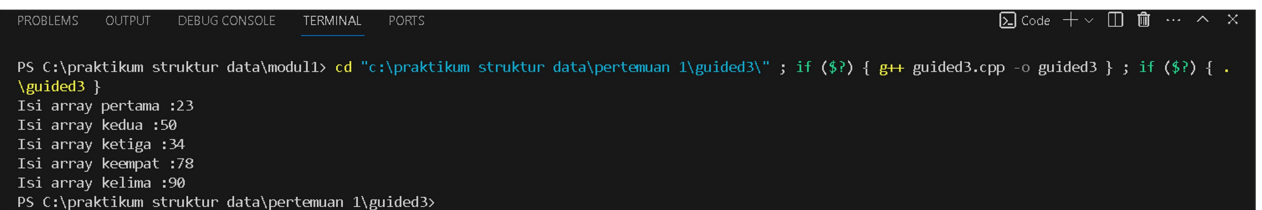
Program ini menggunakan bahasa pemrograman c++ untuk mendefinisikan struktur data `Mahasiswa` yang memiliki tiga anggota: `name` untuk menyimpan nama mahasiswa, `address` untuk menyimpan alamat mahasiswa, dan `age` untuk menyimpan usia mahasiswa. Selanjutnya, program utama (`main`) mendeklarasikan dua variabel struktur `Mahasiswa`, yaitu `mhs1` dan `mhs2`. Kemudian, nilai-nilai untuk masing-masing anggota dari kedua variabel tersebut diinisialisasi. Setelah itu, program mencetak informasi tentang `mhs1` dan `mhs2` ke layar dengan menggunakan fungsi `printf`, termasuk nama, alamat, dan usia masing-masing mahasiswa. Akhirnya, program mengembalikan nilai 0 untuk menandakan bahwa eksekusi program telah berhasil. Program ini dapat digunakan untuk menyimpan dan menampilkan informasi dasar tentang dua mahasiswa.

3. Guided 3

Source code

```
#include <iostream>
using namespace std;
int main()
{
    //deklarasi dan inisialisasi array
    int nilai[5];
    nilai[0] = 23;
    nilai[1] = 50;
    nilai[2] = 34;
    nilai[3] = 78;
    nilai[4] = 90;
    //mencetak array
    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;
}
```

SCREENSHOT PROGRAM



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\praktikum struktur data\modul1> cd "c:\praktikum struktur data\pertemuan 1\guided3\" ; if ($?) { g++ guided3.cpp -o guided3 } ; if ($?) { .
\guided3 }
Isi array pertama :23
Isi array kedua :50
Isi array ketiga :34
Isi array keempat :78
Isi array kelima :90
PS C:\praktikum struktur data\pertemuan 1\guided3>
```


DESKRIPSI PROGRAM

Program ini merupakan contoh penggunaan array. Pada awalnya, sebuah array bernama `nilai` dengan panjang 5 elemen dideklarasikan dan diinisialisasi. Setiap elemen array kemudian diisi dengan nilai tertentu menggunakan indeks array. Setelah itu, program mencetak nilai dari setiap elemen array ke layar menggunakan pernyataan `cout`. Setiap nilai dicetak dengan disertakan pesan yang menjelaskan posisi elemen dalam array, misalnya "Isi array pertama:", "Isi array kedua:", dan seterusnya. Akhirnya, program mengembalikan nilai 0 untuk menandakan bahwa eksekusi program telah berhasil. Program ini bertujuan untuk menunjukkan cara menyimpan beberapa nilai dalam satu variabel array dan cara mengakses nilai-nilai tersebut. Dengan demikian, program ini memberikan contoh sederhana tentang bagaimana menggunakan array dalam C++ untuk menyimpan dan mengakses beberapa nilai dengan menggunakan indeksnya.

LATIHAN KELAS - UNGUIDED

1. Unguided 1

Source code

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

float hitungLuasSegitiga(float alas, float tinggi) {
    return 0.5 * alas * tinggi; // Rumus luas segitiga
}

float hitungKelilingSegitiga(float sisi1, float sisi2, float sisi3) {
    return sisi1 + sisi2 + sisi3; // Rumus keliling segitiga
}

int main() {
    float alas, tinggi, sisi1, sisi2, sisi3;

    cout << "Masukkan panjang alas segitiga: ";
    cin >> alas;

    cout << "Masukkan tinggi segitiga: ";
    cin >> tinggi;

    cout << "Masukkan panjang sisi 1 segitiga: ";
    cin >> sisi1;

    cout << "Masukkan panjang sisi 2 segitiga: ";
    cin >> sisi2;

    cout << "Masukkan panjang sisi 3 segitiga: ";
    cin >> sisi3;

    float luas = hitungLuasSegitiga(alas, tinggi);
    float keliling = hitungKelilingSegitiga(sisi1, sisi2, sisi3);

    cout << "Luas segitiga: " << luas << endl;
    cout << "Keliling segitiga: " << keliling << endl;

    return 0;
}
```

SCREENSHOOT PROGRAM

```
PS C:\praktikum struktur data\pertemuan 1\unguided1> cd "c:\praktikum struktur data\pertemuan 1\unguided1\" ; if ($?) { g++ unguided1.cpp -o unguided1 ; if ($?) { .\unguided1 } }
Masukkan panjang alas segitiga: 4
Masukkan tinggi segitiga: 4
Masukkan panjang sisi 1 segitiga: 4
Masukkan panjang sisi 2 segitiga: 4
Masukkan panjang sisi 3 segitiga: 4
Luas segitiga: 8
Keliling segitiga: 12
PS C:\praktikum struktur data\pertemuan 1\unguided1>
```

DESKRIPSI PROGRAM

Program ini adalah contoh sederhana dalam bahasa C++ untuk menghitung luas dan keliling segitiga.

Deklarasi Fungsi-fungsi: Dua fungsi `hitungLuasSegitiga` dan `hitungKelilingSegitiga` didefinisikan di awal program. Fungsi `hitungLuasSegitiga` mengambil dua parameter, yaitu alas dan tinggi segitiga, dan mengembalikan nilai luas segitiga sesuai dengan rumus $0.5 * \text{alas} * \text{tinggi}$. Fungsi `hitungKelilingSegitiga` mengambil tiga parameter, yaitu panjang ketiga sisi segitiga, dan mengembalikan nilai keliling segitiga dengan menjumlahkan panjang ketiga sisinya.

Input Pengguna: Pengguna diminta untuk memasukkan panjang alas, tinggi, dan panjang ketiga sisi segitiga menggunakan pernyataan `cout` dan `cin`.

Pemanggilan Fungsi: Setelah input dari pengguna diterima, fungsi `hitungLuasSegitiga` dan `hitungKelilingSegitiga` dipanggil dengan parameter yang sesuai (alas dan tinggi untuk luas, dan panjang ketiga sisi untuk keliling). Hasil perhitungan disimpan dalam variabel `luas` dan `keliling`.

Output: Hasil perhitungan luas dan keliling segitiga dicetak ke layar menggunakan pernyataan cout.

Pengembalian Nilai: Program mengembalikan nilai 0, menandakan bahwa program telah berakhir dengan sukses.

Program ini memberikan contoh sederhana tentang bagaimana menggunakan fungsi untuk menghitung luas dan keliling segitiga berdasarkan input yang diberikan pengguna.

2. Unguided 2

Source code

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

// class
class Mahasiswa {
private:
    string nama;
    string jurusan;
    int umur;
public:
    Mahasiswa(string n, string j, int u) {
        nama = n;
        jurusan = j;
        umur = u;
    }

    void displayInfo() {
        cout << "Nama: " << nama << endl;
        cout << "Jurusan: " << jurusan << endl;
        cout << "Umur: " << umur << endl;
    }
};

int main() {
    Mahasiswa mhs1("Tegar", "Teknik Informatika", 19);
    mhs1.displayInfo();

    return 0;
}
```

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

// struct
struct Mahasiswa {
    string nama;
    string jurusan;
    int umur;

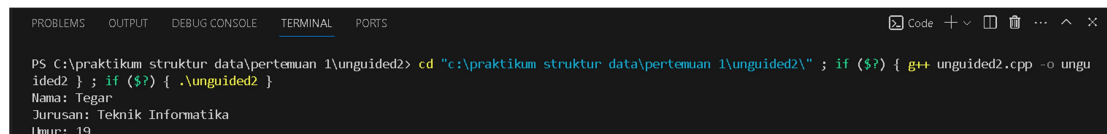
    void displayInfo() {
        cout << "Nama: " << nama << endl;
        cout << "Jurusan: " << jurusan << endl;
        cout << "Umur: " << umur << endl;
    }
};

int main() {
    Mahasiswa mhs1 = {"Tegar", "Teknik Informatika", 19};
    mhs1.displayInfo();

    return 0;
}
```

SCREENSHOT PROGRAM

class



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\praktikum struktur data\pertemuan 1\unguided2> cd "c:\praktikum struktur data\pertemuan 1\unguided2\" ; if ($?) { g++ unguided2.cpp -o unguided2 } ; if ($?) { .\unguided2 }
Nama: Tegar
Jurusan: Teknik Informatika
Umur: 19
```

Struct



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\praktikum struktur data\pertemuan 1\UNGUIDED2_STRUCT> cd "c:\praktikum struktur data\pertemuan 1\UNGUIDED2_STRUCT\" ; if ($?) { g++ UNGUIDED2_STRUCT.CPP -o UNGUIDED2_STRUCT } ; if ($?) { .\UNGUIDED2_STRUCT }
Nama: Tegar
Jurusan: Teknik Informatika
Umur: 19
PS C:\praktikum struktur data\pertemuan 1\UNGUIDED2_STRUCT>
```

DESKRIPSI PROGRAM

Class adalah struktur data yang memungkinkan kita untuk menggabungkan data bersama dengan fungsi yang beroperasi pada data tersebut. Dalam class, data disebut sebagai anggota (member) dan fungsi disebut sebagai metode. Class adalah salah satu konsep dasar dalam pemrograman berorientasi objek (OOP). Dengan class, kita dapat membuat objek-objek yang merupakan instance dari class tersebut.

Struct (singkatan dari structure) adalah kumpulan variabel yang dapat berisi data dari jenis yang berbeda, yang digabungkan menjadi satu kesatuan. Struct dalam bahasa C++ juga dapat memiliki fungsi-fungsi yang disebut metode, mirip dengan class. Struct secara default memiliki visibilitas anggota (member) yang public, artinya anggota-anggota tersebut dapat diakses dari luar struktur.

3. Unguided 3

Source code

```
#include <iostream>
#include <map>
#include <string>
using namespace std;

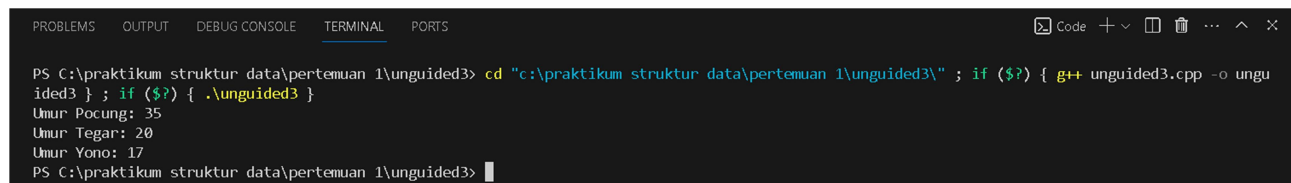
int main() {
    map<string, int> umur;

    umur["Pocung"] = 35;
    umur["Tegar"] = 20;
    umur["Yono"] = 17;

    cout << "Umur Pocung: " << umur["Pocung"] << endl;
    cout << "Umur Tegar: " << umur["Tegar"] << endl;
    cout << "Umur Yono: " << umur["Yono"] << endl;

    return 0;
}
```

SCREENSHOT PROGRAM



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\praktikum struktur data\pertemuan 1\unguided3> cd "c:\praktikum struktur data\pertemuan 1\unguided3\" ; if ($?) { g++ unguided3.cpp -o unguided3 ; if ($?) { .\unguided3 }
Umur Pocung: 35
Umur Tegar: 20
Umur Yono: 17
PS C:\praktikum struktur data\pertemuan 1\unguided3>
```

DESKRIPSI PROGRAM

Program di atas menggunakan map untuk memetakan string (nama) ke integer (umur). Fungsi map memungkinkan kita untuk menyimpan pasangan nilai (key, value) dan melakukan pencarian berdasarkan kunci dengan efisiensi tinggi. Pada contoh di atas, kita menambahkan beberapa data ke dalam map dan kemudian mengakses data tersebut berdasarkan kunci (nama). Hasilnya adalah mencetak umur dari orang-orang yang disimpan di dalam map.

Perbedaan antara array dan map:

Array: Array adalah struktur data yang menyimpan kumpulan elemen data dengan indeks berurutan, di mana setiap elemen dapat diakses menggunakan indeks numerik. Misalnya, `array[0]`, `array[1]`, dst. Array biasanya digunakan ketika kita memiliki kumpulan data yang jumlahnya tetap dan indeksnya dapat dihitung dengan mudah.

Map: Map adalah struktur data yang menyimpan kumpulan pasangan kunci-nilai, di mana kunci dan nilainya dapat berupa tipe data yang berbeda. Kunci digunakan untuk mengakses nilai terkait. Map biasanya digunakan ketika kita memiliki data yang tidak memiliki indeks berurutan atau ketika kita perlu mencocokkan kunci dengan nilai tertentu. Dalam contoh ini, kita menggunakan nama orang sebagai kunci untuk mengakses umur mereka.

BAB IV

KESIMPULAN

Laporan ini bertujuan untuk memperkenalkan mahasiswa pada konsep dasar tipe data dalam pemrograman, termasuk tipe data primitif, abstrak, dan koleksi. Praktikum ini juga bertujuan untuk memberikan pemahaman tentang pengaplikasian tipe data tersebut dalam bahasa pemrograman yang telah ditentukan, serta penggunaannya dalam alat-alat yang digunakan dalam pengembangan perangkat lunak. Dari implementasi program-program tersebut, mahasiswa diharapkan dapat memahami dan menguasai konsep tipe data serta struktur data yang diperlukan dalam pemrograman. Kesimpulan singkat dari laporan ini adalah praktikum ini memberikan landasan yang kuat bagi mahasiswa untuk memahami dan menerapkan konsep dasar tipe data dalam pemrograman, serta meningkatkan pemahaman mereka tentang struktur data yang digunakan dalam pengembangan perangkat lunak.