BAB III
TIPE DATA BENTUKAN

A. TUJUAN

1. Mahasiswa mampu mengetahui dan memahami berbagai jenis tipe data bentukan
2. Mahasiswa mampu memahami class
3. Mahasiswa mampu memahami struct
4. Mahasiswa mampu memahami union
5. Mahasiswa mampu memahami enumeration
6. Mahasiswa mampu memahami typedef
7. Mahasiswa mampu mengimplementasikan tipe data bentukan berdasarkan kasus yang diberikan menggunakan bahasa pemrograman C++.

B. ALAT DAN BAHAN

1. Laptop
2. Mouse
3. IDE C++ (Visual Studio Code)

C. KESEHATAN DAN KESELAMATAN KERJA

1. Hati-hatilah dalam memakai perangkat elektronik
2. Pastikan kabel listrik terpasang dan dalam kondisi baik
3. Lakukan praktikum dalam posisi duduk yang benar
4. Jauhkan kabel listrik dari sentuhan anda
5. Gunakan alas kaki celana Panjang dan kemeja
6. Gunakan kacamata anti radiasi layar

D. TEORI

1. Tipe Data Bentukan

Tipe data bentukan merupakan tipe data yang dibentuk atau dibuat sendiri sesuai kebutuhan dalam program yang akan kita buat. Dalam bahasa asing, tipe ini dikenal dengan istilah User Defined Types. Tipe data bentukan diperoleh dari tipe data yang sudah ada (tipe data primitive) namun dianggap terpisah dan tidak seperti tipe data dasar tersebut. Adapun jenis tipe data bentukan, yaitu:

- Class
- Structure
- Union

- Enumeration
- Typedef

2. Class

a. Deklarasi Class

Blok C++ yang mengarah ke pemrograman Berorientasi Objek adalah Class (Kelas). Class adalah tipe data yang dibentuk oleh pengguna (tipe data bentukan) yang menyimpan data di dalam blok dan fungsi anggota (varaibel member atau member function) yang ada di dalamnya. Class dapat diakses dan digunakan dengan cara membuat objek dari class tersebut. Class seperti cetak biru dari suatu objek.

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.

Fitur class adalah fitur OOP pada C++ mirip seperti fitur struktur data struct pada C, keduanya dapat menampung variabel sebagai anggota. Tapi perbedaan class dan struct tersebut adalah pada class memiliki kemampuan lebih, seperti dimungkinkan untuk memuat method, inheritance dan lain-lain. Adapun sintaks untuk membuat sebuah kelas adalah sebagai berikut:

Sintaks I:

```
Class nama_class {
   Tipe_akses:
   Variabel/data_member;
   Fungsi member(){}
};
```

Sintaks II:

```
Class nama_class {
   Tipe_akses:
   Variabel/data_member;
   Fungsi member(){}
}nama_objek;
```

Keterangan:

- **nama_class**: tempat dimana anda dapat memberikan nama pada class tersebut.
- **Tipe_akses**: tempat dimana anda dapat mendifinisikan hak akses kepada anggota yang ada dibawahnya. Jika tidak didefinisikan maka anggota class secara otomatis memiliki hak akses private.
- **data_member**: tempat dimana anda dapat mendirikan sebuah variabel atau function sebagai anggota dari class.
- **nama_objek**: tempat dimana anda dapat mendirikan object-object dengan menggunakan class tersebut, hal ini merupakan opsional tapi jika class tidak diberi nama class: maka diwajibkan mendirikan object, karena kita tidak akan bisa membuat object dengan class di luar dari deklarasi class tersebut.

Class dan object merupakan fitur yang sangat membantu untuk membangun sebuah program besar, menjadikan kode program yang ditulis oleh programmer mudah untuk dimengerti, lebih terstruktur, dan juga mudah dalam pemeliharaan program.

Terdapat banyak fasilitas yang disediakan oleh class, yaitu dapat menampung member variabel, function, constructor, desctructor, overloading dan lain-lain. Diluar definisi class kita juga dimungkinkan untuk membuat relalasi seperti inheritance dan overriding.

b. Mendeklarasikan Object di Luar Class

Mendirikan object sama seperti kita mendirikan variabel, yang berbeda adalah nama dari class akan digunakan sebagai tipe data dari object. Adapun sintaks untuk mendeklarasikan sebuah objek dari suatu class adalah sebagai berikut:

Sintaks I:

```
nama_kelas nama_object;
```

Sintaks II:

```
nama_kelas nama_object1,nama_object2;
```

Sintaks III:

```
class nama_kelas nama_object;
```

c. Mengakses Object pada Class

Setelah kita berhasil mendirikan object. Object tersebut akan sepenuhnya memiliki bentuk berdasarkan rancangan pada class yang dipakai. Untuk mengakses object dan member dari object tersebut kita membutuhkan "Member Access Operator ." yang diletakan di antara nama object dan nama member. Adapun sintaks untuk mendeklarasikan sebuah objek dari suatu class adalah sebagai berikut:

Sintaks I:

```
nama_kelas.nama_object;
```

d. Mendeklarasikan Object di Luar Class

1. Non-static Member Initialization

   Merupakan cara untuk memberikan nilai awal (Inisialisasi) di saat mendirikan variabel sebagai member class. Maka nilai awal itu akan berlaku pada semua object yang menggunakan class tersebut. dan cara ini hanya tersedia pada C++11 ke atas.

2. Uniform Initialization

   Merupakan cara kedua untuk inisialisasi member pada object. Dengan cara memberikan nilai awal kepada member class saat mendirikan object, inisialisasi harus disebutkan dengan urut berdasarkan urutan deklarasi member. inisialisasi ini hanya akan berlaku pada object itu sendiri. Tapi kelemahanya, cara ini hanya bisa dilakukan jika semua member dari class bersifat public.

e. Access Modifier

Access Modifier (kadang juga disebut Access Specifier) seperti ditunjukkan pada Tabel 3.1 adalah salah satu fitur penting dalam Object Oriented Programming (OOP) untuk melakukan Data Hiding (menyembunyikan data). Fitur ini memungkinkan kita untuk mengatur hak akses dari member class, digunakan agar tidak sembarangan perintah dapat mengakses, atau tidak bisa di akses secara langsung.

Fitur ini memiliki 3 tipe Access Modifier, yaitu:

- Public

  Public adalah label yang berfungsi untuk menentukan sifat akses ke semua member yang mengikutinya (di bawahnya), sehingga memiliki sifat dapat di akses dari manapun. Dapat di akses dari dalam class itu sendiri, dari anak class (derived class) dan juga dari luar class.

- Private

  Private adalah label yang berfungsi untuk menentukan sifat akses ke semua member yang mengikutinya menjadi memiliki sifat yang tidak dapat di akses dari manapun kecuali melalui friend function dan dari dalam class itu sendiri.

- Protected

  Protected adalah label yang berfungsi untuk menentukan sifat akses semua member yang mengikutinya, sehingga memiliki sifat yang tidak dapat diakses dari luar class tapi masih dapat di akses dari dalam class maupun anak class (derived class).

Tabel 3.1. Tipe akses

Access Modifier	Dalam class	Subclass (Anak class)	World (luar kelas)
Public	Ya	Ya	Ya
Protected	Ya	Ya	Tidak
Private	Ya	Tidak	Tidak

3. Structure

a. Deklarasi Struct

Sesuai namanya, structure adalah tipe data bentukan yang menyimpan lebih dari satu variable bertipe sama maupun berbeda. Dengan kemampuan itu struktur dapat digunakan untuk mengelompokkan beberapa data dengan tipe data berbeda dalam sebuah tipe data baru. Dengan menggunakan structure, kita dapat membentuk tipe data baru yang merupakan gabungan dari beberapa variabel. Adapun sintaks untuk membuat sebuah structure adalah sebagai berikut:

Sintaks I:

```
struct nama_struct{
   tipe_data variabel1; //member
   tipe_data variabel2; //member
```

```
    ….
};
```

Sintaks II:

```
struct nama_struct {
    tipe_data variabel1; //member
    tipe_data variabel2; //member
    ….
}nama_object;
```

Keterangan:

- **nama_struct**: Tempat dimana anda dapat memberikan nama pada struct tersebut.
- **nama_object**: Merupakan deklarasi yang menggunakan struct tersebut menjadi tipe data dari deklarasi tersebut. kita dapat mendirikan banyak object di tempat tersebut, masing-masing dipisahkan dengan tanda koma. Object selalu diletakan setelah penutup block dan sebelum semicolon.

b. Mengakses Member pada Struct

Member adalah variabel yang didirikan di dalam struct, dijadikan sebagai satu bagian dari struct. Kita tidak bisa menggunakan member-member dari struct sebelum didirikan object yang didirikan menggunakan data structure tersebut. karena struct hanyalah sebuah rancangan dari tipe structure.

Setelah kita berhasil membuat sebuah deklarasi object. untuk mengakses member struct dari object tersebut kita membutuhkan Member Access Operator. di antara nama object dan nama anggota variabel struct. Adapun sintaks untuk mendeklarasikan sebuah objek dari suatu class adalah sebagai berikut:

Sintaks I:

```
nama_object.nama_member;
```

c. Cara Inisialisasi Object pada Struct

1. Non-static Member

   Suatu tindakan memberikan nilai awal member structure kepada object dengan cara memberikan nilai di saat mendirikan member dari struct. Dengan cara tersebut, nilai inisialisasi akan berlaku ke semua object yang menggunakan tipe structure tersebut.

   Adapun sintaks untuk mengakses elemen array adalah sebagai berikut:

   Sintaks I:

```
struct nama_struct{
    tipe_data nama_variabel = nilai_inisialisasi;
};
```

2. Initializer List

   Cara inisialisasi dengan memberikan nilai awal disaat mendirikan object, nilai tersebut hanya akan berlaku pada object itu sendiri.

   Sintaks I:

```
struct nama_struct nama_object = {nilai_member1,
   nilai_member2, ...};
```

3. Uniform Initialization

   Merupakan inisialisasi yang masih sama seperti initializer list, hanya saja tidak menggunakan tanda =.

   Sintaks I:

```
nama_struct nama_object {nilai_member1, nilai_member2,
   ...};
```

4. Union

Seperti Structures, union adalah tipe data yang ditentukan pengguna. Dalam union, semua anggota berbagi lokasi memori yang sama. Misalnya dalam program berikut, baik x dan y berbagi lokasi yang sama. Jika kita mengubah x, kita dapat melihat perubahan yang tercermin dalam y.

Sintaks I:

```
union nama_union {
   tipe_data variabel1;
   tipe_data variabel2;
   ….
};
```

5. Enumeration

Tipe data enumerasi adalah tipe data yang nilainya terbatas pada nilai-nilai yang didefenisikan sendiri. Tipe enumerasi digunakan untuk membentuk tipe data yang nilainya bersifat pasti. Misalnya untuk mendefenisikan tipe jenis kelamin, nama hari, nama warna dan lain sebagainya. Nilai-nilai ini ditentukan oleh programmer pada saat menyatakan tipe yang disebutkan.

Kata kunci enum digunakan untuk mendeklarasikan tipe enumerasi setelah itu nama tipe enumerasi ditulis kemudian di dalam kurung kurawal, nilai yang mungkin ditentukan. Setelah mendefinisikan variabel tipe Enumerated dibuat. Itu dapat dibuat dalam dua jenis:

- Itu bisa dideklarasikan saat mendeklarasikan tipe enumerasi, cukup tambahkan nama variabel sebelum titik koma.
- Selain itu, kita dapat membuat variabel tipe enumerasi sama dengan variabel normal.

Sintaks I:

```
enum nama_enumerasi{nilai1, nilai2, nilai3…..nilaiN};
```

6. Typedef

C ++ memungkinkan Anda untuk mendefinisikan nama tipe data baru secara eksplisit dengan menggunakan kata kunci typedef. Menggunakan typedef sebenarnya tidak membuat kelas data baru, melainkan mendefinisikan nama untuk tipe yang sudah ada. Ini dapat meningkatkan portabilitas (kemampuan suatu program untuk digunakan di berbagai jenis mesin; yaitu, mini,

mainframe, mikro, dll; tanpa banyak perubahan dalam kode) dari suatu program karena hanya pernyataan typedef yang harus diubah.

Tipe alias adalah nama berbeda yang dengannya suatu tipe dapat diidentifikasi. Dalam C ++, semua type yang valid dapat dialiaskan sehingga dapat disebut dengan pengidentifikasi yang berbeda. Adapun sintaks untuk mendeklarasikan sebagai berikut:

Sintaks I:

```
typedef tipe_data nama_baru_tipe;
```

E. PRAKTIKUM

Praktikum 3.1. Class, Struct, dan Typedef

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

using namespace std;

class Mahasiswa {
  public:
    string motto ="Saya mahasiswa Teknik Komputer. Saya Bangga!";

    typedef struct dataStatus {
      string nim;
      string nama;
      string prodi;
    }dtStatusmhs ;

    dtStatusmhs dtStatusmaba;

    void cetakData(string nim, string nama, string prodi, string motto){
      cout<<"=== Data Mahasiswa ==="<< nim << endl;
      cout<<"NIM Mahasiswa :"<< nim << endl;
      cout<<"Nama Mahasiswa :"<< nama << endl;
      cout<<"Prodi Mahasiswa :"<< prodi << endl;
      cout<<"Motto Mahasiswa :"<< motto << endl;
    };

};

int main( int argc, char const *argv[]) {
```

```
  Mahasiswa tekomB;

  cout<<"Masukkan NIM Mahasiswa :"<<endl;
  getline (cin, tekomB.dtStatusmaba.nim);
  cout<<"Masukkan nama mahasiswa :"<<endl;
  getline (cin, tekomB.dtStatusmaba.nama);
  cout<<"Masukkan prodi mahasiswa :"<<endl;
  getline(cin, tekomB.dtStatusmaba.prodi);
  tekomB.cetakData ( tekomB.dtStatusmaba.nim, tekomB.dtStatusmaba.nama,
tekomB.dtStatusmaba.prodi, tekomB.motto);

  return 0;
}
```

Praktikum 3.2. Class file terpisah

1. Ketikkan kode berikut pada Main.cpp

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

using namespace std;

int pilihan;
float panjang;
float lebar;
float diameter;

void inputPersegi(){
    cout<<"Masukkan panjang sisi (cm): "<<endl;
    cin>>panjang;
};

void inputPersegipanjang(){
    cout<<"Masukkan panjang sisi (cm): "<<endl;
    cin>>panjang;
    cout<<"Masukkan lebar sisi (cm): "<<endl;
    cin>>lebar;
};

void inputLingkaran(){
    cout<<"Masukkan diameter (cm): "<<endl;
    cin>>diameter;
};

int main( int argc, char const *argv[]) {
```

```
  cout <<"=== PROGRAM MENGHITUNG LUAS BANGUN DATAR ==="<<endl;
  cout <<"Silahkan memilih bangun datar:"<<endl;
  cout <<"1. Persegi"<<endl;
  cout <<"2. Persegi Panjang"<<endl;
  cout <<"3. Lingkaran"<<endl;
  cout <<"Silahkan input angka pilihan:"<<endl;
  cin >> pilihan;

  if (pilihan == 1){
    cout <<"Anda memiluh persegi"<<endl;
    Bangundatar newPersegi;
    inputPersegi();
    newPersegi.tampilLuas(newPersegi.luasPersegi(panjang));
  } else if (pilihan==2){
    cout <<"Anda memilih persegi panjang"<<endl;
    Bangundatar newPersegipanjang;
    inputPersegipanjang();
    newPersegipanjang.tampilLuas(newPersegipanjang.luasPersegipanjang(panj
ang, lebar));
  } else if (pilihan == 3){
    cout <<"Anda memilih lingkaran"<<endl;
    Bangundatar newLingkaran;
    inputLingkaran();
    newLingkaran.tampilLuas(newLingkaran.luasLingkaran(diameter));
  } else {
    cout <<"Anda salah input!"<<endl;
  }

  return 0;
}
```

2. Buat file baru bernama “Bangundatar.cpp” dan “Bangundatar.h” pada folder src
3. Pada file Bangundatar.cpp, ketikkan kode berikut.

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


float Bangundatar::luasPersegi(float panjang){
    float luasPersegi = panjang*panjang;
    return luasPersegi;
};

float Bangundatar::luasPersegipanjang (float panjang, float lebar){
    float luasPersegiPanjang = panjang * lebar;
    return luasPersegiPanjang;
```

```
};

float Bangundatar::luasLingkaran(float diameter){
    const float PHI = 3.14f;
    float jariJari = diameter/2;
    float luasLingkaran = PHI * (jariJari*jariJari);
    return luasLingkaran;
};

void Bangundatar::tampilLuas (float luas){
    std::cout<< "Luas yang diperoleh adalah: " << luas << " cm2" <<
std::endl;
};
```

4. Pada file Bangundatar.h, ketikkan kode berikut.

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

class Bangundatar {
public:
    float luasPersegi(float panjang);
    float luasPersegipanjang (float panjang, float lebar);
    float luasLingkaran(float diameter);
    void tampilLuas(float luas);
};
```

Praktikum 3.3. Class file terpisah (inheritance dan keyword this)

1. Ketikkan kode berikut pada Main.cpp

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

using namespace std;

class Subbangundatar : public Bangundatar{
  public:
  float lebar;
  void luasPersegipanjang (float panjang, float lebar){
    this->panjang = panjang;
    this->lebar = lebar;
    float luas = this->panjang * this->lebar;
```

```
    cout <<"Luas persegi panjang adalah: "<< luas << " cm2" << endl;
  }
};

int main( int argc, char const *argv[]) {
  int pilihan;
  cout <<"=== PROGRAM MENGHITUNG LUAS BANGUN DATAR ==="<<endl;
  cout <<"Silahkan memilih bangun datar:"<<endl;
  cout <<"1. Persegi"<<endl;
  cout <<"2. Persegi Panjang"<<endl;
  cout <<"Silahkan input angka pilihan:"<<endl;
  cin >> pilihan;

  if (pilihan == 1){
    float panjang;
    cout <<"Anda memilih persegi"<<endl;
    Subbangundatar newPersegi;
    cout<<"Masukkan panjang persegi (cm): "<<endl;
    cin >> panjang;
    newPersegi.luasPersegi(panjang);

  } else if (pilihan==2){
    float panjang;
    float lebar;
    cout <<"Anda memilih persegi panjang"<<endl;
    Subbangundatar newPersegipanjang;
    cout<<"Masukkan panjang persegi (cm): "<<endl;
    cin >> panjang;
    cout<<"Masukkan lebar persegi (cm): "<<endl;
    cin >> lebar;
    newPersegipanjang.luasPersegipanjang(panjang, lebar);
  } else {
    cout <<"Anda salah input!"<<endl;
  }

  return 0;
}
```

2. Buat file baru bernama “Bangundatar.cpp” dan “Bangundatar.h” pada folder src
3. Pada file Bangundatar.cpp, ketikkan kode berikut.

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

```
void Bangundatar::luasPersegi(float panjang) {
    this->panjang = panjang;
    float luas =  this->panjang* this->panjang;
    cout <<"Luas persegi adalah: "<< luas << endl;
};
```

4. Pada file Bangundatar.h, ketikkan kode berikut.

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

class Bangundatar
{
      public:
      int panjang;
      void luasPersegi(float panjang);

};
```

5. Jalankan program!

Praktikum 3.4. Class, enum, dan keyword this

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

class Datadiri {
  public:

    string nama = "Ahmad Albar";
    int usia = 26;
    enum Gender { Male,
    Female };
    Gender gender = Male;

    void tampil(){
      cout<<this->nama<<endl;
      cout<<this->usia<<endl;
      if (this->gender==0){
        cout<<"Pria"<<endl;
      } else if (this->gender==1){
        cout<<"Wanita"<<endl;
      } else {
        cout<<"Salah input!"<<endl;
      }
    }
};
```

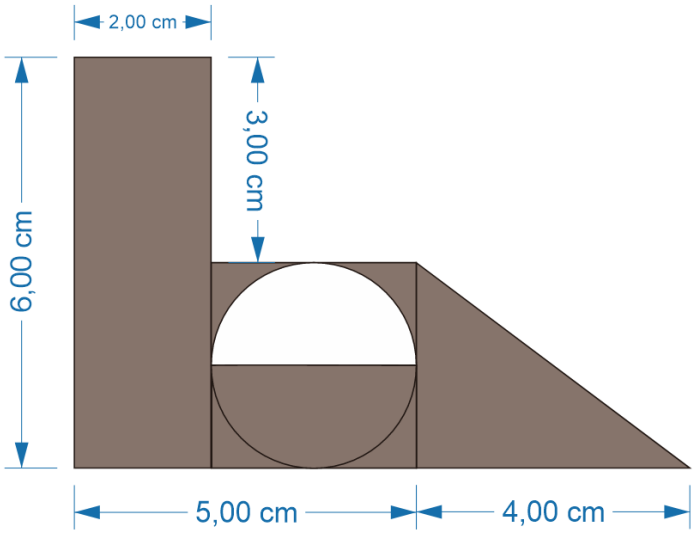
```
int main( int argc, char const *argv[]) {

    Datadiri datadiri;
    datadiri.tampil();

  return 0;
}
```

F. LATIHAN

1. Perhatikan Gambar berikut ini



Hitunglah luas area yang berwarna coklat?

Catatan!

1. Buat flowchart untuk menghitung luas area berwarna coklat (putih tidak termasuk)!
2. Buat program berdasarkan flowhart yang telah anda buat untuk menghitung luas area yang berwarna coklat tersebut!
3. Program wajib menggunakan class yang berisi method luas berbagai jenis bangun datar yang dibutuhkan!

CATATAN