**MEMBUAT PROGRAM CONSOLE SEDERHANA DENGAN BAHASA PEMROGRAMAN C#**

PRAKTIKUM 2

Laporan ini dibuat untuk memenuhi persyaratan kelulusan

mata kuliah pemrograman II



NAMA : WILDAN AZRIL ARVANY

NPM : 1204011

KELAS : 2A

PRODI : D-IV TEKNIK INFORMATIKA

**PROGRAM STUDI D IV TEKNIK INFORMATIKA**

**POLITEKNIK POS INDONESIA**

**2021**

**DAFTAR ISI**

**DAFTAR ISI1**

**TABEL AKTIVITAS2**

**BAB I PENDAHULUAN3**

1.1 Enkapsulasi3

1.2 Class3

1.3 Deklarasi Class4

1.4 Atribut dan Method4

1.5 Class Member5

1.6 Object5

1.7 Access Modifier6

**BAB II PEMBAHASAN7**

2.1 Percobaan7

2.2 Latihan12

**BAB III PENUTUP15**

3.1 Kesimpulan15

3.2 Saran16

**DAFTAR PUSTAKA17**

**TABEL AKTIVITAS**

|  |  |
| --- | --- |
| **TANGGAL** | **CATATAN** |
| 12 November 2021 | Mempelajari Modul dan Mengerjakan Laporan Bab 1 |
| 13 November 2021 | Mengerjakan Laporan Bab 1 dan Bab 3 |
| 14 November 2021 | Mengerjakan Percobaan |
| 15 November 2021 | Mengerjakan Latihan dan Video |

**BAB I**

**PENDAHULUAN**

* 1. **ENKAPSULASI**

Enkapsulasi adalah suatu cara untuk menyembunyikan informasi detail (information hiding) dari suatu kelas. Information hiding adalah proses penyembunyian informasi dari suatu kelas dengan memberikan akses kontrol private pada atribut atau method sehingga anggota atribut atau method tersebut tidak dapat diakses dari luar kelas. Tanpa information hiding, anggota kelas seperti atribut dan method dapat secara langsung diakses dari luar kelas yang melingkupinya, namun dengan information hiding perlu adanya interface untuk menjadi perantara agar atribut dan method yang sudah disembunyikan tersebut tetap dapat diakses dan/atau dimodifikasi dari luar kelas.

Enkapsulasi adalah teknik pembuatan kelas pribadi (private) dan menyediakan akses melalui metode publik. Kelas pribadi tidak dapat diakses oleh siapapun di luar kelas, sehingga enkapsulasi dirujuk sebagai data bersembunyi. Enkapsulasi sebagai pelindung yang mencegah kode dan data yang diakses secara acak oleh kode lainnya yang didefinisikan di luar kelas. Akses ke data dan kodenya dikontrol menggunakan interface. Manfaat utama enkapsulasi adalah kemampuan untuk mengubah kode tanpa melanggar kode orang lain yang menggunakan kode tersebut.

* 1. **CLASS**

Class adalah rancangan atau sketsa dari sebuah object ,sebelum membuat object kita harus membuat rancangan dari objek tersebut terlebih dahulu,secara umum class memiliki 2 macam anggota , yaitu method dan field, field adalah atribut yang dimiliki objek dari rancangan (class) tersebut.

Class merupakan suatu tipe data yang menggabungkan data dan fungsi untuk mengakses data. Sebagai contoh suatu kelas kalkulator yang megandung data bilangan1 dan bilangan2 dan memiliki fungsi untuk mengakses data tersebut, seperti : inisialisasi(), penjumlahan(), pengurangan(), perkalian() dan pembagian().

Data dan fungsi yang berada dalam sebuah class dinamakan anggota. Secara lebih khusus , data yang terletak dalam kelas dinamakan anggota data dan fungsi yang berada dalam kelas disebut anggota fungsi atau fungsi anggota atau terkadang disebut metode.

* 1. **DEKLARASI CLASS**

C# menyediakan beberapa tipe data bawaan seperti int, double, bool, dll. Selain tipe data bawaan yang telah disediakan, c# juga bisa membuat tipe data yang dapat definsisikan sendiri, salah satunya dengan menggunakan tipe class. Deklarasi Class sendiri mempunyai pengertian yaitu penjabaran lebih lanjut dari suatu tipe data yang akan di buat.

* 1. **ATRIBUTE DAN METHOD**

Atribut adalah data yang membedakan antara object satu dengan yang lainnya. Dalam class Atribut sering disebut dengan variabel. Atribut dibagi menjadi 2 yaitu : Instance variabel dan Class variabel. Variable yang berupa attribut ini juga disebut dengan variable global (kebalikan dari variable lokal) karena variabel global tersebut bisa diakses di setiap behavioral. Attribute merupakan karakteristik dari suatu class. Attribut ini berupa suatu variable yang terletak tepat di dalam class. Cara mendeklarasikan attribut sama dengan cara mendeklarasikan variable karena attribut juga merupakan variable.

Method merupakan sebutan untuk behavioral/function. Method selalu memiliku kurung lengkung atau “()”, kurung lengkung tersebut bisa juga disemati suatu variable atau parameter. Parameter sendiri adalah sebutan dari variabel yang terletak dalam kurung lengkung suatu method. Aturan penamaan method sama dengan aturan penamaan variable. Pembahasan tersebut bisa dilihat di penjelasan tentang variabel. Method sendiri ada dua jenis: (1) void method yang tidak mengembalikan suatu nilai. Dilihat sekilas, void method dapat diidentifikasi dengan adanya kata kunci “void” di depan nama method. dan (2) return method yang mengembalikan nilai. Method ini bisa diidentifikasi dengan adanya data type di depan nama method-nya dan kata kunci return di dalam method-nya.

* 1. **CLASS MEMBER**

Class pada c# memiliki anggota yang mewakili data dan perilaku mereka. Class Member mencakup semua anggota yang dideklarasikan di class, bersama dengan semua anggota (kecuali konstruktor dan finalizer) yang dideklarasikan di semua kelas dalam hierarki pewarisannya. Private Member Class diwariskan tetapi tidak dapat diakses dari kelas turunan.

Class Member dilihat dari jenis :

* Constants (Konstanta) mewakili nilai konstan
* Fields (Bidang) yang mewakili variabel
* Methods (Metode) yang menyediakan layanan seperti perhitungan atau tindakan lain pada anggotanya
* Properties (Properti) yang mendefinisikan fitur kelas dan menyertakan tindakan untuk mengambil dan memodifikasinya
* Events (Acara) yang dihasilkan untuk berkomunikasi antara kelas/objek yang berbeda
* Indexers (Pengindeks) yang membantu mengakses instance kelas yang mirip dengan array
* Operators (Operator) yang mendefinisikan semantik ketika digunakan dalam ekspresi dengan instance kelas
* Instance untuk menginisialisasi anggota instance kelas
* Statis untuk menginisialisasi kelas itu sendiri
* Dectructors (Destruktor) untuk mengeksekusi tindakan yang perlu dilakukan sebelum instance kelas dibuang
* Types (Tipe) yang lokal ke kelas (tipe bersarang)

Tidak seperti Java di mana tingkat aksesibilitas default anggota kelas adalah publik, ini bersifat pribadi di C#.

* 1. **OBJECT**

Object adalah data yang telah terstruktur sesuai dengan yang didefinisikan dalam sebuah class. Object adalah instance (hasil salinan/cetakan) dari classnya. Object atau Objek adalah hasil cetak dari class, atau hasil ‘konkrit’ dari class. Proses mencetak objek dari class ini disebut dengan instansiasi. Hasil cetakan class akan disimpan dalam variabel untuk selanjutnya digunakan dalam proses program.

* 1. **ACCESS MODIFIER**

Access Modifier (Access Specifier) adalah salah satu fitur penting dalam Object Oriented Programming (OOP) untuk melakukan Data Hiding (Menyembunyikan Data). Fitur ini memungkinkan untuk mengatur hak akses dari member class, digunakan agar tidak sembarangan perintah dapat mengakses, atau tidak bisa di akses secara langsung. Access modifier menentukan apakah suatu class, method, field, property, event atau hal-hal lainnya dapat diakses di luar class induknya.

Macam – macam dari Access Modifier yaitu :

* public membuat member yang bersangkutan dapat diakses dari mana saja.
* private membuat member yang bersangkutan hanya dapat diakses dari dalam class itu sendiri.
* protected membuat member yang bersangkutan hanya dapat diakses dari class itu sendiri atau class lain yang merupakan turunan (inherits) dari class yang bersangkutan.
* internal membuat member yang bersangkutan hanya dapat diakses dari assembly yang sama, atau dengan kata lain, aplikasi yang sama.
* protected internal menggabungkan protected dan internal, yang berarti member yang bersangkutan hanya dapat diakses dari class itu sendiri atau dari class lain dari dalam assembly yang sama yang merupakan turunan dari class yang bersangkutan.

**BAB II**

**PEMBAHASAN**

**2.1 PERCOBAAN**

|  |  |
| --- | --- |
| **Repository Github** | <https://github.com/azrilarva21/TUGAS_PEMROGRAMAN_2_PART2/tree/main/P2_1_1204011> |
| **Last Commit** | **15 November 2021** |
| **Youtube** | <https://youtu.be/VhM9eiiuSyc> |

|  |  |
| --- | --- |
| using System.Threading.Tasks;**;** | |
| **Penjelasan** | Threading Tasks adalah suatu kelas yang memungkinkan untuk membuat tugas dan menjalankannya secara asinkron. Tugas tersebut merupakan objek yang mewakili beberapa pekerjaan yang harus dilakukan, serta dapat memberi informasi jika pekerjaan selesai dan operasi mengembalikan hasil |

|  |  |
| --- | --- |
| namespace P1\_1\_1204011 | |
| **Penjelasan** | Namespace pada C# berfungsi untuk mengatur banyak kelas sehingga mudah menangani aplikasi |

|  |  |
| --- | --- |
| class Program**;** | |
| **Penjelasan** | Kelas yang menggambarkan beberapa objek layaknya blueprint untuk membuat suatu objek |

|  |  |
| --- | --- |
| public void Main(string[] args) | |
| **Penjelasan** | Suatu fungsi yang harus ada dalam setiap program, apabila fungsi ini tidak di masukkan maka program tidak bisa dijalankan. Fungsi main() merupakan fungsi utama yang akan dieksekusi pertama kali. Oleh sebab itu, biasanya programming banyak menulis kode program didalam fungsi main(). |

|  |  |
| --- | --- |
| Console.WriteLine("Product 1 is a {0} called \"{1}\" and has {2} pages"); | |
| **Penjelasan** | Untuk menampilkan pernyataan yang ada didalam tanda kurung. |

|  |  |
| --- | --- |
| Book\_1204011 product1 = new Book\_1204011("Book","C# Object Oriented Solution", "300"); | |
| **Penjelasan** | Suatu fungsi untuk Membuat instance baru. Dalam kode diatas menampilkan instance bernama product1 dari class Book\_1204011 dengan attribut Book, C# Object Oriented Solution, 300 |

|  |  |
| --- | --- |
| public class Product\_1204011  {  private string myType;  private string myTitle;  public Product\_1204011()  {  }  public Product\_1204011(string type, string title)  {  this.myType = type;  this.myType = title;  }  public string MyType  {  get  {  return myType;  }  set  {  myType = value;  }  }  public string MyTitle  {  get  {  return myTitle;  }  set  {  myTitle = value;  }  }  } | |
| **Penjelasan** | Membuat sebuah class bernama Product\_1204011 dengan attribut string bernama myType dan myTItle, kemudian membuat constructor dari class tersebut dengan parameter string mytype dan string mytitle yang akan mengisi value dari parameter tersebut kedalam variabel myType dan myTitle. Kemudian dibawah consturctor ada 2 property yang menujukkan masing masing ke variabel myType dan myTitle. |

|  |  |
| --- | --- |
| public class DVD\_1204011 : Product\_1204011  {  protected string duration;  public DVD\_1204011(string title, string duration)  {  this.MyTitle = title;  this.Duration = duration;  this.MyType = "DVD";  Console.WriteLine("Ini dari class DVD");  }  public string Duration  {  get  {  return duration;  }  set  {  duration = value;  }  }  } | |
| **Penjelasan** | Membuat sebuah class bernama DVD\_1204011 dan men-inherit property dari Product\_1204011. Kemudian membuat variabel protected string bernama duration. Setelah itu, membuat constructor dari class tersebut dengan parameter title dan duration yang bertipe string. Kemudian menuliskan baris “Ini dari class DVD”. Dibawah constructor, ada sebuah property yang digunakan untuk mengakses dan mengatur value dari duration.  . |

|  |  |
| --- | --- |
| public class DVD\_1204011 : Product\_1204011  {  protected string duration;  public DVD\_1204011(string title, string duration)  {  this.MyTitle = title;  this.Duration = duration;  this.MyType = "DVD";  Console.WriteLine("Ini dari class DVD");  }  public string Duration  {  get  {  return duration;  }  set  {  duration = value;  }  }  } | |
| **Penjelasan** | Membuat sebuah class bernama Book\_1204011 dan men-inherit property dari Product\_1204011. Kemudian membuat variabel protected string bernama pageCount. Setelah itu, membuat constructor dari class tersebut dengan parameter mytype,mytile, dan pagecount yang bertipe string. Kemudian, mytype dan mytile dimasukkan ke class member kelas asal dengan keyword base. Dibawah constructor, ada sebuah property yang digunakan untuk mengakses dan mengatur value dari pageCount. |

Jawaban Pertanyaan :

**Bagian I :**

**No. 14**

Pesan kesalahan tersebut muncul karena pada 'Product\_1204011' tidak mengandung definisi untuk 'MyType' dan tidak ada metode ekstensi 'MyType' yang menerima argumen pertama tipe 'Product\_1204011' dapat ditemukan. Konsep OOP dalam fenomena tersebut yaitu *access modifier* yang mana MyType menyembunyikan datanya, karena data pada MyType ditandai dengan *Protected* yang berada di class Product, maka dari itu data MyType tidak akan muncul pada ProductTest.

**No.23**

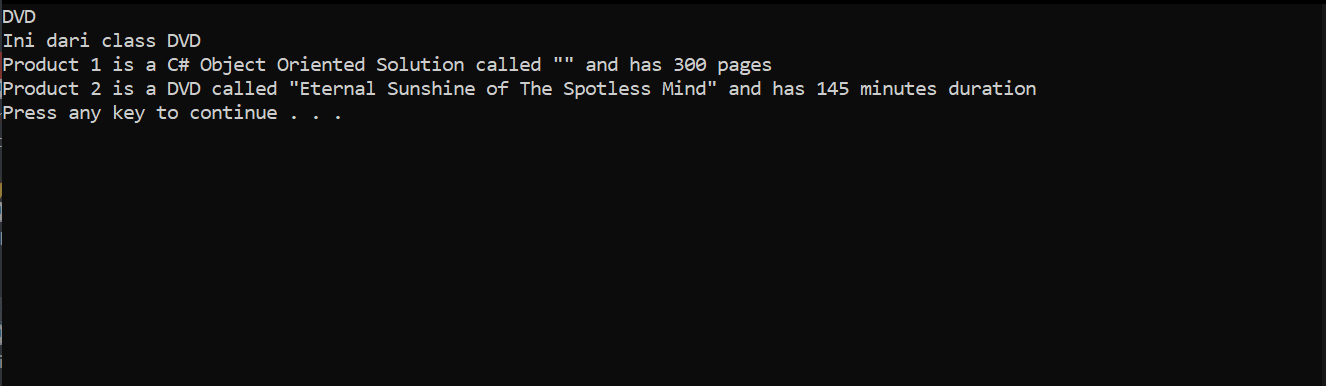
Pada layar muncul kata “DVD” Karena property MyType sudah dibuat dan pada koding tersebut mengakses ke object myProduct yang memilki value “DVD” untuk property myType.

**Bagian II**

**No.9**

Title Buku pada layar tersebut tidak muncul karena property asal dari super class DVD tersebut belum di atur/ dipasangkan.

**Hasil Output PERCOBAAN**

****

**2.2 LATIHAN**

|  |  |
| --- | --- |
| **Repository Github** | <https://github.com/azrilarva21/TUGAS_PEMROGRAMAN_2_PART2/tree/main/P2_2_1204011> |
| **Last Commit** | **15 November 2021** |
| **Youtube** | <https://youtu.be/sp7Pwkfv2KA> |

|  |  |
| --- | --- |
| Karakter hero1 = new Karakter("Downey Jr", "Iron Man", "Avenger"); | |
| **Penjelasan** | suatu fungsi untuk Membuat sebuah instance. Dalam kode diatas instance yang dibuat diambil dari class Karakter dengan parameter “Downey Jr" , “Iron Man”, “Avenger” |

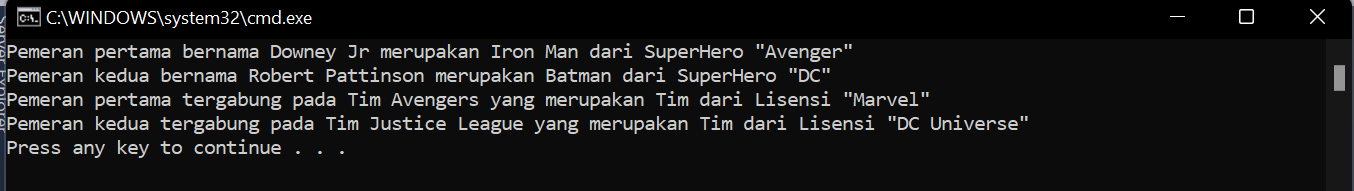
|  |  |
| --- | --- |
| Console.WriteLine("Pemeran pertama bernama {0} merupakan {1} dari SuperHero \"{2}\"", hero1.Nama, hero1.Tipe, hero1.Tipe\_Karakter); | |
| **Penjelasan** | Suatu fungsi untuk menampilkan pesan yang terdapat di dalam kurung. Dalam kode diatas pesan yang akan ditampilkan yaitu Pemeran pertama bernama {0} ( nilai 0 akan diisi dengan hero1.nama ) merupakan {1} (nilai 1 akan diisi dengan hero2.Tipe) dari SuperHero {2} (nilai 2 akan diisi dengan hero1.Tipe\_Karakter). |

|  |  |
| --- | --- |
| public class Aktor  {  private string nama;  private string tipe;  public Aktor(string nama, string tipe)  {  this.nama = nama;  this.tipe = tipe;  }  public string Nama  {  get  {  return nama;  }  set  {  nama = value;  }  }  public string Tipe  {  get  {  return tipe;  }  set  {  tipe = value;  }  }  } | |
| **Penjelasan** | Kode di atas merupakan proses Membuat sebuah class bernama **Aktor** yang memiliki attribute private string nama dan tipe. Selain itu, juga memiliki constructor yang memiliki parameter nama dan tipe yang digunakan untuk memasukkan value ke attribute nama dan tipe. Dibawahnya ada property Nama dan Tipe yang digunakan untuk memasukkan value nama dan value tipe. |

|  |  |
| --- | --- |
| public class Karakter : Aktor  {  protected string tipe\_karakter;  public Karakter(string nama, string tipe, string tipe\_karakter):  base(nama, tipe)  {  this.tipe\_karakter = tipe\_karakter;  }  public string Tipe\_Karakter  {  get  {  return tipe\_karakter;  }  set  {  tipe\_karakter = value;  }  }  } | |
| **Penjelasan** | Membuat sebuah class bernama **Karakter** yang memiliki attribute protected string nama tipe\_karakter. Dibawah itu, ada constructor yang memiliki parameter nama tipe, dan tipe\_karakter yang digunakan untuk memasukkan value ke tipe\_karakter dan keyword base yang digunakan untuk memasukkan ke class member kelas asal. Dibawahnya ada property Tipe\_Karakter yang digunakan untuk memasukkan value tipe\_karakter |

|  |  |
| --- | --- |
| public class Tim : Aktor  {  protected string lisensi;  public Tim(string nama, string lisensi) : base (nama, "Tim")  {  this.lisensi = lisensi;  }  public string Lisensi  {  get  {  return lisensi;  }  set  {  lisensi = value;  }  }  } | |
| **Penjelasan** | Membuat sebuah class bernama **Tim** yang memiliki attribute protected string nama lisensi. Dibawah itu, ada constructor yang memiliki parameter nama dan lisensi yang digunakan untuk memasukkan value ke Lisensi dan keyword base yang digunakan untuk memasukkan ke class member kelas asal. Dibawahnya ada property Lisensi yang digunakan untuk memasukkan value lisensi. |

**Hasil Output Latihan**

****

**BAB III**

**PENUTUP**

**3.1 KESIMPULAN**

Enkapsulasi adalah teknik pembuatan kelas pribadi (private) dan menyediakan akses melalui metode publik. Manfaat utama enkapsulasi adalah kemampuan untuk mengubah kode tanpa melanggar kode orang lain yang menggunakan kode tersebut.

Class adalah rancangan atau sketsa dari sebuah object ,sebelum membuat object kita harus membuat rancangan dari objek tersebut terlebih dahulu,secara umum class memiliki 2 macam anggota , yaitu method dan field, field adalah atribut yang dimiliki objek dari rancangan (class) tersebut.

Deklarasi Class mempunyai pengertian yaitu penjabaran lebih lanjut dari suatu tipe data yang akan di buat.

Attribute merupakan karakteristik dari suatu class. Attribut ini berupa suatu variable yang terletak tepat di dalam class. Cara mendeklarasikan attribut sama dengan cara mendeklarasikan variable karena attribut juga merupakan variable.

Method merupakan sebutan untuk behavioral/function. Method ini bisa diidentifikasi dengan adanya data type di depan nama method-nya dan kata kunci return di dalam method-nya.

Class pada c# memiliki anggota yang mewakili data dan perilaku mereka. Class Member mencakup semua anggota yang dideklarasikan di class, bersama dengan semua anggota (kecuali konstruktor dan finalizer) yang dideklarasikan di semua kelas dalam hierarki pewarisannya.

Object adalah data yang telah terstruktur sesuai dengan yang didefinisikan dalam sebuah class. Object adalah instance (hasil salinan/cetakan) dari classnya.

Access Modifier (Access Specifier) adalah fitur untuk melakukan Data Hiding (Menyembunyikan Data). Access modifier menentukan apakah suatu class, method, field, property, event atau hal-hal lainnya dapat diakses di luar class induknya.

**3.2 SARAN**

Saran yang bisa penulis berikan perlu adanya pembelajaran lebih lanjut akan pembelajaran mengenai Pemrograman II sebagai salah satu cara memaksimalakan pembelajaran dalam mata kuliah.

**DAFTAR PUSTAKA**

Artikel Doavers Development Team, 2016

<https://www.doavers.com/blog/belajar-pemrograman-berbasis-objek-oop-di-c-bagian-2>

Mudjihartono, Rekayasa Pembalikan Kode Berorientasi Objek ke Desain Kelas Dengan Struktur Data Graf, Jurnal Buana Informatika, 2010

<https://ojs.uajy.ac.id/index.php/jbi/article/view/301/381>

Artikel ajengtriansari, Class,Attribute dan Methode, 2012

<https://ajengtriansari.wordpress.com/2012/10/11/71/>

Artikel Dasar Komputer Buat Pemula, 2016

<https://pccontrol.wordpress.com/dasar-c/>

Artikel Class Members, tecnopedia, 2011

<https://www.techopedia.com/definition/25589/class-members-c-sharp>

Artikel Class dan Objek pada c#, 2013

[https://icodeformoney.com/tuts/Class-dan-objek-pada-CSharp#](https://icodeformoney.com/tuts/Class-dan-objek-pada-CSharp)

Artikel Belajar c++, 2019

<https://www.belajarcpp.com/tutorial/cpp/access-modifier/>