Asrianda, S. Kom, M. Kom

TEORI & IMPLEMENTASI
PEMOGRAMAN
C#

UNIMAL PRESS

TEORI DAN IMPLEMENTASI
PEMOGRAMAN C#


universitas
MALIKUSSALEH

Asrianda, S. Kom, M. Kom

TEORI DAN IMPLEMENTASI
PEMOGRAMAN C#

UNIMAL PRESS

Judul: **TEORI DAN IMPLEMENTASI PEMOGRAMAN C#**
xii + 144 hal., 15 cm x 23 cm

Cetakan Pertama: November, 2018
Hak Cipta © dilindungi Undang-undang. *All Rights Reserved*

Penulis:
Asrianda, S. Kom, M. Kom

Perancang Sampul:
Penata Letak: Eriyanto
Pracetak dan Produksi: **Unimal Press**

Penerbit:

UNIMAL PRESS

Unimal Press
Jl. Sulawesi No.1-2
Kampus Bukit Indah Lhokseumawe 24351
PO.Box. 141. Telp. 0645-41373. Fax. 0645-44450
Laman: www.unimal.ac.id/unimalpress.
Email: unimalpress@gmail.com

ISBN: 978-602-464-053-8

Dilarang keras memfotocopy atau memperbanyak sebahagian atau seluruh buku ini tanpa seizin tertulis dari Penerbit

Kata Pengantar

Dengan mengucapkan puji dan syukur kehadirat Allah SWT, dengan rahmat dan karunia-Nya yang telah memberi hidayah kepada Penulis untuk menyelesaikan buku yang berjudul "Teori dan Implementasi Bahasa Pemograman C#" tidak lupa selawat dan salam kepada Rasullullah SAW.

Rasa terima kasih Penulis ucapkan kepada Almarhum dan Almarhummah Ayahanda Muhammad dan Ibunda Hasanah, Almarhum Abanda Marzuki, Kakanda Fauziah, Abanda Fakhrurrazie, Abanda Muzakkir, Al Chaidar dan keponakan penulis Dara Nurfika Sari yang telah banyak membantu Penulis dalam menyelesaikan Buku ini. Dan juga saya ucapkan banyak terima kasih kepada teman-teman di Program Studi Teknik Informatika Universitas Malikussaleh yang telah memberi semangat dan inspirasi kepada Penulis untuk menyelesaikan buku ini.

Dengan buku ini Penulis ingin membagi pengalaman yang sering Penulis alami dalam menyusun buku referensi pemograman Visual Studio 2010, berguna untuk mahasiswa di Universitas Malikussaleh yang minimnya bahan pustaka dan buku pembelajaran dalam menguasai bahasa pemograman. Buku ini bukan hal yang baru tetapi hasil kumpulan referensi yang telah ada baik di dunia maya maupun buku yang telah banyak beredar di pasaran., dan Penulis juga merasa ilmu yang Penulis dapatkan selama ini belum ada apa-apanya di dunia Pemograman, minimal dengan buku ini dapat membuka wawasan bagi para mahasiswa, juga masyarakat umum dan para teman-teman sejawat Dosen yang ingin mempelajari Bahasa Pemograman C# dan dapat mengembangkan lebih lanjut lagi.

Kritik dan saran dari pembaca sekalian sangatlah membantu bagaimana buku ini harus disusun dan disajikan lebih baik lagi.

Lhokseumawe, November 2018

Penulis

Daftar Isi

Daftar Gambar

∞

BAB 1.
PENGENALAN PEMOGRAMAN C#

1.1 Framework .NET

Sejak pertama kali diperkenalkan di tahun 2000, versi alpha Framewo.NET sudah menjadi perhatian dunia pemrograma.Framework.NET versi 1 keluar pada akhir 2001. Banyak yang mengatakan bahwa Framework.NET memulai era pemrograman baru. Gartner Group memperkirakan dalam jangka waktu yang singkat dunia pemrograman akan didominasi Framework.NET. Di bawah ini akan diuraikan beberapa kelebihan Framework.NET (sumber pustaka Mempelajari C Bahasa Pemrograman Modern):

- **Platform Independence :** Program-program yang dibuat untuk berjalan di atas Framework.NET dapat dijalankan di komputer apapun, asalkan di komputer yang bersangkutan terinstall implementasi Framework.NET. Implementasi Framework.NET untuk Linux yang dinamakan Mono sudah dapat digunakan (http://www.go-mono.com. Microsoft telah membuat Framework.NET untuk sistem lain misalnya untuk Mac.
- **Language Independence dan Cross Language Interoperability:** Tidak seperti platform Java (JVM) yang terkait dengan bahasa Java, platform.NET tidak terkaitdengan bahasa pemograman apapun. Contoh-contoh bahasa yang bisa dipakai untuk pemrograman.NET adalah C#, Managed C++, Visual Basic.NET, Jscript.NET, Visual J++.NET, COBOL.NET, Eiffel#, Phyton.NET, Pascal.NET, dan Perl.NET. Hal yang sangat menarik adalah kemampuan berinteraksi program yang dibuat dengan bahasa berbeda secara langsung, Program-program managed juga dapat berinteraksi dengan program-program unmanaged (dan sebaliknya), walaupun kecepatannya tidak maksimal.
- **Kecepatan :** Program-program .NET walaupun portable akan berjalan dengan kecepatan tinggi, bahkan dapat lebih cepat dari program-program unmanaged.
- **Keamanan :** Karena semua program .NET berjalan di bawah pengawasan CLR, maka keamanan menjadi lebih terjamin.

- **Produktifitas Tinggi :** Konsep OOP yang tertanam kuat memungkinkan pembuatan program yang dapat dengan mudah dikembangkan. Kekayaan class library .NET juga sangat berperan dalam produktifitas.
- **Support untuk Network/Internet:** Framework .NET dirancang dengan internet "inmind". Interaksi komponen yang terletak di dua komputer yang berbeda dapatdilakukan secara mudah.
- **.NET adalah Platform yang Terbuka:** Siapapun boleh membuat bahasa dan compiler yang menghasilkan program.NET, bahkan membuat implementasi.NET itu sendiri. Spesifikasi-spesifikasi yang berhubungan dengan Framework.NET diserahkan oleh Microsoft kepada ECMA (suatu badan standar internasional), tidak seperti Sun Microsystems yang menolak untuk menyerahkan Java kepada badan standar walaupun sudah didesak banyak pihak. Framework.NET juga banyak menggunakan teknologi-tekonologi yang sudah dipakai secara luas seperti XML,Unicode,HTTP, dan TCP/IP

1.2 Lahirnya C#

Microsoft membuat C# seiring dengan pembuatan Framework .NET. Chief Architect dalam pembuatan C# adalah Anders Hejlsberg yang sebelumnya berperandalam pembuatan Borland Delphi dan Turbo Pascal. C# menjanjikan produktifitas dan kemudahan yang ada di Visual Basic dengan kemampuan dan fleksibilitas yang ada di C/C++. Pemograman C# sesuai kaidah bahasanya " C# (*pronounced "C Sharp")* is a simple, modern, object oriented, and type-safe programming language. It will immediately be familiar to C and C++ programmers. C# combines the high productivity of Rapid Application Development (RAD) languages andthe raw power of C++" (sumber pustaka Mempelajari_C_Bahasa_Pemrograman_Modern).

Untuk mencapai produktifitas tinggi ini konsep-konsep sulit C++ disederhanakan dan fitur-fitur baru ditambahkan. Hal ini mungkin terasa mirip dengan Java, karena itulah C# bisa dianggap sebagai sepupu Java.

C# adalah salah satu dari banyak bahasa yang bisa dipakai untuk pemrograman.NET. Kelebihan utama bahasa ini adalah sintaksnya yang mirip C, namun lebih mudahdan lebih bersih. Untuk perbandingan penulis cantumkan sedikit informasi mengenai Managed C++ dan Visual Basic.NET:

- **Managed C++:** Managed C++ adalah ekstensi terhadap C++ untuk membuat program.NET. Salah satu keunikan Managed C++ adalah kita bisa mencampur kode-kode managed dengan unmanaged dalam bahasa pemograman yang dibuat. Ini sangat berguna bagi pihak-pihak yang sudah memiliki banyak kode C++ namun ingin bermigrasi ke platform.NET. Dalam pemrograman Managed C++ masih akan terikat dengan konsep-konsep sulit C++ sehingga produktifitas akan lebih rendah dibanding jika menggunakan C#.
- **Visual Basic.NET:** Perbedaan antara C# dengan Visual Basic .NET yang langsung terlihat adalah sintaksnya. C# memiliki beberapa fitur yang tidak ada di Visual Basic.NET sehingga C# sedikit lebih fleksibel. Visual Basic .NET cukup berbeda dengan Visual Basic 6, Visual Basic .NET adalah bahasa yang sepenuhnya berorientasi objek dan dibuat untuk pemrograman .NET.

Tentang kecepatan program yang dihasilkan, semua bahasa.NET menghasilkan program.NET yang berkecepatan tinggi.Perbedaan kecepatan yang ada sangat kecil bahkan pada umumnya bisa dianggap tidak ada

1.3 Microsoft Visual Studio 2010

Microsoft Visual Studio adalah sebuah lingkungan pengembangan terpadu (IDE) dari Microsoft. Hal ini digunakan untuk mengembangkan program komputer untuk sistem operasi Microsoft Windows super famili, serta situs web, aplikasi web dan layanan web. Visual Studio menggunakan Microsoft platform pengembangan perangkat lunak seperti API Windows, Windows Forms Windows Presentation Foundation, Windows Stor dan Microsoft Silverlight.

Hal ini dapat menghasilkan baik kode asli dan kode yang dikelola . Visual Studio mencakup kode editor pendukung IntelliSense serta refactoring kode. Terintegrasi debugger bekerja baik sebagai source-level debugger dan debugger mesin-tingkat. Built-in tools termasuk desainer bentuk untuk membangun GUI aplikasi, web designer , kelas desainer, dan skema database desainer. Menerima plug-in yang meningkatkan fungsionalitas pada hampir setiap tingkat-termasuk menambahkan dukungan untuk sumber-kontrol sistem (seperti *Subversion* dan Visual *SourceSafe*) dan menambahkan toolsets baru seperti editor dan desainer visual untuk bahasa domain-spesifik atau toolsets untuk aspek-aspek lain dari siklus pengembangan perangkat lunak (seperti Team Foundation Server klien: Tim Explorer).

Visual Studio mendukung berbagai bahasa pemrograman dan memungkinkan kode editor dan debugger untuk mendukung hampir semua bahasa pemrograman, memberikan layanan bahasa spesifik ada. Built-in bahasa termasuk C , C + + dan C + + / CLI (melalui Visual C + +), VB.NET (melalui Visual Basic. NET), C # (via Visual C #), dan F # (seperti Visual Studio 2010). Dukungan untuk bahasa lain seperti M , Python , dan Ruby antara lain tersedia melalui layanan bahasa diinstal secara terpisah.

Ini juga mendukung XML / XSLT , HTML / XHTML , JavaScript dan CSS. Individu versi bahasa-spesifik Visual Studio juga ada yang menyediakan layanan bahasa yang lebih terbatas bagi pengguna: Microsoft Visual Basic, Visual J #, Visual C #, dan Visual C + + (sumber pustaka: KITAB BELAJAR PEMOGRAMMAN C#).

1.4 Tampilan Microsoft Visual Studio 2010

Microsoft Visual Studio 2010 Profesional adalah alat penting untuk individu melakukan tugas-tugas dasar pemograman. Ini menyederhanakan penciptaan, debugging, dan penyebaran aplikasi pada berbagai platform, termasuk SharePoint dan Cloud. Visual Studio 2010 Profesional dilengkapi dengan dukungan terpadu untuk pengembangan uji-didorong, serta alat debugging yang membantu memastikan solusi berkualitas tinggi. Menulis kode aplikasi sering membutuhkan banyak memiliki desainer dan editor terbuka sekali.

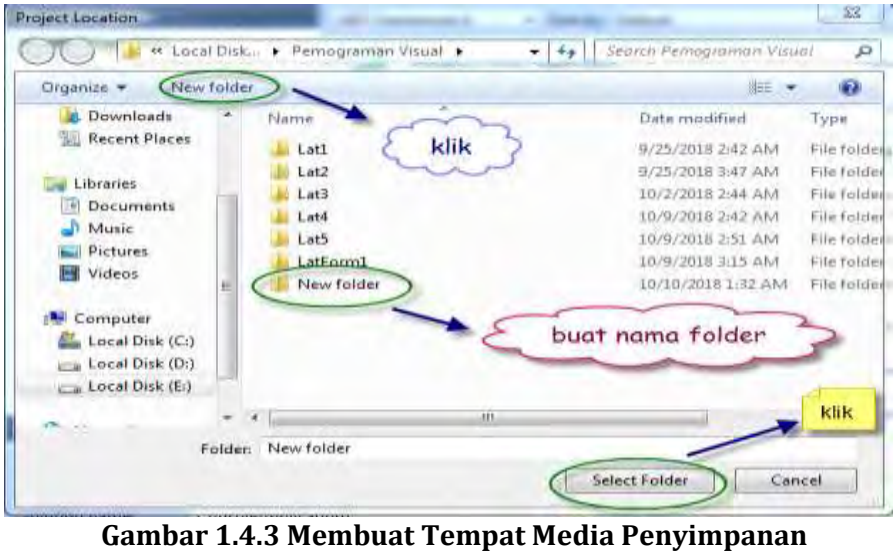
Untuk memulai aplikasi Microsoft visual studio, kamu mulai dengan klik menu **File**, dan klik **New Project** untuk menampilkan kotak dialog. Perhatikan gambar 1 yang sedang menampilkan sebuah kotak dialog yang dihasilkan dari New Project.

Gambar 1.4.1. Menu Tampilan Visual Studio 2010

Pada tampilan sebelah kiri, itu adalah daftar template. Pilih sesuai kebutuhan yang akan dibuat. Tapi untuk mengikuti tutorial ini, gunakan **Console Application** pada template yang ada disebelah kanan.

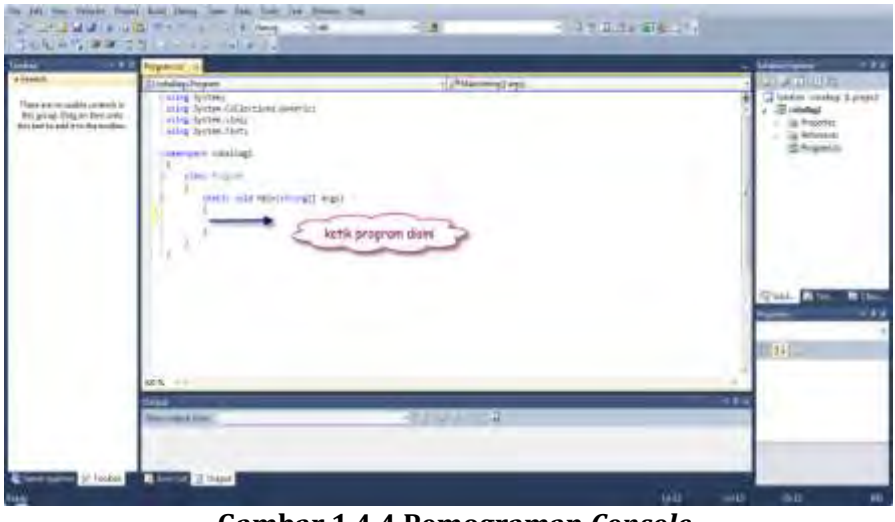
Gambar 1.4.2 Console Aplication

Setelah membuat nama program, nama program itu dibuat secara default menjadi **ConsoleApplication1**, nama ini dapat diganti sesuai keinginan pemograman. Kemudian memilih media tempat penyimpanan. Klik tombol **OK** untuk melanjutkan.



Gambar 1.4.3 Membuat Tempat Media Penyimpanan

Tunggu beberapa detik, kecepatan dalam menjalankan program tergantung seberapa besar ram yang dimiliki. Jika mengikuti perintah dengan benar, akan muncul tampilan seperti gambar berikut.



Gambar 1.4.4 Pemograman *Console*

Tampilan yang muncul seperti gambar diatas tidak selalu sama dengan versi lain. Jika menggunakan versi dibawah aplikasi ini, akan ada fitur-fitur yang tidak tersedia seperti gambar diatas. Jika menggunakan versi ini, seperti Microsoft Visual Studio 2010.

∞

BAB 2.
DASAR PEMOGRAMAN C#

2.1 Variabel

Suatu variabel hanyalah suatu penampung data atau nilai di dalam program. Di dalam dunia pemrograman, setiap variabel memiliki tipe data dan dalam kebanyakan bahasa pemrograman, termasuk C#, tipe data suatu variabel ditentukan ketika variabel tersebut dinyatakan atau diisi. Terdapat beberapa cara untuk menyatakan variabel:

- [tipe data] [identifier];
- [tipe data] [identifier] = [nilai];
- [tipe data] [identifier 1], [identifier 2], [identifier N];
- [tipe data] [identifier 1] = [nilai 1], [identifier 2] = [nilai 2], [identifier N] = [nilai N];

Sebagai contoh

- int jumlahSemuaBarang;
- string namaPelanggan = "Fandi";
- double harga1, harga2, harga3;
- int jumlah1 = 10, jumlah2 = 7, jumlah3 = 12;

Contoh pertama dan ketiga di atas hanya menyatakan variabel saja, tidak memberikan nilai apapun untuk variabel variabel tersebut. Jika suatu variabel kosong dibaca oleh program, akan terjadi *runtime error*. Variabel-variabel kosong seperti ini harus diberi nilai sebelum dibaca.

Tipe-tipe data yang digunakan di atas adalah tipe data yang paling sering digunakan, int adalah bilangan bulat, string adalah kumpulan karakter, dan double adalah bilangan real. Kita akan belajar mengenai tipe data pada bagian selanjutnya dari tutorial ini.

2.2 Identifier

Identifier adalah suatu nama yang mewakili hal-hal di dalam program. Identifier dapat mewakili suatu variabel, konstanta, metoda atau fungsi, properties, class, enum atau struct. Pada C#, berlaku ketentuan-ketentuan berikut pada Identifier:

- Harus unik, tidak boleh terduplikasi.

- Bersifat *case sensitive*, Nama dan nama merupakan dua identifier yang berbeda.
- Harus dimulai dengan huruf atau *underscore*, angka diperbolehkan setelah karakter pertama.
- Tidak mengandung spasi. Jika terdiri dari lebih satu kata, disarankan menggunakan underscore sebagai pengganti spasi atau menggunakan *CamelCase*.
- Tidak mengandung karakter khusus (@, $, #), ataupun operator (+, -, /, *, dsb)
- Bukan merupakan keyword (for, if, class, using, continue, break, dsb)

2.3 Tipe Data

Meskipun banyak tipe data pada C#, tipe-tipe data yang paling sering digunakan adalah *int, double, string, bool* dan *object*. Masing-masing memiliki kegunaannya sendiri:

- **int**, digunakan untuk menyatakan suatu bilangan bulat, positif maupun negatif, sebagai contoh jumlah murid dalam satu kelas.
- **double**, digunakan ketika kita perlu menyatakan suatu bilangan pecahan, misalkan nilai, harga atau berat. Akan tetapi, double adalah data dengan ketelitian 15-16 digit. Sehingga terkadang tipe data **float** dengan ketelitian 7 digit lebih disukai untuk contoh contoh tadi, dengan alasan menghemat memori.
- **string**, digunakan ketika menyatakan sekumpulan karakter, baik itu nama, kalimat, password ataupun kode. String sesungguhnya merupakan suatu array dari **char**.
- **bool**, bernilai true atau false. Paling sering digunakan untuk menyatakan suatu kondisi, misalnya IsEnabled, IsVisible, SedangHujan, SedangLapar, dan lain lain.
- **object**, adalah class dasar dari semua tipe data dalam C#. Digunakan untuk menyatakan tipe data yang tidak pasti.

Tipe-tipe data lainnya adalah:

- **Mirip dengan int**: sbyte, short, long, untuk tipe bertanda, byte, ushort, uint, ulong untuk tipe yang tak bertanda
- **Mirip dengan double**: float dan decimal
- **char**, untuk menyimpan satu karakter

2.4 Konversi tipe data antar angka pada C#

Terkadang kita dapat langsung menugaskan nilai antar tipe data yang berbeda. Misalnya jika merubah suatu nilai *int* ke variabel *long*, tidak ada perintah khusus yang dibutuhkan. Jika kita menugaskan tipe data yang lebih kecil ke tipe data yang lebih besar, C# akan secara implisit mengkonversi nilai tersebut. Akan tetapi jika tipe data yang lebih besar ditugaskan ke tipe data yang lebih kecil, diperlukan konversi tipe data. Sebagai contoh:

```
1  double harga = 32;
2  double diskon = 0.2; //20% off
3  double hargaDiskon = harga - (harga * diskon);
4
5  Console.WriteLine("Harga dasar: " + harga);
6  Console.WriteLine("Diskon     : " + diskon * 100 + "%");
7  Console.WriteLine("Harga      : " + hargaDiskon);
8  //Harga dasar: 32
9  //Diskon     : 20%
10 //Harga      : 25.6
11
12 Console.ReadKey();
```

Penting diketahui bahwa jika suatu tipe data yang lebih besar dikonversi ke tipe data yang lebih kecil, dapat terjadi kehilangan data. Hal-hal ini juga berlaku ketika mengkonversi antara tipe data bilangan bulat dengan tipe data *floating point*. Selalu ingat bahwa berkurangnya ketelitian dapat terjadi, harus berhati-hati dalam memilih tipe data yang sesuai.

2.5 Konversi tipe data string pada C#

Tipe yang paling banyak digunakan untuk input, output dan penyimpanan sebetulnya adalah string. Untuk mengkonversi tipe data apapun ke string sesungguhnya cukup sederhana karena setiap tipe data di C# memiliki metode *ToString()*. Sebagai contoh:

```
1  bool AdaMatahari = true;
2  int angka = 15;
3  double pecahan = 70.88;
4
5  Console.WriteLine(AdaMatahari.ToString()); //True
6  Console.WriteLine(angka.ToString());   //15
7  Console.WriteLine(pecahan.ToString()); //70.88
8
9  Console.ReadKey();
```

Di sisi lain, agak sulit mengkonversi dari string ke tipe lain. Semua tipe data numerik memiliki metoda *Parse* dan *TryParse*. Kita

menggunakan ini untuk mengkonversi string ke tipe data numerik. Kita menggunakan Parse ketika kita SANGAT yakin mengenai string yang bersangkutan, ketika kita tidak yakin, kita gunakan *TryParse*. Sebagai contoh:

```
string strBool = Console.ReadLine();
bool aBool = Boolean.Parse(strBool);
Console.WriteLine(aBool);

string strInt = Console.ReadLine();
Console.WriteLine(Int32.Parse(strInt));

string strDouble = Console.ReadLine();
double aDouble;
bool BisaParse = double.TryParse(strDouble, out aDouble);
if (BisaParse) Console.WriteLine(aDouble); else Console.WriteLine("To
Console.ReadKey();
```

Coba masukkan **"True"** atau **"FaLsE"** pada *prompt* pertama, dan *input* yang *valid* pada *prompt* kedua dan ketiga. Program akan mengeluarkan kembali *input* yang diterima. Kemudian coba masukkan *string* yang bukan *boolean* ke dalam prompt pertama, akan terjadi *runtime error*. Dan terakhir, coba *input string* yang bukan angka pada prompt ketiga, program akan menampilkan "Tolong input suatu angka". Perlu diperhatikan bahwa *runtime error* pada saat *debugging* berarti *Critical Error* pada software yang sudah jadi. Hal ini harus dihindari semampunya.

2.6 Literasi

Istilah literal atau literal *constant* merujuk pada suatu nilai pada program yang tidak dapat diubah. Kita menggunakan literal untuk menyatakan nilai-nilai. Literal-literal pada C# dijelaskan dalam tabel di berikut:

Tipe data	Literal	Contoh
bool	true atau false	true
int	Bilangan bulat	0, -200, 87329
int	Awalan 0x untuk menyatakan bilangan heksadesimal	0xF, 0xaeaeae
double	Bilangan pecahan dengan titik untuk menyatakan koma	72.59034
double	e untuk menyatakan pangkat sepuluh	6.02e23, 1.602e-19
string	Karakter-karakter yang diapit oleh petik dua	"6f8c11# ^_^ <3"
char	Satu karakter yang diapit oleh petik satu	'K'

Gambar 2.1 Literal (https://icodeformoney.com)

Untuk tipe data string, terdapat karakter karakter yang tidak dapat diketik langsung ke dalam petik dua. Untuk mewakili karakter-karakter tersebut, digunakan *escape sequences*. *Escape sequence* yang umum digunakan antara lain:

Escape Sequence	Karakter yang diwakili
\'	Single quote
\"	Double quote
\\	Backslash
\0	Null, not the C# null value
\a	Bell
\b	Backspace
\f	Form feed
\n	Newline
\r	Carriage Return
\t	Horizontal Tab
\v	Vertical Tab

Gambar 2.2 *Escape sequence* (https://icodeformoney.com)

Sebagai contoh apabila kita ingin menampung `mysql < "path\ke\file\sqldump"` ke dalam suatu string:

```
string command = "mysql < \"path\\ke\\file\\sqldump\"";
```

(sumber pustaka : https://icodeformoney.com/)

2.7 Batas nilai tipe-tipe data numerik pada C#

Bagian ini tidak terlalu penting, jika Anda lebih menghargai waktu dibanding pengetahuan, lewati bagian ini. Salah satu alasan untuk menggunakan tipe data lainnya adalah karena batasan nilai. Batasan nilai pada tiap tipe data di C# dipaparkan pada tabel berikut:

Tipe data	Penjelasan	Batas nilai
sbyte	Bilangan bulat 8 bit bertanda	−128 sampai 127
short	Bilangan bulat 16 bit bertanda	−32,768 sampai 32,767
int	Bilangan bulat 32 bit bertanda	−2,147,483,648 sampai 2,147,483,647
long	Bilangan bulat 64 bit bertanda	−9,223,372,036,854,775,808 sampai 9,223,372,036,854,775,807
byte	Bilangan bulat 8 bit tak bertanda	0 sampai 255
ushort	Bilangan bulat 16 bit tak bertanda	0 sampai 65,535
uint	Bilangan bulat 32 bit tak bertanda	0 sampai 4,294,967,295
ulong	Bilangan bulat 64 bit tak bertanda	0 sampai 18,446,744,073,709,551,615
float	Floating point berketelitian 7 digit	±1.5e−45 sampai ±3.4e38
double	Floating point berketelitian 15-16 digit	±5.0e−324 sampai ±1.7e308
decimal	Bilangan dengan 28-29 angka penting	$(-7.9 \times 10^{28}$ sampai $7.9 \times 10^{28}) / (10^{0}$ to $^{28})$

Gambar 2.3 Batas tipe data numerik (https://icodeformoney.com)

Bilangan bulat bertanda sesungguhnya menggunakan 1 bit dari ukuran data mereka untuk menyatakan tanda (negatif atau positif). Itulah kenapa batas nilai positif dari tipe tak bertanda dua kali lebih besar dari tipe yang bertanda dengan ukuran data yang sama. Akan tetapi bilangan bulat bertanda sudah jarang digunakan kecuali ketika berurusan dengan antarmuka *hardware*, pengolahan citra, atau pemrograman tingkat rendah lainnya. Dan juga, sangat disarankan hindari menggunakan tipe data *decimal* karena tipe data ini tidak lazim pada bahasa pemrograman yang lain.

2.8 Jenis-jenis Operator

Ada banyak operator bawaan pada C# yang dapat dikategorikan ke dalam operator aritmatika, operator penugasan, operator pembanding, operator logika dan operator bitwise. Akan tetapi di sini kita tidak akan membahas semua operator tersebut. Kita hanya akan membahas seperlunya saja. Pertama-tama, saya ingin memperkenalkan operator penugasan yang telah kita gunakan beberapa kali sebelumnya, yaitu operator "=". Kita menggunakan operator ini untuk memberi nilai ke suatu variabel. Sebagai contoh:

```
int anInt = 30;
```

Pernyataan di atas akan mengakibatkan anInt menyimpan nilai 30. Berikutnya adalah operator aritmatika, yaitu:

Sebagai contoh, misalkan kita punya a = 15, b = 10

Operator	Keterangan	Contoh	Hasil
+	Penjumlahan	a + b	25
-	Pengurangan	a - b	5
*	Perkalian	a * b	150
/	Pembagian	a / b	1
%	Modulus	a % b	5
++	Increment	a++	16
--	Decrement	a--	14
-	Minus	-a	-15

Gambar 2.4 Operator Aritmatika (https://icodeformoney.com)

Perhatikan bahwa pada contoh pembagian di atas, 15 dibagi 10 seharusnya 1,5. Hal ini benar untuk tipe data double. Tetapi pada contoh di atas diasumsikan a dan b adalah integer yang mana akan membulatkan ke bawah setiap bilangan pecahan positif. Terdapat prioritas dalam operator-operator di atas. Operator increment, decrement dan minus akan dihitung duluan, kemudian baru modulus, perkalian dan pembagian, terakhir barulah operator pengurangan dan penjumlahan. Jika perlu untuk mengubah urutan eksekusi, tanda kurung dapat digunakan. Di bawah ini adalah contoh sederhana operasi aritmatika:

```
double harga = 32;
double diskon = 0.2; //20% off
double hargaDiskon = harga - (harga * diskon);

Console.WriteLine("Harga dasar: " + harga);
Console.WriteLine("Diskon     : " + diskon * 100 + "%");
Console.WriteLine("Harga      : " + hargaDiskon);
//Harga dasar: 32
//Diskon     : 20%
//Harga      : 25.6

Console.ReadKey();
```

Hasil perhitungan hargaDiskon sesungguhnya akan tetapi sama meskipun tanpa tanda kurung. Akan tetapi seringkali, lebih mudah untuk menggunakan tanda kurung daripada harus meninjau kembali urutan prioritas operator. Pernyataan yang bersangkutan juga akan lebih mudah dibaca dengan menyertakan tanda kurung. Perhatikan bahwa tanda "+" di dalam metoda *Console.WriteLine* bukanlah operator aritmatika, melainkan operator penyambungan string. Ketika + digunakan di antara angka, ia menjadi operator aritmatika, akan tetapi ketika digunakan di antara string dengan tipe data apa saja, ia menjadi operator penyambungan string.

Terdapat juga operator yang di sebut operator majemuk. Terkadang kita ingin menambahkan sesuatu ke nilai yang sebelumnya, sebagai contoh, kita ingin menaikkan harga menjadi 34, kita dapat menuliskan harga = harga + 2; Tetapi dengan operator majemuk, kita cukup menuliskan harga += 2; yang akan menghasilkan hasil yang sama. Operator-operator majemuk antara lain:

Sebagai contoh jika a = 15

Operator	Keterangan	Contoh	Hasil
+=	Penambahan dan penugasan	a += 30	45
-=	Pengurangan dan penugasan	a -= 3	12
*=	Perkalian dan penugasan	a *= 5	75
/=	Pembagian dan penugasan	a /= 5	3
%=	Modulus dan penugasan	a %= 6	3

Gambar 2.5 Operasi Operator Aritmatika (https://icodeformoney.com)

∞

BAB 3.
STATEMENT DI PEMOGRAMAN C#

3.1 Pemograman *Console*

Struktur program C# yang paling dasar, terdiri dari tiga bagian:

1. Bagian deklarasi pustaka
2. Bagian Class
3. Bagian Fungsi atau Method

```
//-- Deklarasi Pustaka ---
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace coballagi
{
    //-- Bagian Class ---
    class Program
    {
        //-- Bagian Fungsi --
        static void Main(string[] args)
        {
            Console.WriteLine("Program pertama saya");
            Console.ReadKey();
        }
    }
}
```

1. Bagian Pustaka

Ini adalah bagian paling atas dari program C#. Pada bagian ini, kita menuliskan pustaka ***(library)*** yang dibutuhkan dalam program. Apa itu pustaka?

Pustaka berisi sekumpulan fungsi, method, class, objek, konstanta, dan variabel yang bisa kita gunakan ulang di dalam program. Sebagai contoh:

```
//-- Deklarasi Pustaka ---
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

Pustaka ini kita butuhkan untuk menggunakan class Console dan method *WriteLine().*

2. Bagian Class

Bahasa pemrograman C# adalah bahasa pemrograman yang menggunakan paradigma OOP *(Object Oriented Programming)* atau pemrograman berorientasikan objek. Setiap kita membuat program C#, kita harus membungkus fungsi dan variabel di dalam **class**.

"Class adalah sebuah rancangan atau *blue print* dari objek."

3. Baigan Fungsi

Pada bagian ini, kita bisa menuliskan fungsi-fungsi dari program. Fungsi yang harus ada di dalam setiap program adalah fungsi **Main().** Kalau tidak ada fungsi ini, program tidak akan bisa dijalankan. Karena fungsi **Main()** merupakan fungsi utama yang akan dieksekusi pertama kali. Oleh sebab itu, kita biasanya akan banyak menulis kode program di dalam fungsi **Main().**

3.2 Struktur Program C#

Penulisan statemen dan ekspresi dalam C# harus diakhiri dengan titik koma (;). Kalau tidak maka nanti akan error.

Contoh:

```
//-- Deklarasi Pustaka ---
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace coballagi
{
    //-- Bagian Class ---
    class Program
    {
        //-- Bagian Fungsi --
        static void Main(string[] args)
        {
            Console.WriteLine("Program pertama saya");
            Console.WriteLine("Saya Belajar Pemograman C#");
            Console.ReadKey();
        }
    }
}
```

3.3. Penulisan Blok Kode

Blok kode di dalam C# dibungkus menggunakan kurung kurawal **{ ... }**.

Contoh

```
//-- Deklarasi Pustaka ---
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace coballagi
{
    //-- Bagian Class ---
    class Program
    {
        //-- Bagian Fungsi --
        static void Main(string[] args)
        {
            Console.WriteLine("Program pertama saya");
            Console.WriteLine("Saya Belajar Pemograman C#");
            Console.ReadKey();
        }
    }
}
```

Biasanya blok kode digunakan untuk membungkus lebih dari satu statement. Jika *statement* hanya ada satu baris, maka bisa kita tidak bungkus dengan tanda kurung kurawa.

3.4 Sintak dasar Pemograman C#

Sintak dasar dalam pemograman C# adalah sebagai berikut:

1. **Nama File :** secara default dalam program yang kita buat adalah Program.cs. walaupun kita bisa menggantinya dengan apa saja seperti programpertama.cs. Akhiran atau extention yang di pakai bahwa suatu file berisi sorce code C# adalah .cs
2. **Comment**
   Comment dapat digunakan sebagai dokumentasi, untuk memberikan informasi atau keterangan mengenai program yang kita buat dan memahami dalam membantu program yang kita buat.

   ✓ ***Comment Single Line***
   Comment Single line di awali dengan tanda // hanya berlaku untuk satu baris code dan berakhir pada baris tersebut.

   Contoh:
   System.Console.WriteLine ("Halo Dunia!"); //akan menuliskan pesan di layar

   ✓ ***Comment Multi Line***

Comment Multi Line di awali dengan tanda /* dan di akhiri dengan */

Contoh:

```
/* Comment jenis ini
dapat menutupi lebih dari satu baris */
```

3. **Method Main**

Method Main() bisa di katakan sebagai pintu masuk program anda. Kalau anda mencoba mengganti nama Main() dengan nama lain, compiler akan mengeluh bahwa tidak ada entry point atau pintu masuk. Perlu juga diketahui bahwa isi atau tubuh method harus diawali dengan "{" dan diakhiri dengan "}". Method Main() dapat disebut juga sebagai fungsi utama.

Contoh:

```
static void Main(string[] args)
{
 //Tubuh atau isi method
}
```

4. **Class**

Method di dalam C# tidak bisa berdiri sendiri, namun harus menjadi bagian dari suatu class. Nama class yang menyelimuti **Main()** tidak menjadi masalah. Seperti method, tubuh suatu class dimulai dengan **{** dan diakhiri dengan **}**.

Contoh:

```
class Hallo
 {
 // tubuh atau isi class
 }
```

5. **System.Console.WriteLine() dan System.Console.Write()**

Method WriteLine() tersebut berada di dalam class yang bernama Console (sebagaimana method Main() berada dalam suatu class). Class Console sendiri dikelompokkan ke dalam namespace yang bernama System. Namespace System menampung semua class library.NET (Console adalah salah satu dari class library.NET). Jadi kita memanggil method WriteLine() dengan menuliskan mulai dari namespacenya sampai ke method itu sendiri.

Contoh:

System.Console.WriteLine("Halo Lingkup Informatika!");

//Akan menyisipkan baris baru

System.Console.Write("Sedang belajar C#!"); // tanpa menyisipkan baris baru

System.Console.Write("Gampang-gampang susah."); // tanpa menyisipkan baris baru

Output dari penggalan program di atas sebagai berikut:

Halo Lingkup Informatika!

Sedang belajar C#! Gampang-gampang susah.

6. Readline()

Method static **ReadLine()** dari class **Console** memungkinkan kita menuliskan input sampai enter ditekan. Input yang kita berikan akan dikembalikan ke program dalam bentuk string. **String** yang dikembalikan tersebut dapat kita jadikan argument bagi **Parse().**

Contoh:

```
class Input
        {
                static void Main()
                {
                        int var1, var2;
                        Console.WriteLine("Program
                Penjumlahan");
                        Console.WriteLine();
                        Console.Write("Masukkan angka
                pertama: ");
                        var1 = int.Parse(Console.ReadLine());
                        Console.Write("Masukkan angka
                kedua: ");
                        var2 = int.Parse(Console.ReadLine());
                        Console.WriteLine();
                        Console.WriteLine("Jumlahnya adalah
                {0}.", var1 + var2);
                }
        }
```

7. **Parse()**
   Type data yang telah dipakai seperti **int** dan **double** juga merupakan sejenis *class*. Type type dasar yang didefinisikan Framework .NET memiliki *method static* yang bernama **Parse()**. Method tersebut menerima sebuah argument **string** dan mengembalikan hasil olahannya. Hasil olahannya berupa tipe yang bersangkutan, dengan nilai yang terkandung di dalam string argument.

   1. **Namespace**
      Namespace adalah kata kunci untuk mendefinisikan ruang lingkup atau batasan program dan menghindari konflik nama. Misalnya **source code** A.cs dan **source code** B.cs sama-sama membuat class yang bernama **namakelas**. Kedua class yang namanya sama tersebut dapat digunakan di suatu program asalkan terletak di namespace yang berbeda. Kegunaan namespace untuk mengelompokan elemen-elemennya (misalnya class) bisa dimisalkan seperti kegunaan folder untuk mengelompokkan file-file.

   2. **Placeholder**
      Placeholder adalah data yang di ikat atau di gabungkan ke dalam sebuah pernyataan secara terpisah dari data-data yang lain.

      Contoh:

```
int permen = 10;
int snack = 5;
//placeholder {0} dan {1}
Console.WriteLine("Saya memliki {0} permen dan {1} snack", permen, snack);
Console.ReadLine();
```

3.5 Statement Kondisi

Statement IF digunakan untuk mengeksekusi beberapa kode program apabila mempunyai kondisi **True** atau **False.** Statement **IF** menentukan kondisi ekspresi yang akan dievaluasi. Apabila kondisi benar, pernyataan dalam kurung kurawa "{}" akan dieksekusi. Apabila kondisi salah, maka akan di abaikan Kemudian komputer akan

melanjutkan program yang berada setelah tubuh dari statement **IF** tersebut.

a. Bentuk umum statement **IF**:

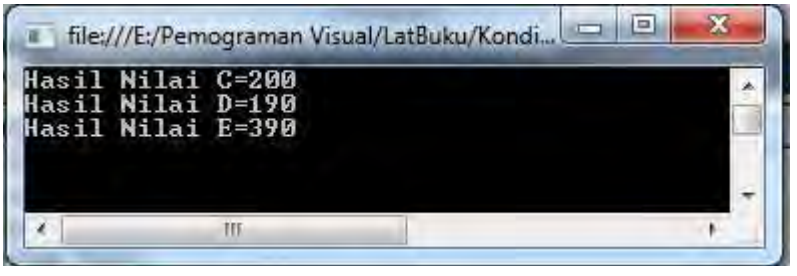
```
if ( kondisi )
{
 Perintah benar;
}
else
{
 Perintah salah;
}
```

Contoh program:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace KondisiIF_1
{
 class Program
 {
 static void Main(string[] args)
 {
 int a = 10, b = 20, c, d, e;
 c = a * b;
 if (c > 100)
 {
d = c - a;
 }
 else
 {
d = c - b;
 }
 e = d + c;
 Console.WriteLine("Hasil Nilai C=" + c);
 Console.WriteLine("Hasil Nilai D=" + d);
 Console.WriteLine("Hasil Nilai E=" + e);
 Console.ReadKey();
 }
 }
```

}

Hasil:



b. Bentuk **IF bersarang**

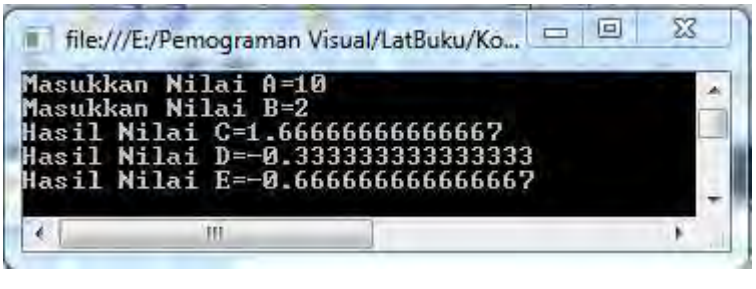
```
if (kondisi_1)
{
Perintah_1;
}
else if (kondisi_2)
{
Perintah_2:
}
else
{
 Perintah salah;
}
```

Contoh program:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace KondisiIF_2
{
 class Program
 {
 static void Main(string[] args)
 {
 double a, b, c, d, e;
 Console.Write("Masukkan Nilai A=");
 a = double.Parse(Console.ReadLine());
```

```
 Console.Write("Masukkan Nilai B=");
 b = double.Parse(Console.ReadLine());
 c = (a * b) / (a + b);
 if (c > 100)
 {
d = c - a;
 }
 else if (c > 50)
 {
d = c + a;
 }
 else
 {
d = c - b;
 }
 e = (d / a) * (a * b);
 Console.WriteLine("Hasil Nilai C="+c);
 Console.WriteLine("Hasil Nilai D=" + d);
 Console.WriteLine("Hasil Nilai E=" + e);
 Console.ReadKey();
 }
 }
}
```

Hasil:



3.6 Perulangan

a. While Statement

Pernyataan while, do-while, for dan foreach merupakan pernyataan yang termasuk dalam keluarga perulangan. Pernyataan while memiliki mekanisme yaitu suatu statement yang dijalankan

secara berulang selama kondisi dalam while masih terpenuhi.Bentuk umum Statement **While**

While (kondisi)

{

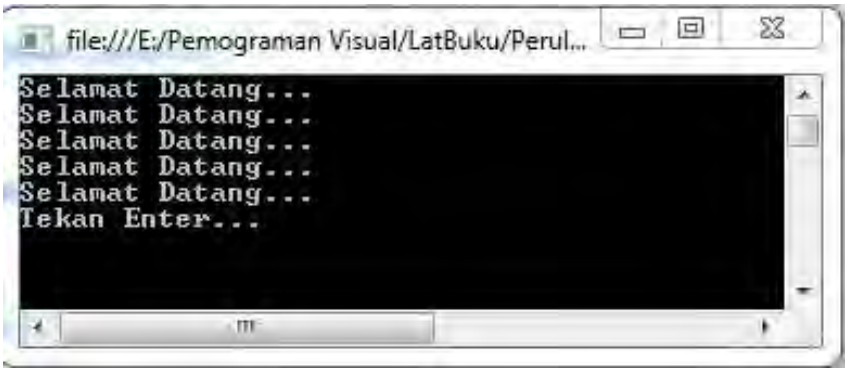
Perintah;

}

Contoh program:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace PerulanganWhile_1
{
 class Program
 {
 static void Main(string[] args)
 {
 int a = 10;
 while (a > 5)
 {
Console.WriteLine("Selamat Datang...");
a = a - 1;
 }
 Console.Write("Tekan Enter...");
 Console.ReadKey();
 }
 }
}
```

Hasil:



Hasil diatas adalah untuk menampilkan Selamat Datang sebanyak 5 kali, pertama memberikan nilai awal a=10, lakukan perulangan jika a>5. Cetak Selamat Datang, kemudian lakukan perngurangan a=a-1, a=10-1→9, apakah 9 lebih besar 5 jika ya cetak lagi, a=9-1→8, apakah 8 lebih besar dari 5 jika ya tampilkan dilayar.

b. Do While Statement

Pernyataan ***Do-While*** memiliki mekanisme blok kode yang terdapat dalam lingkup **do** dieksekusi dahulu, kemudian setelah itu baru dieksekusi oleh **while**, bila kondisi dalam **while** terpenuhi tapi jika tidak maka blok kode dalam lingkup **do** akan berlanjut ke perintah berikutnya.

Bentuk umum Statement **Do** - **While**

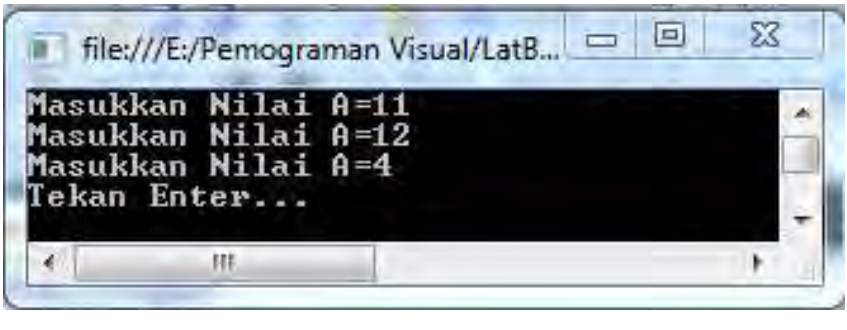
```
do
{
Perintah;
}
while (kondisi)
```

Contoh program:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
namespace DoWhile_1
{
 class Program
 {
 static void Main(string[] args)
 {
 int a;
 do
 {
Console.Write("Masukkan Nilai A=");
a=int.Parse(Console.ReadLine());

 }
 while(a>10);
 Console.Write("Tekan Enter...");
 Console.ReadKey();
 }
 }
}
```

Hasil:



Hasil program diatas yaitu mengeluarkan pertanyaan yang harus diisi pengguna (program masukan), jika pengguna memasukkan angka a<10 maka blok kode dalam do akan berlanjut ke blok kode yang terdapat dibawahnya yaitu Tekan Enter…, namun jika pengguna memasukkan angka a>10, maka while akan di eksekusi alias pertanyaan yang terdapat dalam blok kode do diulang kembali.

c. For Statement

Pernyataan **For** memiliki mekanisme yang hampir mirip dengan pernyataan ***While*** dimana suatu blok kode akan di iterasi sampai mencapai kondisi tertentu, namun yang membedakan antara **for** dengan **while** adalah sintaks dalam **for** sudah di ***built-in*** dan juga menggunakan operator *increment* atau *decrement*.

Bentuk umum Statement **For**

for (nilai awal;kondisi;perulangan/penambahan)

{

Perintah;

}

Contoh program:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace For_1
{
 class Program
 {
 static void Main(string[] args)
 {
 for (int a = 1; a <= 5; a++)
 {
Console.WriteLine("ini adalah nilai angka=" + a);
 }
 Console.Write("Tekan Enter...");
 Console.ReadKey();
 }
 }
}
```

Hasil:

Hasil program yang ditampilkan adalah muncul kalimat "ini adalah nilai angka=1" sampai "ini adalah nilai angka=5" secara berurutan.

d. Foreach Statement

Pernyataan ***Foreach*** memiliki mekanisme hampir sama seperti **for**, *statement* ini digunaakan untuk melakukan perulangan pada setiap isi atau elemen pada *array*. ***foreach*** memungkinkan untuk melakukan penelusuran atas item dalam sekumpulan / koleksi data, *statement* ini banyak digunakan dalam melakukan penelusuran pada collection.

Bentuk umum Statement **Foreach**

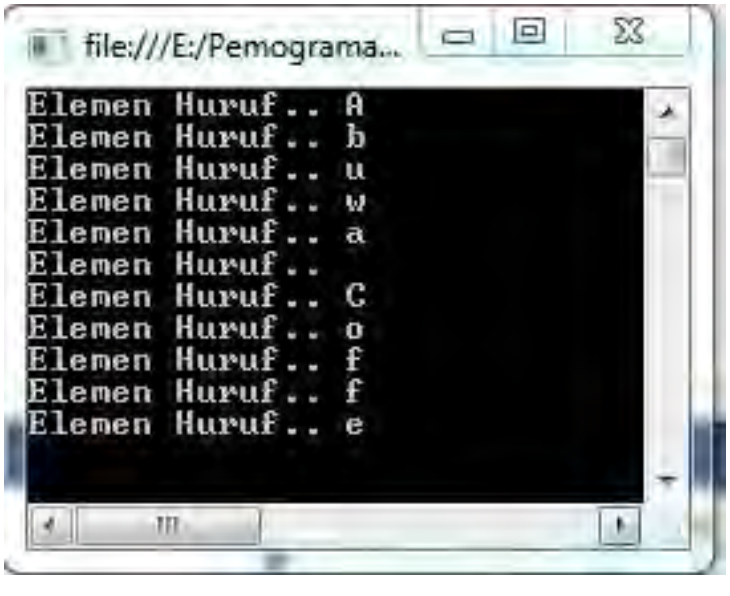
```
Foreach (tipe variabel dalam koleksi data)
{
 Perintah;
}
```

Contoh program_1:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ForEach_1
```

```
{
 class Program
 {
 static void Main(string[] args)
 {
 string huruf = "Abuwa Coffe";
 foreach (char huruf1 in huruf) //-- convert huruf ke tipe
data char dengan
  //-- variabel huruf1
 {
 Console.WriteLine("Elemen Huruf.. " + huruf1);
 }
 Console.ReadKey();
 }
 }
}
```

Hasil:



Tampilan program adalah keluarnya kalimat "Elemen Huruf .. A" sampai "Elemen Huruf .. e" secara berurutan. Pertanyaannya kenapa program tidak error dan tetap mengeluarkan tampilan perulangan. String yang digunakan tidak berada dalam *arrays*, sedangkan jika menggunakan char dalam program hasilnya akan error. Jika dilihat kode program akan mengconvert string menjadi char agar setiap elemen dalam string tersebut dapat diulangi satu-persatu.

Contoh program_1:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ForEach_2
{
 class Program
 {
 static void Main(string[] args)
 {
 int[] nilai = { 2, 3, 2, 6, 4, 2, 8, 7 };
 foreach (int angka in nilai)
 {
Console.WriteLine("Ini adalah Angka = " + angka);
 }
 Console.ReadKey();
 }
 }
}
```

Hasil:

3.7 Array

Array digunakan untuk menyimpan lebih dari satu nilai dengan tipe data yang sama dalam satu variabel. Sebuah array memiliki sifat sebagai berikut:

1. Setiap elemen dari array harus memiliki tipe data yang sama.
2. Sebuah array dapat berupa *Single-Dimensi, Multidimensional* atau *Jagged.*
3. Jumlah dimensi dan panjang masing-masing dimensi ditetapkan ketika ketika array dibuat. Nilai-nilai tersebut tidak dapat diubah jika sudah dideklarasikan.
4. Nilai *default* dari elemen array numerik adalah 0, dan elemen string diatur ke *null.*
5. *Jagged array* adalah array di dalam array, artinya elemen dari array tersebut berupa array juga.
6. Array adalah sebuah *zero indexed*, artinya elemen dari array itu diindex dari 0 sampai jumlah elemen-1. Misalkan jumlah elemen 5, maka indeknya adalah 0, 1, 2, 3, 4.

Array dalam pemograman C# bekerja layaknya array dalam bahasa pemrograman secara umum. Ada beberapa perbedaan yang anda harus perhatikan, ketika mendeklarasikan sebuah array, tanda kurung siku **[ ]** ditaruh di belakang tipe data bukan di belakang nama variabel. Jika ditempatkan di belakang variabel, maka akan muncul error pada pemograman C#.

Bentuk Umum:

```
int[] nilai;  //Bukan int nilai[];
```

Cara menentukan panjang array atau jumlah elemen dalam suatu array.

```
int[] nilai; //Mendeklarasikan array;

nilai = new int[5] //Menentukan panjang array sebanyak 5
```

A. Mendeklarasikan Array

Pemograman C# mendukung *Single-Dimensional Array*, *Multidimensional Array*, dan *Jagged Array*. Berikut cara mendeklarasikan berbagai jenis array:

- **Single-dimensional**
  Contoh:

array:int[] nilai;

- **Multidimensional array**
  Contoh:

  string[,] nama;

- **Jagged array (array di dalam array)**
  Contoh:

  int[][] angka;

Mendeklarasikan array belum membuat sebuah array, karena menentukan panjang arraynya. Untuk menentukan panjang array kita bisa menambahkan:

a) Single-dimensional Array:

```
int[] angka = new int[5];
```

b) Multidimensional array:

```
string[,] nama = new string[5,4];
```

c) Jagged array:

```
byte[][] angka= new byte[5][];
for (int x = 0; x < angka.Length; x++)
{
 angka[x] = new byte[4];
}
```

B. Inisialisasi Array

Pemograman C# menyediakan berbagai cara dalam mengisi nilai dari setiap elemen array, dimana pengisian elemen diapit dengan tanda kurung kurawa (**{}**). Berikut cara-cara pengisian array:

a) Single-dimentional array:

```
int[] nilai = new int[5] { 1, 2, 3, 4, 5 };
string[] nama = new string[3] { "John", "Jesica", "David" };
```

atau

```
int[] nilai = new int[] { 1, 2, 3, 4, 5 };
string[] nama = new string[] { "John", "Jesica", "David" };
```

atau

```
int[] nilai = { 1, 2, 3, 4, 5 };
string[] nama = { "John", "Jesica", "David" };
```

b) Multidimentional array:

```
int[,] nilai = new int[3, 2] { { 1, 2 }, { 3, 4 }, { 5, 6 } };
string[,] nama = new string[2,2] { {"John", "Jessica"},
{"David", "Marry"} };
```

atau

```
int[,] nilai = new int[,] { { 1, 2 }, { 3, 4 }, { 5, 6 } };
string[,] nama = new string[,] { {"John", "Jessica"},
{"David", "Marry"} };
```

atau

```
int[,] nilai = { { 1, 2 }, { 3, 4 }, { 5, 6 } };
string[,] nama = { {"John", "Jessica"}, {"David", "Marry"} };
```

c) Jagged Array

```
int[][] angka = new int[2][] { new int[] { 1, 2, 3 }, new
int[] { 4, 5, 6 } };
```

atau

```
int[][] angka = new int[][] { new int[] { 1, 2, 3 }, new int[]
{ 4, 5, 6 } };
```

atau

```
int[][] angka = { new int[] { 1, 2, 3 }, new int[] { 4, 5, 6 } };
```

C. Mengakses Anggota Array

Berikut adalah cara untuk mengambil nilai yang ke 4 dari sebuah *single-dimentional array*:

```
static void Main(string[] args)
{
```

```
int[] angka = { 2, 4, 5, 2, 1, 7, 8, 9 };
Console.WriteLine(angka[5]);
Console.ReadKey();
}
```

Hasil:

7 //-- karean array dimulai dari 0,1,2,3,4,5 jadi angka kelima adalah 7

Berikut ini cara untuk mengambil elemen pada posisi 4,3 dari sebuah **multi dimentional array:**

```
static void Main(string[] args)
 {
 int[,] angka={{1,2},{3,4},{5,6}};
 Console.WriteLine(angka[1,0]);
 Console.ReadKey();
 }
```

Hasil:

3

Int[,] angka yang mempunyai array ke → 0 {1,2}, array ke → 1 {3,4} dan array ke 2 → {5,6}. Dan array {3,4} dimulai dari array 0 dan array 1, jadi hasil dari angka[1,0] adalah terletak di array ke – 1 dan terletak di kolom array 0 yaitu array {3,4} yang array 0-nya adalah angka 3.

Berikut adalah cara untuk mengambil elemen pada posisi 0, 0 dan 1, 2 dari sebuah jagged array:

```
int[][] angka = { new int[] { 1, 2 }, new int[] { 3 , 4, 5 } };
System.Console.WriteLine(angka[0][0]);
System.Console.WriteLine(angka[1][2]);
```

Hasil:

1
5

Angka[0][0] terletak pada array ke – angka[0] yaitu {1,2}, dan nilai array [0] yaitu nilainya . Dan angka[1][2] terletak pada array ke-1 yaitu {3,4,5} yang dimulai pada array 0,1,2 dan [2] terletak pada array 5.

Berikut adalah cara untuk mengambil elemen pada posisi 2, 1 dari sebuah *multi dimentional array*:

```
int[,] angka = { { 1, 2 }, { 3, 4 }, { 5, 6 } };
System.Console.WriteLine(angka[2,1]);
```

Hasil:

```
6
```

Angka[2,1] terletak di array ke -2 yaitu array {5,6}, dimana {1,2} sebagai array ke-0, {3,4} sebagai array ke-1, dan {5,6} sebagai array ke-2. Dan nilai array 1 adalah elemen array {5,6} yaitu elemen 0 adalah 5 dan elemen 1 adalah 6.

Berikut adalah cara untuk mengambil elemen pada posisi 0, 0 dan 1, 2 dari sebuah **jagged array**:

```
int[][] angka = { new int[] { 1, 2 }, new int[] { 3 , 4, 5 } };
System.Console.WriteLine(angka[0][0]);
System.Console.WriteLine(angka[1][2]);
```

Hasil:

```
1
5
```

D. Array adalah Object

Pemograman C#, array sebenarnya merupakan **object. System.Array** adalah base dari semua jenis array. Dalam menggunakan properti, dan *class* lainnya yang memiliki ***System.Array***. Misalnya kita gunakan **property length** untuk menentukan **size** atau panjang dari suatu array.

Contoh program:

```
int[] angka = { 1, 2, 3, 4, 5, 6, 7, 8, 1 };
System.Console.WriteLine(angka.Length);
```

Hasil:

```
9
```

∞

BAB 4.
Object Oriented Programing

4.1 Pengertian OOP

Object Oriented Programming atau Pemrograman Berbasis Objek, adalah salah satu cara membuat program berguna untuk memecah alur program menjadi modul-modul sederhana yang disebut dengan objek. Setiap objek akan memiliki fungsi dan tugas tersendiri. OOP berbeda dengan prosedural programming yang memecah program menjadi fungsi-fungsi atau prosedural.

C# (dibaca C-Sharp) merupakan suatu bahasa pemrograman yang menggunakan konsep Pemrograman berbasis Objek (OOP). OOP memposisikan semua bagian yang terlibat dalam program sebagai objek. Program dapat dibentuk dan dijalankan melalui interaksi antara objek yang satu dengan objek yang lainnya.

Dalam paradigma OOP, solusi dibangun dari 1 objek atau lebih yang saling bekerja sama untuk menyelesaikan masalah. Objek merupakan benda (nyata dan tidak nyata). Setiap objek memiliki nilai yang melekat pada setiap objek tersebut. Nilai yang melekat pada objek tersebut disebut dengan properti. Setiap objek memiliki kemampuan untuk melakukan sesuatu. Setiap objek bisa memiliki kemampuan untuk melakukan suatu aksi. Aksi yang dapat dilakukan oleh sebuah objek disebut dengan *method*.

Contohnya sebuah objek bernama sepeda. Sepeda memiliki nilai yang melekat dengan sepeda tersebut misalnya merek, harga, jenis sepeda, warna, ukuran roda, jumlah rem. Sedangkan aksi yang bisa dilakukan oleh sepeda contohnya berhenti, jalan, ganti gigi, mengerem. Aksi lebih cenderung menggunakan kata kerja sedangkan properti lebih ke kata benda dan kata sifat.

4.2 *Class*, Objek dan *Atribute*

Objek digunakan dalam C# merupakan bentuk nyata sebuah kelas (kelompok). Hal ini disebut dengan *Classification* (klasifikasi), yaitu mengatur informasi dan tingkah laku yang sama ke dalam entitas yang berarti. Contohnya, semua mobil memiliki tingkah laku dan properti yang sama sehingga mobil tersebut “diklasifikasikan” ke dalam kelas (kelompok) Mobil.

Objek dalam C# merupakan bentuk nyata (**instance**) dari kelas. Kelas merupakan cetakan dari objek. Dari kelas tersebut dapat dibuat

objek. Dalam kelas sudah terdapat informasi berupa properti dan tingkah laku objek yang nanti dibuat berdasarkan kelas tersebut. Properti berupa variabel kelas yang digunakan untuk menyimpan data dan tingkah laku berupa fungsi yang dapat dijalankan.

Banyak objek dapat dibuat dari sebuah kelas. Dalam C# untuk kelas dibuat dengan menggunakan keyword class. Format penulisan kelas dalam C# adalah sebagai berikut;

```
class <Nama Kelas>
{
   <Tipe> <Nama Properti>
   <Tipe> <Nama Fungsi> ( [Daftar Argumen])
     {
         Statement_1;
         Statement_2;
     }
}
```

Berikut ini adalah contoh sebuah kelas yaitu kelas Segitiga

```
Class Segitiga
{
    Decimal alas;
    Decimal tinggi;

    Public void hitungLuas()
    {
      Console.WriteLine("Disini seharusnya terjadi
      hitung luas");
    }

    Public void tampilkanInfo()
    {
      Console.WriteLine("Alas segitiga "+ alas);
      Console.WriteLine("Tinggi segitiga "+ tinggi);
    }
}
```

Sebuah objek dibuat dari kelas yang telah ada. Gunakan keyword **new** untuk membuat objek dari kelas.

Bentuk umum pembuatan **class**:

[nama_kelas] [nama_objek] = ***new*** *[nama_kelas]****()***

Contoh untuk membuat objek dari kelas Segitiga adalah sebagai berikut:

Segitiga s1 = new Segitiga();

Atau dapat juga ditulis sebagai berikut;

Segitiga s1;
s1 = new Segitiga();

Statement diatas bisa dijelaskan sebagai berikut. "Deklarasikan sebuah variabel bernama s1 bertipe Segitiga, kemudian buat sebuah objek dari kelas Segitiga dan simpan objek tersebut ke dalam variabel s1. Untuk selanjutnya objek yang dihasilkan disebut dengan objek s1.

Kelas bisa diperlakukan layaknya sebuah tipe data, sehingga sebuah variabel bisa memiliki tipe data berupa nama kelas. Format deklarasi variabel <Tipe Data> <nama Variabel>, juga bisa digunakan digunakan dengan menggunakan kelas menjadi <Class> <nama variabel/nama objek>

Dari sebuah kelas dapat dibuat banyak objek. Contoh membuat objek baru dari kelas yang sama dapat dilihat pada contoh dibawah ini.

Segitiga s1 = new Segitiga();
Segitiga s2 = new Segitiga();
Segitiga segitiga = new Segitiga();

4.3 Property

Properti adalah variabel dalam kelas/objek. Properti digunakan untuk menyimpan data atau nilai yang ada dalam objek tersebut. Properti dideklarasikan dalam kelas dan dapat digunakan di semua method dalam kelas tersebut. Deklarasi properti dalam kelas mengikuti format berikut ini.

[Level akses] <Tipe data> <nama properti>;

Level akses adalah hak akses objek lain terhadap properti tersebut. Level akses juga digunakan terhadap method. Dalam C# ada 4 hak akses yang dapat digunakan yaitu :

1. Public–Level akses public memungkinkan sebuah properti atau method sebuah kelas dapat diakses oleh objek lain baik dari assembly yang sama maupun assembly yang berbeda.

2. Private–Sebuah properti atau method yang memiliki akses level private hanya bisa diakses oleh kode (objek atau kelas) yang berasal dari kelas yang sama (atau struct yang sama).
3. Protected–Properti atau method yang memiliki akses level protected hanya bisa diakses oleh kode (objek atau kelas) yang sama dan kode turunannya. Kita akan membahas turunan kelas pada pembahasan selanjutnya.
4. Internal–Hak akses internal membatasi akses terhadap properti atau method hanya bisa diakses oleh kode yang berasal dari assembly yang sama, dan tidak dari assembly yang berbeda.

Keempat akses level itulah yang digunakan untuk menentukan akses level sebuah properti dan method dalam sebuah kelas. Deklarasi akses level bersifat opsional. Jika deklarasi akses level tidak dinyatakan maka akses level properti atau method tersebut menjadi private.

Contoh deklarasi sebuah properti pada contoh kelas diatas adalah:

String nama;
Public Int jumlahKaki;

Pada contoh diatas properti **jumlahKaki** merupakan properti bertipe **integer** dan memiliki akses level **public**. Sedangkan properti nama, karena tidak dideklarasikan akses levelnya, memiliki akses level **private**. Disarankan deklarasi properti memiliki nilai awal. Nilai awal sebuah properti dapat dideklarasikan saat deklarasi properti (sama seperti deklarasi variabel). Contoh deklarasi properti yang memiliki nilai awal dapat dilihat pada contoh berikut ini.

String nama = "Yu";
Public int jumlahSisi = 3;

4.4 Method

Method dalam kelas menunjukkan aksi yang dapat dilakukan oleh kelas (objek) tersebut. Jika properti digunakan untuk menyimpan nilai, maka method digunakan untuk menyimpan isntruksi (statement) yang akan dieksekusi saat method tersebut dijalankan. Sebuah kelas dapat memiliki 1 atau lebih method, atau tidak memiliki

method sama sekali. Deklarasi method dalam kelas mengikuti format berikut ini.

```
[akses level] <tipe data> <nama method> ([daftar agumen])
    {
        Statement01;
        Statement02;
        ...
    }
```

Akses level yang berlaku terhadap sebuah method sama seperti akses level yang berlaku terhadap sebuah properti. Tipe data yang bisa digunakan untuk sebuah method merupakan tipe data yang sama yang dapat digunakan terhadap properti. Hanya saja pada method ada tipe data tambahan yang dapat digunakan yaitu tipe data **void**. Tipe data void, ini merupakan tipe data yang tidak menyimpan nilai apapun (hampa).

Tipe data pada method digunakan untuk mengindikasikan nilai yang akan dikembalikan atau diberikan oleh method tersebut. Jika tipe datanya void, maka method tersebut tidak mengembalikan nilai apapun. Jika tipe datanya bukan void maka method tersebut akan mengembalikan sebuah nilai. Oleh karena method tersebut akan mengembalikan sebuah nilai maka di dalam implementasi method tersebut harus ada statement return yang untuk mengembalikan nilai sesuai tipe data method tersebut.

Daftar argumen merupakan input yang akan digunakan method tersebut. Input tersebut didelarasikan dalam bentuk variabel yang bisa digunakan dalam method tersebut. Daftar argumen ini bersifat opsional (boleh ada, boleh tidak ada).

Contoh Method dalam sebuah kelas, dapat dilihat pada potongan kode sumber berikut ini.

```
Class Segitiga
{
  Public double hitungLuas(double alas, double tinggi)
     {
         double luas = alas * tinggi;
         return luas;
     }
}
```

Pada contoh diatas dideklarasikan sebuah method bernama hitungLuas. *Method* ini memiliki akses *level public* yang berarti bisa diakses oleh objek atau kelas lain. *Method* ini juga memiliki tipe data double yang berarti method ini akan menghasilkan dan mengembalikan sebuah nilai bertipe *double*. Oleh karena tipe method ini bukan *void*, maka dalam method tersebut harus ada statement return yang digunakan untuk mengembalikan nilai bertipe *double*. Method hitungLuas ini juga memiliki 2 buah argumen yaitu alas dan tinggi. Kedua argumen ini bertipe double dan digunakan sebagai input data dalam method tersebut. Kedua argumen ini digunakan untuk menghitung luas segitiga.

4.5 *Polymorphisme* (*Overloading*)

Polimorfisme bisa diartikan satu bentuk banyak aksi, sekilas mirip dengan *inheritance* tetapi dalam polimorfisme suatu objek dapat melakukan tindakan yang secara prinsip sama tapi secara proses dan outputnya berbeda. Polimorfisme mengizinkan kelas induk untuk mendefinisikan sebuah method general (bersifat umum) untuk semua kelas turunannya, dan selanjutnya kelas-kelas turunan dapat memperbarui implementasi dari *method* tersebut secara lebih spesifik sesuai dengan karakteristiknya masing-masing.

Polymorphisme secara harfiah berarti banyak bentuk. *Polymorphisme* memungkinkan sebuah method dijalankan dengan cara yang berbeda-beda. Secara umum polymorphisme terbagi menjadi 2 jenis yaitu:

1. ***Overloading*** : Penggunaan satu nama untuk beberapa method. *Overloading* mensyaratkan setiap method dengan nama yang sama memiliki daftar argumen yang berbeda. Method yang akan dieksekusi tergantung cara memanggil method tersebut.
2. ***Overriding*** : Penggunaan satu nama method yang sama dengan nama method yang ada pada kelas induknya. *Overriding* ini akan dibahas lebih lanjut pada pembahasan selanjutnya.

Contoh **Overloading** dapat dilihat pada kode kelas dibawah ini.

```
class Segitiga
{
    //Menghitung luas dengan menggunakan alas dan tinggi
    segitiga
    public double hitungLuas(double alas, double tinggi)
        {
            double luas = alas * tinggi;
```

```
            return luas;
        }
    //Menghitung luas dengan menggunakan ketiga sisi segitiga
    public double hitungLuas(double a, double b, double c)
        {
            double S = (a + b + c)/2;
            double luas = Math.Sqrt(S * (S -a) * (S -b) * (S -c));
            return luas;
        }
}
```

Kelas segitiga diatas memiliki dua buah method dengan nama hitungLuas. Fungsi kedua method ini sama-sama untuk menghitung luas segitiga. Ada beberapa cara untuk menghitung segitiga yaitu dengan menggunakan persamaan L = a * t / 2 atau dengan cara menghitung luas dengan menggunakan panjang ketiga sisi-sisi segitiga tersebut dengan menggunakan rumus S. Luas dihitung dengan menggunakan persamaan berikut :

$$L = \sqrt{S(S-a)(S-b)(S-c)}$$

dan

$$S = \frac{(a+b+c)}{2}$$

Dengan demikian terdapat 2 buah method yang namanya sama, tapi memiliki implementasi yang berbeda. Bagaimana menentukan method yang akan dijalankan nantinya? Tergantung dari parameter yang digunakan. Perhatikan kode berikut ini;

```
Segitiga = new Segitiga();
luas = segitiga.hitungLuas(4.0, 2.0);
luas = segitiga.hitungLuas(3.0, 2., 3.0);
luas = segitiga.hitungLuas(3, 8, 4, 0);
```

Pada contoh diatas, terdapat 3 kali pemanggilan method hitungLuas dengan argumen yang berbeda. Pada pemanggilan pertama (baris 2), method hitungLuas dipanggil dengan menggunakan 2 buah argumen bertipe *double*. Secara otomatis method hitungLuas yang pertama dijalankan yaitu method yang menghitung luas dengan menggunakan alas dan tinggi. Pada pemanggilan method hitungLuas

yang kedua (baris ke 3), method hitungLuas dipanggil dengan menggunakan 3 argumen.

Maka method yang dipanggil adalah method kedua yaitu menghitung luas dengan menggunakan rumus S. Pemanggilan method hitungLuas yang ketiga dilakukan dengan memberikan 4 argumen yang akan menyebabkan error. Hal ini terjadi karena dalam kelas Segitigayang sudah dideklarasikan sebelumnya tidak ada method hitungLuas yang menggunakan 4 buah argumen. Pemanggilan *method overloading* tergantung pada argumen yang diberikan terhadap method tersebut. Tidak hanya memperhatikan jumlah argumen yang diberikan, tapi juga tipe data argumen.

∞

BAB 5.
PENGENALAN WINDOWS FORM

5.1 Komponen Visual Studio 2010

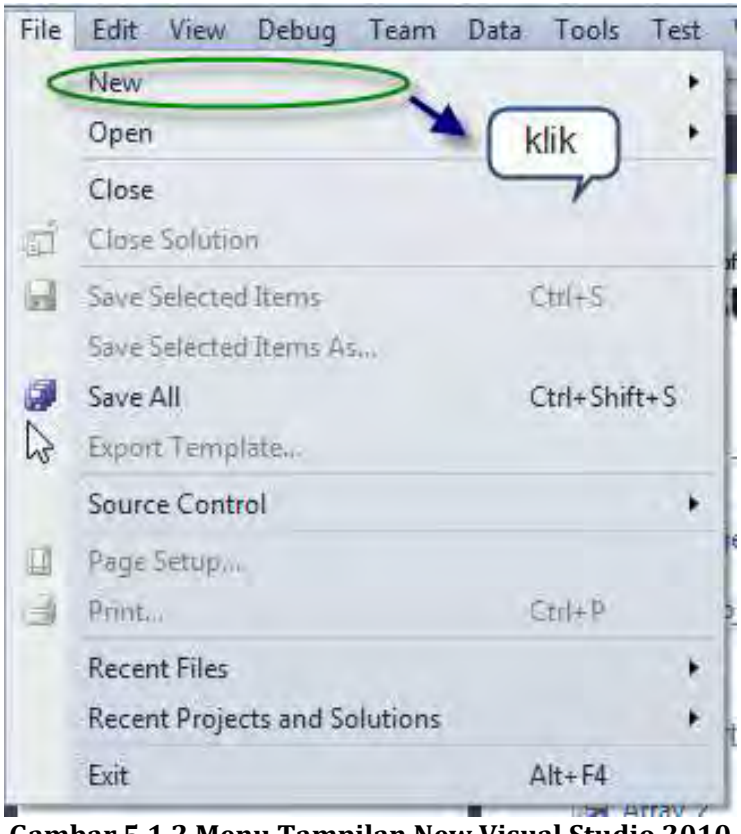
Tahapan dalam menjalankan Visual Studio 2010, beserta tahapan dalam membuat sebuah Project Visual C# 2010.

1. Klik tombol Start → All Program → Microsoft Visual Studio 2010 → Microsoft Visual Studio 2010

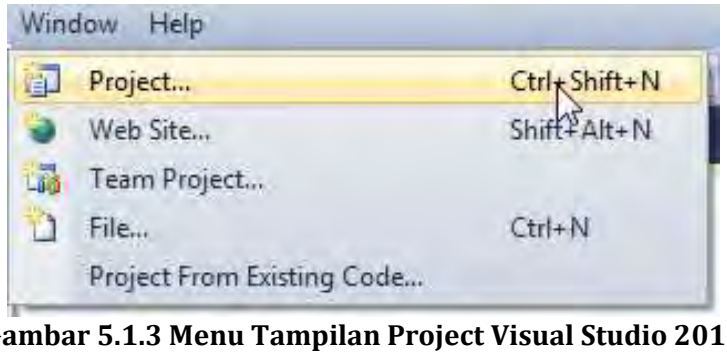
2. Tunggu beberapa saat sampai keluar tamplian seperti pada gambar berikut:

Gambar 5.1.1 Menu Tampilan Visual Studio 2010

3. Klik Menu File → New → Project atau anda juga bisa juga langsung mengklik New

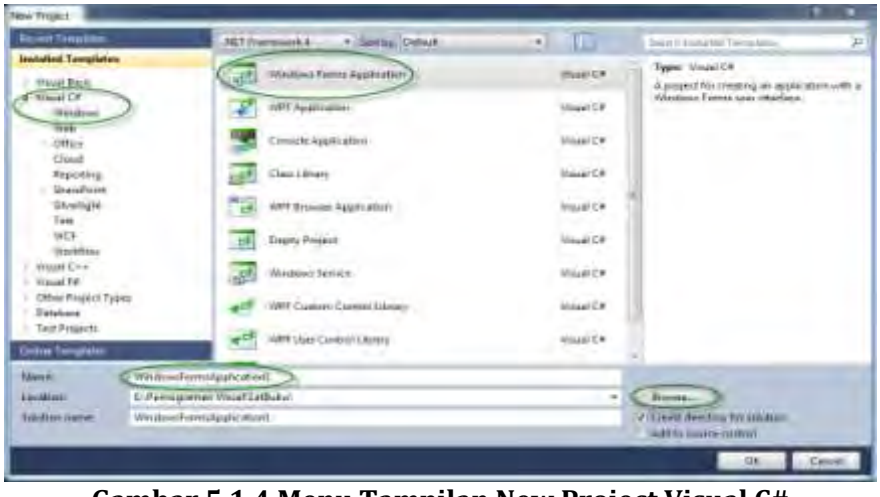


Gambar 5.1.2 Menu Tampilan New Visual Studio 2010



Gambar 5.1.3 Menu Tampilan Project Visual Studio 2010

4. Kemudian akan muncul jendela baru dengan tampilan sebagai berikut:



Gambar 5.1.4 Menu Tampilan New Project Visual C#

5. Pada bagian sebelah kiri pilih **Installed Templates** kemudian klik **Windows Form Application** pada bagian tengah.

6. Selanjutkan beri **name project** pada bagian **Name**, sebagai contoh pada gambar di bawah ini kami beri Lat_1.

5.2 Pengenalan IDE Visual C# 2010

IDE (*Integrated Developement Environment*) merupakan sebuah "layanan satu pintu" yang bisa digunakan oleh programmer untuk melakukan, desain, *coding, debugging*, dan kompilasi program dalam sebuah tool yang terintegrasi. IDE terdiri dari beberapa macam jenisnya yang terdiri dari :

1. **Menu Bar**, dipakai dalam memilih tugas-tugas tertentu seperti membuka *project*, yang terdiri dari menu *file, edit, view* dan sebagainya.

2. **Main Toolbar**, *shortcut* untuk menu yang sering dipakai pada menu bar.

Gambar 5.2.1 [1] Menu Bar, [2] ToolBar

Menu Bar terdiri dari banyak pilihan Menu yaitu:

a) **File**, menu untuk menutup aplikasi, membuka, membuat, menutup, dan menyimpan file Visual C# atau proyek Visual Studio 2010.

b) **Edit**, menyediakan fungsi umum pada item tertentu, *Undo, Redo, Cut, Copy, Paste*, dan *Delete*.

c) **View**, menyediakan akses cepat untuk membuka dan menutup jendela yang ada pada *IDE*, seperti jendela *Solution Explorer*, jendela *Properties*, jendela *Output*, jendela *Toolbox* dan sebagainya. Jika membutuhkan jendela yang belum nampak pada IDE, Anda dapat membuka melalui menu *View*.

d) **Project**, menu *Project* menyediakan bermacam-macam file untuk aplikasi Anda, misalnya *form* dan *class*.

e) **Build**, menu ini sangat penting ketika Anda telah menyelesaikan aplikasi yang Anda buat dan ingin menjalankannya tanpa IDE Visual C# 2010, seperti aplikasi lain dalam bentuk *.EXE.

f) **Debug**, menu ini menyediakan tool untuk menjalankan (*start*) dan menghentikan (*stop*) aplikasi dengan IDE Visual C# 2010. Selain itu Anda juga dapat mencari kesalahan dan melihat apa yang terjadi pada kode yang Anda tulis.

g) **Data**, Anda membutuhkan menu ini jika ingin berhubungan dengan database. Anda dapat mengelola *data source* dan *preview* Anda.

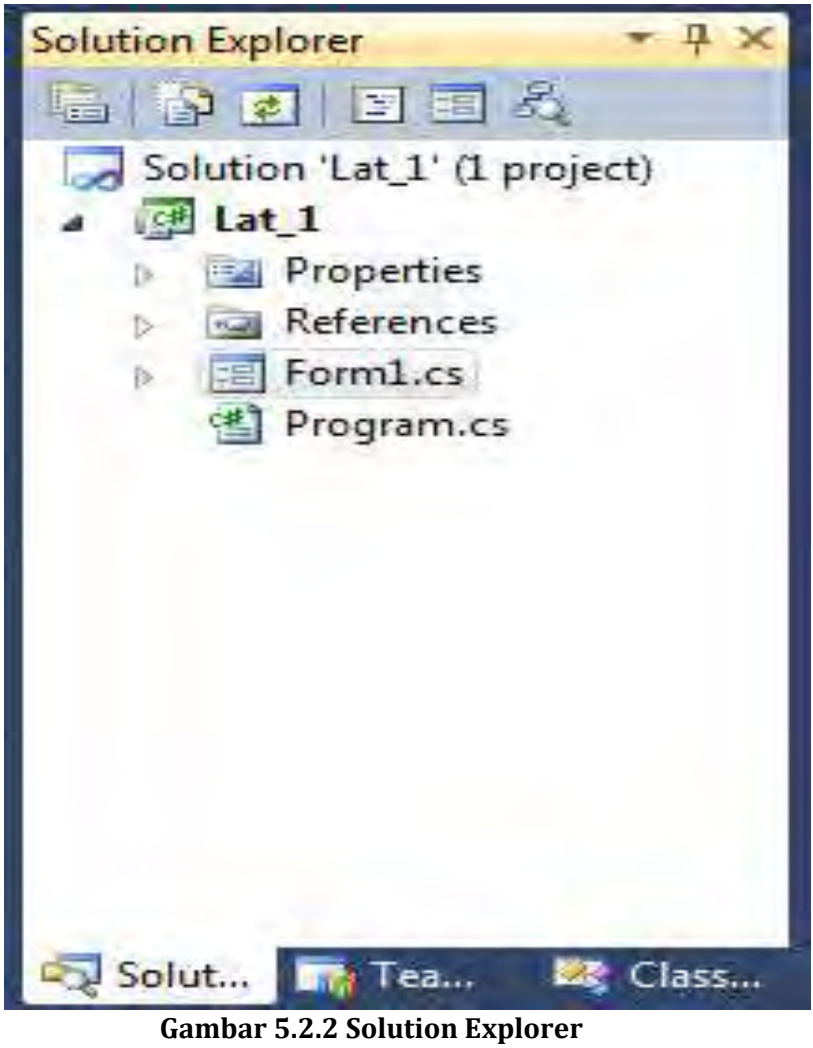
h) **Tools**, menu ini menyediakan perintah untuk melakukan konfigurasi IDE Visual Studio 2010.

i) **Test**, menu ini menyediakan pilihan untuk menciptakan dan melihat unit percobaan untuk aplikasi yang Anda kembangkan sebagai bahan latihan penulisan *source code* pada bermacam-macam situasi.

j) **Window**, *menu Window* menjadi standar pada sebuah aplikasi yang mengijinkan membuka jendela pada satu waktu seperti word dan excel. Anda dapat berganti jendela yang aktif melalui menu ini.

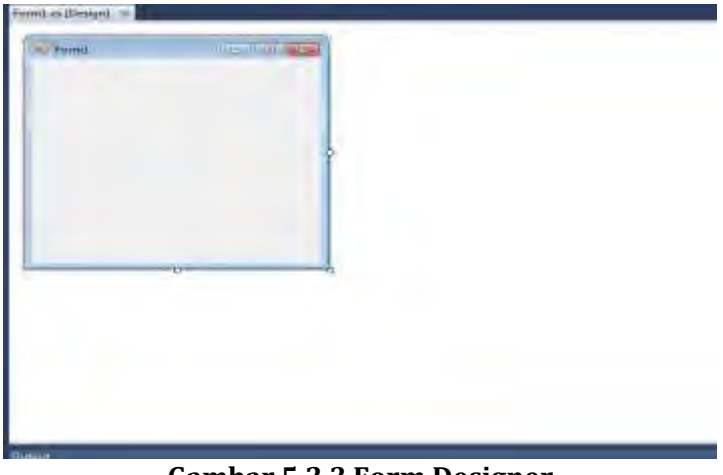
k) **Help**, *menu Help* menyediakan akses untuk membuka dokumentasi Visual Studio 2010, dan Informasi lain seperti Jendela *About*, dan lain-lain.

3. **Solution Explorer**, window yang berisi **struktur tree** dari **project** yang sedang dikerjakan



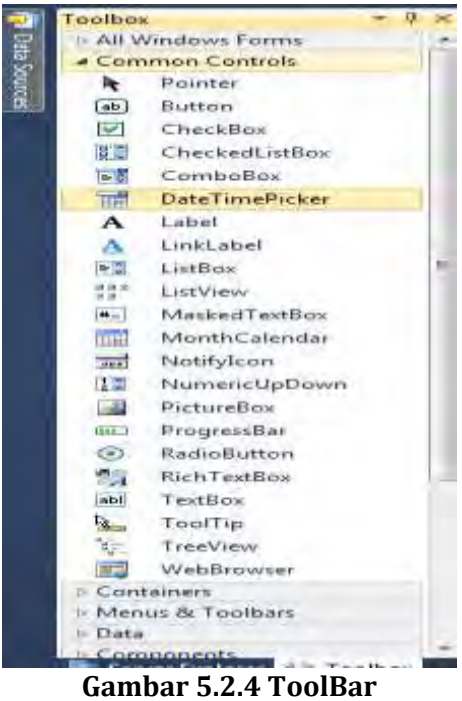
Gambar 5.2.2 Solution Explorer

4. **Form Designer, window** untuk merancang **user interface** dari aplikasi



Gambar 5.2.3 Form Designer

5. **Toolbox**, window yang berisi komponen-komponen yang digunakan untuk memperkaya **user interface.**



Gambar 5.2.4 ToolBar

Fungsi-Fungsi Komponen-Komponen Pada **Toolbox**:

a. **Fungsi *Tab Common Control***

1) *Pointer*: Digunakan untuk memindahkan atau mengubah ukuran kontrol dan form.
2) *Button*: Kontrol ini digunakan untuk melakukan tindakan ketika diakses.
3) *CheckBox*: Kontrol yang memiliki nilai *True* atau *False*.
4) *CheckedListBox*: Daftar kotak centang disamping item.
5) *ComboBox:* Kombinasi antara kotak list dan kotak teks yang memungkinkan untuk dipilih serta mengeditnya.
6) *DateTimePicker*: Menampilkan kalender untuk memilih hari dan tanggal.
7) *Label*: Menampilkan teks label.
8) *LinkLabel*: Menampilkan label dengan teks link.
9) *ListBox*: Kontrol yang berisi beberapa item.
10) *ListView*: Hampir sama seperti kontrol *ListBox*, tetapi dengan tambahan untuk membuat ikon dan judul.
11) *MaskedTextBox*: Menggunakan Mask untuk membedakan input teks yang tepat dan tidak tepat.
12) *MonthCalendar*: Dapat memilih tanggal saat runtime.
13) *NotifyIcon*: Menampilkan ikon pada Windows Tray.
14) *NumericUpDown*: Memungkinkan untuk memasukkan integer desimal tertentu dalam kisaran tertentu.
15) *PictureBox*: Menampilkan file gambar.
16) *ProgressBar*: Menampilkan proses dari sebuah task.
17) *RadioButton*: Memungkinkan untuk memilih pilihan dari sekelompok pilihan.
18) *RichTextBox*: Memungkinkan untuk mengedit dan menambahkan rich text.
19) *TextBox*: Kontrol yang digunakan untuk menampilkan atau memasukkan teks.
20) *ToolTip:* Menampilkan teks tooltip.

21) *TreeView*: Menampilkan hubungan antar node.

22) *WebBrowser*: Memungkinkan untuk membuka dokumen HTML di dalam form.

b. Fungsi Tab Containers

1) *Grupbox*: untuk mengorganisasikan melalui jendela entri data dan dapat diberi judul.kita dapat mengatur grub satu dengan grup yang lainnya.

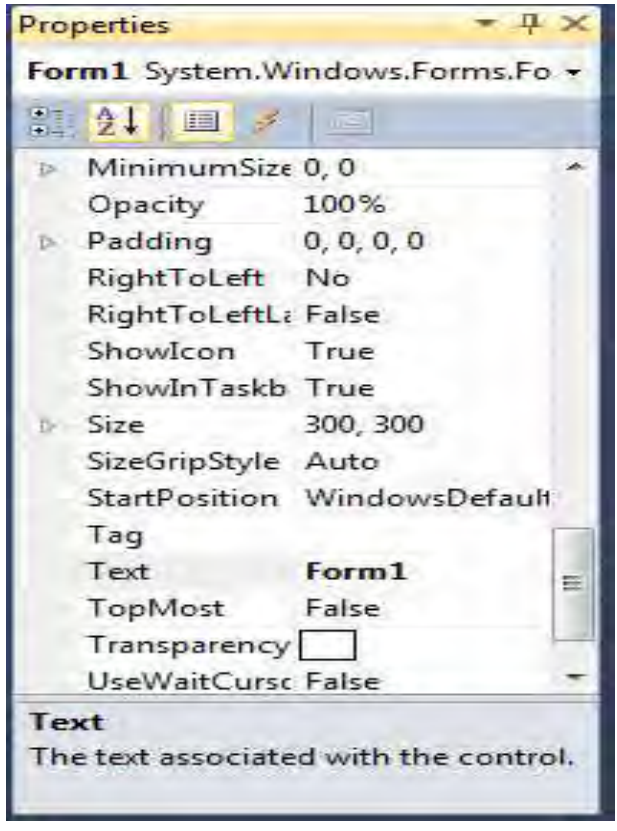
2) *Panel*: untuk mengorganisasikan melalui jendela entri data, hampir sama dengan *grubbox* tetapi tidak dapat diberi judul.

3) *Tabcontrol* : Untuk meletakkan halaman lebih dari satu jendela entri data (*TabPages*)

6. Code Editor, untuk menuliskan *source code* dari *program.source code* dapat ditampilkan dengan mengklik 2 kali pada form.

Gambar 5.2.5 *Code Editor* (tempat penulisan *source code*)

7. Properties, untuk melihat / mengedit sifat dari *object* yang sedang dipilih. Disinilah dapat mengubah *name, text*, jenis huruf, *background form* dan sebagainya.



Gambar 5.2.6 Menu Properties

c. ***Properti (Properties)*** yang ada pada komponen beserta kegunaanya **:**

Properti **(Properties)** yang ada pada komponen beserta kegunaanya :

a) *BackColor:* Mengatur warna latar belakang

b) *Font:* Untuk mengatur jenis tulisan, ukurannya dan gayanya (tebal,miring,garisbawah)

c) *ForeColor:* Mengatur warna tulisan object pada label

d) *Text:* Nama yang ditampilkan di *form* yang dilihat oleh *user*.

e) *Enable:* Menentukan sebuah object apakah dapat dimasukkan perintah yang dilakukan oleh *user*

f) *Visible:* Menentukan sebuah objek apakah dapat dilihat atau tidak selama proses dijalankan

g) *(Name):* Digunakan untuk identifikasi pada *form*. Hanya dapat dilihat oleh *programmer* untuk kebutuhan koding, tapi tidak dapat dilihat oleh user

h) *Size:* Menentukan ukuran sebuah objek (panjang dan lebar)

∞

BAB 6.
PEMOGRAMAN VISUAL C# 2010

6.1 Parsing Data

Dalam sebuah komunikasi protokol, ada data yang dikirim, biasanya data dikirim dalam satu paket. Dalam proses pembacaan data terlebih dahulu harus di pisah-pisahkan sesuai dengan jenis datanya. Proses pemisahan tersebut dinamakan Parsing Data. Cara memprogram parsing data menggunakan visual studio dengan pemograman C#. Harus dipersiapkan dahulu Visual Studio atau Sharp Develop atau software sejenisnya. Perintah dalam melakukan parsing data adalah sebagai berikut:

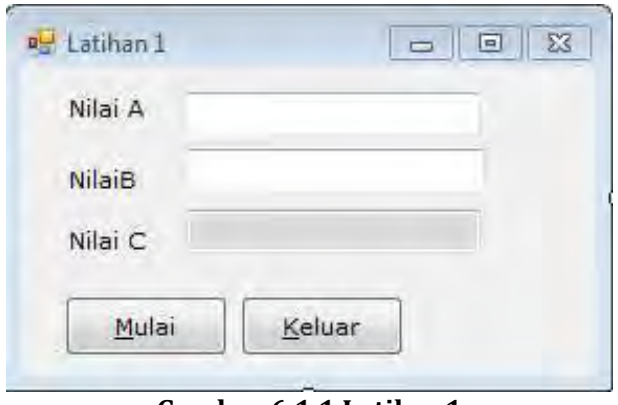
Convert.<metode>

Ada beberapa metode dalam melakukan parsing data :

Numeric Type	Metode
decimal	ToDecimal(String)
float	ToSingle(String)
double	ToDouble(String)
short	ToInt16(String)
int	ToInt32(String)
long	ToInt64(String)
ushort	ToUInt16(String)
uint	ToUInt32(String)
ulong	ToUInt64(String)

Latihan 1 :

1. Hasil C = Nilai A * Nilai B
2. Jika di klick tombol Mulai maka seluruh kotak isian akan kosong dan kursor di Nilai A
3. Jika di klik tombol Keluar maka akan keluar dari program



Gambar 6.1.1 Latihan1

Ubahlah isian properti dari setiap objek/control seperti dibawah ini :

Nama Objek/Control	Properties	
	Properti	Nilai yang diisikan
Form	Name Text	Form1 Latihan 1
Label	Text	Nilai A
TextBox	Name Text	txtA Kosongkan
Label	Text	Nilai B
TextBox	Name Text	txtB Kosongkan
Label	Text	Hasil C
TextBox	Name BackColor Enabled	txtC &H80000000& False
Button	Name Text	btnMulai &Mulai
Button	Name Text	btnKeluar &Keluar

1. Lakukan Penulisan Program dengan menDouble Klick control txtA

```
private void txtA_TextChanged(object sender, EventArgs e)
{
    int a=0, b=0, c;
    a = int.Parse(txtA.Text.Trim() != string.Empty ? txtA.Text.Trim() : "0");
    b = int.Parse(txtB.Text.Trim() != string.Empty ? txtB.Text.Trim() : "0");
    c = a * b;
    txtC.Text = Convert.ToString(c);
}
```

Gambar 6.1.2 Kode *Procedure Event Changed* txt A

Keterangan :

Event Change berfungsi kalau ada perubahan procedure baris dibawahnya akan dijalankan, **int.Parse()** berfungsi untuk merubah tipe data string menjadi tipe data integer. **String.Empty()** berfungsi untuk menghilangkan string nilai string atau bernilai kosong.

2. Lakukan double Klick control txtB

```
private void txtB_TextChanged(object sender, EventArgs e)
{
    int a, b, c;
    a = int.Parse(txtA.Text.Trim() != string.Empty ? txtA.Text.Trim() : "0");
    b = int.Parse(txtB.Text.Trim() !=string.Empty ? txtB.Text.Trim() : "0");
    c = a * b;
    txtC.Text = Convert.ToString(c);
}
```

Gambar 6.1.3 Kode *Procedure Event Changed* txt B

3. Double Klick kembali di btnMulai

```
private void btnMulai_Click(object sender, EventArgs e)
{
    txtA.Clear();
    txtB.Clear();
    txtC.Clear();
    txtA.Focus();
}
```

Gambar 6.1.4 Kode *Procedure Event* Click btn Mulai

Keterangan :

a. Event Click adalah jika Tindakan Mouse diClick maka program dibawahnya akan dijalankan
b. txtA.Clear() adalah untuk mengosongkan/membersihkan tombol txtA

c. txtB.Clear() adalah untuk mengosongkan/membersihkan tombol txtB
d. txtC.Clear() adalah untuk mengosongkan/membersihkan tombol txtC
e. txtA.Focus() adalah untuk meletakkan kursor di txtA

4. Double Klick btnKeluar

```
private void btnKeluar_Click(object sender, EventArgs e)
{
    this.Close();
}
```

Gambar 6.1.5 Kode *Procedure Event* Click btn Keluar

Latihan 2:

1. Jika di Klik Status="Kawin" akan keluar di Keterangan="Kawin"

   Jika di Klik Status="Tidak Kawin" akan keluar di Keterangan="Tidak Kawin"

2. Kalau ditekan Enter Kotak isiannya akan turun kebawah
3. Tombol Mulai seluruh kotak isian akan kosong
4. Tombol Keluar akan keluar dari program

Gambar 6.1.6 Latihan 2

Ubahlah isian properti dari setiap objek/control seperti dibawah ini :

Nama Objek/Control	Properties	
	Properti	Nilai yang diisikan
Form	Name	Form1
	Text	Latihan 2
Label	Text	NIK
TextBox	Name	txtNik
	Text	Kosongkan
Label	Text	Nama Karyawan
TextBox	Name	txtNama
	Text	Kosongkan
Label	Text	Alamat
TextBox	Name	txtAlamat
Label	Text	Keterangan
TextBox	Name	txtKet
	BackColor	MenuBar
	Enabled	False
GroupBox1	Text	[Jenis Kelamin]
RadioButton	Name	rdKawin
	Text	Kawin
RadioButton	Name	rdTKawin
	Text	Tidak Kawin
Button	Name	btnMulai
	Text	&Mulai
Button	**Name**	btnKeluar
	Text	&Keluar

Kode Program:

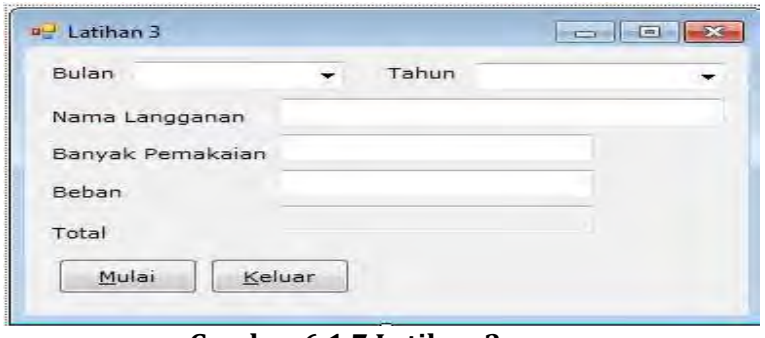
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Lat2
{
 public partial class Form1 : Form
```

```
 {
 public Form1()
 {
 InitializeComponent();
 }

 private void rdKawin_CheckedChanged(object sender, EventArgs
e)
 {
 txtKet.Text = "Kawin";
 }

 private void rdTKawin_CheckedChanged(object sender,
EventArgs e)
 {
 txtKet.Text = "Tidak Kawin";
 }

 private void txtNik_TextChanged(object sender, EventArgs e)
 {
 if (rdKawin.Checked == true)
 {
txtKet.Text = "Kawin";
 }
 else
 {
txtKet.Text = "Tidak Kawin";
 }
 }

 private void txtNama_TextChanged(object sender, EventArgs e)
 {
 if (rdKawin.Checked == true)
 {
txtKet.Text = "Kawin";
 }
 else
 {
txtKet.Text = "Tidak Kawin";
 }
 }
```

```
 private void txtAlamat_TextChanged(object sender, EventArgs e)
 {
 if (rdKawin.Checked == true)
 {
txtKet.Text = "Kawin";
 }
 else
 {
txtKet.Text = "Tidak Kawin";
 }
 }

 private void btnMulai_Click(object sender, EventArgs e)
 {
 txtNik.Clear();
 txtNama.Clear();
 txtAlamat.Clear();
 txtKet.Clear();
 txtNik.Focus();
 }

 private void btnKeluar_Click(object sender, EventArgs e)
 {
 this.Close();
 }
 }
}
```

Latihan 3:

1. Total = Banyak Pemakaian * Beban
2. Bulan akan muncul nama bulan, dan Tahun akan muncul dari 1980 sampai 2010



Gambar 6.1.7 Latihan 3

Ubahlah isian properti dari setiap objek/control seperti dibawah ini :

Nama Objek/Control	Properties	
	Properti	Nilai yang diisikan
Form	Name	Form1
	Text	Latihan 3
Label	Text	Bulan
combobox	Name	cboBulan
	TabIndex	0
Label	Text	Tahun
combobox	Name	cboTahun
	TabIndex	1
Label	Text	Nama Langganan
TextBox	Name	txtNama
	TabIndex	2
Label	Text	Banyak Pemakaian
TextBox	Name	txtBanyak
	TabIndex	3
Label	Text	Beban
TextBox	Name	txtBeban
	TabIndex	4
Label	Text	Total
TextBox	Name	txtTot
	BackColor	MenuBar
	Enabled	False
Button	Name	btnMulai
	Text	&Mulai
Button	Name	btnKeluar
	Text	&Keluar

Kode program:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Lat_3
{
 public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();
 }

 private void Form1_Load(object sender, EventArgs e)
 {
 bulan();
 tahun();
 }
 private void bulan()
 {
 for (int i = 1; i <= 12; i++)
 {


cboBulan.Items.Add(System.Globalization.DateTimeFormatInfo.Curr
entInfo.GetMonthName(i));
 cboBulan.SelectedIndex = 0;
 }
 }
 private void tahun()
 {
 for (int i = 1940; i <= DateTime.Now.Year; i++)
 {
 cboTahun.Items.Add(i);
 cboTahun.SelectedIndex = 0;
```

```
 }

 }

 private void btnMulai_Click(object sender, EventArgs e)
 {
 txtNama.Clear();
 txtBanyak.Clear();
 txtBeban.Clear();
 txtTotal.Clear();
 txtNama.Focus();
 }

 private void btnKeluar_Click(object sender, EventArgs e)
 {
 this.Close();
 }

 private void txtNama_KeyPress(object sender, KeyPressEventArgs
e)
 {
 if (e.KeyChar == 13)
 {
 SendKeys.Send("{tab}");
 }
 }

 private void cboBulan_KeyDown(object sender, KeyEventArgs e)
 {
 if (e.KeyCode == Keys.Enter)
 {
 SendKeys.Send("{tab}");
 }
 }

 private void cboTahun_KeyDown(object sender, KeyEventArgs e)
 {
 if (e.KeyCode == Keys.Enter)
 {
 SendKeys.Send("{tab}");
 }
```

```
 }

 private void txtBanyak_KeyPress(object sender, KeyPressEventArgs
e)
 {
 if (e.KeyChar == 13)
 {
 SendKeys.Send("{tab}");
 }
 }
 private void txtBeban_KeyPress(object sender, KeyPressEventArgs
e)
 {
 if (e.KeyChar == 13)
 {
 SendKeys.Send("{tab}");
 }
 }
 private void total()
 {
 float banyak, beban, tot;
 banyak = float.Parse(txtBanyak.Text.Trim() !=string.Empty ?
txtBanyak.Text.Trim() :"0");
 beban = float.Parse(txtBeban.Text.Trim() != string.Empty ?
txtBeban.Text.Trim() : "0");
 tot = banyak * beban;
 txtTotal.Text = tot.ToString();
 }

 private void txtBanyak_TextChanged(object sender, EventArgs e)
 {
 total();
 }

 private void txtBeban_TextChanged(object sender, EventArgs e)
 {
 total();
 }

  }
}
```

Latihan 4

1. Jumlah = Biaya * Banyak
2. Jika Potongan=DVD maka DVD= Jumlah*0.1
3. Jika Potongan=Langganan maka Lgn=Jumlah*0.1
   Jika Potongan=ID Card maka IDCard=Jumlah*0.1
4. Bonus=DVD+Lgn+IDCard
5. Total=Jumlah-Bonus

Gambar 6.1.8 Latihan 4

Ubahlah isian properti dari setiap objek/control seperti dibawah ini :

Nama Objek/Control	Properties	
	Properti	Nilai yang diisikan
Form	Name	Form1
	Text	Latihan 4
Label	Text	Kode Film
TextBox	Name	txtKode
	TabIndex	0
Label	Text	Judul Film
TextBox	Name	txtJudul
	TabIndex	1

Label	Text	Biaya Sewa
TextBox	Name	txtBiaya
	TabIndex	2
Label	Text	Banyak
TextBox	Name	txtBanyak
	TabIndex	3
Label	Text	Jumlah
TextBox	Name	txtJumlah
	TabIndex	4
	BackColor	MenuBar
	Enabled	False
Label	Text	Diskon
TextBox	Name	txtDiskon
	BackColor	MenuBar
	Enabled	False
Label	Text	Total
TextBox	Name	txtTotal
	BackColor	MenuBar
	Enabled	False
GroupBox	Text	[Potongan]
CheckBox	Name	chkDVD
	Text	DVD
CheckBox	Name	chkLangganan
	Text	Langganan
CheckBox	Name	chkID
	Text	ID Card
Button	Name	btnMulai
	Text	&Mulai
Button	Name	btnKeluar
	Text	&Keluar

Kode program:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
```

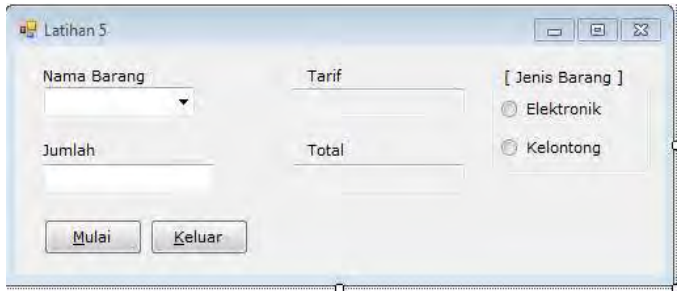
```
namespace Lat_4
{
  public partial class Form1 : Form
  {
  public Form1()
  {
  InitializeComponent();
  }

  private void Form1_Load(object sender, EventArgs e)
  {
  }
  private void Hitung()
  {
  double biaya, banyak, jumlah, diskon, total, dvd=0, lang=0, id=0;
  biaya = double.Parse(txtBiaya.Text.Trim() != string.Empty ?
txtBiaya.Text.Trim() : "0");
  banyak = double.Parse(txtBanyak.Text.Trim() != string.Empty ?
txtBanyak.Text.Trim() : "0");
  jumlah = biaya * banyak;
  if (chkDVD.Checked == true)
  {
 dvd = jumlah * 0.1;
  }
  if (chkID.Checked == true)
  {
 id = jumlah * 0.1;
  }
  if (chkLangganan.Checked == true)
  {
 lang = jumlah * 0.1;
  }
  diskon = dvd + id + lang;
  total = jumlah - diskon;
  txtJumlah.Text = jumlah.ToString();
  txtDiskon.Text = diskon.ToString();
  txtTotal.Text = total.ToString();
  }

  private void txtBiaya_TextChanged(object sender, EventArgs e)
  {
```

```
 Hitung();
 }

 private void txtBanyak_TextChanged(object sender, EventArgs e)
 {
 Hitung();
 }

 private void chkDVD_CheckedChanged(object sender, EventArgs e)
 {
 Hitung();
 }

 private void chkLangganan_CheckedChanged(object sender,
EventArgs e)
 {
 Hitung();
 }

 private void chkID_CheckedChanged(object sender, EventArgs e)
 {
 Hitung();
 }

 private void txtKode_KeyPress(object sender, KeyPressEventArgs
e)
 {
 if (e.KeyChar == 13)
 {
 SendKeys.Send("{tab}");
 }
 }

 private void txtJudul_KeyPress(object sender, KeyPressEventArgs
e)
 {
 if (e.KeyChar == 13)
 {
 SendKeys.Send("{tab}");
 }
 }
```

```
  private void txtBiaya_KeyPress(object sender, KeyPressEventArgs
e)
  {
  if (e.KeyChar == 13)
  {
 SendKeys.Send("{tab}");
  }
  }

  private void txtBanyak_KeyPress(object sender,
KeyPressEventArgs e)
  {
  if (e.KeyChar == 13)
  {
 SendKeys.Send("{tab}");
  }
  }
  }
}
```

Latihan 5:

1. Jika Jenis Barang =”Kelontong” maka
   - ❖ Nama Barang — Tarif
     - Panci — 56000
     - Sendok — 6000
     - Piring — 7800
2. Jika Jenis Barang=”Elektronika” maka
   - ❖ Nama Barang — Tarif
     - TV 21 Inchi — 1970000
     - Mini Compo — 1500000
     - DVD Player — 780000
3. Total = Tarif * Jumlah



Gambar 6.1.9 Latihan 5

Ubahlah isian properti dari setiap objek/control seperti dibawah ini :

Nama Objek/Control	Properties	
	Properti	Nilai yang diisikan
Form	Name	Form1
	Text	Latihan 5
Label	Text	Nama Barang
ComboBox	Name	cboNama
	TabIndex	0
Label	Text	Jumlah
TextBox	Name	txtJumlah
	TabIndex	1
Label	Text	Tarif
TextBox	Name	txtTarif
	TabIndex	2
	BackColor	MenuBar
	Enabled	False
Label	Text	Total
TextBox	Name	txtTotal
	TabIndex	3
	BackColor	MenuBar
	Enabled	False
GroupBox	Text	[Jenis Barang]
RadioButton	Name	rdElek
	Text	Elektronika
RadioButton	Name	rdKel
	Text	Kelontong
Button	Name	btnMulai
	Text	&Mulai
Button	Name	btnKeluar
	Text	&Keluar

Kode program:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Lat_5
{
 public partial class Form1 : Form
 {

 public Form1()
 {
 InitializeComponent();
 }

 private void Form1_Load(object sender, EventArgs e)
 {
 Hitung();
 }

 private void Hitung()
 {
 float total, tarif, jumlah;
 if (rdElek.Checked == true)
 {
cboNama.DisplayMember = "Text";
cboNama.ValueMember = "Value";
DataTable tb = new DataTable();
tb.Columns.Add("Text", Type.GetType("System.String"));
tb.Columns.Add("Value", Type.GetType("System.Int64"));
tb.Rows.Add("TV 21 inchi", 197000);
tb.Rows.Add("Mini Compo", 150000);
tb.Rows.Add("DVD Player", 780000);
cboNama.DataSource = tb;
 }
 else
```

```
 {
 cboNama.DisplayMember = "Text";
 cboNama.ValueMember = "Value";
 DataTable tb = new DataTable();
 tb.Columns.Add("Text", Type.GetType("System.String"));
 tb.Columns.Add("Value", Type.GetType("System.Int64"));
 tb.Rows.Add("Panci", 56000);
 tb.Rows.Add("Sendok", 6000);
 tb.Rows.Add("Piring", 7800);
 cboNama.DataSource = tb;
 }
 tarif = float.Parse(txtTarif.Text.Trim() !=string.Empty
?txtTarif.Text.Trim(): "0");
 jumlah = float.Parse(txtJumlah.Text.Trim() !=string.Empty
?txtJumlah.Text.Trim() : "0");
 total = jumlah * tarif;
 txtTotal.Text = total.ToString();
 }

 private void rdElek_Click(object sender, EventArgs e)
 {
 Hitung();
 }

 private void cboNama_SelectedIndexChanged(object sender,
EventArgs e)
 {
 txtTarif.Text = cboNama.SelectedValue.ToString();
 }

 private void txtJumlah_TextChanged(object sender, EventArgs e)
 {
 Hitung();
 }

 private void rdKel_Click(object sender, EventArgs e)
 {
 Hitung();
 }

 private void btnMulai_Click(object sender, EventArgs e)
```

```
 {
 txtJumlah.Clear();
 txtTotal.Clear();
 txtTarif.Clear();
 txtJumlah.Focus();
 }

 private void btnKeluar_Click(object sender, EventArgs e)
 {
 this.Close();
 }
 }
}
```

Latihan 6:

1. Jika Jenis="Film" maka
   - ❖ Nama Barang Harga
     - Kawin Lagi - 50000
     - Asmara Berdarah - 35000
     - Darah Biru - 20000

   Jika Jenis="Musik" maka
   - ❖ Nama Barang Harga
     - Bergek: Boh Hate 2 - 45000
     - Puja Sharma - 20000
     - Apache 13: Bek Panik - 25000
     - Kutidhieng - 23000
2. Jumlah=Harga*Banyak
3. Jika Banyak>10 maka Potongan=Jumlah*0.1
   Jika Banyak<0 maka Potongan=0
4. Jika Bonus="Member" maka Member=Jumlah*0.05
   Jika Bonus = "Hari Besar" maka Hari Besar = Jumlah*0.05
   Jika Bonus = "Ulang Tahun" maka Ulang Tahun = Jumlah*0.05
5. Diskon=Potongan+Member+Hari Besar+Ulang Tahun
6. Total Bayar=Jumlah-Diskon
7. Uang Kembali=Uang Bayar – Total Bayar

Gambar 6.1.10 Latihan 6

Ubahlah isian properti dari setiap objek/control seperti dibawah ini :

Nama Objek/Control	Properties	
	Properti	Nilai yang diisikan
Form	Name	Form1
	Text	Latihan 6
Label	Text	Nama Pembeli
TextBox	Name	txtNama
	TabIndex	0
Label	Text	Nama Barang
ComboBox	Name	cboNama
	TabIndex	1
Label	Text	Harga
TextBox	Name	txtHarga
	TabIndex	2
	BackColor	MenuBar
	Enabled	False
Label	Text	Banyak
TextBox	Name	txtBanyak
	TabIndex	3
Label	Text	Jumlah
TextBox	Name	txtJumlah
	TabIndex	4
	BackColor	MenuBar
	Enabled	False
Label	Text	Diskon
TextBox	Name	txtDiskon
	TabIndex	5

	BackColor	MenuBar
	Enabled	False
Label	Text	Total Bayar
TextBox	Name	txtTotal
	TabIndex	6
	BackColor	MenuBar
	Enabled	False
Label	Text	Uang Bayar
TextBox	Name	txtBayar
	TabIndex	7
Label	Text	Uang Kembali
TextBox	Name	txtKembali
	TabIndex	8
	BackColor	MenuBar
	Enabled	False
GroupBox	Text	[Jenis Barang]
RadioButton	Name	rdMusik
	Text	Musik
RadioButton	Name	rdFilm
	Text	Film
GroupBox	Text	[Bonus]
CheckBox	Name	chkMember
	Text	Member
CheckBox	Name	chkHari
	Text	Hari Besar
CheckBox	Name	chkUlang
	Text	Ulang Tahun
Button	Name	btnMulai
	Text	&Mulai
Button	Name	btnKeluar
	Text	&Keluar

Kode program:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
```

```
namespace Lat_6
{
 public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();
 }

 private void btnMulai_Click(object sender, EventArgs e)
 {
 txtNama.Clear();
 txtBanyak.Clear();
 txtHarga.Clear();
 txtJumlah.Clear();
 txtDiskon.Clear();
 txtTotal.Clear();
 txtBayar.Clear();
 txtKembali.Clear();
 txtNama.Focus();
 }

 private void TampilBarang()
 {
 if (rdFilm.Checked == true)
 {
cboBarang.DisplayMember = "Text";
cboBarang.ValueMember = "Value";
DataTable tb = new DataTable();
tb.Columns.Add("Text", Type.GetType("System.String"));
tb.Columns.Add("Value", Type.GetType("System.Int64"));
tb.Rows.Add("Kawin Lagi", 50000);
tb.Rows.Add("Asmara Berdarah", 35000);
tb.Rows.Add("Darah Biru", 20000);
cboBarang.DataSource = tb;
 }
 else
 {
cboBarang.DisplayMember = "TExt";
cboBarang.ValueMember = "Value";
DataTable tb = new DataTable();
```

```
 tb.Columns.Add("Text", Type.GetType("System.String"));
 tb.Columns.Add("Value", Type.GetType("System.Int64"));
 tb.Rows.Add("Bergek: Boh Hate 2", 45000);
 tb.Rows.Add("Puja Sharma", 20000);
 tb.Rows.Add("Apache 13: Bek Panik", 25000);
 tb.Rows.Add("Kutidhieng", 23000);
 cboBarang.DataSource = tb;
  }
  }
  private void Hitung()
  {
  double harga, banyak, jumlah,
potongan,diskon,hari=0,member=0,besar=0;
            double total,bayar,kembali;
  harga = double.Parse(txtHarga.Text.Trim() !=string.Empty ?
txtHarga.Text.Trim() : "0");
  banyak = double.Parse(txtBanyak.Text.Trim() != string.Empty ?
txtBanyak.Text.Trim() : "0");
  bayar = double.Parse(txtBayar.Text.Trim() != string.Empty ?
txtBayar.Text.Trim() : "0");
  jumlah = harga * banyak;
  if (banyak > 10)
  {
 potongan = jumlah * 0.1;
  }
  else
  {
 potongan = 0;
  }
  if (chkHari.Checked == true)
  {
 hari = jumlah * 0.05;
  }
  if (chkMember.Checked == true)
  {
 member = jumlah * 0.05;
  }
  if (chkUlang.Checked == true)
  {
 besar = jumlah * 0.05;
  }
```

```
  diskon = potongan+hari + member + besar;
  total = jumlah - diskon;
  kembali = bayar - total;
  txtJumlah.Text = jumlah.ToString();
  txtDiskon.Text = diskon.ToString();
  txtTotal.Text = total.ToString();
  txtKembali.Text = kembali.ToString();
  }

  private void cboBarang_SelectedIndexChanged(object sender,
EventArgs e)
  {
  txtHarga.Text = cboBarang.SelectedValue.ToString();
  }

  private void rdFilm_CheckedChanged(object sender, EventArgs e)
  {
  TampilBarang();
  }

  private void rdMusik_CheckedChanged(object sender, EventArgs
e)
  {
  TampilBarang();
  }

  private void txtBanyak_TextChanged(object sender, EventArgs e)
  {
  Hitung();
  }

  private void txtBayar_TextChanged(object sender, EventArgs e)
  {
  Hitung();
  }

  private void chkMember_CheckedChanged(object sender,
EventArgs e)
  {
  Hitung();
  }
```

```
  private void chkHari_CheckedChanged(object sender, EventArgs e)
  {
  Hitung();
  }

  private void chkUlang_CheckedChanged(object sender, EventArgs
e)
  {
  Hitung();
  }

  private void btnKeluar_Click(object sender, EventArgs e)
  {
  this.Close();
  }
  }
}
```

∞

BAB 7.
DATABASE

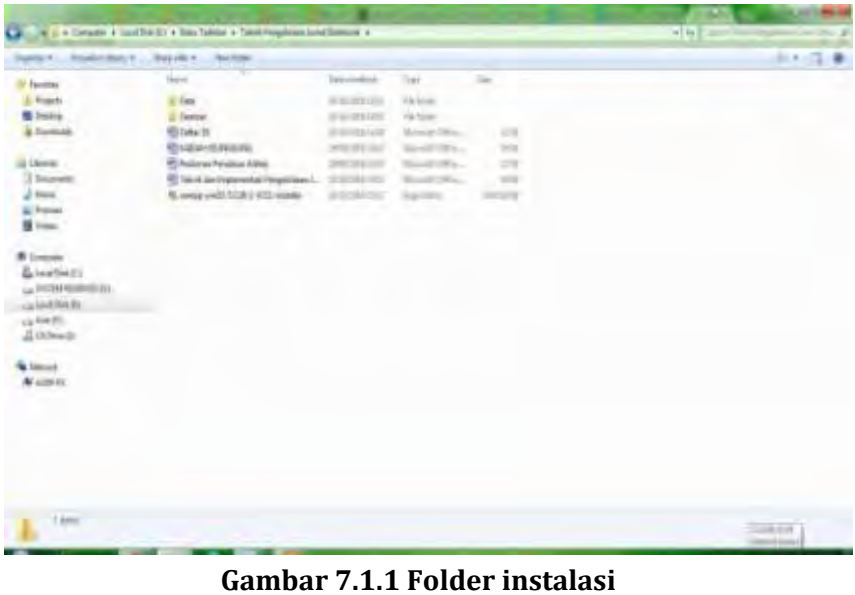
7.1 Instalasi MySQL

Aplikasi XAMPP ini dibuat oleh *Apache Friends* dan installer-nya bisa langsung diunduh dari situs mereka. Isi aplikasinya juga sudah sangat komplit, antara lain:

1. Apache
2. MySQL
3. PHP
4. phpMyAdmin
5. FileZilla FTP Server
6. Tomcat
7. XAMPP Control Panel

Proses instalasi Xampp di Microsoft Windows sebagai berikut:

1. Download aplikasi Xampp terbaru di https://www.apachefriends.org/download.html
2. Sebelum melakukan instalasi Xampp, **Anti Virus di non aktifkan**
3. Jalankan file **xampp-win32-5.5.38-1-VC11-installer** (cari versi lebih tinggi) di folder tempat penyimpanan

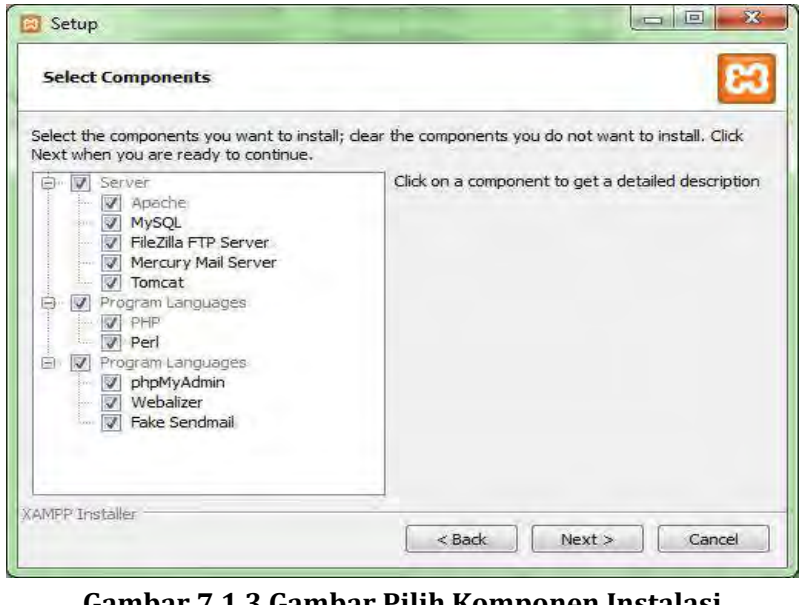


Gambar 7.1.1 Folder instalasi

4. Klik dua kali file xampp-win32-5.5.38-1-VC11-installer
5. Muncul menu seperti berikut ini

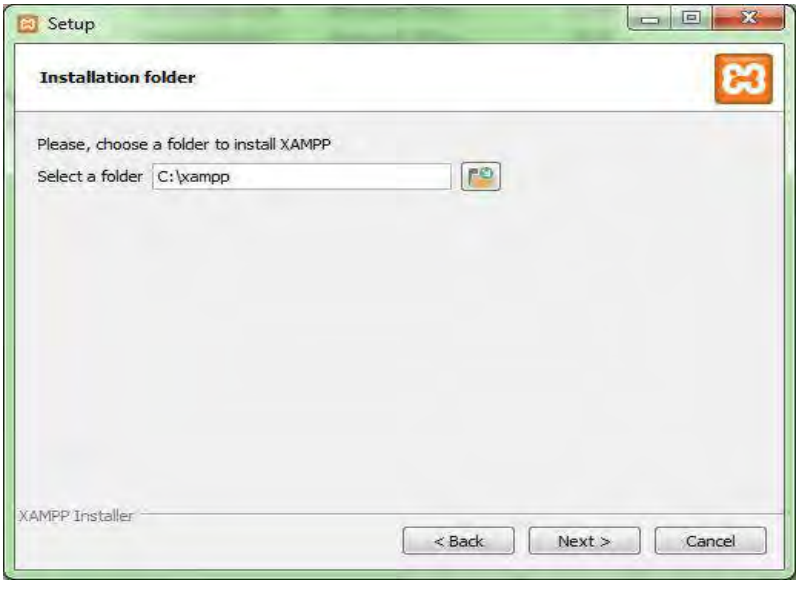
Gambar 7.1.2 Gambar Setup XAMPP

6. Kemudian Klik Next, muncul menu seperti dibawah ini



Gambar 7.1.3 Gambar Pilih Komponen Instalasi

7. Kemudian Klik Next, muncul lagi menu seperti di bawah ini

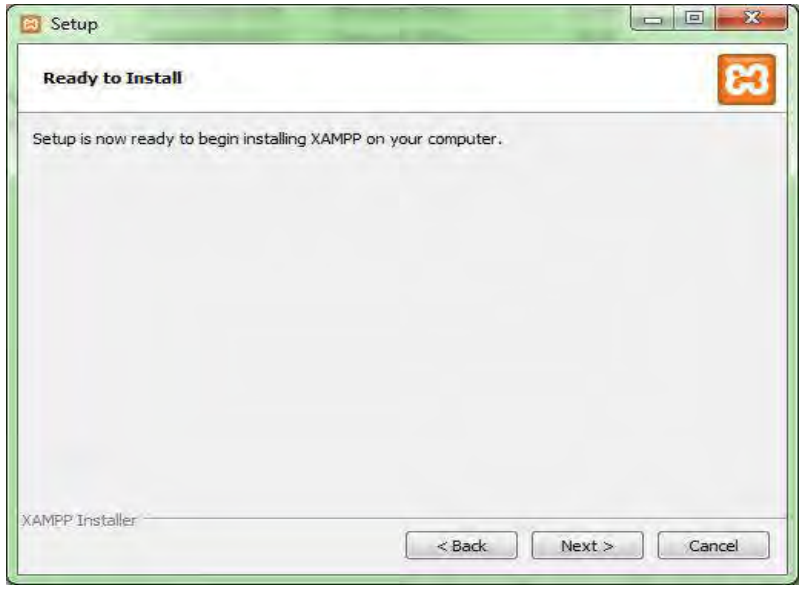


Gambar 7.1.4 Gambar Tempat Penyimpanan Instalasi

8. Kemudian Klik Next, atau pilih Folder lainnya, lebih baik abaikan

Gambar 7.1.5 Gambar Paket Dukungan XAMPP

9. Kemudian Klik Next, akan muncul menu berikut ini



Gambar 7.1.6 Gambar Instalasi Siap Dilakukan

10. Kemudian Klik Next lagi, muncul menu seperti dibawah ini

Gambar 7.1.7 Gambar Proses Instalasi XAMPP

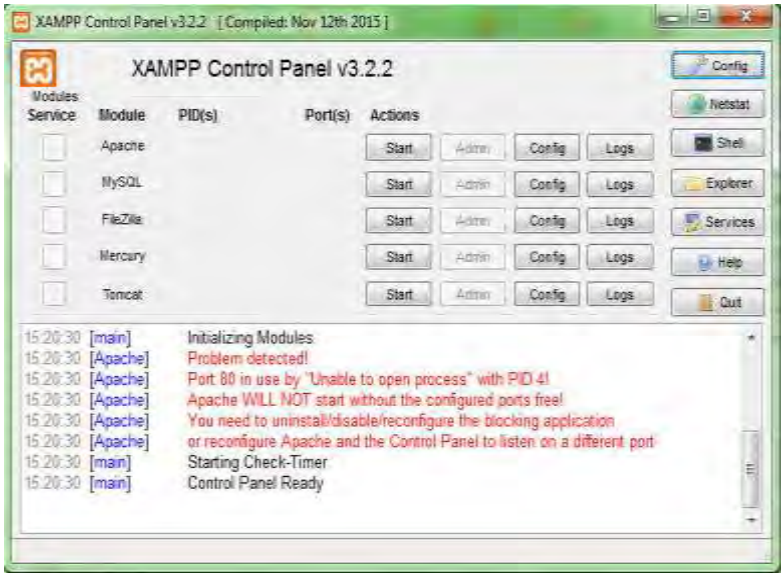
11. Tunggu sampai proses instalasi berhasil dan muncul menu seperti dibawah ini

Gambar 7.1.8 Gambar Instalasi Selesai

12. Kemudian Klik Finish, Proses instalasi Selesai dan muncul Menu seperti dibawah ini lagi dan pilih bendera Amerika klik **Save**

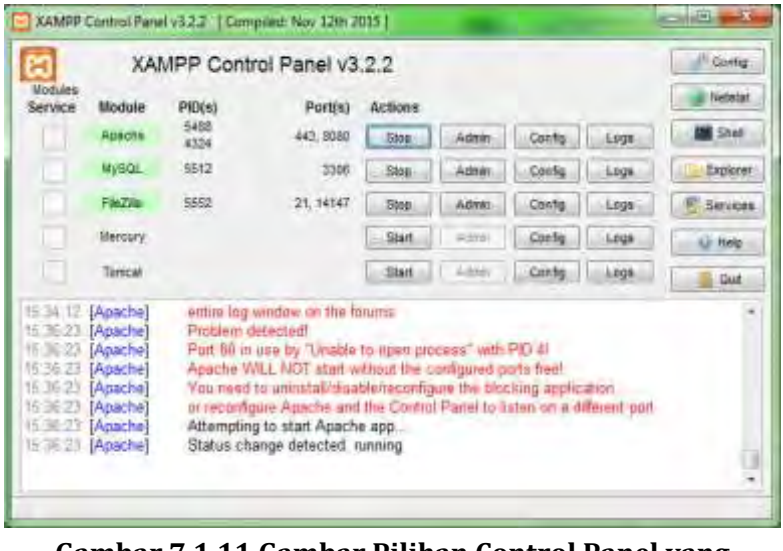
Gambar 7.1.9 Gambar Pilihan Penggunaa Bahasa

13. Kemudian akan muncul Xampp Control Panel



Gambar 7.1.10 Gambar XAMPP Control Panel

14. Klik tombol Start di pilihan Module Apache
15. Klik tombol Start di pilihan Module MySQL, lihat seperti gambar dibawah ini



Gambar 7.1.11 Gambar Pilihan Control Panel yang Dijalankan

Cara Menjalankan Aplikasi XAMPP:

1. Bukalah aplikasi XAMPP, bisa melalui S**tart Menu** atau **Desktop,** dan klik icon XAMPP. Atau, jika Anda membukanya begitu proses instalasi selesai maka klik **Yes** seperti yang terlihat pada gambar di atas.
2. Setelah terbuka, silahkan klik tombol **Start** pada kolom **Action** sehingga tombol tersebut berubah menjadi **Stop.** Dengan mengklik tombol tersebut, artinya itulah aplikasi yang dijalankan. Biasanya jika saya menggunakan XAMPP, yang saya start hanyalah aplikasi Apache dan MySQL, karena saya tidak memerlukan aplikasi seperti Filezilla, dan lain-lain.
3. Lihat Gambar di atas
4. Buka Browser Web pada address bar ketik : http://localhost atau 127.0.0.1
5. Lihat Gambar di bawah ini

Gambar 7.1.12 Gambar Laman XAMPP Berhasil

6. Selamat Instalasi Xampp berhasil.

7.2 Pengenalan MySQL

SQL (*Structured Query Language*) merupakan bahasa yang digunakan dalam mengakses data di basis data relasional. Bahasa ini merupakan bahasa standar yang digunakan dalam manajemen basis data relasional. Hampir semua server basis data yang ada mendukung bahasa ini untuk melakukan manajemen basis data. SQL terdiri dari dua bahasa, yaitu *Data Definition Language* (DDL) dan *Data Manipulation Language* (DML). Implementasi DDL dan DML berbeda untuk tiap sistem manajemen basis data, secara umum implementasi tiap bahasa ini memiliki bentuk standar yang ditetapkan.

Sebagai sebuah bahasa, SQL telah distandarisasi dan mengalami beberapa perubahan atau penyempurnaan. SQL muncul pertama kali pada tahun 1970 dengan nama *Sequel* (nama yang masih sering digunakan hingga saat ini). Standarisasi yang pertama dibuat pada tahun 1986 oleh ANSI (*American National Standards Institute*) dan ISO (*International Standard Organization*), yang disebut SQL-86. Pada tahun 1989 SQL-86 diperbaharui menjadi SQL-89. Standar terakhir yang dibuat adalah SQL-92.

Fungsi dari *Structured Query Language* adalah sebagai berikut:

a. SQL dapat mengeksekusi query terhadap database
b. SQL dapat mengambil data dari database
c. SQL dapat menyisipkan catatan dalam database
d. SQL dapat memperbarui catatan dalam database
e. SQL dapat menghapus catatan dari database
f. SQL dapat membuat database baru
g. SQL dapat membuat tabel baru dalam database
h. SQL dapat menciptakan prosedur yang tersimpan dalam database
i. SQL dapat membuat tampilan dalam database
j. SQL dapat mengatur hak akses pada tabel, prosedur, dan pandangan

7.2.1 Data *Definition Language*

Data Definition Language (DDL) merupakan sub bahasa SQL yang digunakan untuk membangun kerangka database. Kelompok perintah yang berfungsi untuk mendefinisikan atribut-atribut database, tabel, atribut kolom, batasan-batasan terhadap suatu atribut serta hubungan antar tabel. Dibawah ini yang termasuk kelompok DDL ini adalah

a. CREATE digunakan untuk membuat Database dan Table
   1) Bentuk umum membuat database:
      CREATE DATABASE nama_database;

contoh:
CREATE DATABASE akademik

2) Bentuk umum untuk membuat tabel
CREATE TABLE table_name (column1 datatype, column2 datatype, column3 datatype);

Pengertiannya:
- Table_name adalah nama dari tabel yang ingin dibuat.
- column_1 adalah berisi definisi dari nama_field yang akan dibuat untuk table
- datatype adalah query opsional untuk mendefinisikan tipe tabel untuk tabel yang akan digunakan, seperti MyISAM maupun InnoDB.

contoh:
CREATE TABLE matakuliah (kd_mtk varchar(9), nm_mtk varchar(30), sks int)

b. ALTER digunakan untuk melakukan perubahan struktur table yang telah dibuat, baik menambah field (ADD), mengganti nama field (Change) ataupun mengganti nama (Rename), dan menghapus field (Drop).

1) Bentuk umum script menambah field / kolom table:
ALTER table nama_table ADD field_baru tipe_data (panjang_data);

contoh:
ALTER table matakuliah ADD semester varchar (2);

2) Bentuk umum script menghapus field pada table
ALTER table nama_table DROP nama_field;

contoh:
ALTER table siswa DROP semester;

3) Bentuk umum script mengganti nama table:
RENAME table nama_table TO table baru;

contoh:
RENAME table matakuliah TO mata_kuliah;

4) Bentuk umum script mengganti nama filed
ALTER table nama_table change field_lama field_baru tipe_data (panjang data);

contoh:

ALTER table matakuliah change kd_mtk kode_mtk varchar (9);

c. DROP digunakan untuk menghapus Database Dan Table
   1) Bentuk umum script untuk menghapus database
      DROP DATABASE nama_database;

      contoh:
      DROP DATABASE akademik;

   2) Bentuk umum script menghapus table:
      DROP TABLE nama_table;

      contoh:
      DROP TABLE matakuliah;

7.2.2 Fungsi Data *Manipulation Language*

Data Manipulation Language (DML) merupakan satu paket DBMS yang memperbolehkan pemakai untuk mengakses atau memanipulasi data sebagaimana yang telah diorganisasikan sebelumnya dalam model data yang tepat. *Data Manipulation Language* (DML) merupakan kumpulan perintah SQL yang digunakan untuk proses pengolahan isi data di dalam *table* seperti memasukkan, merubah dan menghapus data, data yang dimanipulasi tidak terkait dengan perubahan struktur dan definisi tipe data dari objek database. DML dapat melakukan perintah sebagai berikut :

1. Mengambil informasi yang tersimpan dalam basis data.
2. Menyisipkan informasi baru dalam basis data.
3. Menghapus informasi dari tabel.
4. Merubah data dalam basis data

Ada dua tipe DML yaitu prosedural dan non prosedural. Prosedural DML membutuhkan pemakai untuk menspesifikasikan data yang dibutuhkan dan bagaimana cara mendapatkannya,. non prosedural DML membutuhkan pemakai untuk menspesifikasikan data apa yang dibutuhkan tanpa tahu bagaimana cara mendapatkannya.

Data Manipulation Language berfungsi untuk memanipulasi database seperti: menambah data, merubah atau mengganti data dan menghapus data. Perintah *Data Manipulation Language* tidak terkait dengan struktur dan metadata dari obyek yang berada pada tabel database. Perintah-perintah yang paling sering digunakan pada *Data Manipulation Language* adalah:

1. *Insert* berfungsi untuk menambah data atau record pada database
   Bentuk umum:

   insert into nama_table (field_1,field_2, field_n) values (data_1,data_2,data_n)

   contoh:
   insert into matakuliah(kd_mtk,nm_mtk,sks) values ('kom-001','Basis Data',3)

2. *Delete* berfungsi untuk menghapus data pada database
   Bentuk umum:

   delete from nama_table where field_1=data_1

   contoh:
   delete from matakuliah where kd_mtk='kom-001'

3. *Update* yaitu perintah yang berfungsi untuk merubah maupun memperbarui data pada database
   Bentuk umum:

   UPDATE nama_table set field_1=data_baru_1 where field_1=data_lama_1

   contoh:
   update matakuliah set nm_mtk='Basis Data I' where kd_mtk='kom-001'

4. *Select* yaitu perintah yang digunakan untuk menampilkan data dari suatu tabel pada database.
   a. Bentuk umum untuk menampilkan beberapa kolom dari suatu table

   SELECT field_1, field_n FROM table_name;

   contoh:

   select kd_mtk, nm_mtk from matakuliah

   b. Bentuk umum untuk menampilkan semua kolom dari suatu tabel :

   SELECT * FROM table_name;

   contoh:

   select * from matakuliah

   c. Perintah WHERE digunakan untuk menampilkan record dengan kriteria tertentu dalam suatu table bentuk umum dalam penulisan WHERE :

   SELECT field_1, field_n FROM table_name WHERE field_1=data_1

contoh:

select kd_mtk,nm_mtk_sks from matakuliah where kd_mtk='kom-001'

7.2.3 *Data Control Language* (DCL)

Data Control Language (DCL) merupakan sub bahasa SQL yang digunakan untuk melakukan pengontrolan data dan server databasenya. *Data Control Language* adalah bagian inti dari *Structured Query Language* (SQL) yang mempunyai kemampuan untuk mengatur hak akses terhadap sebuah basis data. *Data Control Language* terbagi dua :

a. GRANT berfungsi untuk memberikan Hak Akses
Bentuk umum :

GRANT priviliges ON tbname TO user;

contoh :
GRANT select, insert, update, delete ON matakuliah TO bergek;

Pengertian:

Perintah GRANT di atas menunjukkan bahwa user bergek diberikan hak akses untuk menampilkan, menambah, memodifikasi dan menghapus data pada table matakuliah.

b. REVOKE berfungsi untuk mencabut Hak Akses
Bentuk umum :

REVOKE priviliges ON tbname from user;

Contoh :
REVOKE insert, update, delete ON matakuliah FROM bergek;

Pengertian:

Perintah REVOKE di atas menunjukkan bahwa sebagian hak akses dari bergek dicabut kembali. Hak akses yang dicabut adalah hak untuk menambah, memodifikasi dan menghapus data. Sementara user bergek masih bisa menampilkan data, karena hak select tidak dicabut.

∞

BAB 8.
KONEKSI MySQL

8.1 Instalasi MySQL .NET Connector

Database yang paling banyak digunakan adalah MySQL, ada beberapa perintah yang akan menggunakan class-class: MySql Connection, MySql Command, MySql Data Reader, dan MySql Data Adapter. Untuk itu sangat diperlukan perangkat lunak RDBMS (Relational DBMS) MySQL. Database MySQL digunakan secara free dan dapat download dari salah satu website di bawah ini:

- MySQL: Jika Anda hanya ingin menginstall MySQL saja. **http://dev.mysql.com/downloads/mysql/**

- MariaDb: fork dari MySQL yang diprakarsai oleh pengagas MySQL. https://mariadb.org/

- XAMPP: salah satu dari paket aplikasi WAMP (Windows Apache MySQL PHP). Dianjurkan apabila Anda juga ingin mencoba pemrograman
  PHP. **http://www.apachefriends.org/en/xampp.html**

- Appserv: paket aplikasi WAMP lainnya. Bagus, tetapi tidak dianjurkan karena sepertinya tidak diupdate lagi. **http://www.appservnetwork.com/**

Selain MySQL, juga memerlukan MySQL .NET Connector, yang hanya tersedia di: **http://dev.mysql.com/downloads/connector/net/6.6.html#downloads.** Jika menggunakan Visual Studio 2012 atau 2013, *download* versi terbaru (6.7.4 atau di atasnya). Akan tetapi jika masih menggunakan Visual Studio 2010, *download* versi 6.5.4 atau sub versinya. Versi tersebut sesunguhnya tidak jauh berbeda, agar dapat menggunakan *Data Source Configuration Wizard*, jika ingin mencobanya sewaktu waktu. Akan tetapi untuk dapat terhubung ke MySQL tanpa wizard tesebut. Perlu diketahui juga bahwa wizard tersebut tidak tersedia pada Visual Studio 2010 Express.

1. Install dulu MySQL .NET Connector

Gambar 8.1.1 MySQL Setup

2. Klik Next untuk melanjutkan instalasi, selanjutnya akan tampak layar instalasi

Gambar 8.1.2 MySQL Setup

3. Terdapat tiga pilihan instalasi, ***Typical*** cocok untuk berbagai macam kasus. ***Custom*** berguna untuk memilih komponen apa saja dari Connector/Net yang akan di install. Dan ***Complete*** akan

menginstall seluruh komponen Connector/Net, namun akan menghabiskan banyak ruang

4. Untuk instalasi ini pilih pilihan **Typical**, klik pilihan dan mulai InstalasiConncetor/Net, akan muncul perintah berikut ini

Gambar 8.1.3 MySQL Setup Finish

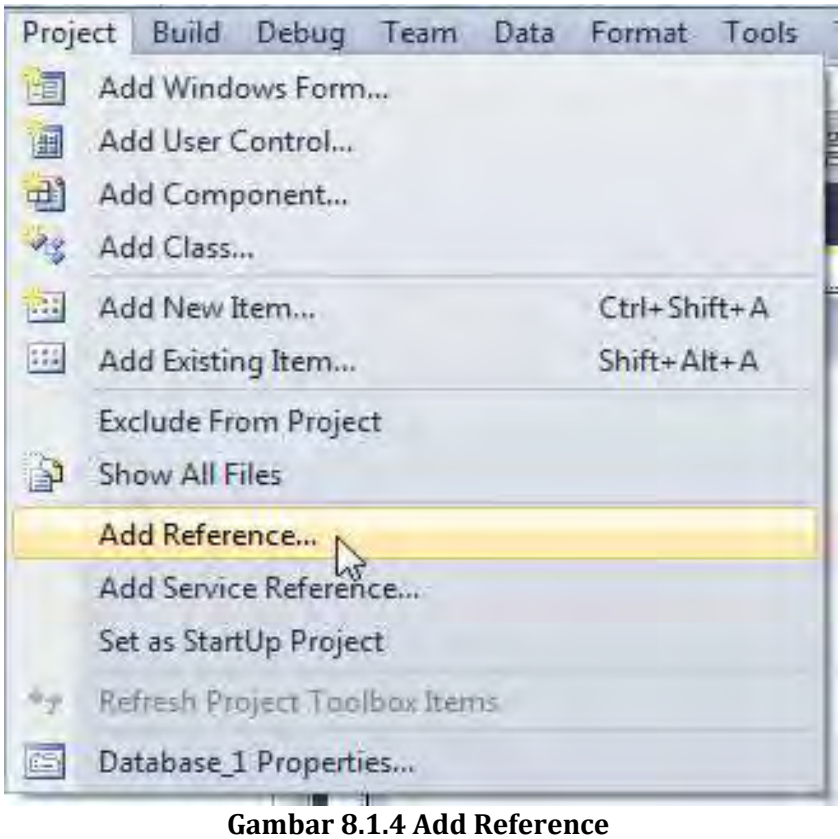
5. Klik Finish, untuk menyelesaikan instalasi Berurusan dengan database pada C#, pada dasarnya kita memerlukan paling tidak 4 dari class-class berikut:
   1) **Connection** untuk menghubungkan dengan database.
   2) **Command** untuk menjalankan query pada database.
   3) **DataReader** untuk mengenumerasi (membaca satu persatu) hasil query dari database (sebagai **DataRow**).
   4) **DataAdapter** untuk mempermudah berbagai macam operasi database, misalnya membaca data dari database ke aplikasi.
   5) **DataTable** untuk menyimpan data dari database dalam bentuk tabel.
   6) **DataSet** untuk mengelola sekumpulan DataTable.

4 class pertama dari *class-class* di atas tidak benar-benar ada. *Class-class* tersebut merupakan *class* yang spesifik terhadap database tertentu. Sebetulnya *class-class* tersebut di atas merupakan *class-class* turunan dari *class* induk masing-masing fungsi. Sebagai contoh, ketika

untuk menyebutkan *class Connection* di atas, tergantung dari jenis databasenya, yang dimaksud adalah: **OleDbConnection, SqlConnection, MySqlConnection, OdbcConnection**, yang merupakan class-class turunan dari class DbConnection. Penggunaan class-class tersebut sangat mirip. Perbedaan antara mereka, tentunya ada pada syntax SQL.

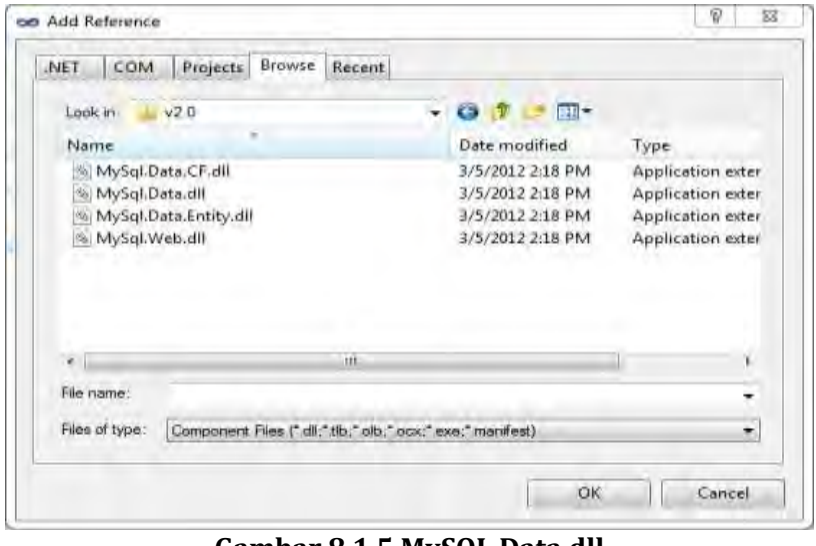
Buat project baru dengan langkah-langkah sebagai berikut:

1. Tambahkan reference dengan cara klik menu Project->Add reference

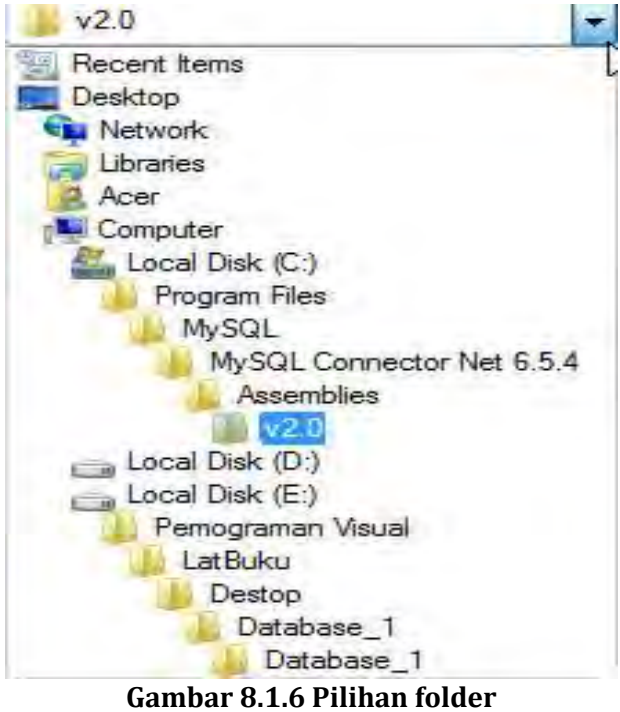


Gambar 8.1.4 Add Reference

2. Akan muncul menu Add Reference, lalu cari arahkan ke file **mysql.data.dll** kelokasi folder dimana Mysql connector diinstall , yaitu di C:\Program Files\MySQL\MySQL Connector Net 6.3.5\Assemblies



Gambar 8.1.5 MySQL.Data.dll



Gambar 8.1.6 Pilihan folder

3. Jika sudah berhasil menambahkan referensi Mysql, klik tombol OK

8.2 Aristektur MySQL Connector.NET

Pada saat menginstal Visual Studio .NET untuk pertama kalinya, terlebih dahulu di diinstal kebutuhan-kebutuhan pendukung yang diperlukan. Salah satu namespace yang ditemukan di dalam class-class .NET Framework adalah System.Data. *Namespace* ini berisikan class-class untuk berinteraksi dengan sumber data, dimana *namespace system* bertindak sebagai *root* untuk seluruh *class .NET Framework*.

Arsitektur MySQL Connector.NET menggambarkan class-class yang membentuk data provider. Meskipun Connector.NET bekerja dalam lingkungan .NET Framework, tetapi secara eksplisit tidak berada di bawah namespace System. Data provider ini memiliki *namespace root* sendiri, yaitu MySql, dan menyediakan namespace *MySql.Data* yang berfungsi sama dengan System.Data.

a. Namespace MySql.Data.MySqlClient

Dalam rangka mendukung operasi-operasi database, data provider Connector.NET menyediakan dua buah namespace utama, yaitu namespace *MySql.Data.MySqlClient* dan *MySql.Data.Types*. Di dalam pengolahan database, sering digunakan namespace *MySql.Data.MySqlClient*. Adapun namespace yang kedua lebih berkaitan dengan pengolahan tipe data antara Visual Basic .NET dan MySQL. Namespace *MySQL.Data.MySqlClient* terdapat class-class dasar yang diinstantiasi menjadi objek-objek serta digunakan untuk mengolah database melalui aplikasi Pemograman C#. seperti diilustrasikan OLE DB, namespace ini kedudukannya setingkat dengan namespace *System.Data.SqlClient* untuk SQL Server. Connector .NET berusaha untuk mengimplementasikan data provider ADO.NET.

b. Class MySqlConnection

Class MySqlConnection merepresentasikan koneksi ke database MySQL Server. Objek yang dihasilkan dari instantiasi dari class ini menggambarkan *session* ke sumber data, yaitu database MySQL. Beberapa properti penting yang sering digunakan untuk mendukung fungsionalitas *MySqlConnection* adalah sebagai berikut.

1) **ConnectionString** adalah berfungsi untuk mendapatkan atau menetapkan string koneksi yang digunakan untuk menghubungkan aplikasi ke database MySQL Server.
2) **ConnectionTimeout** berguna untuk mendapatkan nama database yang ada saat itu atau database yang digunakan setelah koneksi terbuka.

3) **Database** berfungsi ntuk mendapatkan nama database yang ada saat itu atau database yang digunakan setelah koneksi terbuka.
4) **DataSource** berfungsi untuk mendapatkan nama MySQL Server yang akan dihubungkan.
5) **Class MySqlConnection** bertugas untuk menyediakan *method-method* yang berkaitan dengan operasi pada database. Method-method ini digunakan untuk membuka dan menutup koneksi, memilih database, serta mengelola objek koneksi yang telah dibuat. Diantaranya:
   - **BeginTransaction** berfungsi untuk memulai transaksi, yang berarti bahwa transaksi diaktifkan.
   - **ChangeDatabase** berfungsi untuk mengubah database yang aktif saat itu, dan dilanjutkan dengan memilih database lainnya.
   - **Close** berfungsi untuk menutup koneksi database, dan meliputi semua database aktif yang terbuka.
   - **Dispose** berfungsi untuk melepaskan sumber daya yang digunakan oleh objek *MySqlConnection*.
   - **Open** berfungsi untuk membuka koneksi ke database dengan pengaturan properti yang telah ditentukan dalam ConnectionString.

Dalam implementasi koneksi ke database, **class MySqlConnection** menyediakan dua macam konstruktor. Pertama, konstruktor tanpa parameter, sedangkan yang kedua menggunakan parameter string berupa koneksi.

1) Class MySqlCommand

Class MySqlCommand merepresentasikan pernyataan SQL yang akan dieksekusi. Class ini cukup sederhana pernggunaannya, tetapi mampu melakukan berbagai operasi database, seperti penambahan, pengubahan serta penghapusan record. Ketika ingin melakukan operasi database secara praktis, tepat sekali jika menggunakan kemampuan dari *MySqlCommand*. Dalam medukung fungsionalitas class ini, ada beberapa properti yang dapat digunakan adalah sebagai berikut:

1. **CommandText** berfungsi untuk mendapatkan atau menentukan pernyataan SQL yang akan dieksekusi.
2. **CommandType** berfungsi untuk mendapatkan atau menetapkan suatu nilai yang menyatakan bagaimana properti CommandText diterjemahkan.

3. **Connection** berfungsi untuk mendapatkan atau menetapkan objek MySqlConnection yang akan digunakan oleh MySqlCommand.

2) **Class MySqlCommand** menyediakan method-method yang sebagian merupakan milik class ini sendiri dan ada juga yang diwarisi dari class tertentu. Secara garis besar, fungsionalitas dari MySqlCommand digambarkan oleh tiga method berikut ini.
   1. **ExecuteNonQuery** berguna untuk mengeksekusi pernyataan SQL seperti **INSERT, UPDATE, dan DELETE**. Ada pun hasil yang dikembalikan adalah jumlah dari baris data.
   2. **ExecuteReader** berguna untuk mengirimkan CommandText ke objek Connection dan membangun *MySqlReader*.
   3. **ExecuteScalar** berguna Mengeksekusi query, dan mengembalikan kolom pertama dari baris pertama di dalam hasil yang dikembalikan oleh query. Implementasi method ini identik dengan komputasi data, misalnya untuk mencari jumalah record di dalam tabel.

   Pada saat ingin melakukan manipulasi data didalam tabel, MySqlCommand merupakan pilihan tepat yang sudah menyediakannya. Meskipun method-method MySqlCommand mampu digunakan secara independen, akan tetapi sering juga digabungkan dari method classs lain.

3) **Class MySqlDataAdapter** merepresentasikan kumpulan perintah data dan koneksi database yang digunakan untuk mengisi suatu dataset dan memodifikasi database MySQL. Instantiasi *class MySqlDataAdapter* memungkinkan untuk pengolahan data dengan memanfaatkan komponen-komponen data milik ADO.NET. Pada prinsipnya, *MySqlDataAdapter* bertindak sebagai jembatan antara objek *DataSet* dan MySQL untuk mendapatkan kembali serta menyimpan data. Properti-properti yang tersedia pada *MySqlDataAdapter* antara lain:
   1. **DeleteCommand** berfungsi untuk mendapatkan atau menetapkan pernyataan SQL yang digunakan dalam menghapus record dari dataset.
   2. **InsertCommand** befungsi untuk mendapatkan atau menetapkan peryantaan SQL yang digunakan dalam melakukan penambahan record baru ke dalam tabel.
   3. **SelectCommand** berfungsi untuk mendapatkan atau menetapkan pernyataan SQL pengubahan data. Umumnya objek

MySqlDataAdapter akan digunakan bersama-sama dengan objek MySqlConnection dan MySqlCommand.

4) Class MySQLCommandBuilder

Secara otomatis objek *MySqlCommandBuilder* mampu menghasilkan perintah *single-table* untuk menyesuaikan perubahan yang dibuat ke suatu DataSet. Contoh sederhana yang menjelaskan fungsionalitas objek ini adalah ketika tampilan data yang dibuat melalui objek DataSet juga memungkinkan untuk dimodifikasi, dan perubahan diterapkan secara langsung. Properti-properti yang tersedia untuk mendukung fungsionalitas objek *MySqlCommandBuilder* ini antara lain:

a. **DataAdapter p**roperti public ini digunakan untuk mendapatkan atau menetapkan objek MySqlDataAdapter pada pernyataan SQL yang dihasilkan secara otomatis.
b. **QuotePrefix** berfungsi untuk mendapatkan atau menetapkan karakter awal yang akan digunakan ketika menentukan objek database spesifik.

Seperti halnya class-class lainnya, *MySqlCommandBuilder* juga menyediakan method-method penting seperti berikut:

a. **GetDeleteCommand** berfungsi untuk medapatkan objek MySqlCommand yang dihasilkan secara otomatis dan diperlukan dalam penghapusan pada database.
b. **GetInsertCommand** berfungsi untuk mendapatkan objek MySqlCommand yang dihasilkan secar otomatis dan diperlukan dalam penambahan data pada database.
c. **GetUpdateCommand** menndapatkan objek MySqlCommand yang diperlukan untuk melakukan pengubahan pada database.

Objek MySqlCommandBuilder mampu meregistrasi dirinya sendiri sebagai listener unutk event *OnRowUpdating*. Secara normal hanya diperkenankan untuk menghubungkan satu objek MySqlDataAdapter atau MySqlCommandBuilder pada satu waktu yang bersamaan.

5) **Class MySqlDataReader** menyediakan kemampuan untuk melakukan pembacaan dari database MySQL. Pembacaan tidak hanya terbatas pada data di dalam tabel, akan tetapi juga meliputi skema tabel maupun database. Properti-properti public yang disediakan oleh MySqlDataReader adalah sebagai berikut:

a. **Depth** berfungsi untuk mendapatkan nilai yang menyatakan kedalaman row barsarang pada row saat itu.
b. **FieldCount** berfungsi untuk mendapatkan jumlah kolom dari current row.
c. **HasRows** berfungsi untuk mendapatkan nilai yang menyatakan apakah MySqlDataReader berisi satu atau lebih baris.
d. **IsClosed** berfungsi untuk mendapatkan nilai yang menyatakan apakah data reader tertutup.
e. **Item** berfungsi untuk mendapatkan nilai dari suatu kolom dalam format aslinya.
f. **RecordAffected** berfungsi untuk mendapatkan banyaknya baris yang di ubah, dimasukkan, atau dihapus oleh eksekusi pernyataan SQL.
g. **GetValue** untuk mengoptimalkan performansi, MySqlDataReader menghindari pembuatan objek atau pembuatan salinan data yang tidak diperlukan. Sebagai hasilnya, beberapa pemanggilan ke method semacam GetValue akan mengembalikan suatu acuan ke objek yang sama.

8.3 Objek-Objek ADO.NET

Objek DataSet adalah komponen utama dari arsitektur disconnected ADO.NET, dan merupakan objek di dalam memori yang dapat mengisi tabel, *view*, dan *relationship*. Pada prinsipnya, objek *DataSet* adalah suatu cache di dalam memori dari sumber data yang diambil, Secara eksplisit, *DataSet* didesain untuk pengaksesan data independen dari berbagai sumber data.

Objek *DataSet* bisa diciptakan dengan cara melakukan instantiasi class DataSet. Selain itu, juga diperbolehkan membuat DataSet dari DataSet yang sudah ada. Cara seperti ini dikenal dengan clone, yakni untuk mendapatkan copy dari DataSet. Normalnya, ketika meng-copy DataSet, hanya mendapatkan skema atau struktur relasional. Selain DataSet, ada beberapa objek ADO.NET yang akan sangat diperlukan dalam pengolahan data, baik untuk medukung objek *DataSet* ataupun digunakan secara independen, antara lain:

1) DataTable

Objek *DataTabel* merepresentasikan sebuah tabel di dalam memori dari hasil query data. Objek *DataTable* berisi satu atau beberapa data objek *DataColumn* dan *DataRow*. Dapat berisi satu atau beberapa objek *Constraint* digunakan untuk mengelola

integritas data dalam tabel. Data tersimpan dalam *DataTable* dimungkinkan untuk modifikasi, seperti menambah, mengubah, atau menghapus.

2) DataColumn

Objek *DataColumn* merupakan dasar yang membangun blok berguna untuk menciptakan skema *DataTable*. Prinsipnya, objek merepresentasikan skema kolom dalam suatu DataTable. Data pada setiap kolom, termasuk nama dan tipe, serta objek *DataRow* didefiniskan dalam objek *DataColumn*.

3) DataRow

Pada dasarnya, objek *DataRow* dan *DataColumn* merupakan komponen utama dari *DataTable*. Objek *DataRow* memungkinkan dilakukannya operasi penambahan data, pengubahan data, dan penghapusan nilai di dalam DataTable. Operasi-operasi modifikasi dapat dilakukan dengan mudah melalui *method-method* yang tersedia., misalnya *BeginEdit, EndEdit, Delete* dan sebagainya.

4) DataView

Fungsi utama *DataView* adalah untuk memungkinkan *binding data* pada *Windows Form* dan *Web Form*. *Binding data* sangat membantu, ketika melakukan pengambilan data. *DataView* juga dapat digunakan untuk menampilkan *subset* data dari *DataTable*. Guna mendukung pengolahan data, objek *DataView* memiliki kemampuan dalam melakukan penyaringan, pengurutan, pencarian, pengubahan, dan navigasi data. Pekerjaan ini dapat dilakukan dengan mudah melalui properti-properti dan method-method yang tersedia, misalnya *RowFilter, Sort, Find, FindRows*, dan masih banyak lagi.

Latihan:

Gambar 8.4.1

Ubahlah isian properti dari setiap objek/control seperti dibawah ini :

Nama Objek/Control	Properties	
	Properti	Nilai yang diisikan
Form	Name	Form1
	Text	Koneksi Database
Button	Name	btnKoneski
	Text	&Koneksi

Kode Program:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using MySql.Data;
using MySql.Data.MySqlClient;

namespace Database_1
{
 public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();
 }

 private void Form1_Load(object sender, EventArgs e)
 {

 }

 private void btnKoneksi_Click(object sender, EventArgs e)
 {
 try
 {
 string conn =
"Server=localhost;UID=root;Database=akademik";
```

```
MySqlConnection koneksi = new MySqlConnection(conn);
koneksi.Open();
MessageBox.Show("Koneksi Berhasil");
koneksi.Close();
 }
 catch (Exception ex)
 {
MessageBox.Show(ex.Message);
 }
 }
 }
}
```

8.4 Membuat Aplikasi Database

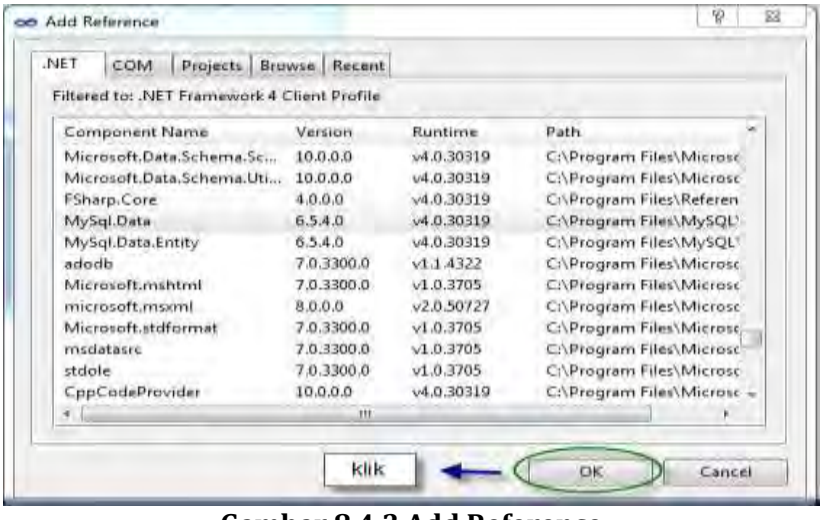
Jika baru pertama menggunakan MySQL dan pemograman C#, dan cara menghubungkan antara pemograman C# dengan MySQL untuk pertama kalinya, maka perlu mengunduh dan menginstal perangkat lunak berikut berikut ini:

1. Setelah mengunduh dan menginstal server Xampp, kemudian buka localhost di browser yang digunakan, dengan cara mengetik localhost atau 127.0.0.1 di bilah address. Buat database **Akademik** dengan nama tabel **Fakultas** .

Field	Type	Null	Key	Default	Extra
kd_fak	varchar(2)	NO	PRI	NULL	
nm_fak	varchar(30)	NO		NULL	

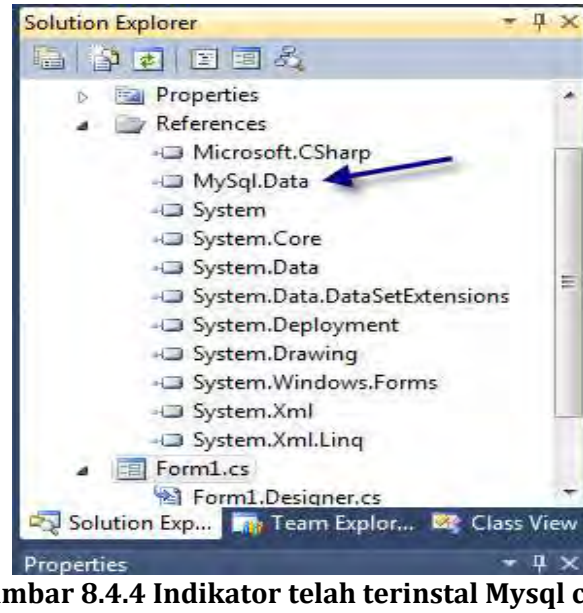
Gambar 8.4.2 Membuat Tabel dan Field

2. Buka Visual Studio 2010 pilih pemogrman C#, klik new project
3. Setelah muncul Form, Klik menu Project -> pilih Add Reference cari MySQL.Data.dll



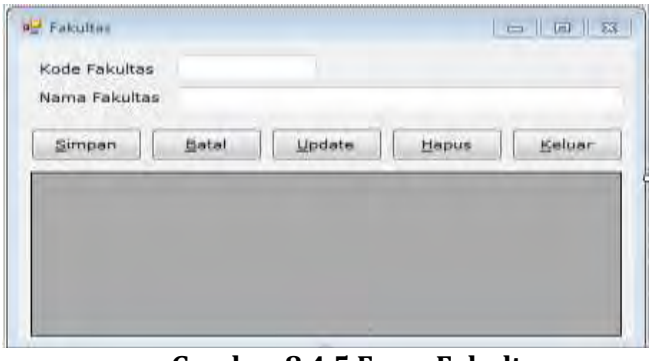
Gambar 8.4.3 Add Reference

4. Jika sudah berhasil menambahkan referensi Mysql di aplikasi akan tampak indikator seperti gambar dibawah ini :



Gambar 8.4.4 Indikator telah terinstal Mysql connector

5. Tambahkan script import seperti **Imports MySql.Data.MySqlClient**
6. Selanjutkan buat buat Form



Gambar 8.4.5 Form Fakultas

Ubahlah isian properti dari setiap objek/control seperti dibawah ini :

Nama Objek/Control	Properties	
	Properti	Nilai yang diisikan
Form	Name	Form1
	Text	Fakultas
Label	Text	Kode Fakultas
TextBox	Name	txtKode
	TabIndex	0
Label	Text	Nama Fakultas
TextBox	Name	txtNama
	TabIndex	1
Button	Name	btnSimpan
	Text	&Simpan
	TabIndex	2
Button	Name	btnBatal
	Text	&Batal
	TabIndex	3
Button	Name	btnUpdate
	Text	&Update
	TabIndex	4
Button	Name	btnHapus
	Text	&Hapus
	TabIndex	5
Button	Name	btnKeluar
	Text	&Keluar
	TabIndex	6
DataGridView	Name	DGTampil
	TabIndex	7

Kode Program:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using MySql.Data;
using MySql.Data.MySqlClient;

namespace Database_2
{
 public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();
 }
 //--Menginisialisasi koneksi MySql
 MySqlConnection konek = new MySqlConnection();
 //--Menginisialisasi semua class
 MySqlCommand cmd = new MySqlCommand();
 MySqlDataAdapter da = new MySqlDataAdapter();
 DataTable dt = new DataTable();

 //--Deklarasi variabel
 string sql;
 int res;

 private void Form1_Load(object sender, EventArgs e)
 {
 konek.ConnectionString =
"server=localhost;uid=root;database=akademik";
 }

 private void btnSimpan_Click(object sender, EventArgs e)
 {
 //--Membuka koneksi database
 konek.Open();
```

```
  //--Membuat query untuk melakukan penyimpanan database
  sql="insert into fakultas(kd_fak,nm_fak) values ('"+txtKode.Text
+ "','" + txtNama.Text + "')";
  //--Menyimpan data untuk dapat di eksekusi
  cmd.Connection = konek;
  cmd.CommandText = sql;
  res = cmd.ExecuteNonQuery();
  Bersih();
  konek.Close();
  TampilData();
  }

  private void Bersih()
  {
  txtKode.Clear();
  txtNama.Clear();
  txtKode.Focus();
  }

  private void Clear()
  {
  foreach (Control ctl in this.Controls) //Untuk setiap Control di
Aplikasi ini
  {
 if (ctl is TextBox) //Jika control adalah textbox maka
 {
 ctl.Text = ""; //Kosongkan Isinya
 }
  }
  }

  private void btnBatal_Click(object sender, EventArgs e)
  {
  Bersih();
  TampilData();
  }

  private void TampilData()
  {
  //--koneksi dan membuka Database
  konek.Open();
```

```
 //--Membuat query untuk memanggil data dari database
 sql = "select kd_fak as Kode_Fakultas,nm_fak as Nama_Fakultas
from fakultas where 1=1";
 //--Menginilisasai SqlCommand yang baru
 cmd = new MySqlCommand();
 //--Menyimpan data untuk dapat dieksekusi
 cmd.Connection = konek;
 cmd.CommandText = sql;
 //--Menginilisasi MySqlDataAdapter yang baru
 da = new MySqlDataAdapter();
 //--Mengambil data query dalam database
 da.SelectCommand = cmd;
 //--Menginilisasi DataTable yang baru
 dt = new DataTable();
 //--Menyegarkan baris dalam sumber data
 da.Fill(dt);
 //--Sumber data ditampilkan dalam DataGridView
 DGTampil.DataSource = dt;
 konek.Close();
 }

 private void CariData()
 {
 try
 {
bool temp = false;
//--Membuat koneksi database
konek.Open();
//--Membuat query untuk memanggil data dari database
sql = "select * from fakultas where kd_fak='" + txtKode.Text + "'";
MySqlCommand cmd = new MySqlCommand(sql, konek);
MySqlDataReader dr = cmd.ExecuteReader();
while (dr.Read())
{
txtNama.Text = dr.GetString(1);
temp = true;
}
if (temp == false)
{
MessageBox.Show("Data Tidak Ada");
txtNama.Text = string.Empty;
```

```
txtNama.Focus();
}
konek.Close();
TampilData();
 }
 catch (Exception ex)
 {
MessageBox.Show(ex.Message);
konek.Close();
 }
 }
 private void txtKode_KeyPress(object sender,
KeyPressEventArgs e)
 {
 if (e.KeyChar == 13)
 {
CariData();
 }
 }

 private void btnUpdate_Click(object sender, EventArgs e)
 {
 try
 {
konek.Open();
sql = "update fakultas set nm_fak='" + txtNama.Text + "' where
kd_fak='" + txtKode.Text + "'";
MySqlCommand cmd = new MySqlCommand(sql, konek);
cmd.ExecuteNonQuery();
konek.Close();
 }
 catch (MySqlException ex)
 {
MessageBox.Show(ex.ToString());
konek.Close();
 }
 Bersih();
 TampilData();
 }

 private void btnHapus_Click(object sender, EventArgs e)
```

```
 {
 try
 {
konek.Open();
sql = "delete from fakultas where kd_fak='" + txtKode.Text + "'";
MySqlCommand cmd = new MySqlCommand(sql, konek);
cmd.ExecuteNonQuery();
konek.Close();
 }
 catch (MySqlException ex)
 {
MessageBox.Show(ex.ToString());
 }
 Bersih();
 TampilData();
 }
 }
}
```

Latihan Database 2:

Gambar 8.4.6 Form Mata Kuliah

Ubahlah isian properti dari setiap objek/control seperti dibawah ini :

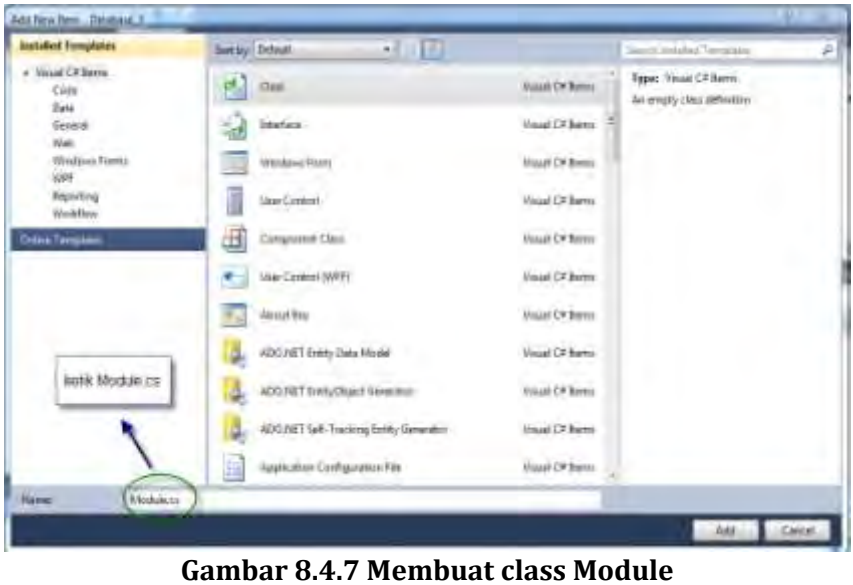
Nama Objek/Control	Properties	
	Properti	Nilai yang diisikan
Form	Name	Form1
	Text	Mata Kuliah
Label	Text	Kode
TextBox	Name	txtKode
	TabIndex	0
Label	Text	Nama Mata Kuliah
TextBox	Name	txtNama
	TabIndex	1
Button	Name	btnSimpan
	Text	&Simpan
	TabIndex	2
Button	Name	btnBatal
	Text	&Batal
	TabIndex	3
Button	Name	btnKeluar
	Text	&Keluar
	TabIndex	3
ListView	Name	ListView1
	TabIndex	4

Setelah Project dan Form telah dibuat, ikuti langkah berikut ini:

1. Klik menu Project pilih Add Class

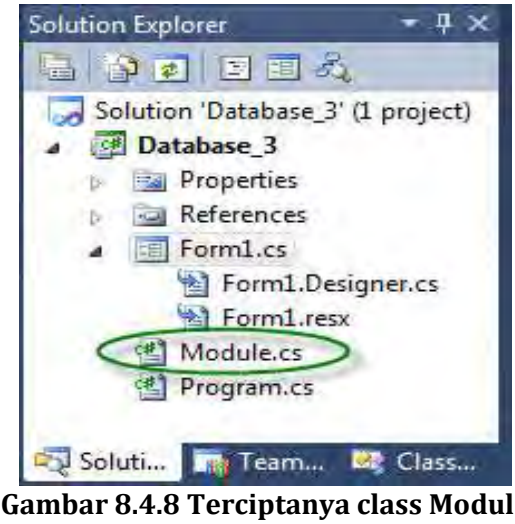
Gambar 8.4.6 *Add Class*

2. Sesudah di klik akan muncul menu pilihan sebagai berikut:



Gambar 8.4.7 Membuat class Module

3. Akan Muncul file Module.cs di Kotak Solution Explorer seperti gambar ini:



Gambar 8.4.8 Terciptanya class Module

4. Ketik Program di Module.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
```

```
using System.Text;
using MySql.Data;
using MySql.Data.MySqlClient;
using System.Threading.Tasks;
using System.Data;
//using System.ComponentModel;
using System.Windows.Forms;

namespace Database_3
{
 class Module
 {
 string konek;
 //MySqlCommand cmd = new MySqlCommand();
 MySqlDataAdapter da = new MySqlDataAdapter();
 // dt = new DataTable();

 public Module()
 {
 konek = "server=localhost;Uid=root;database=akademik";
 }

 //-- Jenis Query
 //--Query dengan pilihan data ->select [field] from [nama
tabel] where [kondisi]

 public void GetDataTable(string sql)
 {
 DataTable result = new DataTable();
 MySqlConnection conn = new MySqlConnection(konek);
 conn.Open();
 MySqlCommand cmd = new MySqlCommand(sql, conn);
 MySqlDataReader dr = cmd.ExecuteReader();
 result.Load(dr);
 conn.Close();
 //return result;
 }
 //--Query untuk Memanipulasi Data
 //-- Menambah Data ->insert into [nama table] (field) values
(data)
```

```
  //--Memodifikasi Data ->update [nama table] set [field]=[data]
where [kondisi]
  //-Menghapus Data ->delete from [nama table] where
[kondisi]

  public void ExecuteQuery(string sql)
  {
  MySqlConnection conn = new MySqlConnection(konek);
  conn.Open();
  MySqlCommand cmd = new MySqlCommand(sql, conn);
  cmd.ExecuteNonQuery();
  conn.Close();
  }
  public void TampiList(ListView lv, string sql)
  {
  MySqlConnection conn = new MySqlConnection(konek);
  conn.Open();
  MySqlCommand cmd = new MySqlCommand(sql, conn);
  MySqlDataReader reader = cmd.ExecuteReader();
  lv.Items.Clear();
  while (reader.Read())
  {
 ListViewItem item = new
ListViewItem(reader.GetString(0).ToString());
 item.SubItems.Add(reader.GetString(1));
 item.SubItems.Add(reader.GetString(2));
 lv.Items.Add(item);
  }
  reader.Close();
  cmd.Dispose();
  conn.Close();
  }
  public void Bersih(Form form)
  {
  foreach (Control control in form.Controls)
  {
 if (control.GetType() == typeof(TextBox))
 {
 control.Text="";
 }
  }
```

```
        }

        }
      }
```

5. Kembali ke Form Matakuliah, Klik tombol Simpan, ketik kode program:

```
private void btnSimpan_Click(object sender, EventArgs e)
{
string sql;
      sql="insert into matakuliah(kd_mtk,nm_mtk,sks) values ('" +
     txtKode.Text + "','" + txtNama.Text +"'," + txtSKS.Text + ")";
Module db = new Module();
db.ExecuteQuery(sql);
Clear();
}
```

6. Klik tombol Batal, ketik kode program berikut ini:

```
private void btnBatal_Click(object sender, EventArgs e)
 {
 Clear();
 }

 private void Clear()
 {
 Module obj = new Module();
 obj.Bersih(this);
 txtKode.Focus();
 Module db = new Module();
 db.TampiList(listView1, "select * from matakuliah");
 }
```

7. Klik Form, muncul procedure Form1_Load, ketik kode program berikut ini:

```
private void Form1_Load(object sender, EventArgs e)
  {
  listView1.GridLines = true;
  listView1.View = View.Details;
  // membuat kolom baru
```

```
    listView1.Columns.Add("Kode Mata Kuliah", 150);
    listView1.Columns.Add("Nama Mata Kuliah", 250);
    listView1.Columns.Add("SKS", 80);
    }
```

8. Kode Program yang lengkapnya sebagai berikut:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using MySql.Data;
using MySql.Data.MySqlClient;

namespace Database_3
{
 public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();
 }

 private void btnSimpan_Click(object sender, EventArgs e)
 {
 string sql;
 sql="insert into matakuliah(kd_mtk,nm_mtk,sks) values ('" +
txtKode.Text + "','" + txtNama.Text +"'," + txtSKS.Text + ")";
 Module db = new Module();
 db.ExecuteQuery(sql);
 Clear();

 }

 private void Form1_Load(object sender, EventArgs e)
 {
 listView1.GridLines = true;
```

```
listView1.View = View.Details;
// membuat kolom baru
listView1.Columns.Add("Kode Mata Kuliah", 150);
listView1.Columns.Add("Nama Mata Kuliah", 250);
listView1.Columns.Add("SKS", 80);
}

private void btnBatal_Click(object sender, EventArgs e)
{
Clear();
}

private void Clear()
{
Module obj = new Module();
obj.Bersih(this);
txtKode.Focus();
Module db = new Module();
db.TampiList(listView1, "select * from matakuliah");
}

private void btnKeluar_Click(object sender, EventArgs e)
{
this.Close();
}

private void Cari()
{
string sql = "select * from matakuliah where kd_mtk='" +
txtKode.Text + "'";
Module db = new Module();
}

private void txtKode_KeyPress(object sender,
KeyPressEventArgs e)
{
if (e.KeyChar == 13)
{
SendKeys.Send("{tab}");
}
}
```

```
  private void txtNama_KeyPress(object sender,
KeyPressEventArgs e)
  {
  if (e.KeyChar == 13)
  {
 SendKeys.Send("{tab}");
  }
  }

  private void txtSKS_KeyPress(object sender,
KeyPressEventArgs e)
  {
  if (e.KeyChar == 13)
  {
 SendKeys.Send("{tab}");
  }
  }

  }
}
```

Tampilan Formnya

Gambar 8.4.9 Tampilan Form Mata Kuliah

∞

BAB 9
LAPORAN

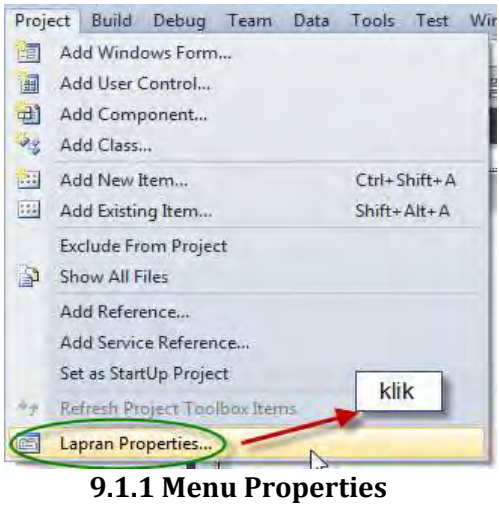
9.1 Koneksi ODBC

Kemudian aplikasi yang telah dibuat, untuk mencetak sebuah laporan dan pengguna dapat mencetak ke printer, sebuah keharusan untuk dibuat oleh seorang pemogram dalam mengembangkan aplikasi di destop. Crystal Report salah satu *software* tambahan di Visual Studio 2010 yang berguna untuk membuat laporan.

Komponen yang digunakan secara default pada Microsoft Visual Studio 2010 adalah komponen Crystal Report. Keuntungan menggunakan Crystal Report adalah kemudahan dalam melakukan perancangan laporan. Untuk mendapatkan laporan yang menarik dan sesuai kebutuhan pemakai, perlu manambahkan report baru menggunakan crystal report. Crystal Report dapat di download secara free di website SAP Crystal Reports. Dengan alamat website:

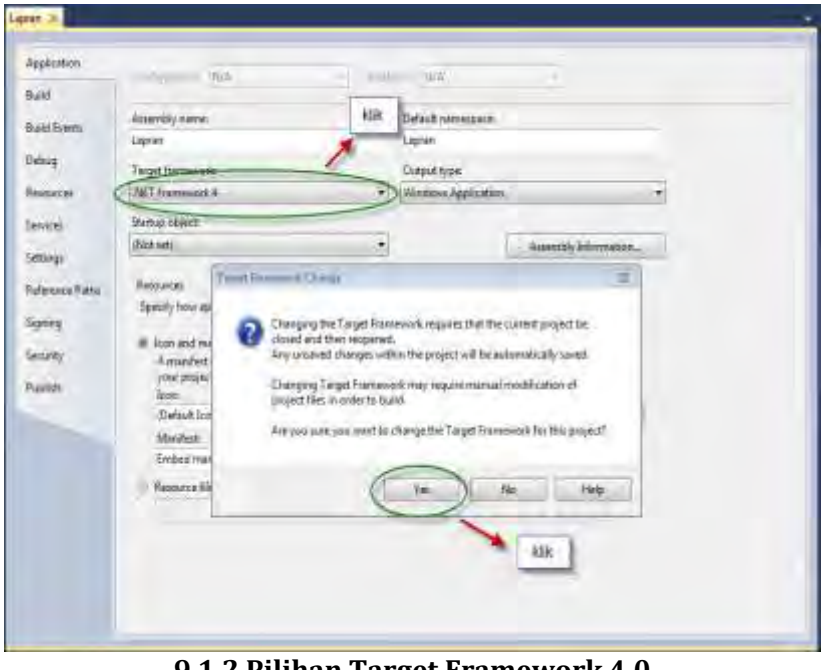
http://downloads.businessobjects.com/akdlm/cr4vs2010/CRforVS_redist_install_32bit_13_0_4.zip. Sebelum menggunakan Crystal Report, MySQL connector yang digunakan adalah **mysql-connector-odbc-5.1.13-win32**, yang dapat didownload di website resmi MySQL. Setelah semua perangkat telah diinstalasi perlu untuk merubah setting di Microsoft Visual Studi C# 2010, kemudian ikuti langkah-langkah berikut ini:

1. Buka project baru, di Form klik menu Project klik prpeperties



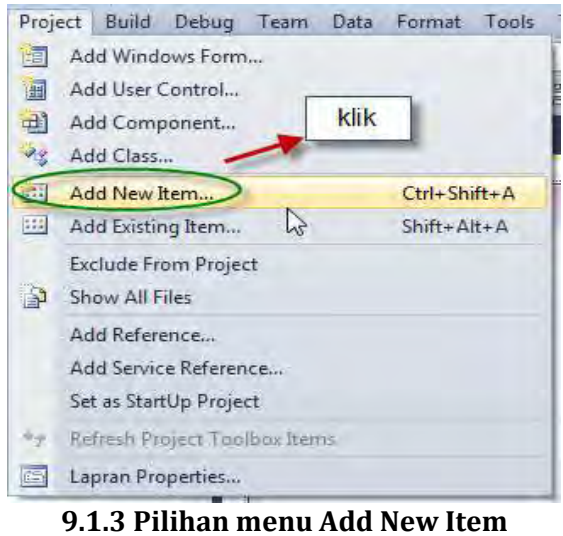
9.1.1 Menu Properties

2. Setelah diklik, muncul pilih di menu target Framework dengan piliha **.NET Framework 4,** akan muncul kotak konfirmasi dan pilih Yes.



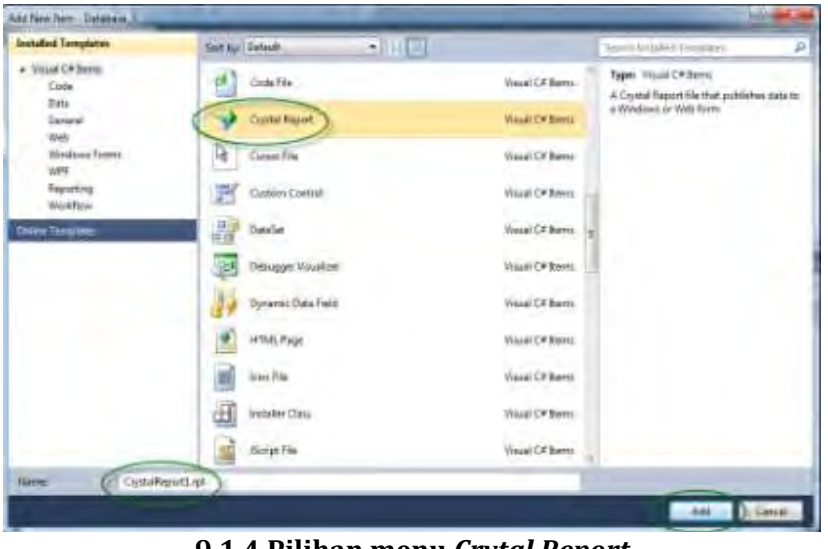
9.1.2 Pilihan Target Framework 4.0

3. Setelah selesai klik menu Project lagi, pilih menu Add New Item



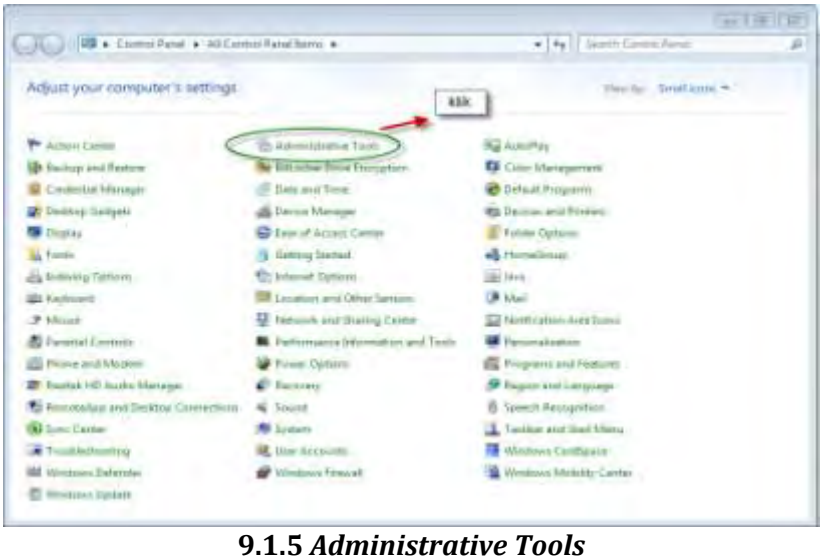
9.1.3 Pilihan menu Add New Item

4. Setelah di klik menu Add New Item, pilih Crystal Report dan kemudian klik Add seperti gambar berikut:



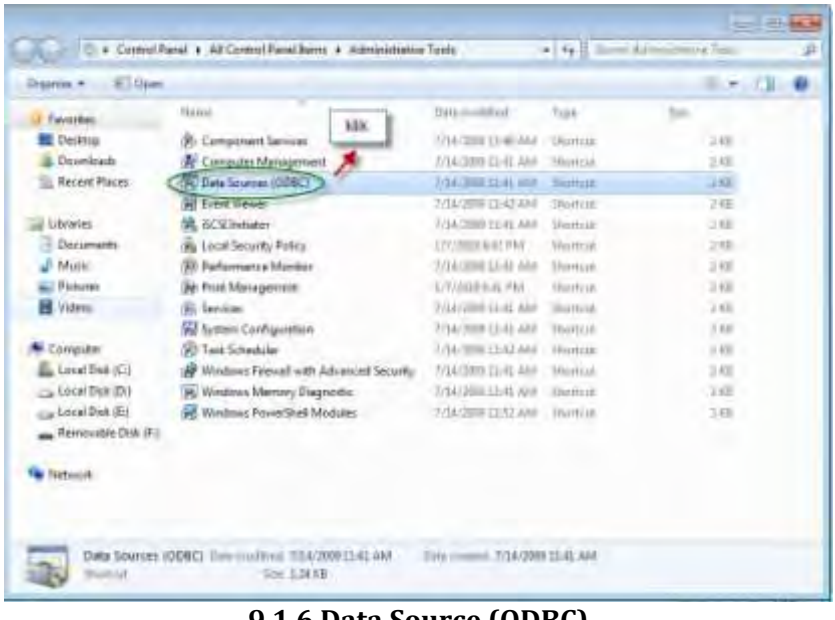
9.1.4 Pilihan menu *Crytal Report*

5. Sebelumnya setting dulu DSN (*Data Source Name*) pada computer untuk melakukan koneksi dengan database MySQL dengan cara klik **Control Panel->ODBC->**pilih menu **Administrative Tools** lalu setting seperti ini:



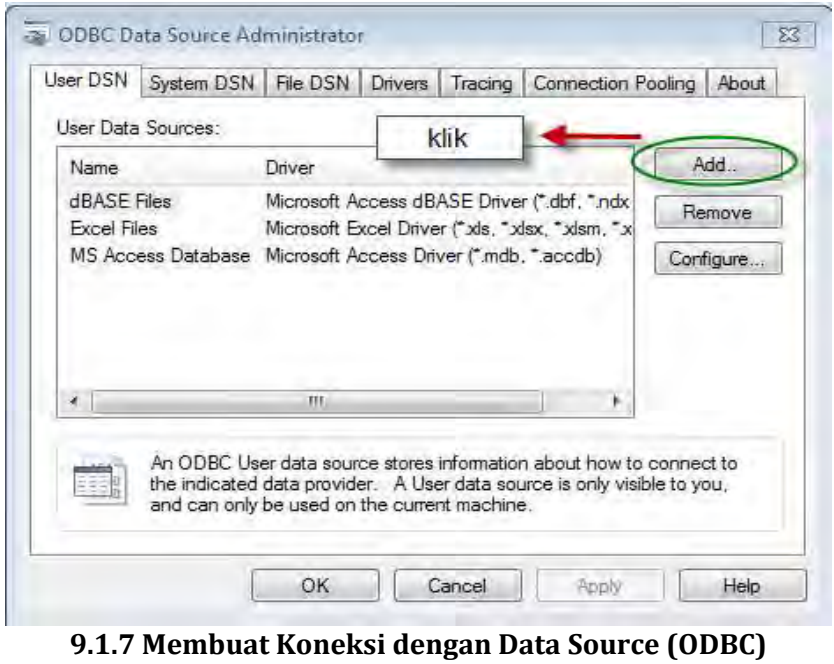
9.1.5 *Administrative Tools*

6. Klik menu Administrative Tools->Data Source (ODBC)



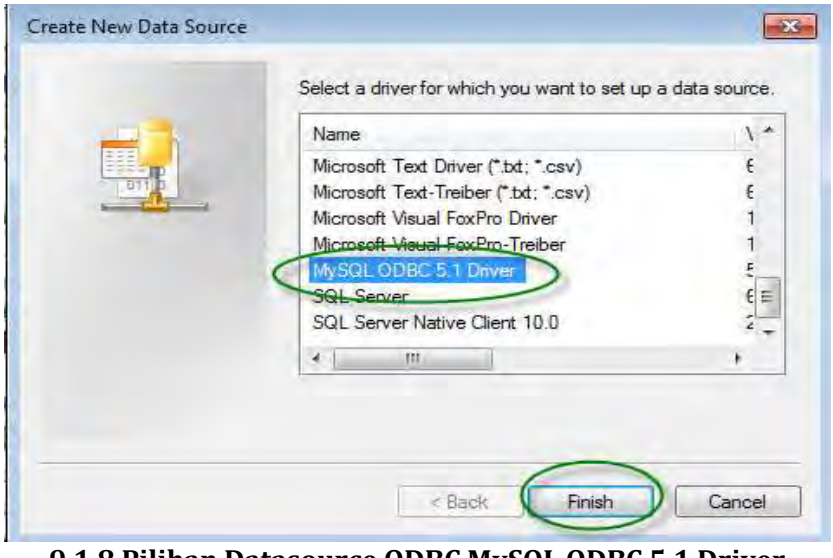
9.1.6 Data Source (ODBC)

7. Setelah di klik pilihan Data Source (ODBC), klik Add:



9.1.7 Membuat Koneksi dengan Data Source (ODBC)

8. Pilih Database Pada Data Source MySQL ODBC 5.1 Driver, dan klik Finish

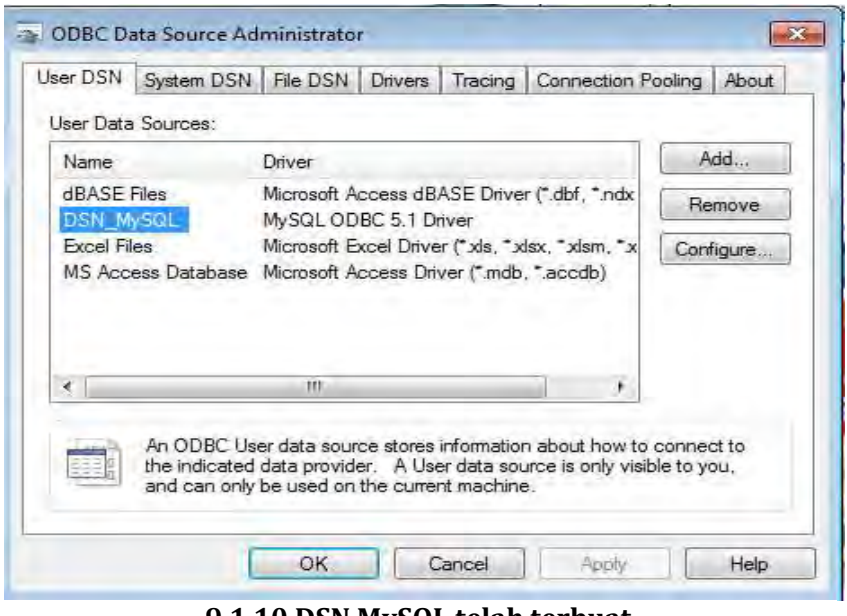


9.1.8 Pilihan Datasource ODBC MySQL ODBC 5.1 Driver

9. Pada window configuration isikan Server, Port, User, Password, dan Database sesuai dengan yang anda miliki, lihat gambar.

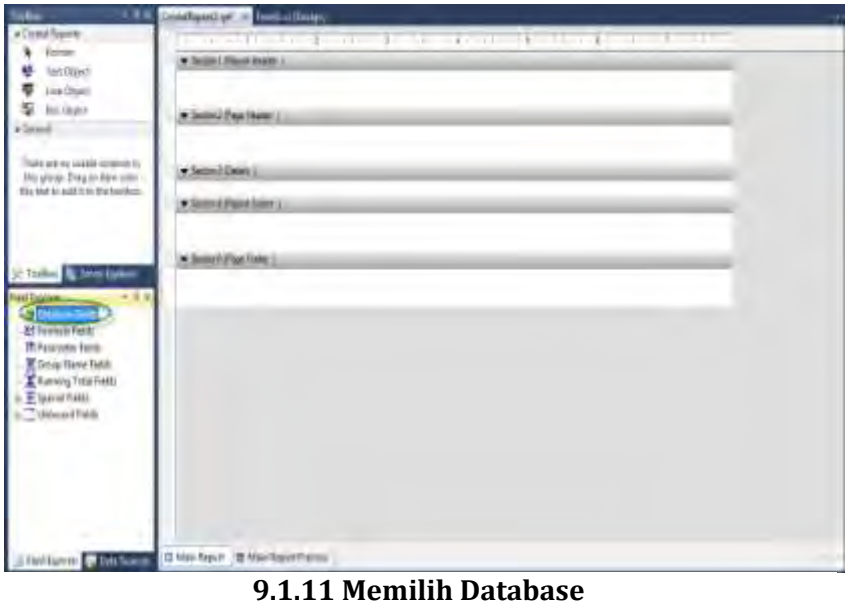
9.1.9 DSN MySQL Konfigurasi

10. Isi Data Source Name DSN_MySQL dan klik tombol OK.



9.1.10 DSN MySQL telah terbuat

11. Kembali lagi ke template Crystal Report, di menu Field Explorer, klik kanan Database Fields.

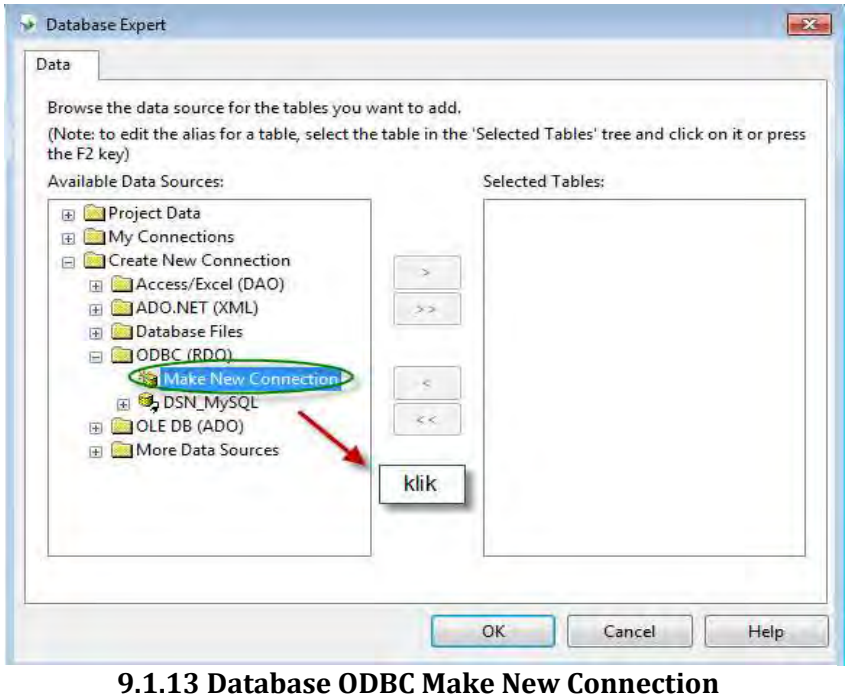


9.1.11 Memilih Database

12. Sesudah di klik kanan pada database field akan muncul menu pilihan sebagai berikut:

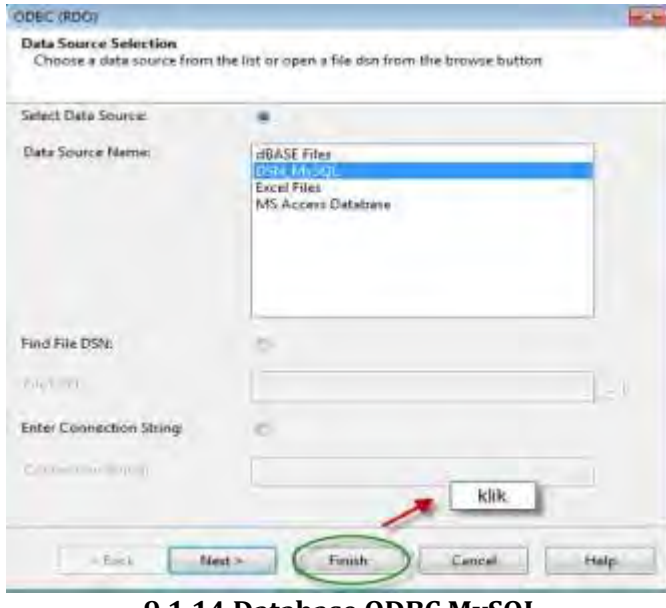
9.1.12 Database Expert

13. Expand Create New Connection > ODBC (RDO).



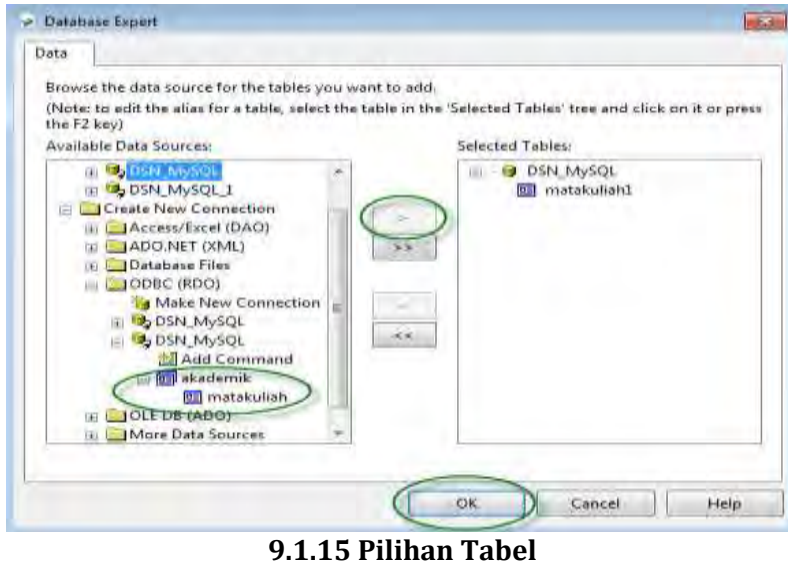
9.1.13 Database ODBC Make New Connection

14. Dan pilih ODBC yang telah kita buat sebelumnya dan klik tombol Finish.



9.1.14 Database ODBC MySQL

15. Tambahkan tabel matakuliah yang telah kita buat dan klik tombol Finish. Kemudian keluar menu Database Expert pilih table yang telah dibuat, kemudian klik OK.



9.1.15 Pilihan Tabel

16. Kembali Masukkan Column-column yang akan ditampilkan pada Report.

9.1.16 Masukkan Field ke Crysatal Report

9.2 Crystal Report

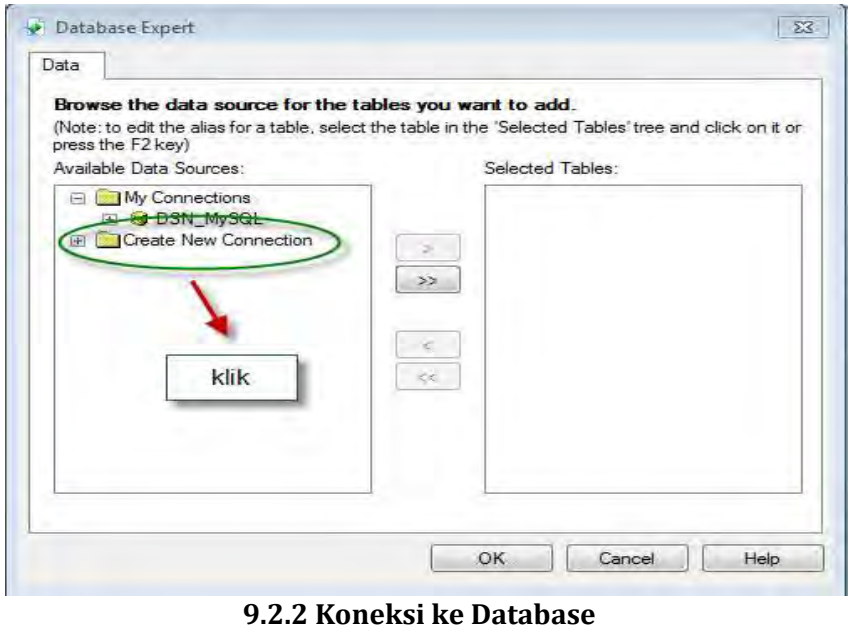
Crystal Report merupakah perangkat lunak untuk membuat laporan baik berupa sumber data dari database buatan Microsoft maupun database buatan lainnya. Menggunakan Crystal Report memungkinkan pengguna akhir menghasilkan laporan yang mencakup visualisasi dengan penerapan bisnis baru ke dalam laporan tergantung kebutuhan perusahaan maupun keinginan pengguna. Crystal Report terhubung ke sumber data termasuk database relasional Oracle, SQL Server, MySQL ataupun sumber data seperti BW, atau dengan data XML.

Crystal report pada Visual Studio 2010 tidak langsung disertakan pada saat kita menginstal visual studio, jadi mau tidak mau kita harus menginstal crystal report secarah terpisah. Untuk mendapatkan Crystal Report anda bisa mendapatkanya melalui *search engine google* dengan kata kuncin SAP Crystal Report for Visual Basic 2010, atau dengan cara lain harus membelinya secara resmi di https://www.sap.com/products.html. Penulis menggunakan Crystal Report 2008.

Sesudah perangkat lunak Crystal Report di instalasi di komputer, jalankan Crystal Report perti berikut ini:

9.2.1 Tampilan Crystal Report 2008

1. Klik Menu File -> New -> Blank Report, akan muncul menu Koneksi ke Database

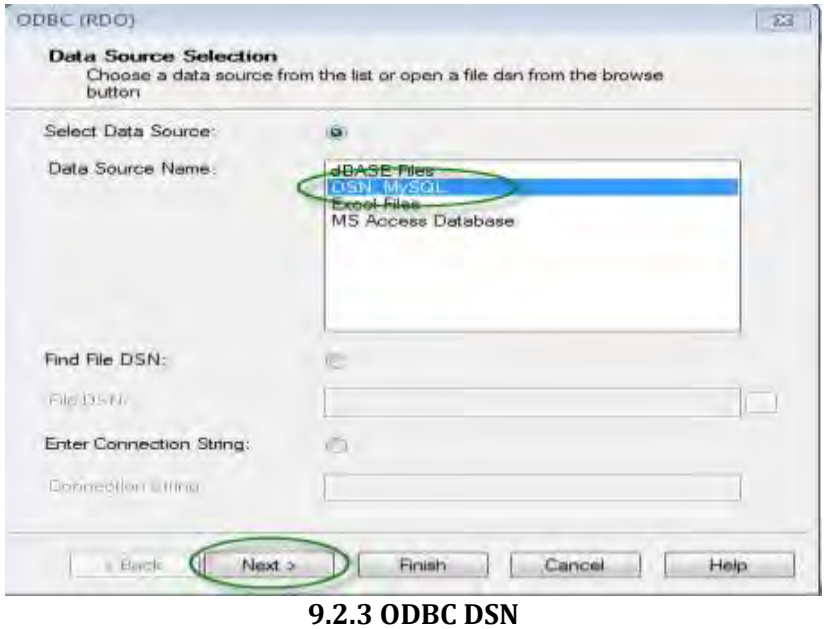


9.2.2 Koneksi ke Database

2. Pilih Folder **Create New Connection**

9.2.3 Koneksi ke ODBC

3. Double klik **Make New Connection,** akan muncul menu sebagai berikut:



9.2.3 ODBC DSN

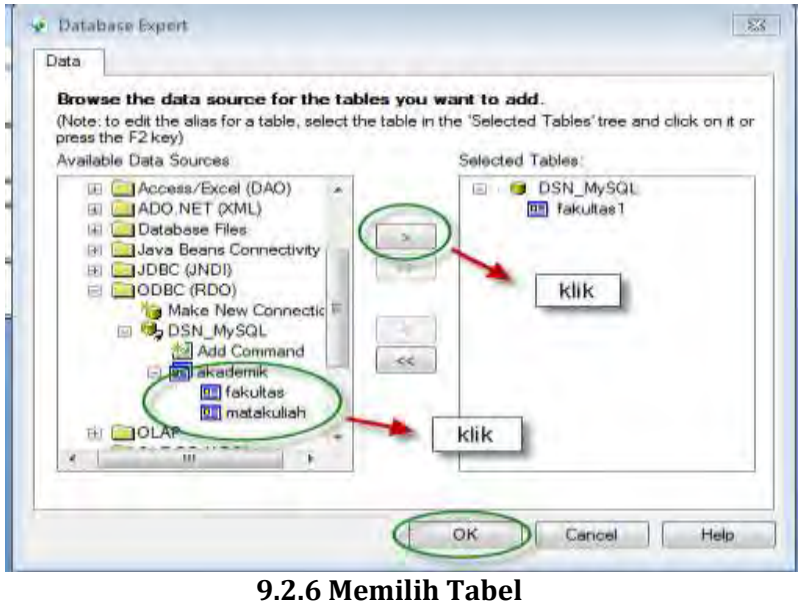
4. Pilih DSN_MySQL yang telah dibuat di Control Panel. Kemudian klik tombol **Next, User ID** ketik *root*, **Database *pilih*** *Akademik*

9.2.4 Koneksi ke Database dan User ID

5. Klik tombol **Finish**, akan muncul Menu Berikut:

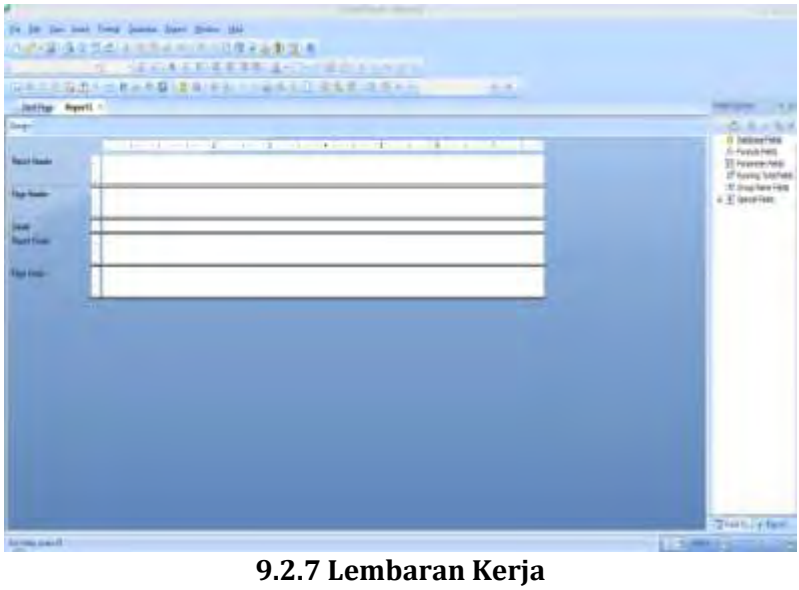
9.2.5 Memilih Database dan Tabel

6. Pilih Folder DSN_MySQL, seperti gambar, klik dan lihat gambar berikut:



9.2.6 Memilih Tabel

7. Pilih Tabel yang diinginkan, letakkan ke kotak sebelah kanan tabel yang akan di tampilkan, kemudian klik tombol OK

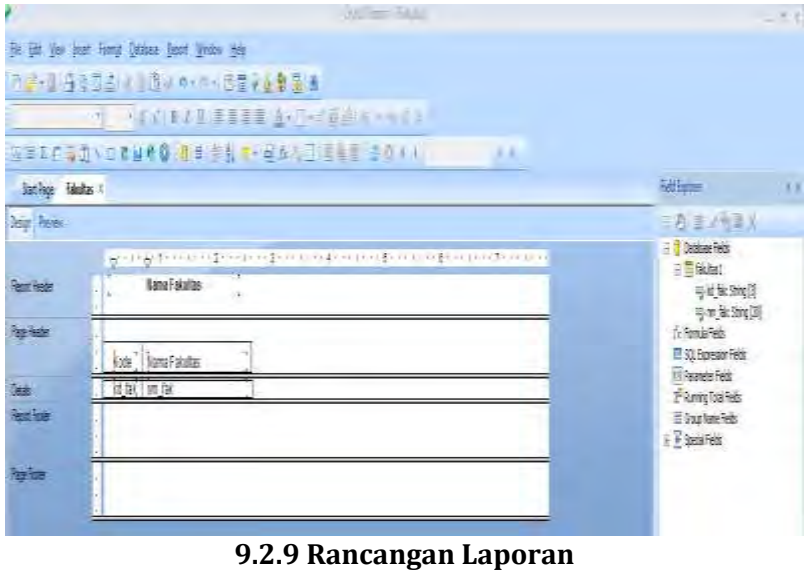


9.2.7 Lembaran Kerja

8. Lihat di Menu Field Explorer, klik dan draw field yang akan ditampilkan

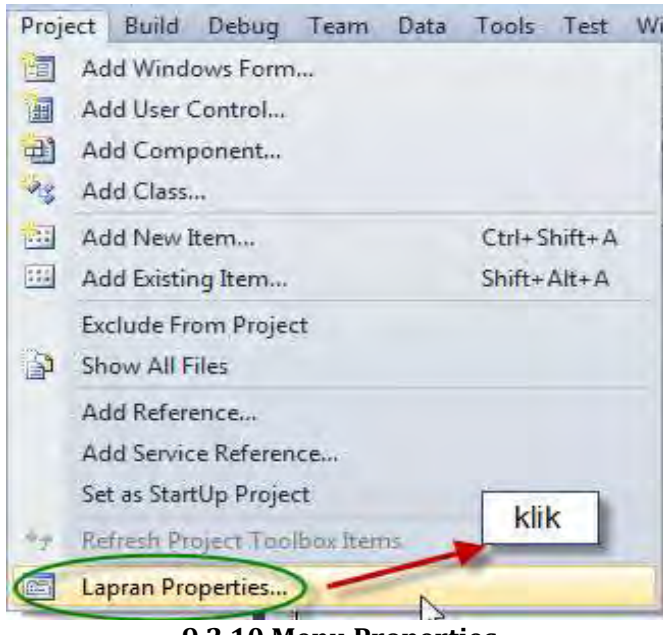
9.2.8 Memilih Field

9. Rancang Tampilan seperti gambar ini, kemudian simpan dengan nama file **Fakultas**.



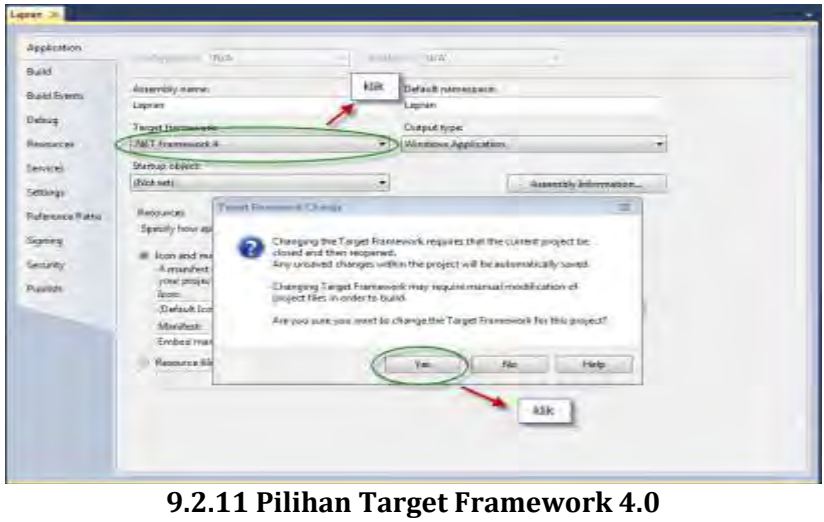
9.2.9 Rancangan Laporan

10. Kembali ke Form dalam project sebagai tempat untuk laporan, pada menu properties rubah *name* LapFakultas dan klik OK.
11. Dalam Form klik menu Project klik prpeperties



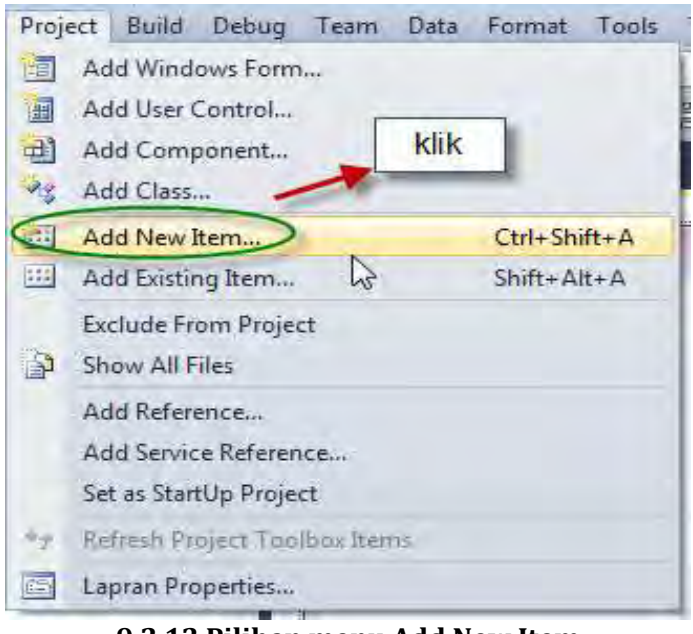
9.2.10 Menu Properties

12. Setelah diklik, muncul pilih di menu target Framework dengan piliha **.NET Framework 4,** akan muncul kotak konfirmasi dan pilih Yes.



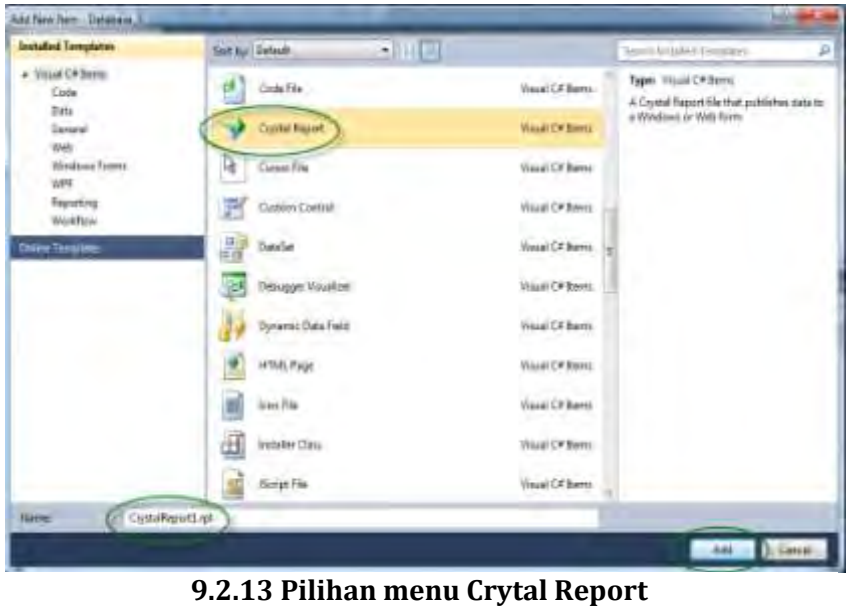
9.2.11 Pilihan Target Framework 4.0

13. Setelah selesai klik menu Project lagi, pilih menu Add New Item



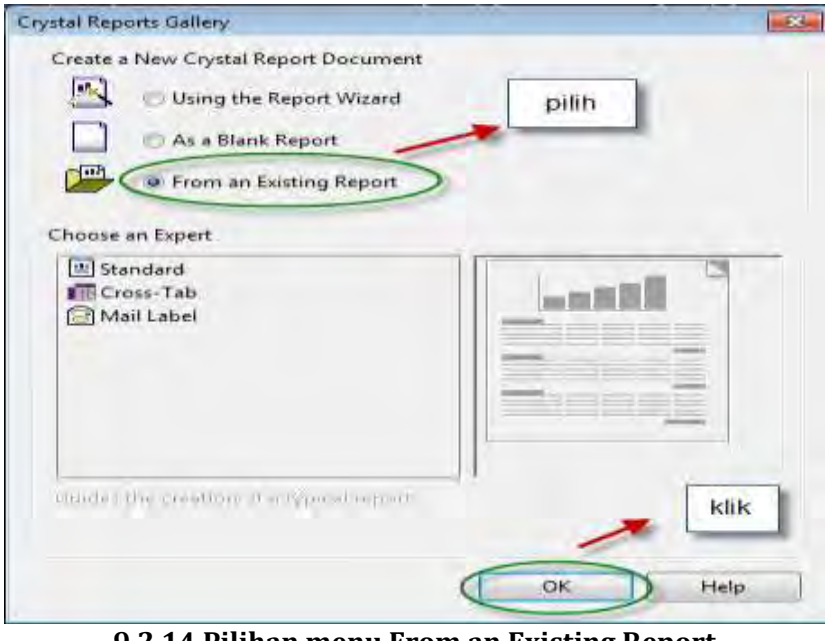
9.2.12 Pilihan menu Add New Item

14. Setelah di klik menu Add New Item, pilih Crystal Report dan kemudian klik Add seperti gambar berikut:



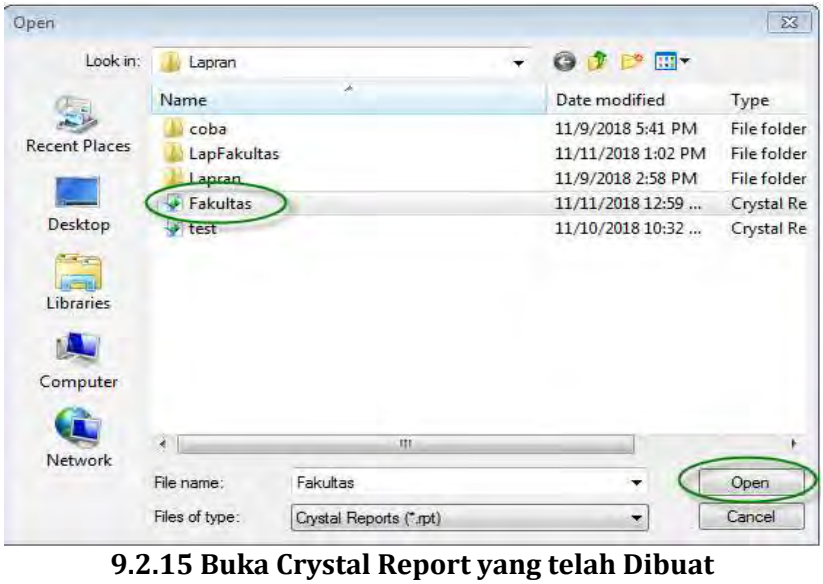
9.2.13 Pilihan menu Crytal Report

15. Klik tombol Add



9.2.14 Pilihan menu From an Existing Report

16. Pilih Pilihan From an Existing Report untuk membuka Laporan yang telah di buat menggunakan Crystal Report 2008

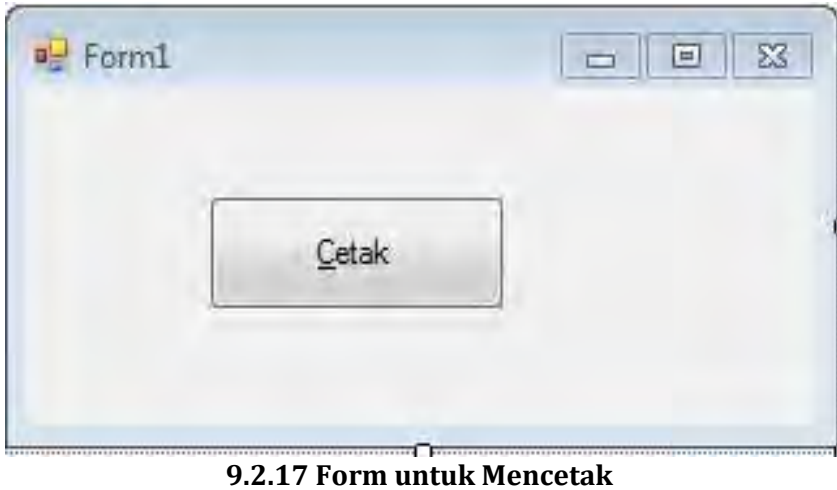


9.2.15 Buka Crystal Report yang telah Dibuat

17. Pilih **File Crystal Report** ***Fakultas.rpt*** yang telah dibuat dengan memilih tombol **Open**

9.2.16 Crystal Report Fakultas

18. Form yang Pertama, tambahkan toolbox Button



9.2.17 Form untuk Mencetak

19. Kemudian Add Form Baru di menu -> Project -> Add Windows Form, lalu tambahkan *Toolbox CrystalReportViewer*

9.2.18CrystalReportViewer

20. Buka Form Pertama, tambahkan pustaka

using CrystalDecisions.CrystalReports.Engine;

21. Tambahkan program di Class Form1

ReportDocument rd=new ReportDocument();

22. Lalu klik CrystalReportViewer untuk memastikan fokusnya dan kembali ke Properties dan atur **modifier** menjadi **Public**

9.2.17 pilih Modifiear

23. Kembali ke Form1, Klik Button Cetak ketik program seperti berikut:

```
private void button1_Click(object sender, EventArgs e)
 {
 Form2 fr = new Form2();
 fr.Show();
 rd.Load(@"E:\Pemograman
Visual\LatBuku\Destop\Lapran\fakultas.rpt");
 fr.crystalReportViewer1.ReportSource=rd;
 fr.crystalReportViewer1.Refresh();
 }
```

Program lengkapnya Sebagai Berikut:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using CrystalDecisions.CrystalReports.Engine;

namespace LapFakultas
{
 public partial class Form1 : Form
 {
 public Form1()
 {
 InitializeComponent();
 }
 ReportDocument rd=new ReportDocument();

 private void button1_Click(object sender, EventArgs e)
 {
 Form2 fr = new Form2();
 fr.Show();
 rd.Load(@"E:\Pemograman Visual\LatBuku\Destop\Lapran\fakultas.rpt");
 fr.crystalReportViewer1.ReportSource=rd;
 fr.crystalReportViewer1.Refresh();
 }
 }
}
```

∞

DAFTAR PUSTAKA

Ali, M. 2014. Kitab Belajar Pemogramman C#. https://www.scribd.com/doc/236623672/Ebook-Pemrograman-C-Lengkap. Di download pada tanggal 9 Oktober 2018

Crystal Report. 2008. Crystal Report Developer for Studio Download. Diperoleh dari laman https://wiki.scn.sap.com/wiki/display/BOBJ/Crystal+Reports%2C+Developer+for+Visual+Studio+Downloads. Didownload pada tanggal 30 Oktober 2018

Dian, M. 2017. Belajar C#: Memahami Sintaks Dasar Bahasa Pemrograman C#. https://www.petanikode.com/cs-sintaks/. Di download pada tanggal 9 Oktober 2018.

Icodeformoney. Belajar C Sharp. Diperoleh dari https://icodeformoney.com/tuts/Mengambil-data-dari-database-pada-CSharp. Di Download pada tanggal 9 Oktober 2018

ik3-2013.blogspot.com. 2014. Visual C#: Membuat Class Untuk Koneksi MYSQL. Diperoleh dari http://ik3-2013.blogspot.com/2014/04/visual-c-membuat-class-untuk-koneksi.html. Di download pada tanggal 12 Oktober 2018

MySQL dot NET Connector. 2018. Diperoleh dari website http://dev.mysql.com/downloads/connector/net/6.6.html#downloads. Di download pada tanggal 4 May 2018

Siswanto. D. 2002. Mempelajari C#: Bahasa Pemrograman Modern. https://www.academia.edu/32837430/Mempelajari_C_Bahasa_Pemrograman_Modern. Di download pada tanggal 9 Oktober 2018

sqlneed.com. 2018. Load into listview from MySql using C Sharp. Diperoleh dari laman https://www.sqlneed.com/2018/01/Load-into-listview-from-MySql-using-CSharp.html. Di download pada tanggal 28 Oktober 2018

stackoverflow.com. 2018. how-to-connect-to-mysql-database. Di peroleh dari laman.

https://stackoverflow.com/questions/21618015/how-to-connect-to-mysql-database. Didownload pada tanggal 9 Oktober 2018

stackoverflow.com. 2015. Displaying a crystal report using c sharp. Diperoleh di laman https://stackoverflow.com/questions/16218371/displaying-a-crystal-report-using-c-sharp. Di download pada tanggal 8 November 2018.

∞

RIWAYAT PENULIS

1	Nama lengkap(dengan gelar)	Asrianda, S. Kom, M. Kom
2	Jenis Kelamin	Laki-laki
3	Jabatan fungsional	Lektor
4	NIP/NIK/Identitas lainnya	197305152006041001
5	NIDN	0015057317
6	Tempat dan tanggal lahir	Lhokseumawe, 15 Mei 1973
7	E-mail	4srianda@gmail.com
8	Nomor telepon/Faks/HP	085260584721
9	Alamat kantor	Jln. Tengku Nie Reulet Aceh utara

A. RIWAYAT PENDIDIKAN

	S-1	S-2	S-3
Nama Perguruan Tinggi	STMIK- AMIK Riau	Universitas Sumatera Utara	-
Bidang Ilmu	Teknik Informatika	Teknik Informatika	
Tahun Masuk-Lulus	2002 - 2005	2010 - 2013	-
Judul Skripsi/Tesis/Disertasi	Desain Sistem Informasi Geografis Kota Pekanbaru	Spesifikasi dan Pengaturan Kontrol Akses dalam Pendataan Masyarakat Miskin di Kabupaten Aceh Utara	
Nama Pembimbing/Promotor	Lusiana, M. Kom/Ts. Syalahudin, M. Kom	Prof. Dr. Muhammad Zarlis/Dr. Sutarman, M. Sc	-

B. PUBLIKASI ARTIKEL ILMIAH DALAM JURNAL DALAM 5 TAHUN TERAKHIR

No	Judul Artikel Ilmiah	Volume/Nomor/Tahun	Nama Jurnal
1	Kontrol Akses dalam Keamanan Data Pendataan Penduduk Miskin	Volume 6, Nomor 1, Mei 2013, ISSN 1979-0236	Jurnal Samudera

No	Judul Artikel Ilmiah	Volume/Nomor/Tahun	Nama Jurnal
2	Spesifikasi dan Pengaturan Kontrol Akses dalam Pendataan Masyarakat Miskin di Kabupaten Aceh Utara	Volume 6, Nomor 2, November 2013, ISSN 1979-0236	Jurnal Samudera
3	Role Pengontrol Dalam Pendataan Data Penduduk Miskin	Vol. 8, No.2, Oktober 2016, ISSN 2302-4838	Jurnal TECHSI
4	Kontrol Akses dan Keamanan Data bagi Penduduk Miskin	Vol 1, No 1, November 2016	Proceeding Seminar Nasional Ilmu Komputer Universitas Almuslim

C. KARYA BUKU DALAM 5 TAHUN TERAKHIR

No	Judul buku	Tahun	Jumlah Halaman	Penerbit
1	Buku Pemograman Visual Basic: Teori dan Implementasi, Unimal Press ISBN 978-602-1373-00-2	2013	108	Unimal Press
2	Pemograman Database, Unimal Press ISBN 978-602-1373-01-9	2012	141	Unimal Press
3	Pengaturan Kontrol Akses bagi Pendataan Data	2016	126	Unimal Press
4	Teknik Dan Implementasi Pengelolaaan Jurnal *Online*	2017	240	Unimal Press

∞

Microsoft membuat C# seiring dengan pembuatan Framework .NET. Chief Architect dalam pembuatan C# adalah Anders Hejlsberg yang sebelumnya berperandalam pembuatan Borland Delphi dan Turbo Pascal. C# menjanjikan produktifitas dan kemudahan yang ada di Visual Basic dengan kemampuan dan fleksibilitas yang ada di C/C++. Pemograman C# sesuai kaidah bahasanya " C# (pronounced "C Sharp") is a simple, modern, object oriented, and type-safe programming language. It will immediately be familiar to C and C++ programmers. C# combines the high productivity of Rapid Application Development (RAD) languages andthe raw power of C++".

Untuk mencapai produktifitas tinggi ini konsep-konsep sulit C++ disederhanakan dan fitur-fitur baru ditambahkan. Hal ini mungkin terasa mirip dengan Java, karena itulah C# bisa dianggap sebagai sepupu Java.

C# adalah salah satu dari banyak bahasa yang bisa dipakai untuk pemrograman.NET. Kelebihan utama bahasa ini adalah sintaksnya yang mirip C, namun lebih mudahdan lebih bersih. Untuk perbandingan penulis cantumkan sedikit informasi mengenai Managed C++ dan Visual Basic.NET:

Dengan buku ini Penulis ingin membagi pengalaman yang sering Penulis alami dalam menyusun buku referensi pemograman Visual Studio 2010, berguna untuk mahasiswa di Universitas Malikussaleh yang minimnya bahan pustaka dan buku pembelajaran dalam menguasai bahasa pemograman. Buku ini bukan hal yang baru tetapi hasil kumpulan referensi yang telah ada baik di dunia maya maupun buku yang telah banyak beredar di pasaran., dan Penulis juga merasa ilmu yang Penulis dapatkan selama ini belum ada apa-apanya di dunia Pemograman, minimal dengan buku ini dapat membuka wawasan bagi para mahasiswa, juga masyarakat umum dan para teman-teman sejawat Dosen yang ingin mempelajari Bahasa Pemograman C# dan dapat mengembangkan lebih lanjut lagi.

ISBN 978-602-464-053-8

UNIMAL PRESS