

PRAKTIKUM 2

Pengantar Visual C++ & Penulisan Program

A. TUJUAN PEMBELAJARAN

1. Mampu memahami struktur penulisan bahasa C
2. Mengerti perintah keluaran di C
3. Mengetahui Lingkungan Visual C++
4. Mengerti cara membuat, meng-compile, dan melakukan running sebuah program C di Visual C++

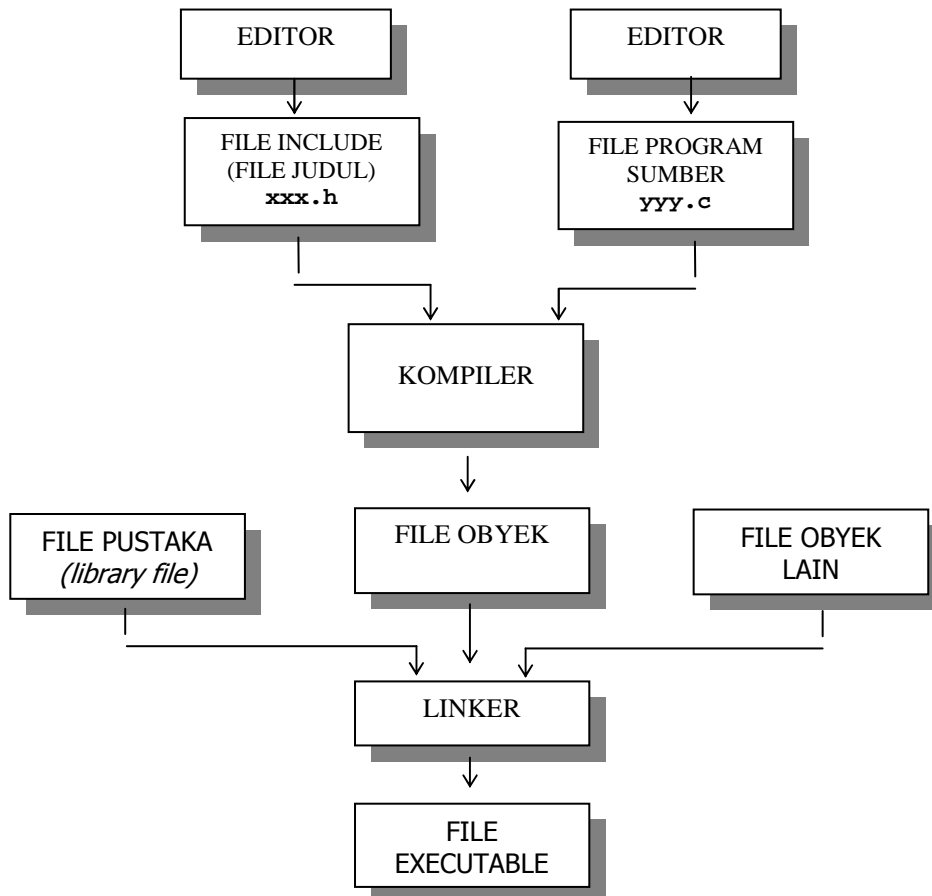
B. DASAR TEORI

Akar dari bahasa C adalah bahasa BCPL yang dikembangkan oleh Martin Richards pada tahun 1967. Bahasa ini memberikan ide kepada Ken Thompson yang kemudian mengembangkan bahasa yang disebut dengan B pada tahun 1970. Perkembangan selanjutnya dari bahasa B adalah bahasa C oleh Dennis Ritchie sekitar tahun 1970-an di Bell Telephone Laboratories Inc. (sekarang adalah AT&T Bell Laboratories). Bahasa C pertama kali digunakan pada komputer Digital Equipment Corporation PDP-11 yang menggunakan sistem operasi UNIX.

Standar bahasa C yang asli adalah standar dari UNIX. Sistem operasi, kompiler C dan seluruh program aplikasi UNIX yang esensial ditulis dalam bahasa C. Kepopuleran bahasa C membuat versi-versi dari bahasa ini banyak dibuat untuk komputer mikro. Untuk membuat versi-versi tersebut menjadi standar, ANSI (*American National Standards Institute*) membentuk suatu komite (*ANSI committee X3J11*) pada tahun 1983 yang kemudian menetapkan standar ANSI untuk bahasa C. Standar ANSI ini didasarkan kepada standar UNIX yang diperluas.

Proses Kompilasi dan Linking Program C

Proses dari bentuk *source program*, yaitu program yang ditulis dalam bahasa C hingga menjadi program yang *executable* ditunjukkan pada Gambar 2.1 di bawah ini.



Gambar 2.1 Proses Kompilasi-Linking dari program C

Struktur Penulisan Program C

Program C pada hakekatnya tersusun atas sejumlah blok fungsi. Sebuah program minimal mengandung sebuah fungsi. Fungsi pertama yang harus ada dalam program C dan sudah ditentukan namanya adalah *main()*. Setiap fungsi terdiri atas satu atau beberapa pernyataan, yang secara keseluruhan dimaksudkan untuk melaksanakan tugas khusus. Bagian pernyataan fungsi (sering disebut tubuh fungsi) diawali dengan tanda kurung kurawal buka ({} dan diakhiri dengan tanda kurung kurawal tutup (}). Di antara kurung kurawal itu dapat dituliskan statemen-statement program C. Namun pada kenyataannya, suatu fungsi bisa saja tidak mengandung pernyataan sama sekali.

Walaupun fungsi tidak memiliki pernyataan, kurung kurawal haruslah tetap ada. Sebab kurung kurawal mengisyaratkan awal dan akhir definisi fungsi. Berikut ini adalah struktur dari program C

```
main()
{
    statemen-statemen;
}
fungsi_fungsi_lain()
{
    statemen-statemen;
}
```

fungsi utama

fungsi-fungsi lain yang ditulis oleh pemrogram

Bahasa C dikatakan sebagai bahasa pemrograman terstruktur karena strukturnya menggunakan fungsi-fungsi sebagai program-program bagiannya (*subroutine*). Fungsi-fungsi yang ada selain fungsi utama (*main()*) merupakan program-program bagian. Fungsi-fungsi ini dapat ditulis setelah fungsi utama atau diletakkan di file pustaka (*library*). Jika fungsi-fungsi diletakkan di file pustaka dan akan dipakai di suatu program, maka nama file judulnya (*header file*) harus dilibatkan dalam program yang menggunakannya dengan *preprocessor directive* berupa *#include*.

Pengenalan Fungsi-Fungsi Dasar

a. Fungsi *main()*

Fungsi *main()* harus ada pada program, sebab fungsi inilah yang menjadi titik awal dan titik akhir eksekusi program. Tanda { di awal fungsi menyatakan awal tubuh fungsi dan sekaligus awal eksekusi program, sedangkan tanda } di akhir fungsi merupakan akhir tubuh fungsi dan sekaligus adalah akhir eksekusi program. Jika program terdiri atas lebih dari satu fungsi, fungsi *main()* biasa ditempatkan pada posisi yang paling atas dalam pendefinisian fungsi. Hal ini hanya merupakan kebiasaan. Tujuannya untuk memudahkan pencarian terhadap program utama bagi pemrogram. Jadi bukanlah merupakan suatu keharusan.

b. Fungsi *printf()*.

Fungsi *printf()* merupakan fungsi yang umum dipakai untuk menampilkan suatu keluaran pada layar peraga. Untuk menampilkan tulisan

```
Selamat belajar bahasa C
```

misalnya, pernyataan yang diperlukan berupa:

```
printf("Selamat belajar bahasa C");
```

Pernyataan di atas berupa pemanggilan fungsi *printf()* dengan argumen atau parameter berupa string. Dalam C suatu konstanta string ditulis dengan diawali dan diakhiri tanda petik-ganda (“”). Perlu juga diketahui pernyataan dalam C selalu diakhiri dengan tanda titik koma (;). Tanda titik koma dipakai sebagai tanda pemberhentian sebuah pernyataan dan bukanlah sebagai pemisah antara dua pernyataan.

Tanda \ pada string yang dilewatkan sebagai argumen *printf()* mempunyai makna yang khusus. Tanda ini bisa digunakan untuk menyatakan karakter khusus seperti karakter baris-baru ataupun karakter *backslash* (miring kiri). Jadi karakter seperti \n sebenarnya menyatakan sebuah karakter. Contoh karakter yang ditulis dengan diawali tanda \ adalah:

\"	menyatakan karakter petik-ganda
\\	menyatakan karakter backslash
\t	menyatakan karakter tab

Dalam bentuk yang lebih umum, format *printf()*

```
printf("string kontrol", daftar argumen);
```

dengan string kontrol dapat berupa satu atau sejumlah karakter yang akan ditampilkan ataupun berupa penentu format yang akan mengatur penampilan dari argumen yang terletak pada daftar argumen. Mengenai penentu format di antaranya berupa:

%d	untuk menampilkan bilangan bulat (integer)
%f	untuk menampilkan bilangan titik-mengambang (pecahan)
%c	untuk menampilkan sebuah karakter
%s	untuk menampilkan sebuah string

Contoh:

```
#include <stdio.h>

main( )
{
    printf("No      : %d\n", 10);
    printf("Nama   : %s\n", "Ali");
    printf("Nilai  : %f\n", 80.5);
    printf("Huruf  : %c\n", 'A');
}
```

Fungsi scanf()

Fungsi ini digunakan untuk memasukkan berbagai jenis data. Misalnya untuk memasukkan data jari-jari lingkaran pada program menghitung luas lingkaran.

```
scanf("%f", &radius);
```

Bentuk *scanf()* sesungguhnya menyerupai fungsi *printf()*. Fungsi ini melibatkan penentu format yang pada dasarnya sama digunakan pada *printf()*. Secara umum bentuk *scanf()* adalah sebagai berikut :

```
scanf("string kontrol", daftar_argumen);
```

Dengan string kontrol dapat berupa :

- Penentu format
- Karakter spasi-putih (*white-space*)
- Karakter bukan spasi-putih

```
scanf("%f", &radius);
```

berarti (bagi komputer) : “bacalah sebuah bilangan real (%f) dan tempatkan ke alamat dari **radius** (&radius)”.

Pengenalan Praprosesor #include

#include merupakan salah satu jenis pengarah praprosesor (*preprocessor directive*). Pengarah praprosesor ini dipakai untuk membaca file yang di antaranya berisi deklarasi fungsi dan definisi konstanta. Beberapa file judul disediakan dalam C. File-file ini mempunyai ciri yaitu namanya diakhiri dengan ekstensi **.h**. Misalnya pada program `#include <stdio.h>` menyatakan pada kompiler agar membaca file bernama *stdio.h* saat pelaksanaan kompilasi.

Bentuk umum `#include`:

```
#include "namafilename"
```

Bentuk pertama (`#include <namafilename>`) mengisyaratkan bahwa pencarian file dilakukan pada direktori khusus, yaitu direktori file *include*. Sedangkan bentuk kedua (`#include "namafilename"`) menyatakan bahwa pencarian file dilakukan pertama kali pada direktori aktif tempat program sumber dan seandainya tidak ditemukan pencarian akan dilanjutkan pada direktori lainnya yang sesuai dengan perintah pada sistem operasi.

Kebanyakan program melibatkan file **stdio.h** (file-judul I/O standard, yang disediakan dalam C). Program yang melibatkan file ini yaitu program yang menggunakan pustaka I/O (input-output) standar seperti *printf()*.

Komentar dalam Program

Untuk keperluan dokumentasi dengan maksud agar program mudah dipahami di suatu saat lain, biasanya pada program disertakan komentar atau keterangan mengenai program. Dalam C, suatu komentar ditulis dengan diawali dengan tanda */** dan diakhiri dengan tanda **/*.

Contoh :

```
/*
    Tanda ini adalah komentar
    untuk multiple lines
*/
#include <stdio.h>

main()
{
    printf("Coba\n");    //Ini komentar satu baris
}
```

Mengenal Visual C++

Dalam praktikum Algoritma dan Pemrograman ini kita menggunakan Microsoft Visual Studio sebagai IDE-nya. IDE merupakan singkatan dari Integrated Development Environment, merupakan lembar kerja terpadu untuk pengembangan program.

Visual Studio merupakan sebuah perangkat lunak lengkap (suite) yang dapat digunakan untuk melakukan pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi console, aplikasi Windows, ataupun aplikasi Web. IDE dari Visual C++ dapat digunakan untuk :

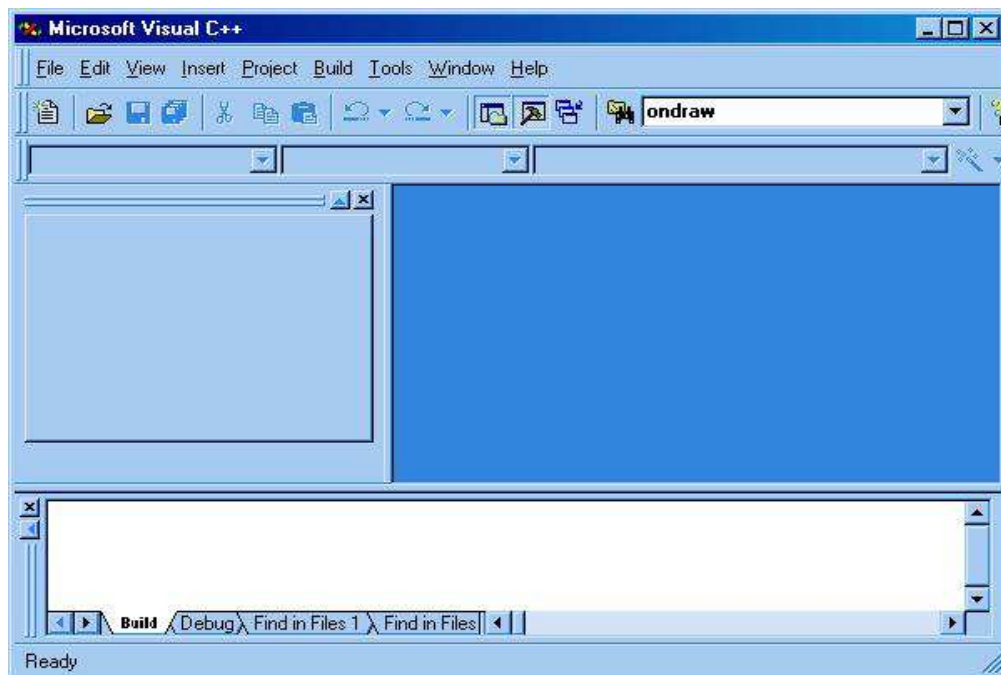
1. menulis naskah program;
2. mengompilasi program (compile);
3. melakukan pengujian terhadap program (Debugging); dll.

Memulai Visual C++

Memulai membuat program dengan visual C++ adalah dengan mengikuti langkah-langkah sebagai berikut:

1. Klik tombol **Start** pada *taskbar* Windows
2. Klik **Programs** dari Start menu
3. Klik **Microsoft Visual C++ 6.0** sehingga akan muncul tampilan awal dari Visual C++ seperti pada Gambar 2.2.

(Jika Visual C++ di-install sebagai produk yang *standalone*, maka akan terlihat Microsoft Visual C++ 6.0 sebagai salah satu pilihan dalam menu Programs. Namun jika di-install sebagai bagian dari Microsoft Visual Studio, maka Visual C++ akan tercantum dalam menu di bawah Microsoft Visual Studio 6.0 dalam menu Programs).



Gambar 2.2 Tampilan awal dari Visual C++


Project dan Workspace dalam Pembuatan Program

- ❖ **Workspace** adalah ruang yang digunakan untuk menempatkan project.
- ❖ Dalam satu workspace dimungkinkan diisi lebih dari satu project.
- ❖ Workspace akan disimpan dengan ekstensi *.dsw (Developer Studio Workspace)*
- ❖ Setiap kali memulai membuat sebuah program, terlebih dahulu harus membuat sebuah project.

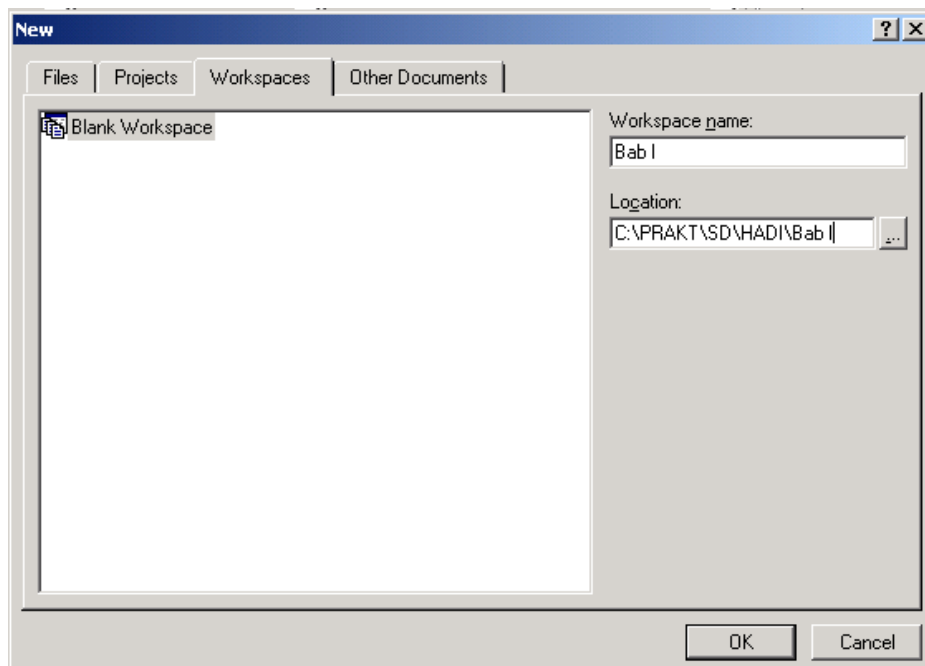
- ❖ File project ini menyimpan berbagai informasi, di antaranya tentang *source code* (kode sumber) mana yang akan dipakai dalam program.
- ❖ Project disimpan dengan ekstensi **.dsp** (*Developer Studio Project*).

Membuat Workspace Baru

1. Klik **New** pada File menu
2. Dalam kotak dialog New, klik **Workspaces**
3. Ketiklah nama *workspace* yang hendak dibuat. Dalam praktikum ini, nama *workspace* mewakili nama bab bahasan, misalnya : **Bab I**
4. Pastikan lokasi tempat *workspace* adalah : **C:\Prakt\SD\user_name**)**
(misalnya **C:\Prakt\SD\Hadi**)
5. Klik **OK**

') Untuk menempatkan *workspace* klik : **Browse 

Tampilan langkah membuat workspace baru ini ditunjukkan pada Gambar 2.3.



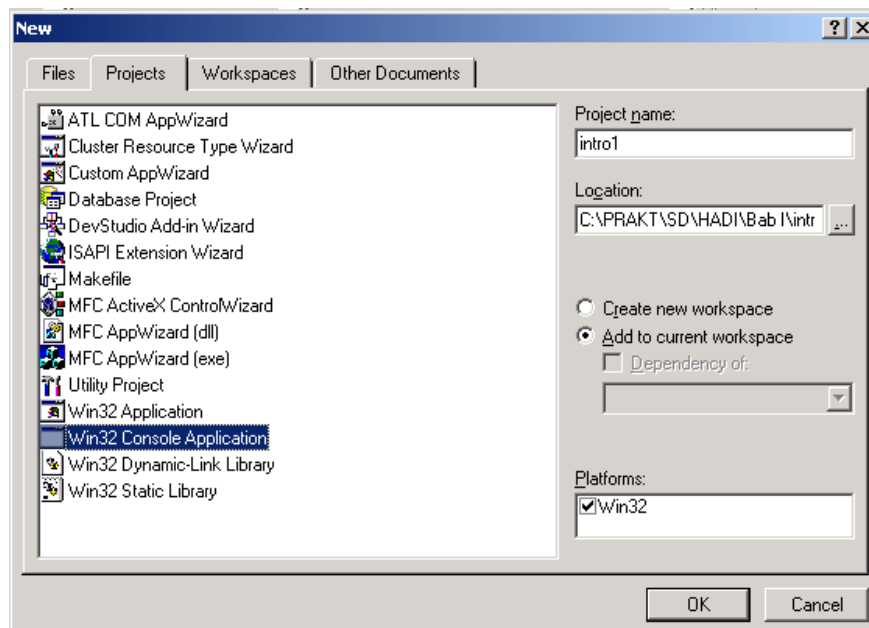
Gambar 2.3 Kotak dialog untuk New Workspaces

Membuat Project Baru

1. Klik **New** pada File menu
2. Dalam kotak dialog New, klik **Projects**

3. Klik **Win32 Console Application**
4. Ketikkan nama *project* yang dikehendaki, dalam hal ini misalnya : **intro1**
5. Pilihlah opsi : **Add to current workspace**, agar *project* yang baru dibuat disimpan di bawah *workspace* Bab I yang telah dibuat sebelumnya
6. Pastikan **Win32** telah terpilih pada kotak **Platforms**
7. Klik **OK**

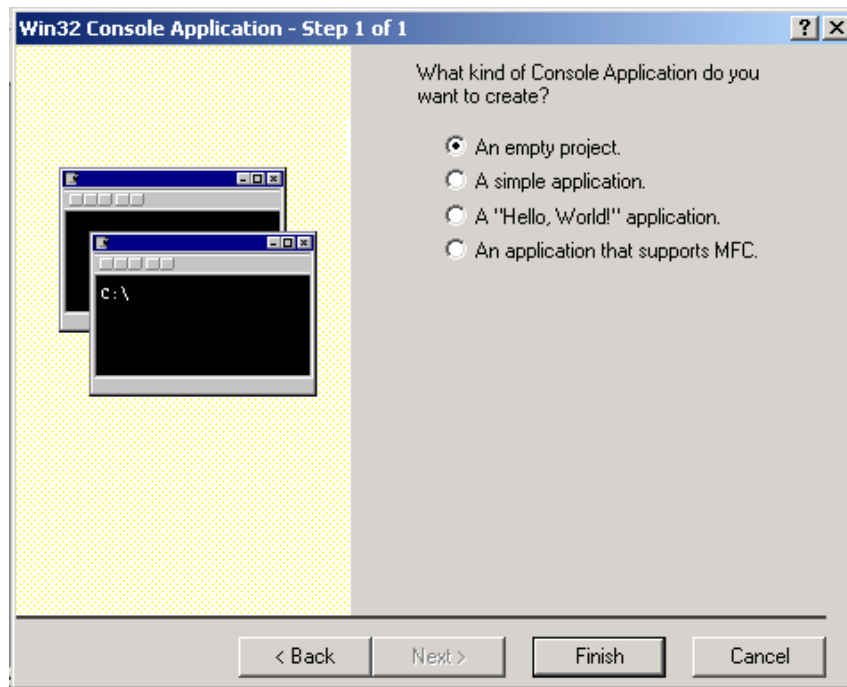
Tampilan langkah membuat project baru ini ditunjukkan pada Gambar 2.4.



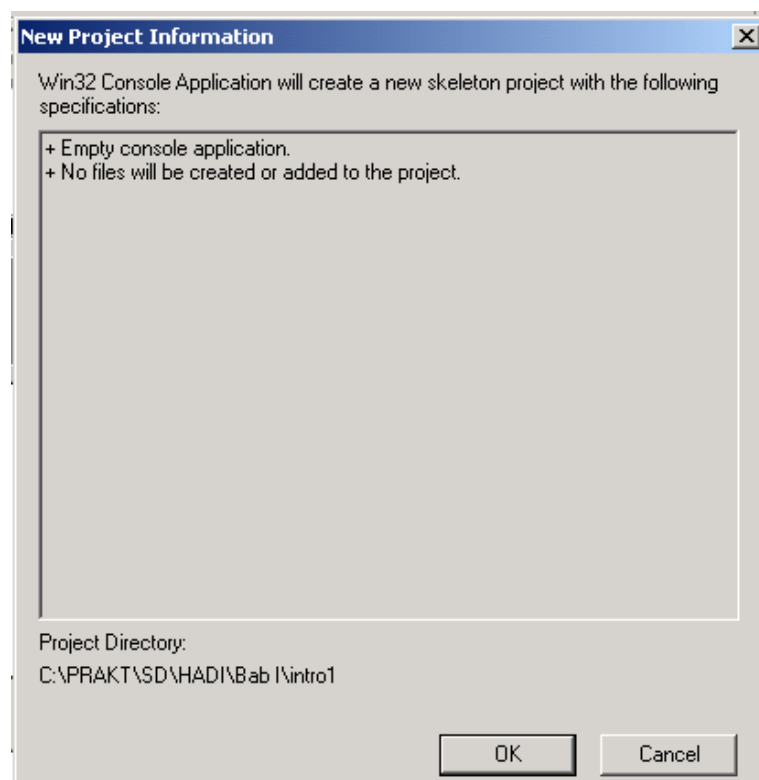
Gambar 2.4 Kotak dialog Step 1 of 3

- ❖ Ikutilah instruksi dalam kotak dialog Wizard yang akan muncul setelah kota dialog New.
- ❖ Untuk sebuah *Win32 console application* hanya ada satu kotak di bawah ini :
 1. Dalam kotak dialog **Step 2 of 3** pilihlah opsi : **An empty project** (Gambar 2.5)
 2. Klik **Finish**
 3. Dalam kotak dialog **New Project Information**, telitilah informasi yang diberikan untuk memastikan apa yang hendak dibuat. Berikutnya klik **OK**.

Tampilan kotak dialog step 2 dari 3 ditunjukkan pada Gambar 2.5 dan tampilan kotak dialog New Project Information ditunjukkan pada Gambar 2.6.



Gambar 2.5 Kotak dialog Step 2 of 3



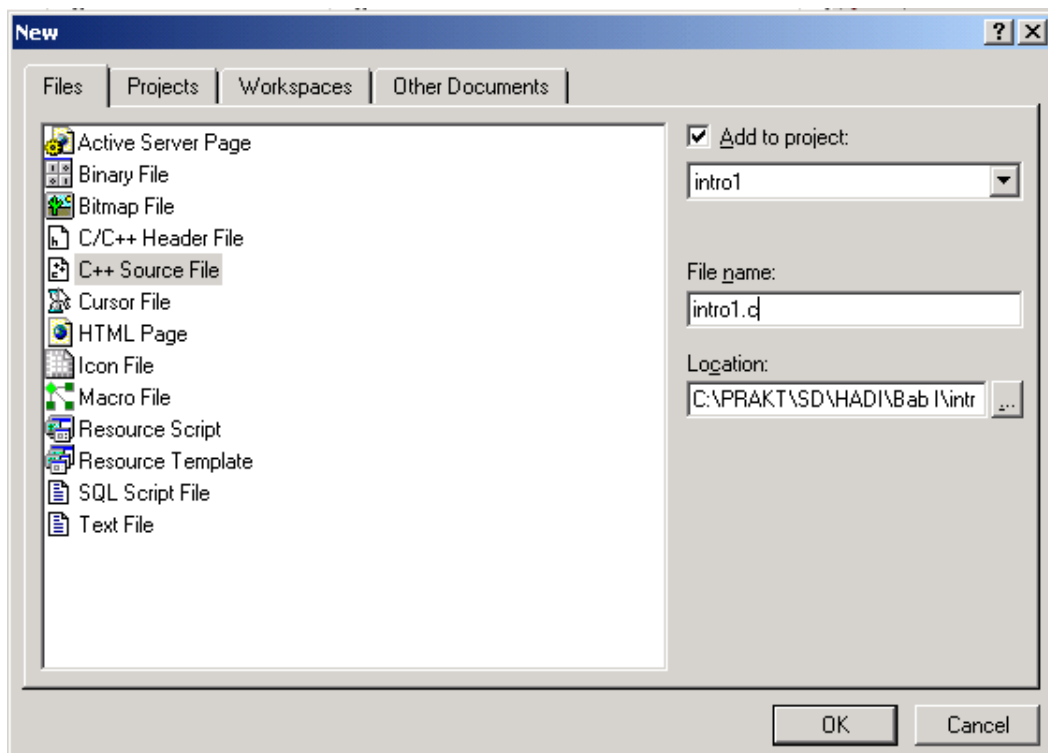
Gambar 2.6 Kotak dialog New Project Information

Membuat File Baru

1. Klik **New** pada File menu

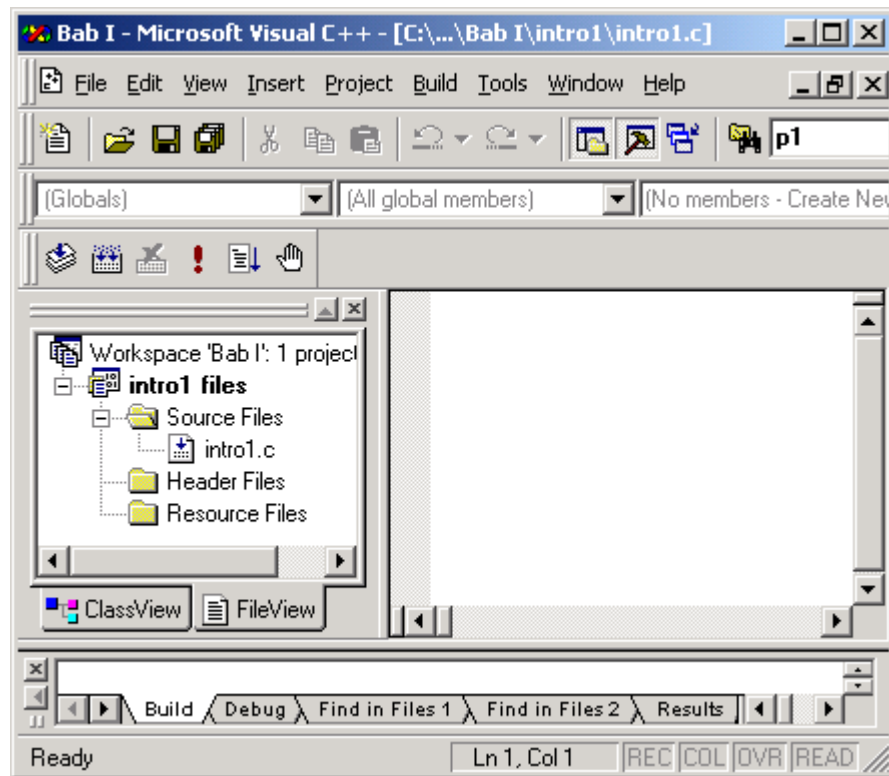
2. Dalam kotak dialog New, klik **Files**
3. Klik **C++ Source File**
4. Ketikkan nama file yang dikehendaki, dalam hal ini misalnya : **intro1.c** (jangan lupa untuk selalu menambahkan ekstensi **.c**. Bila user hanya menyetikkan nama file tanpa ekstensi, dalam hal ini akan dianggap sebagai file dengan ekstensi **.cpp** /C++)
5. Pastikan **Add to project** telah terpilih
6. Pastikan *project* tempat file akan diletakkan dan lokasi penyimpanan file (dalam hal ini adalah : **C:\Prakt\SD\Hadi\Bab I\intro1**).
7. Klik **OK**

Tampilan langkah membuat file baru ini ditunjukkan pada Gambar 2.7.

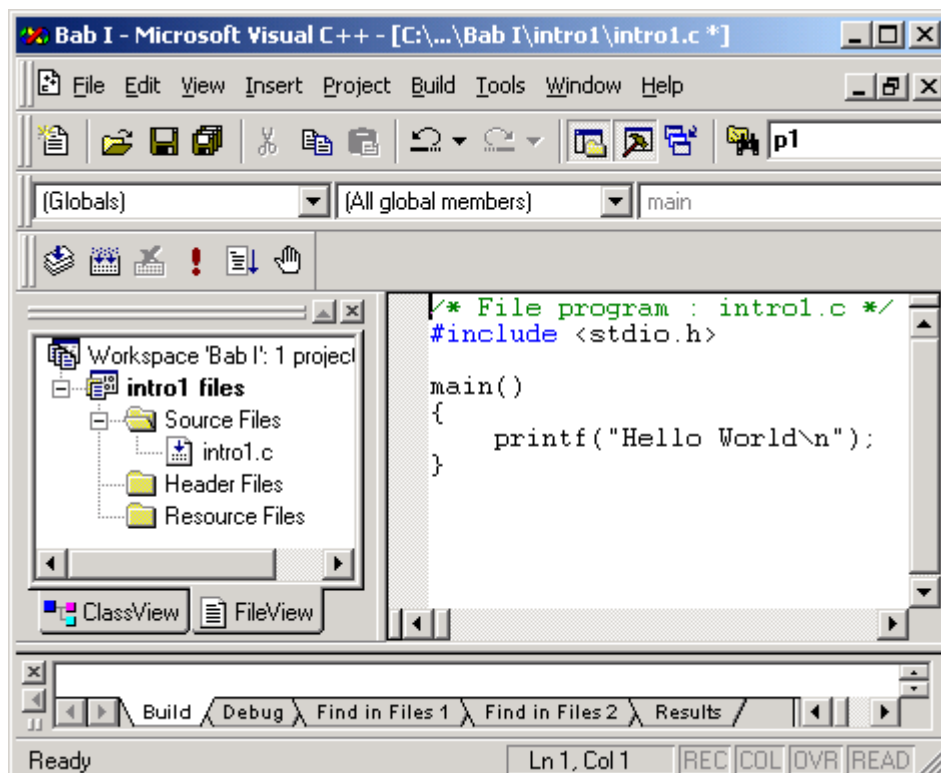


Gambar 2.7 Kotak dialog untuk New File

Hasilnya tampilan jendela workspace setelah proses pembuatan workspace, project dan file baru ditunjukkan pada Gambar 2.8. Tampilan langkah membuat program ditunjukkan pada Gambar 2.9.




Gambar 2.8 Tampilan Jendela Workspace



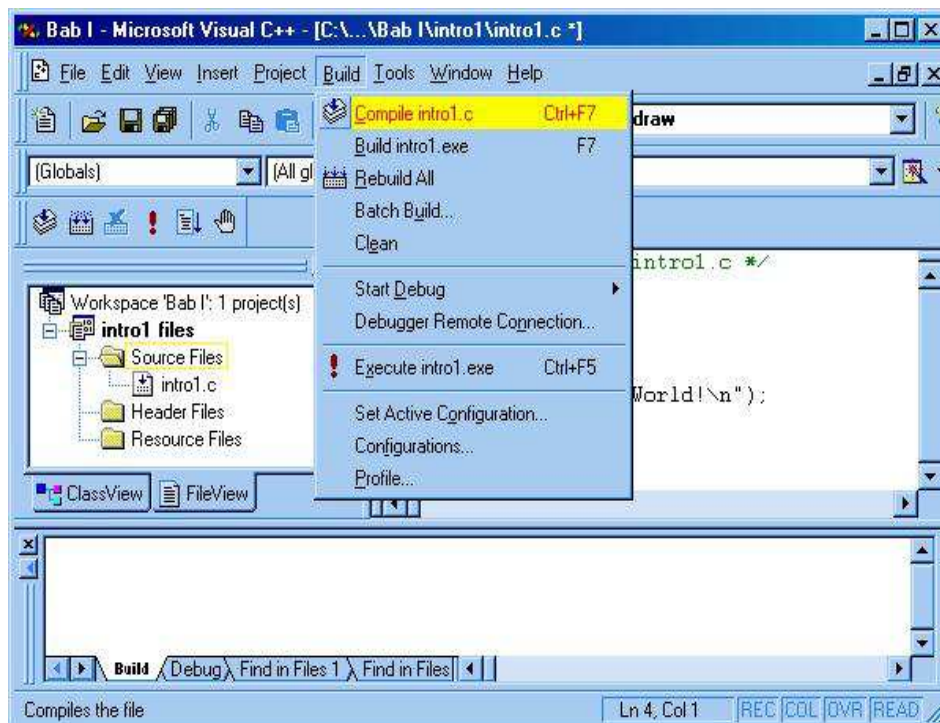
Gambar 2.9 Pembuatan Program intro1.c

Proses Kompilasi

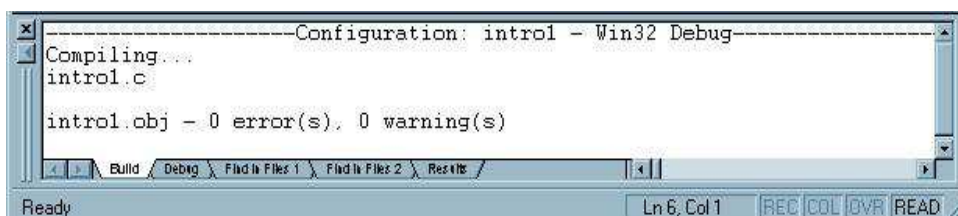
1. Klik menu **Build**
2. Klik opsi **Compile intro1.c** atau secara singkat dengan cara menekan **Ctrl+F7** atau tekan icon bergambar 

Jika program tersebut tidak mengandung kesalahan sintaks sama sekali, maka proses kompilasi akan memberikan hasil berupa pesan: **intro1.obj - 0 error(s), 0 warning(s)** .

Tampilan proses kompilasi ini ditunjukkan pada Gambar 2.10 dan tampilan pesan ketika proses kompilasi berhasil dilakukan ditunjukkan pada Gambar 2.11.



Gambar 2.10 Menu Pull-down untuk Melakukan proses Kompilasi



Gambar 2.11 Tampilan Pesan Ketika Proses Kompilasi sukses dilakukan

C. TUGAS PENDAHULUAN

Tuliskan desain algoritma dan flowchart untuk soal-soal di bawah ini :

1. Mencetak nama, kelas dan nrp masing-masing
2. Menghitung penjumlahan $1024 + 4096$ dan mencetak hasilnya
3. Mengisi nilai 2 variabel int, menjumlahkan kedua isi variabel tersebut dan mencetak hasilnya
4. Mengisi nilai sebuah variabel float, mengalikan isi variabel tersebut dengan 50 dan mencetak hasilnya
5. Menampilkan nilai sebuah bilangan float dengan tanpa menentukan format tampilannya (default)
6. Menerima masukan sebuah variabel dengan format int dan char kemudian menampilkannya kembali

D. PERCOBAAN

Implementasikan semua desain yang telah dibuat dalam tugas pendahuluan menggunakan bahasa pemrograman C

E. LAPORAN RESMI

1. Untuk setiap listing program dari percobaan-percobaan di atas, ambil *capture* outputnya.
2. Tuliskan kesimpulan dari percobaan yang telah anda lakukan.
3. Kerjakan soal-soal di bawah ini, dan sertakan jawaban Anda pada Laporan Resmi
 - a. Berapakah nilai jawaban yang ditampilkan oleh program di bawah ini :

```

main()
{
    int jawab, hasil;
    jawab = 100;
    hasil = jawab - 10;

    printf("Jawabannya adalah %d\n", hasil + 6);
}

```

b. Apakah keluaran dari potongan program di bawah ini

```

main()
{
    int value1, value2, sum;

    value1 = 35;
    value2 = 18;
    sum = value1 + value2;

    printf("The sum of %d and %d is %d\n", value1,value2,sum);
}

```

c. Program di bawah ini tidak berhasil di-compile karena masih terdapat beberapa kesalahan. Temukan paling sedikit 6 buah kesalahannya. Selanjutnya tampilkan keluaran, setelah program ini berhasil dijalankan.

```

main ()
{
    INT jumlah;

    /* PERHITUNGAN HASIL
    jumlah = 25 + 37 - 19;

    /* TAMPILKAN HASIL
    printf("Berapa hasil perhitungan 25 + 37 - 19 ?\n");
    printf("Jawabannya adalah %d\n" jumlah);
}

```

d. Buatlah program yang menerima masukan dua buah bilangan. Tampilkan keluaran berupa jumlah, rata-rata dan kuadrat dari kedua bilangan yang dimasukkan.

e. Program di bawah ini seharusnya menampilkan keluaran satu baris sbb :

```
c * c = 25,000000
```

Namun, belum berhasil karena masih ada beberapa kesalahan. Temukan minimal 3 kesalahan dalam program tersebut.

```
#include <Studio.h>
main ()
{
    float a, b, c;

    a = 3;
    b = 4.0;

    c = a * a + b * b
    printf("c * c = %d", c);
}
```