Logika Dan Algoritma Pemrograman

Logika Pemrograman merupakan sebuah cara penyelesaian suatu masalah pemrograman dengan prinsip dan aturan tertentu sehingga dapat menghasilkan output yang sesuai dengan apa yang diinginkan sedangkan algoritma pemrograman adalah urutan-urutan atau langkah-langkah sistematis yang dibuat dalam menyelesaikan masalah tersebut menggunakan aplikasi codeblocks.

Buku ini disusun dengan maksud untuk memberikan dasar pemahaman, tuntunan dan juga sebagai pedoman dasar bagi para pemula yang ingin mendalami pemrograman maupun yang masih duduk di bangku kuliah untuk memahami konsep dasarnya, dengan penjelasan yang sistematis, mudah untuk dipahami dan dilengkapi dengan latihan yang harus dikerjakan. Buku ini juga dapat dijadikan referensi serta bahan diskusi bersama untuk dapat terus diperbaiki baik secara tulisan maupun pembahasannya.

Bei Harira Irawan, S.Kom, M.M., M.Kom. atau yang akrab dengan sapaan Pak Bei, merupakan seorang dosen kelahiran Tegal, 12 Agustus 1980. Saat ini beliau mengajar di Program Studi Bisnis Digital Universitas Pancasakti Tegal. Beliau memiliki beberapa karya buku lainnya, diantaranya Konsep Dasar OOP Java, Aplikasi Absen RFID dan Penggajian VB.Net 2012, Aplikasi Bengkel Menggunakan Java Netbeans 8.2, Pemrograman Alice-3 Berbasis Objek Oriented, dan lain sebagainya.Selain aktif mengajar, Pak Bei juga seorang praktisi di bidang Komputer, Networking, dan Database. Didukung keahlian yang mumpuni dalam bidang manajemen, Pak Bei mendirikan sebuah IT Consulting yang melayani Software Development, Public Training, dan General IT Workshop yang diperuntukan bagi pelajar maupun masyarakat umum, beliau merupakan seorang founder dari HariraTREND (hariratrend.com).

Deddy Prihadi, S.E., M.Kom merupakan seorang Dosen Tetap Universitas Pancasakti Tegal yang bertugas di Fakultas Ekonomi dan Bisnis, tepatnya di Program Studi Bisnis Digital. Pengalamannya sebagai dosen tidak terbatas pada mengajar, beliau juga mendapat kepercayaan untuk mengemban amanah menjadi Asesor Talent Scouting di institusi pemerintahan. Pak Deddy, begitu panggilan akrabnya, yang lahir di Surabaya pada 1 Maret 1971, juga berkompeten di bidang Digital Marketing dan Customer Service Officer, terbukti dengan sertifikasi yang telah beliau peroleh.

UNIVERSITAS PANCASAKTI TEGAL

Jalan Halmahera KM 1 Minteragen
Kecamatan Tegal Timur, Kota Tegal, 52121
https://www.upstegal.ac.id/

JABE

ISBN 978-623-99921-4-9

9 786239 992149

Bei Harira Irawan, S.Kom, M.M., M.Kom. | Deddy Prihadi, S.E., M.Kom

Logika Dan Algoritma Pemrograman

Bei Harira Irawan, S.Kom, M.M., M.Kom.

Deddy Prihadi, S.E., M.Kom

Logika Dan Algoritma Pemrograman

Lingkup Pembahasan :

- Konsep Dasar Logika
- Instalasi Code::Blocks
- Variabel, Tipe Data dan Konstanta
- Ekspresi, Operator dan Operand
- Input / Output Pada C++
- Pencabangan Dengan IF dan Switch
- Looping, Array, Function, Parameter

Logika dan Algoritma Pemrograman

Bei Harira Irawan, S.Kom, M.M., M.Kom

Deddy Prihadi, S.E., M.Kom

Logika dan Algoritma Pemrograman

Bei Harira Irawan, S.Kom, M.M., M.Kom
Deddy Prihadi, S.E., M.Kom

Editor : Agnes Dwita Susilowati, S.E., M.Kom
Proofreader : Yustia Hapsari, S.Kom, M.Kom

Cetakan Pertama: November 2022

ISBN: 978-623-99921-4-9

PENERBIT PT JABE KREASI INDONESIA

Jalan Bima Sakti B/B-1/65 Desa Mekar Mukti
Phone: (+62) 878 2976 0975
Website: www.jabekreasi.com
E-mail: nscdocumentation@gmail.com

KATA PENGANTAR

Alhamdulillah segala puji syukur kami panjatkan kepada Allah SWT, atas segala limpahan rahmat dan berkah-Nya terutama atas terbitnya buku mengenai pemahaman dasar logika dan implementasi algoritma yang dituangkan kedalam baris-baris pemrograman menggunakan aplikasi Code Blocks. Buku yang didalamnya menjelaskan konsep-konsep dasar memahami lojik dengan disertai analogi-analogi dan bahasa yang mudah sekali dipahami.

Buku ini disusun dengan maksud untuk memberikan dasar pemahaman, tuntunan dan juga sebagai pedoman dasar bagi para pemula yang ingin mendalami pemrograman maupun yang masih duduk di bangku kuliah untuk memahami konsep dasarnya, dengan penjelasan yang sistematis, mudah untuk dipahami dan dilengkapi dengan latihan yang harus dikerjakan. Buku ini juga dapat dijadikan referensi serta bahan diskusi bersama untuk dapat terus diperbaiki baik secara tulisan maupun pembahasannya.

Terima kasih sedalam-dalamnya juga kami sampaikan kepada semua pihak terutama keluarga, rekan-rekan dan mahasiswa-mahasiswa kami yang tentunya tidak dapat disebutkan satu persatu namun atas bantuan serta peran mereka buku ini dapat diterbitkan. Alhamdulillah.

Many Happy Return.

Bei Harira Irawan
Deddy Prihadi

DAFTAR ISI

PENDAHULUAN

Sebelum memulai silahkan simak perintah-perintah berikut:

1. Pastikan kamu membaca perlahan instruksi di setiap Chapter buku ini, belajarlah untuk mulai menyimak perlahan-lahan apa yang kamu baca (banyak orang membaca tapi tidak menyimak apa yang dibacanya).
2. Download dan install aplikasi Code Blocks melalui halaman www.codeblocks.org dimana aplikasi ini akan digunakan sebagai implementasi dalam setiap latihan-latihan pada buku ini. Bahasa pemrograman yang akan digunakan untuk setiap praktek pada buku ini adalah Bahasa Pemrograman C++.
3. Setiap akhir Chapter akan berisi penugasan yang bisa kamu coba untuk kerjakan.
4. Pastikan kamu bersabar dan jangan terburu-buru untuk membaca Chapter selanjutnya sebelum kamu berusaha mencoba membuat dan mempraktekkan setiap *syntax* program pada setiap Chapter di buku ini.
5. Selamat belajar dan mencoba.

1

KONSEP DASAR LOGIKA

A. LOGIKA

Sebuah logika dalam pamahamannya adalah sarana untuk berpikir lebih sistematis dan terstruktur dalam implementasinya. Berpikir logis adalah kemampuan semua makhluk hidup sesuai dengan aturan-aturan berpikir, seperti saat akan menyeberang jalan, kita akan mulai melangkahkan kaki setelah kita rasa kendaraan mulai memiliki jarak yang jauh dengan jarak kita, atau kita tahu bahwa saat akan memarkir motor kita akan mencari tempat yang teduh agar motor kita tidak kepanasan dan masih banyak contoh lainnya. Logis dalam bahasa sehari-hari biasa disebut dengan masuk akal. Semua makhluk hidup dianugerahi Tuhan logika dan akal pikiran baik manusia maupun hewan. Seekor induk ayam yang memiliki anak-anak ayam kecil akan berfikir bahwa jika ada manusia mendekat maka induk ayam akan merasa manusia itu akan mengganggu anak-anaknya sehingga induk ayam akan bersikap protektif dan menyerang siapa saja yang mendekat. Bagi manusia, logika membantu berpikir lurus, efisien, tepat, dan teratur untuk mendapatkan kebenaran dan menghindari kekeliruan. Mendidik manusia bersikap objektif, tegas, dan berani dimana hal tersebut merupakan suatu sikap yang dibutuhkan dalam segala suasana dan tempat.

Dalam pemrograman, logika pemrograman merupakan sebuah cara penyelesaian suatu masalah pemrograman dengan prinsip dan aturan tertentu sehingga dapat menghasilkan output yang sesuai dengan apa yang kita inginkan. Logika yang digunakan dalam pemrograman merupakan kemampuan dasar wajib yang harus dikuasai seorang programmer diawal belajar sehingga akan mudah untuk memahami cara menyelesaikan masalah

pada pemrograman lanjutan lainnya yang lebih rumit. Beberapa logika dasar yang harus dipahami dalam pemrograman antara lain:

1. Logika Operasional (Aritmatika)

   Perhitungan matematika merupakan hal dasar dalam pemrograman dimana setiap pemrograman, perhitungan matematis seperti penjumlahan, pengurangan, perkalian, pembagian dan lainnya akan selalu digunakan dalam menyelesaikan masalah atau mengembangkan suatu program.

2. Logika Operasional (Perbandingan)

   Operator yang digunakan untuk membandingkan nilai dengan nilai lainnya yang hasilnya adalah TRUE atau FALSE. Operator (<) atau kurang dari akan menghasilkan TRUE jika nilai di sebelah kiri lebih kecil dari nilai di sebelah kanan. Jika tidak demikian, maka akan menghasilkan FALSE. Contoh 7 < 10 maka akan menghasilkan nilai TRUE.

3. Logika Fungsi

   Bayangkan jika kamu diminta untuk menghitung gaji karyawan lalu ada 300 orang operator dengan gaji pokok yang sama untuk dihitung gajinya. Tentunya sistem akan menghitung satu persatu gaji karyawan sebanyak 300 orang. Fungsi merupakan sebuah deretan blok kode untuk menyelesaikan sebuah task atau fitur, sehingga apabila kita membutuhkan fitur tersebut, kita dapat dengan mudah menggunakan kode fungsi yang sudah kita buat tadi. Jadi dengan memiliki 1 fungsi rumusan perhitungan gaji, maka fungsi tersebut dapat kita gunakan berulang-ulang untuk menghitung gaji 300 karyawan. Sebuah fungsi sangat dibutuhkan agar memudahkan pengaturan kode dan pelacakan permasalahan (error).

4. Logika Looping

Logika looping atau perulangan digunakan untuk menjalankan kode atau instruksi yang sama berulang-ulang hingga kondisi terpenuhi atau jika kondisi stop/berhenti tercapai. Contoh gampangnya kita akan mencetak 100 baris nama kita masing-masing. Apabila tidak menggunakan looping, maka kita akan menuliskan 100 baris nama kita dalam program. Namun bila kita menggunakan fungsi looping, maka untuk mencetak nama 100 baris bahkan 1000 baris hanya dilakukan dalam 2 atau 3 baris coding saja.

5. Logika Kondisional

   Ketika kamu mulai menulis kode, kamu pasti akan menjalankan berbagai jenis perintah dalam berbagai kondisi tertentu. Di saat seperti ini, logika kondisional adalah cara sederhana yang bisa digunakkan untuk mengecek kondisi spesifik dan menjalankan perintah berdasarkan kondisi tersebut. Contohnya apabila kondisi lampu lalu lintas berwarna merah, maka kendaraanmu harus berhenti. Apabila kondisi lampu berwarna merah berarti dalam kondisi TRUE, maka kendaraan bermotor wajib berhenti, namun apabila bukan merah (hijau), maka kendaraan boleh berjalan.

6. Logika Perbandingan

   Logika perbandingan ini biasanya digunakan untuk membuat persyaratan atas kondisi tertentu yaitu dengan cara mengukur nilai dua atau lebih variabel apakah sama, lebih kecil, lebih besar atau tidak sama. Misalnya perintah, "jika nilai angka mahasiswa 90 maka nilai abjadnya adalah A".

7. Logika Boolean

   Logika boolean dalam pemrograman digunakan untuk membuat persyaratan tertentu dengan hasil atau kondisi akhir bernilai "BENAR" atau "SALAH". Misalnya, penggunaan logika "dan" akan menghasilkan nilai "benar" jika kedua kondisi persyaratan terpenuhi.

Contohnya seseorang dapat diterima bekerja pada suatu perusahaan apabila nilai test wawancara > 80 **dan** menguasai Microsoft Office. Bila orang tersebut mendapat nilai 81 dan menguasai Microsoft Office maka dinyatakan LULUS namun bila nilai test 95 tetapi tidak menguasai Microsoft Office maka dinyatakan GAGAL. Dalam contoh ini kedua kondisi harus benar (nilai > 80 dan menguasai Microsoft Office) baru bisa dinyatakan LULUS. Apabila salah satu berkondisi tidak benar maka dinyatakan GAGAL.

Contoh lain, misalnya penggunaan logika "atau" akan menghasilkan nilai "benar" jika kedua atau salah satu kondisi persyaratannya terpenuhi. Namun sebaliknya akan menghasilkan nilai "salah" jika kedua kondisi persyaratan tidak terpenuhi. Contohnya pada saat mengerem kendaraan bermotor, motor akan berhenti apabila tuas rem kanan atau tuas rem kiri di tekan. Apa yang terjadi apabila hanya tuas rem kanan saja yang ditekan? Apakah motor tetap akan berhenti?

B. ALGORITMA

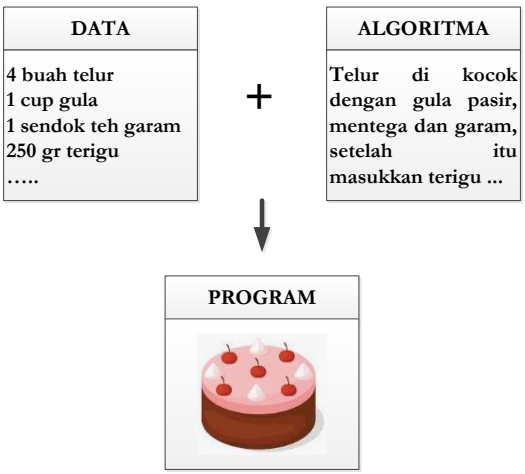
Logika pemrograman tidak dapat dipisahkan dari algoritma pemrograman. Jika logika pemrograman merupakan sebuah cara untuk menyelesaikan suatu masalah dengan prinsip dan aturan tertentu, maka algoritma pemrograman adalah urutan-urutan atau langkah-langkah sistematis yang dibuat dalam menyelesaikan masalah tersebut. Algoritma menurut (Kani, 2020, 1.19) adalah suatu upaya dengan urutan operasi yang disusun secara logis dan sistematis untuk menyelesaikan suatu masalah untuk menghasilkan suatu output tertentu. Dalam algoritma juga dikenal struktur data. Struktur data adalah suatu cara menyimpan atau merepresentasikan data di dalam komputer agar bisa dipakai secara efisien. Pemakaian struktur data yang tepat di dalam proses pemrograman akan

menghasilkan algoritma yang lebih jelas dan tepat, sehingga menjadikan program secara keseluruhan lebih efisien dan sederhana.

Komponen Penting Algoritma Pemrograman adalah sebagai berikut:

Adanya input ⇒ melakukan proses ⇒ menghasilkan output

- Input merupakan variabel atau bahan yang perlu dipersiapkan untuk dimasukkan kedalam sebuah proses agar dapat diolah.
- Proses merupakan cara atau langkah dalam mengolah variabel atau bahan tersebut untuk menghasilkan suatu output.
- Output adalah hasil yang didapat dari hasil proses pengolahan variabel atau bahan. Output yang dihasilkan haruslah merupakan solusi terhadap masalah pemrograman.



Gambar 1.1 Ilustrasi Algoritma Pemrograman

Algoritma pemrograman yang dibuat ditujukan agar dapat dibaca dan dipahami baik oleh manusia sebagai pemrogram (*programmer*) maupun oleh komputer. Terdapat 3 cara menyajikan algoritma pemrograman dalam bahasa yang mudah dan dapat dimengerti manusia.

1. Kalimat Deskriptif

   Kalimat deskriptif atau *Structured English* (SE). Kalimat ini asalnya memang menggunakan Bahasa Inggris, namun dapat dimodifikasi dengan menggunakan Bahasa Indonesia. Algoritma dengan kalimat deskriptif haruslah menggunakan bahasa yang jelas dan sederhana seperti yang biasanya kita gunakan serta mudah dicerna oleh logika. Contoh sebagai berikut:

   Luas Persegi Panjang

   Program Mencari Luas Persegi Panjang

   ALGORITMA:

   1) Masukan lebar dan panjang persegi panjang.
   2) Lakukan operasi perkalian antara panjang dan lebar persegi panjang.
   3) Hasil perkalian merupakan luas dari persegi panjang.
   4) Tampilkan luas persegi panjang.

2. Pseudocode (PC)

   Pseudocode merupakan cara penyajian algoritma pemrograman dengan menggunakan bahasa yang dibuat mirip dengan bahasa dalam sebuah program. Penyajian pseudocode biasanya lebih ringkas daripada kalimat deskriptif. Dalam menuliskan algoritma pseudocode ini dibutuhkan tiga struktur dasar, yaitu Judul, Deskripsi dan Implementasi. Contohnya sebagai berikut:

   Algoritma pseudocode untuk menentukan luas persegi panjang

   Judul : Program Menghitung_luas_persegi_panjang

   Deskripsi :

   var panjang, lebar, luas : integer;

   Implementasi :

   Read(panjang);

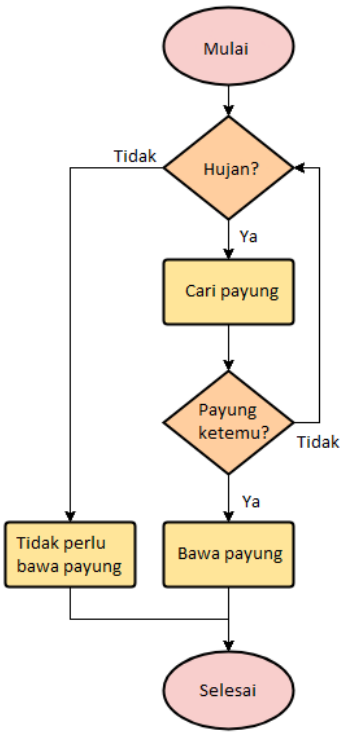
Read(lebar);

luas ← panjang * lebar;

Write (luas);

3. Flowchart (FC)

Flowchart atau disebut bagan alir adalah cara penyajian sebuah algoritma dengan menggunakan gambar skema atau bagan. Flowchart lebih banyak memuat tentang komponen yang terlibat atau peralatan komputer dalam proses pengolahan dan keterkaitannya. Dalam pembuatan flowchart terdapat simbol-simbol yang memiliki fungsi masing-masing dan tidak boleh salah penggunaanya. Sebagai contoh buatlah algoritma dalam bentuk flowchart untuk menentukan perlu membawa payung tidak saat akan berangkat kuliah.



Gambar 1.2 Algoritma Dalam Bentuk Flowchart

C. PEMROGRAMAN

Pemrograman adalah sebuah proses menulis kode, melakukan pengujian, memperbaiki (*debug*) dan memelihara kode dalam membangun suatu program komputer. Kode ini tidak hanya ditulis dalam 1 bahasa pemrograman saja, namun dapat ditulis dalam berbagai bahasa pemrograman. Tujuan dari pemrograman ini adalah untuk memuat suatu program yang dapat dieksekusi dalam melakukan suatu perhitungan atau 'pekerjaan' sesuai dengan keinginan pemrogramnya (*programmer*). Untuk melakukan pemrograman, diperlukan keterampilan dalam membaca suatu algoritma, memiliki logika pemrograman yang baik, menguasai bahasa pemrograman bahkan pada banyak kasus, pengetahuan-pengetahuan lain seperti matematika dan analisa suatu proses kejadian wajib dikuasai.

Pemrograman merupakan suatu seni dalam menggunakan satu atau lebih algoritma dan logika yang saling berhubungan dengan menggunakan suatu bahasa pemrograman tertentu sehingga terciptalah suatu program komputer. Bahasa pemrograman yang berbeda mendukung gaya pemrograman yang berbeda pula. Gaya pemrograman ini biasa disebut paradigma pemrograman.

Bahasa pemrograman komputer terdiri dari dua bagian yaitu bahasa pemrograman tingkat tinggi (*high level language*) dan bahasa pemrograman tingkat rendah (*low level language*). Semakin tinggi suatu bahasa pemrograman komputer, maka bahasa pemrograman komputer tersebut akan makin mudah dipelajari.

D. BAHASA PEMROGRAMAN C++

Sebelum bahasa pemrograman C++ dikembangkan, sebelumnya sudah ada bahasa sejenis yang digunakan programmer yaitu bahasa C yang dikembangkan oleh Dennis Ritchie dari bahasa B. Dalam pemrograman

dengan bahasa C menggunakan 2 konsep, yaitu data dan algoritma (Lihat pada Gambar 1.1).

Bahasa C++ dikembangkan oleh Bejarne Stroustrup tahun 1979 adalah nama sebuah bahasa pemrograman yang sangat terkenal. Bahasa pemrograman C++ sangat unik dibandingkan bahasa pemrograman komputer yang lain, yaitu bersifat *case sensitive* (membedakan antara huruf kecil dan huruf kapital saat dibuat), dimana umumnya hampir semua perintah dalam bahasa pemrograman C++ menggunakan huruf kecil.

E. CODE::BLOCKS

Menurut Wikipedia Code::Blocks adalah suatu program lingkungan pengembangan terpadu bebas, nirlaba, bersumber terbuka dan lintas platform. Program yang ditulis dalam C++ beserta wxWidgets untuk GUI-nya ini bisa digunakan bersama dengan berbagai macam kompilator, contohnya GCC dan Visual C++. Peralatannya yang tersedia tergantung dari "plugin" yang ada dipasang. Sekarang ini, Code::Blocks lebih tersedia sebagai perangkat pengembangan dalam bahasa C dan C++. Silahkan download dan install aplikasi Code Blocks melalui halaman www.codeblocks.org aplikasi ini akan digunakan sebagai implementasi dalam setiap latihan pada buku ini.

F. LATIHAN

Untuk lebih memantapkan pemahaman kamu mengenai konsep dasar logika dan algoritma maka silahkan kamu kerjakan latihan berikut:

1. a <= b, tentukan TRUE atau FALSE dari nilai a dan b sebagai berikut:

   a = 4, b = 3 nilainya = …

   a = 7, b = 10 nilainya = …

   a = 6, b = 6 nilainya = …

2. Diberikan algoritma : Apabila warna putih maka akan menjadi hitam. Apabila warna hitam maka jadi merah, selain warna putih dan hitam maka warna jadi biru. Jika kondisi input warna adalah coklat, maka warna akan menjadi …
3. Tulislah algoritma dalam bentuk Flowchart untuk menentukan suatu bilangan ganjil atau genap …

.

2

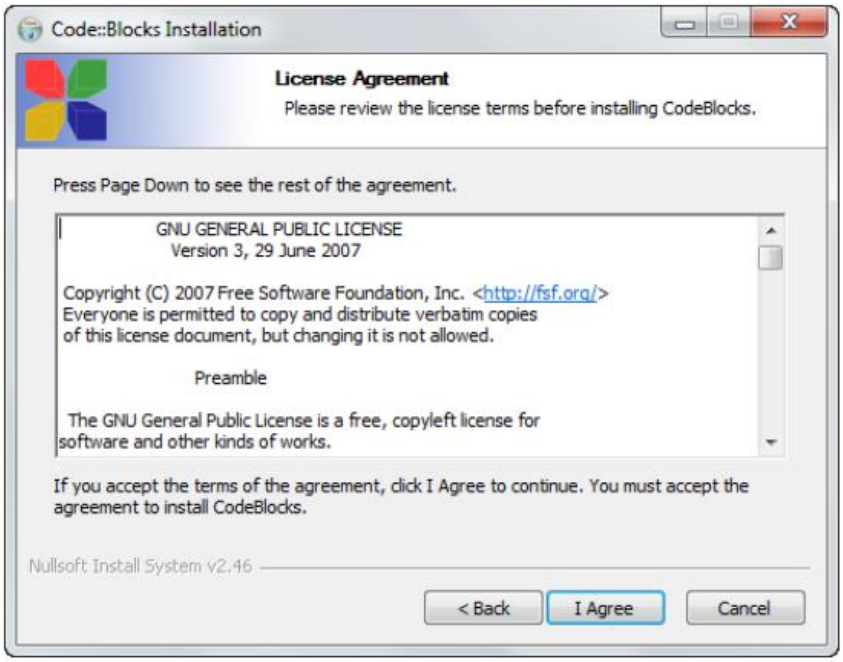
INSTALASI CODE::BLOCKS

A. INSTALASI

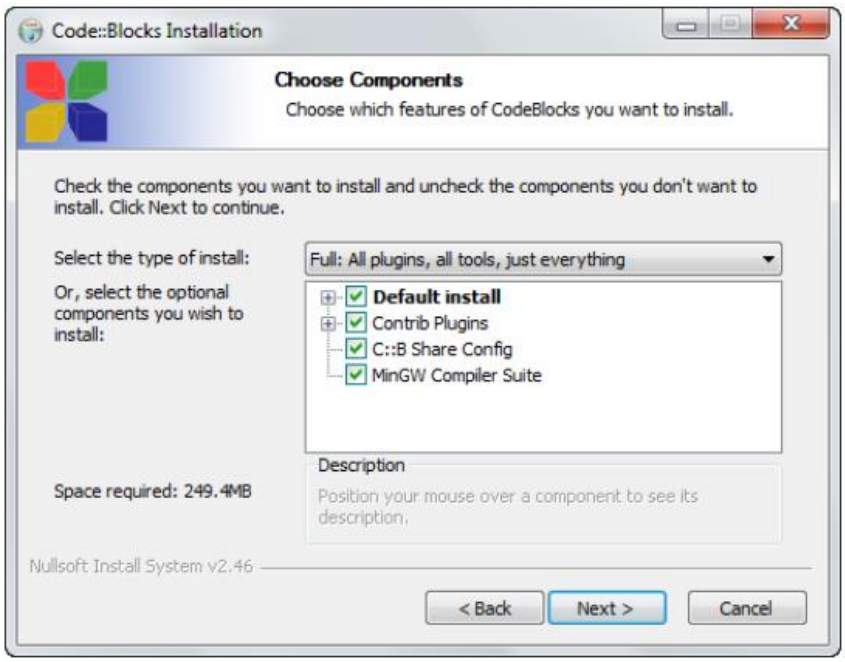
Setelah installer Code::Blocks di download silahkan lakukan instalasi pada laptop masing-masing dengan langkah sebagai berikut:

1. Klik pada codeblocks-13.12mingw-setup (versi dapat mengikuti yang telah di download oleh masing-masing). Beberapa saat kemudian akan muncul tampilan seperti berikut:

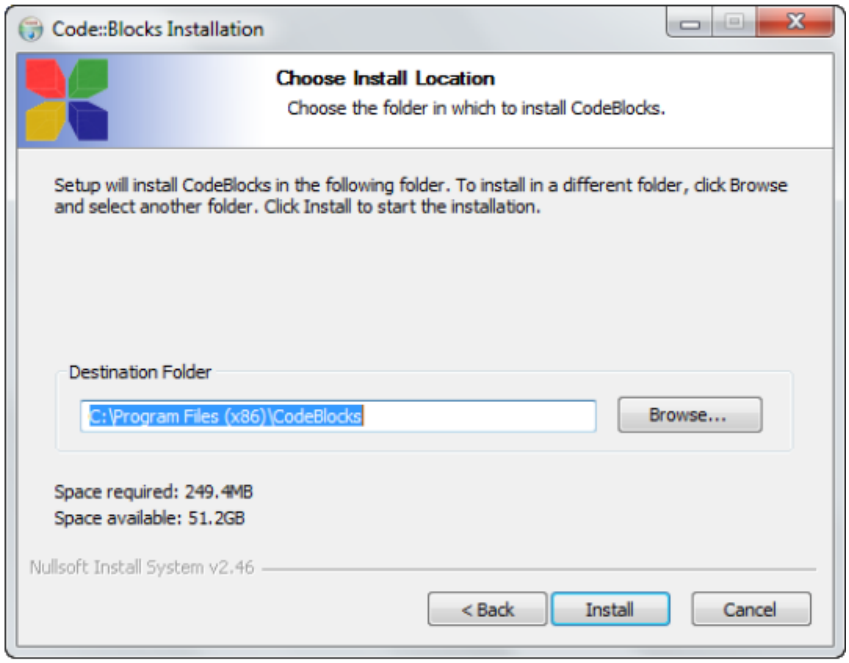
2. Klik pada tombol **NEXT** akan muncul tampilan seperti berikut:



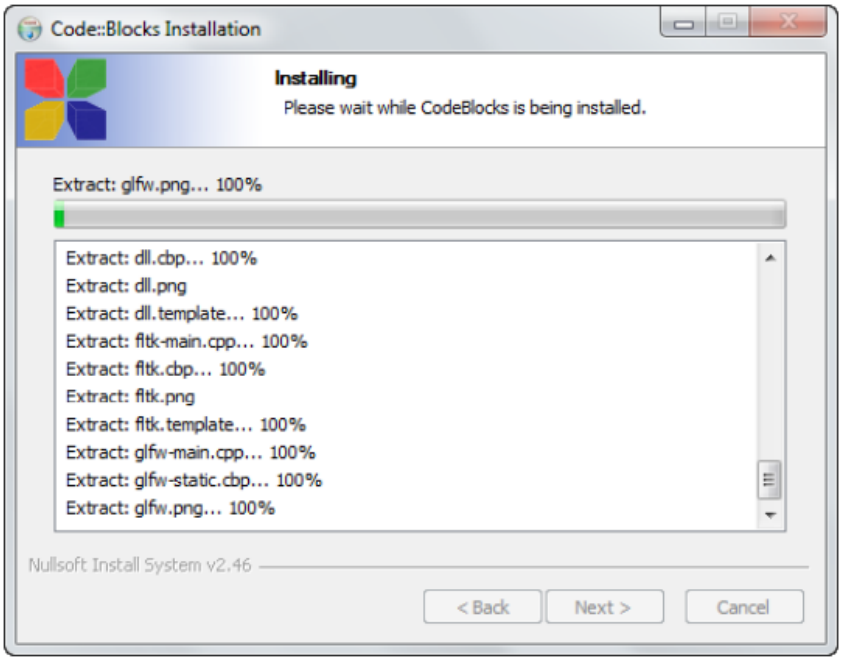
3. Klik pada tombol **I Agree**. Tampilan yang muncul seperti berikut:



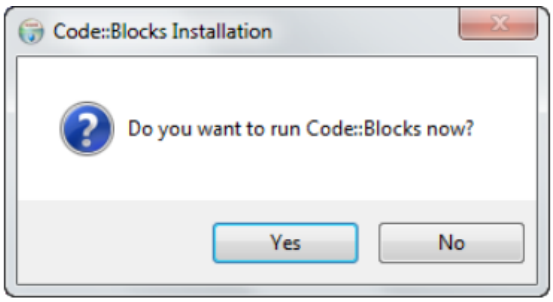
4. Klik pada tombol **NEXT** akan muncul tampilan sebagai berikut:



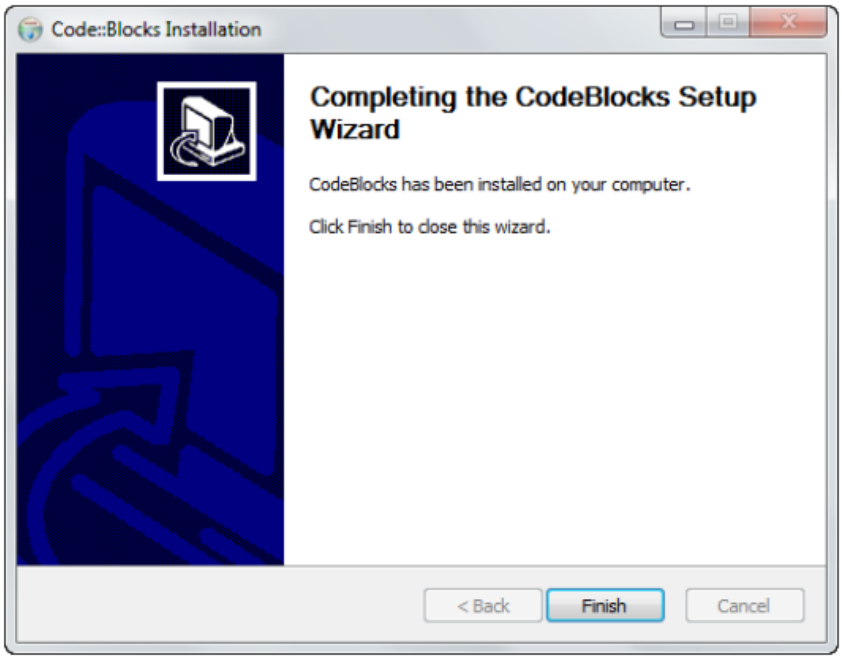
5. Klik pada tombol **INSTALL**, maka proses instalasi dimulai



6. Klik tombol **NO** apabila tampil notifikasi untuk RUN program



7. Klik tombol **NEXT** akan muncul tampilan seperti berikut:

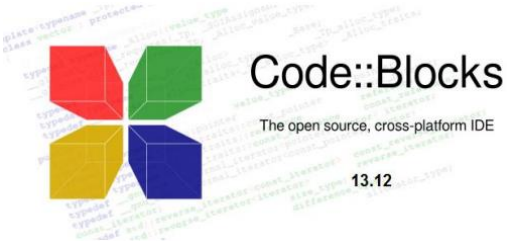


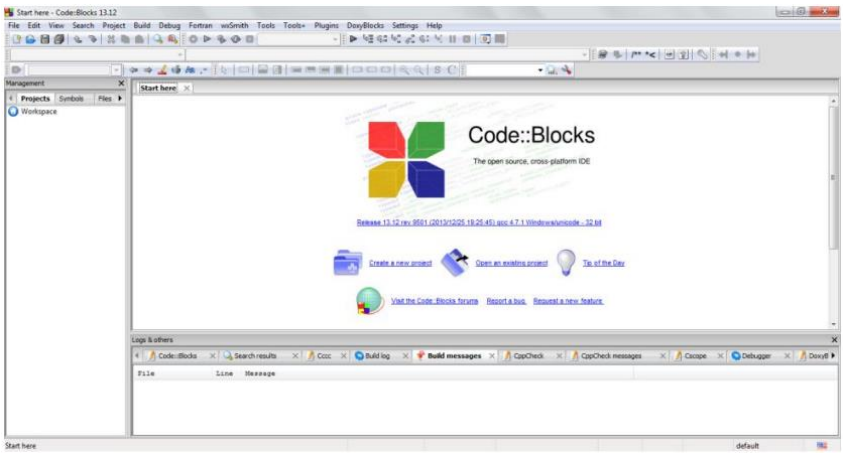
8. Klik pada tombol **FINISH** untuk mengakhiri instalasi CodeBlocks

B. LANGKAH AWAL MENJALANKAN PROGRAM

Setelah program CodeBlocks diinstall, yang pertama harus dilakukan langkah-langkahnya adalah seperti berikut:

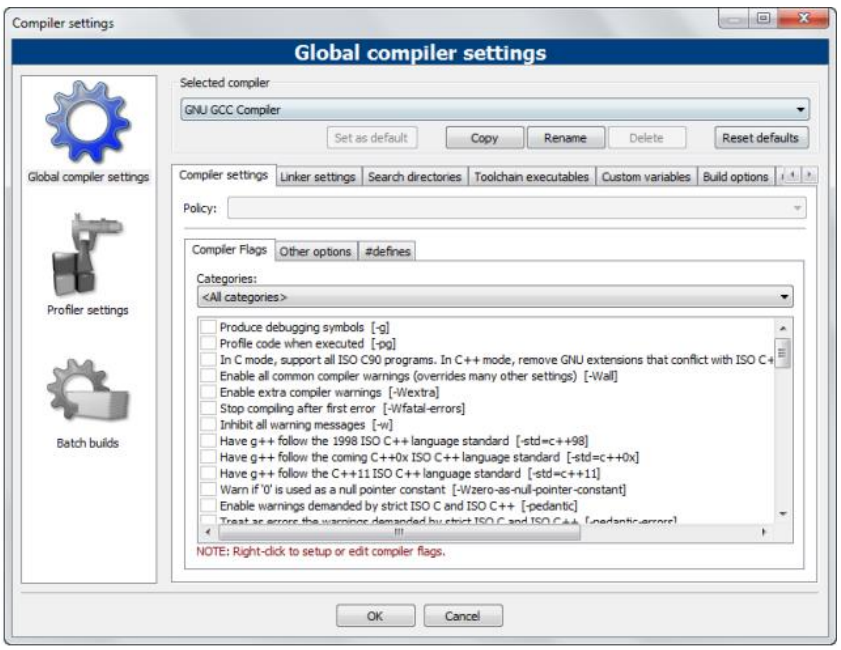
1. Klik pada **Start Windows**
2. Cari program **CodeBlocks** lalu klik program tersebut
3. Setelah di klik maka kita akan mendapat tampilan sebagai berikut



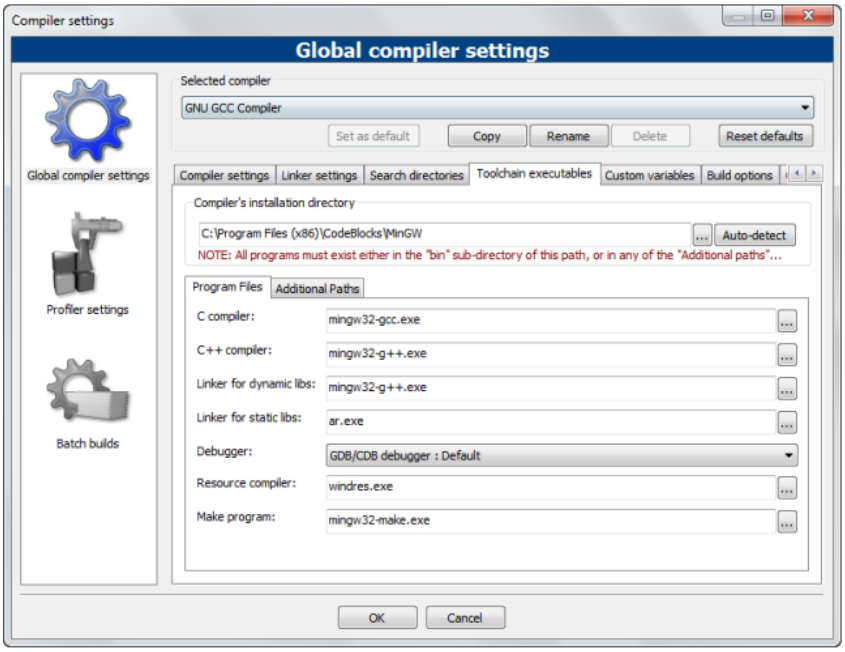


4. Klik pada **Settings**

5. Klik pada **Compiler**, akan muncul kotak dialog seperti berikut:



6. Klik pada tab **Toolchain Executables** dan klik pada **Auto Detect** untuk memastikan bahwa **C:\Program Files (x86)\CodeBlocks\MinGW** terpilih sebagai lokasi compiler



7. Klik tombol **OK**

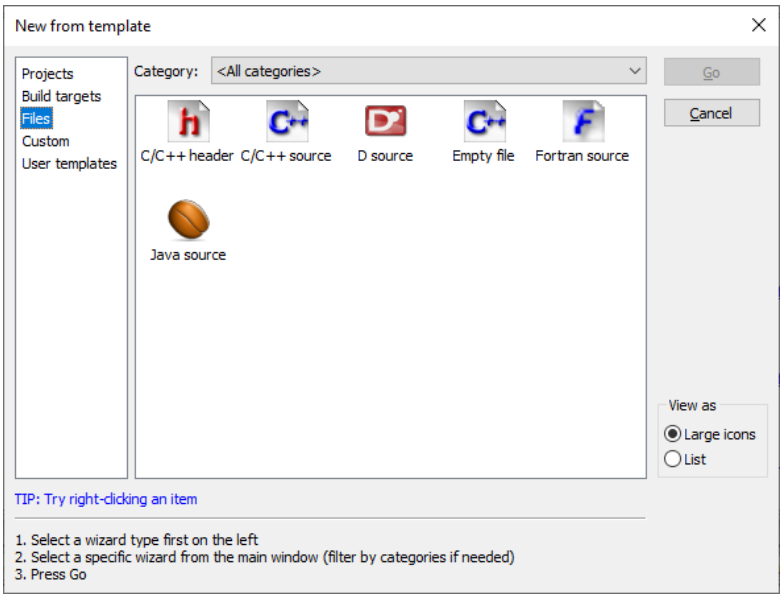
Langkah nomor 5 hingga 7 hanya dilakukan sekali saja. Selanjutnya untuk keperluan latihan dalam pemrograman menggunakan CodeBlocks ini, siapkan folder dengan nama **Pemrograman CodeBlocks** di Drive D: laptop atau komputer masing-masing. Folder tersebut nantinya akan digunakan untuk menyimpan semua data latihan pemrograman C++ menggunakan aplikasi CodeBlocks.

C. MENULIS PROGRAM C/C++

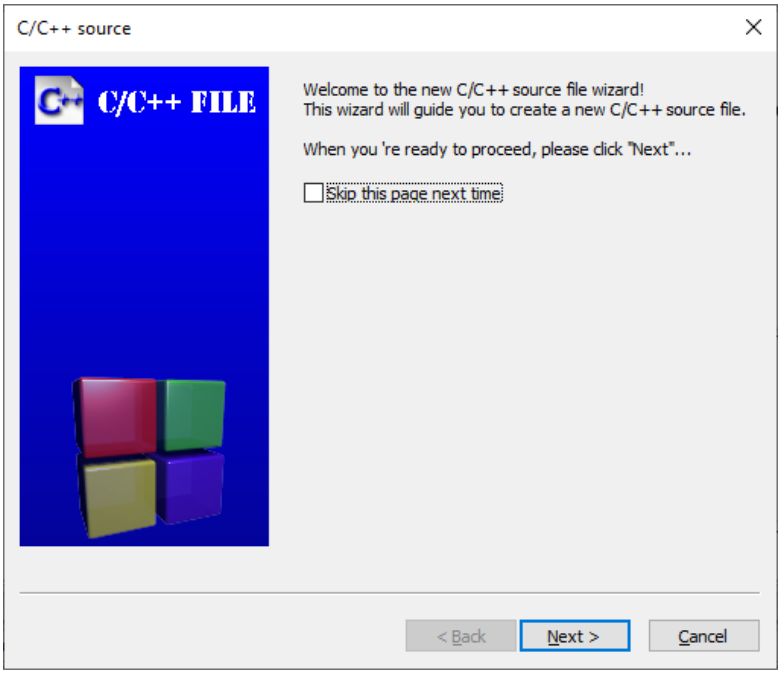
Selanjutnya kita akan mencoba belajar membuat program menggunakan bahasa pemrograman C/C++ dengan cara sebagai berikut:

1. Buka aplikasi CodeBlocks

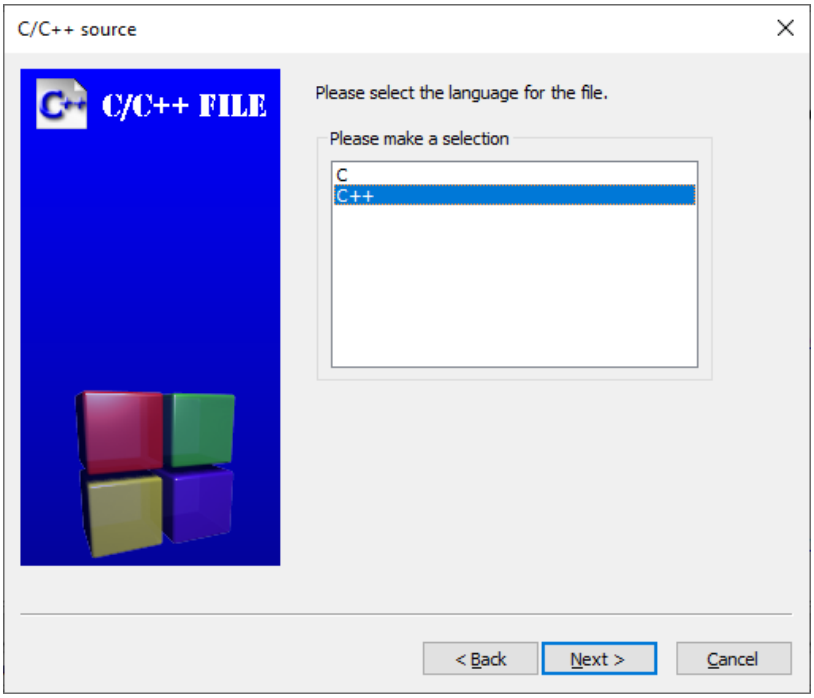
2. Klik pada menu **File** pilih **New** lalu pilih **Project**, setelah muncul jendela **New From Template,** pilih **Files** di sebelah kiri, akan muncul tampilan seperti berikut:



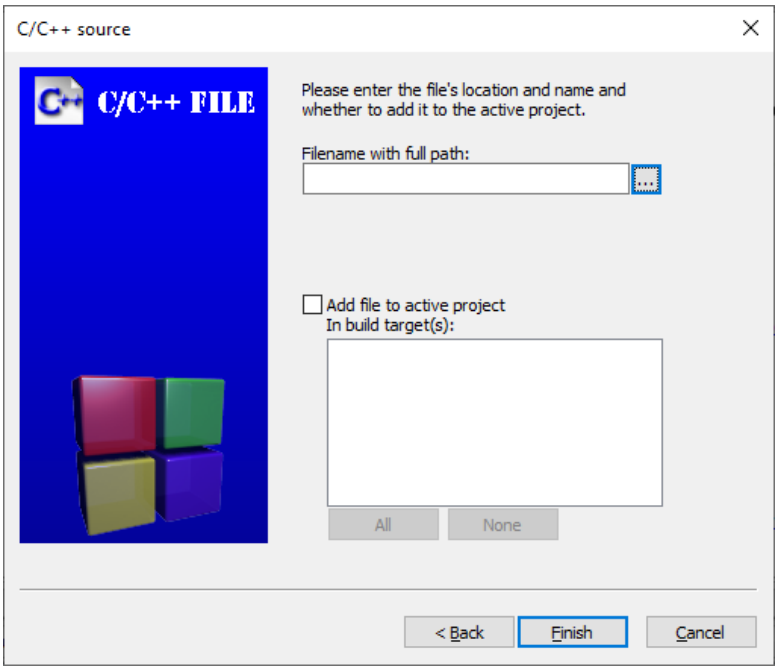
3. Klik pada **C/C++ source** dan kemudian klik pada tombol **GO**, maka akan tampil seperti berikut:



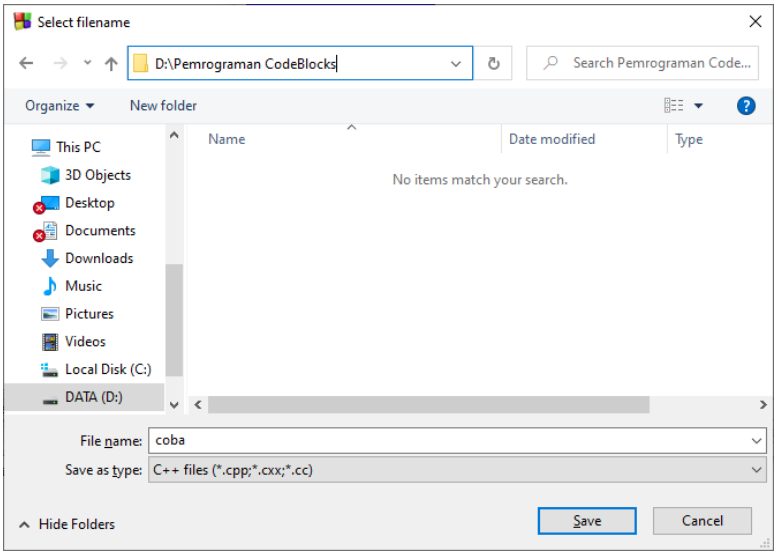
4. Klik **NEXT**, lalu pilihlah pada pilihan **C++**



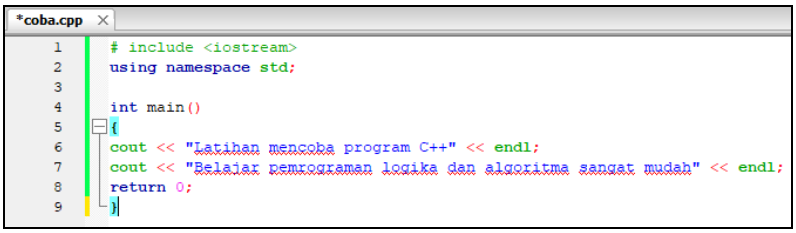
5. Setelah itu klik **NEXT** lagi, maka akan tampil seperti berikut:



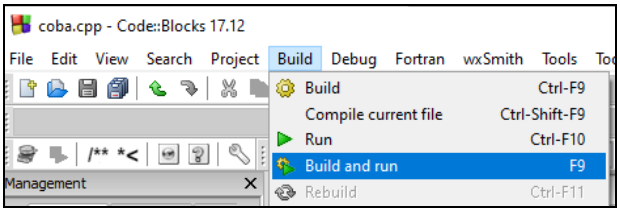
6. Pada lokasi **Filename With Full Path**, klik simbol titik tiga di samping kanan, lalu arahkan ke lokasi nama folder yang sudah kita buat (misal di lokasi **D:/Pemrograman CodeBlocks**) lalu beri nama **coba**. Gambar lengkap seperti berikut lalu klik **SAVE**



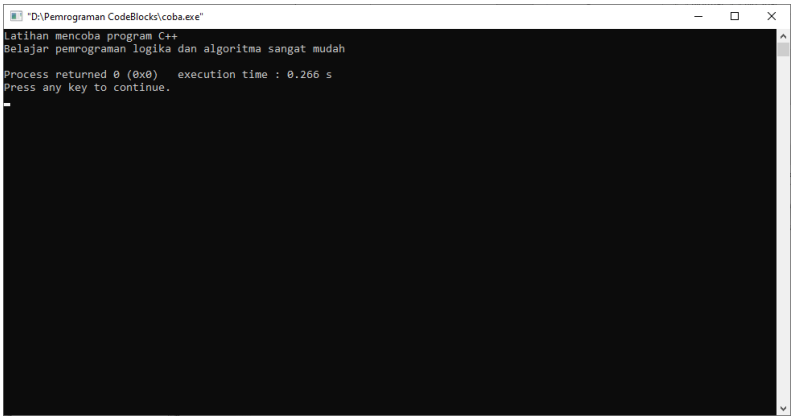
7. Klik tombol **FINISH** untuk memulai membuka jendela program C++
8. Pada tampilan program C++ tuliskan program seperti berikut (penulisan harus benar karena penulisan pada program C++ bersifat case sensistive):



9. Simpan dengan menekan tombol kombinasi **Ctrl+S** atau melalui menu **File** dan kemudian pilih **Save File**
10. Untuk melakukan kompilasi dan menjalankan program, klik pada menu **Build** dan kemudian pilihlah **Build and Run**



11. Berikut tampilan hasil kompilasi program C++ sebagai berikut



12. Untuk menutup jendela hasil kompilasi tersebut tekan sembarang tombol pada keyboard

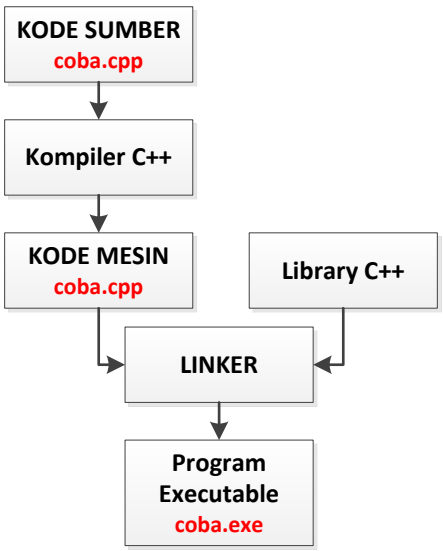
D. LATIHAN

Buatlah program menggunakan C++ untuk menampilkan data Nama, NIM dan Jurusan masing-masing.

3

HOW C++ WORKS?

Pada Chapter 2 kita sudah mencoba membuat program awal menggunakan C++. Pada saat kita mengklik **Build and Run**, kode sumber tersebut akan diterjemahkan ke bentuk **executable** melalui proses **Build** (berekstensi **.exe**) dan kemudian hasil **.exe** tersebut akan dijalankan melalui proses **Run**. Berikut gambaran mekanisme program yang sudah kita Build ini bekerja:

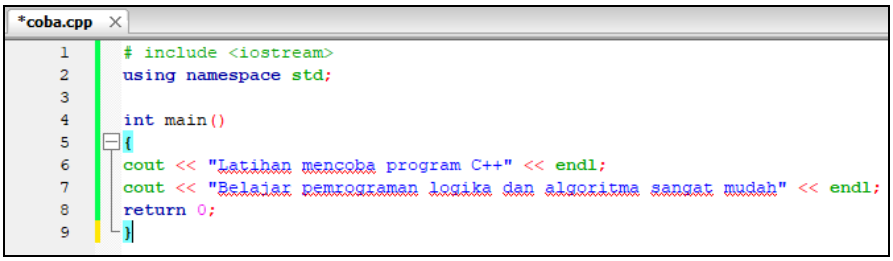


Gambar 3.1 Proses Build Pada C++

Kompiler berfungsi untuk menerjemahkan (mengkompilasi) kode sumber menjadi kode objek (kode mesin). Karena coding program yang kita input merupakan bahasa yang hanya bisa dimengerti manusia dan tidak dimengerti oleh komputer, maka agar komputer dapat memahami apa yang kita tulis pada coding perlu dilakukan kompilasi agar bahasa manusia tadi

diubah ke bahasa mesin. Linker membantu menggabungkan berbagai file objek untuk membuat file yang dapat dieksekusi. Semua file ini mungkin telah dikompilasi dengan assembler terpisah. Tugas utama linker adalah mencari modul yang dipanggil dalam program dan menemukan lokasi memori tempat semua modul disimpan.

Sebuah program C++ selalu diawali dengan **#include <iostream>**. Tanda '#' disebut *preprocessor directive*. *Preprocessor directive* adalah perintah–perintah yang diberikan kepada compiler untuk melakukan definisi, misalnya untuk memasukkan file library, dan lain sebagainya.



Jika kita lihat kembali coding program **coba.cpp** dapat dijelaskan sebagai berikut:

1) # include <iostream>
2) using namespace std;
3) int main()
4) {
5) cout << "Latihan mencoba program C++" << endl;
6) cout << "Belajar pemrograman logika dan algoritma sangat mudah" << endl;
7) return 0;
8) }

1. **Baris 1 :** Pada header program diatas terdapat **include** yang merupakan pengarah preprosesor untuk memanggil file header yang

berisi obyek bawaan dari C yang digunakan dalam program. library iostream adalah header yang dibutuhkan untuk "kegiatan" input dan output itu sendiri.

2. **Baris 2 :** Terdapat **using namespace std;** sebagai tempat mendeklarasikan obyek (fungsi dan data) dimana berfungsi agar kita tidak perlu mengetikan **std::cout** untuk mencetak output namun hanya cukup menggunakan fungsi **cout** saja, karena telah menggunakan using namespace std.
3. **Baris 3 :** Pada kode **main()** adalah blok program yang berfungsi sebagai badan sebuah alur kodingan atau disebut sebagai program utama yang ditulis antara kode dalam kurung kurawal { hingga }. Pada program C++, kedudukan main () adalah sebagai fungsi. Fungsi adalah kumpulan kode yang diperlukan sebagai kesatuan yang menjalankan tugas tertentu. Sebuah fungsi umumnya mempunyai nilai balik (*return value*). Pada program **coba.cpp** diatas, fungsi mempunyai nilai balik yang bertipe int (pada kalimat **return 0** pada baris 7).
4. **Baris 4 :** Bagian yang berawalan tanda kurawal buka { merupakan awal blok program/fungsi.
5. **Baris 5 dan 6 :** Berisi deklarasi variabel local. Pernyataan tersebut melibatkan **cout**, yang digunakan untuk menampilkan suatu nilai. Pada pernyataan yang melibatkan cout yaitu kalimat **"Latihan mencoba program C++"** adalah contoh nilai yang akan ditampilkan ke layar. Nilai tersebut ditulis dalam tanda petik ganda dan dinamakan **string**. String berarti deretan karakter. Sebuah karakter bisa berupa symbol, spasi atau huruf seperti tanda plus dan titik.

   Tanda **<<** adalah tanda yang spesifik dalam cout dimana tanda tersebut menyatakan bahwa nilai yang ada disebelah kanannya akan dikirim ke layar komputer.

Kode **endl** (berasal dari "*end line*"), yang berarti akhir baris, berfungsi untuk memindahkan kursor ke baris berikutnya atau kebagian bawah sehingga nilai yang dikeluarkan ke layar pasti akan berbeda baris dengan nilai sebelumnya yang telah ditampilkan ke layar. Simbol untuk pindah baris baru juga bisa dituliskan **\n**.

6. **Baris 7 :** Pernyataan terakhir **return 0** digunakan untuk memberitahu bahwa kode telah selesai di esekusi dengan exit code 0 dan program berakhir dengan normal. Return 0 merupakan nilai pengembalian (*return value*) terhadap fungsi.
7. **Baris 8 :** Bagian yang berakhiran tanda kurawal tutup } merupakan akhir blok program/fungsi.

Dalam C++ juga sering ditemukan tanda // (*double slash*) yang merupakan tanda penulisan komentar, selain itu C juga masih mengenal tanda komentar /*……*/. Komentar dipakai untuk memberikan penjelasan kepada pembaca kode yaitu merupakan dokumentasi, bisa berupa keterangan bagian tertentu pada kode.

Direktif **#include** digunakan untuk menambahkan pustaka (library). File *.h merupakan file header yang berisi definisi variabel, konstanta, dan fungsi untuk keperluan tertentu. Beberapa file header yang sering digunakan adalah:

1. stdio.h : pustaka standar yang berhubungan input/output
2. conio.h : pustaka operasi konsol (layar monitor dan keyboard)
3. math.h : pustaka operasi matematis
4. string.h : pustaka operasi string
5. iostream.h : pustaka operasi stream

#include harus dituliskan sebelum variabel atau konstanta yang dikandungnya digunakan dalam program. Direktif ini biasanya diletakkan di bagian awal program.

4

VARIABEL, TIPE DATA DAN KONSTANTA

A. VARIABEL DAN TIPE DATA

Variabel adalah sebuah penanda identitas yang digunakan untuk menampung suatu nilai. Variabel akan merujuk kepada sebuah alamat di memory komputer kita (RAM). Ketika kita membuat sebuah variabel, satu 'slot' memory dalam komputer kita akan disiapkan untuk menampung nilai tersebut. Setiap variabel memiliki nama yang dipakai sebagai identitas variabel.

Sesuai dengan namanya, isi variabel bisa berubah sepanjang kode program. Perhatikan contoh penggunaan variabel pada program menghitung luas persegi panjang berikut:

```
int panjang, lebar, luas;
panjang = 10;
lebar = 5;
luas = panjang * lebar;
```

panjang, lebar dan luas adalah variabel dimana variabel panjang diisi angka 10, lebar diisi angka 5 dan luas adalah perkalian panjang dikali lebar. Sedang int adalah tipe data dari variabel tersebut.

Variabel juga biasa dipakai untuk menampung nilai inputan, misalnya apabila kita ingin nilai variabel panjang dan lebar diisi oleh user (pengguna aplikasi). Berikut adalah aturan penamaan variabel dalam bahasa pemrograman C++:

1. Variabel bisa terdiri dari huruf, angka dan karakter *underscore* atau garis bawah (_).
2. Karakter pertama dari variabel hanya boleh berupa huruf dan *underscore* (_), tetapi tidak bisa berupa angka. Meskipun dibolehkan, sebaiknya hindari menggunakan karakter underscore sebagai awal dari variabel karena dapat terjadi bentrok dengan beberapa variabel settingan program.
3. Variabel harus selain dari keyword. Contohnya, kita tidak dapat memakai kata int sebagai nama variabel, karena int merupakan keyword untuk menandakan tipe data integer.
4. Beberapa compiler bahasa C++ ada yang membatasi panjang variabel maksimal 31 karakter. Sebaiknya tidak menulis nama variabel yang memiliki lebih dari 31 karakter panjangnya.

Untuk menulis variabel, dalam hampir semua bahasa pemrograman terdapat 2 langkah yaitu deklarasi dan inisialisasi. Deklarasi merupakan proses untuk memberitahu compiler C++ bahwa kita akan membuat sebuah variabel didalamnya. Bahasa C++ termasuk bahasa pemrograman yang menggunakan konsep **strongly typed programming language**, yang artinya untuk setiap variabel harus ditulis akan berisi tipe data apa. Apakah itu angka bulat (integer), angka pecahan (float/double), huruf (char), atau yang lain.

1. Tipe data integer, yaitu tipe data angka bulat seperti 1, 5 atau 1000. Tipe data ini ditulis dengan keyword **int**.
2. Tipe data double, yaitu tipe data angka pecahan seperti 1.33, 5.90 atau 1000.99. Tipe data ini ditulis dengan keyword **double**.
3. Tipe data character, yaitu tipe data huruf seperti 'A', 'a', atau 'Z'. Tipe data ini ditulis dengan keyword **char**.
4. Tipe data string, yaitu tipe data untuk menampung kumpulan karakter, seperti "Bahasa Pemrograman", "Nama saya adalah" atau

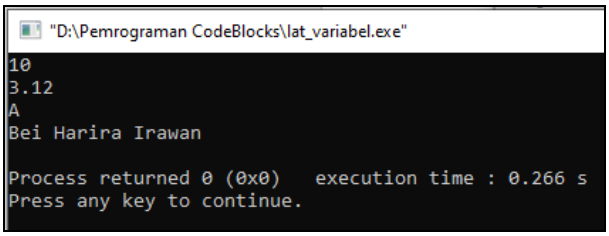
"Belajar bahasa pemrograman C++". Tipe data ini ditulis dengan keyword **string**.

B. DEKLARASI VARIABEL

Buat project baru pada CodeBlocks lalu beri nama **lat_variabel.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/lat_variabel.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main()
{
int nilai;
double ipk;
char kelas;
string nama;
        nilai = 10;
        ipk = 3.12;
        kelas = 'A';
        nama = "Bei Harira Irawan";
                cout << nilai << endl;
                cout << ipk << endl;
                cout << kelas << endl;
                cout << nama << endl;
return 0;
}
```

Hasilnya seperti berikut:



Proses deklarasi dan inisialisasi (pengisian nilai awal) juga dapat dituliskan gabung seperti berikut:

```
#include <iostream>
using namespace std;
int main()
{
int nilai = 10;
double ipk = 3.12;
char kelas = 'A';
string nama = "Bei Harira Irawan";
```

C. KONSTANTA

Konstanta adalah sebuah tempat atau sebuah penampung (*container*) dari suatu nilai. Nilai dari konstanta ini akan bersifat tetap (konstan) dan tidak bisa diubah sepanjang program berjalan. Untuk membuat konstanta dalam bahasa C++ terdapat 2 cara yaitu:

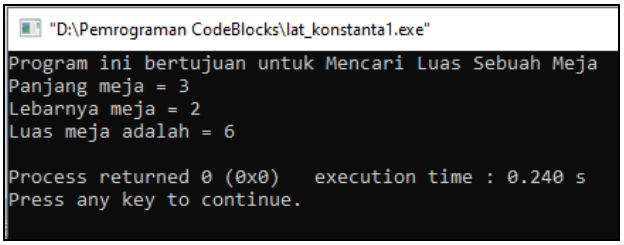
1. Menggunakan keyword **#define**
2. Menggunakan keyword **const**

Sekarang kita coba membuat konstanta menggunakan keyword #define. Buat project baru pada CodeBlocks lalu beri nama **lat_konstanta1.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/lat_konstanta1.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
    #define tujuan "Mencari Luas Sebuah Meja"
```

```
#define panjang 3
#define lebar 2
int main()
{
cout << “Program ini bertujuan untuk ” << tujuan << endl;
cout << “Panjang meja = ” << panjang << endl;
cout << “Lebarnya meja = ” << lebar << endl;
cout << “Luas meja adalah = ” << panjang * lebar << endl;
return 0;
}
```

Hasilnya seperti berikut:



Berikut contoh kode program bahasa C++ untuk membuat konstanta menggunakan keyword const. Buat project baru pada CodeBlocks lalu beri nama **lat_konstanta2.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/lat_konstanta2.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main()
{
const string nama = “Bei Harira Irawan”;
```

```
const int NIM = 212264001;
        cout << “Nama Mahasiswa : ” << nama << endl;
        cout << “NIM Mahasiswa : ” << NIM << endl;
return 0;
}
```

Hasilnya seperti berikut:

D. LATIHAN

Buatlah program menggunakan C++ untuk mengerjakan soal-soal berikut:

1. Menghitung luas sebuah segitiga dengan nilai alas = 3 dan tinggi = 5 (nama file latihan bebas).
2. Desi membeli 2 buku di toko Gramedia dengan harga buku 1 adalah Rp. 55.000,- dan buku 2 adalah Rp. 78.000,-. Saat itu sedang ada potongan harga (diskon) untuk pembelian buku sebesar 20%. Berapakah Desi harus membayar buku yang dibelinya? (nama file latihan bebas).
3. Buat program untuk flowchart berikut:


START
Pendaftaran = Rp. 100.000,-
Buku LKS = Rp. 200.000,-
SPP Juli = Rp. 150.000,-
Seragam = Rp. 600.000,-
Hitung Total Biaya
Jumlah biaya yang harus
dibayar adalah Rp.xxx,-
END

5

EKSPRESI, OPERATOR DAN OPERAND

A. EKSPRESI

Ekspresi atau disebut juga ungkapan adalah suatu rangkaian operator, variabel, fungsi atau konstanta yang ditujukan untuk menghasilkan sebuah nilai dengan tipe tertentu. Pada umumnya ekspresi dalam C++ dapat berupa:

1. Pengenal (variabel)
2. Konstanta
3. Kombinasi antara elemen di atas dengan operator

Contoh ekspresi:

```
A = 1;
B  = A + 5;
C = A * B;
```

B. OPERATOR DAN OPERAND

Operator merupakan sebuah simbol atau bentuk karakter khusus (misalnya + atau *) yang digunakan dalam suatu ekspresi untuk menghasilkan suatu nilai yang biasa dilibatkan dalam program untuk melakukan suatu operasi atau manipulasi data. Misalnya untuk:

1. Menjumlahkan dua buah nilai
2. Memberikan nilai ke suatu variabel
3. Membandingkan kesamaan dua buah nilai

Contoh pada ekspresi 5 + 6

5 dan 6 adalah sebagai operand dan tanda plus (+) adalah sebagai operatornya.

Operator dapat dikelompokkan menjadi beberapa bagian, yaitu:

1. Operator Aritmatika
2. Operator Penjumlahan dan Pengurangan
3. Operator Penugasan
4. Operator Perbandingan
5. Operator Logika
6. Operator Bit

Menurut jumlah operand yang dilibatkan, operator dapat diklasifikasikan menjadi beberapa bagian yaitu :

1. Operator unary : Operator yang melibatkan satu operand
2. Operator binary : Operator yang melibatkan dua operand
3. Operator ternary : Operator yang melibatkan tiga operand

C. OPERATOR ARITMATIKA

Operator aritmatika digunakan dalam pemrograman untuk melakukan operasi aritmatika seperti operasi penjumlahan, pengurangan, perkaliaan, pembagian dan sebagainya. Semua operator aritmatika berlaku baik untuk bilangan bulat (integer) maupun untuk bilangan real (float). Tabel berikut merangkum operator aritmatika dalam bahasa C++:

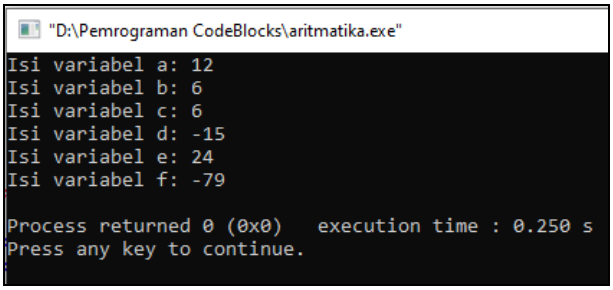
Operator	Penjelasan	Contoh
+	Penambahan	a = 5 + 2
–	Pengurangan	a = 5 – 2
*	Perkalian	a = 5 * 2
/	Pembagian (real/pecahan)	a = 5 / 2
%	Sisa hasil bagi (modulus)	a = 5 % 2

Berikut contoh penggunaan operator aritmatika menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **aritmatika.cpp** dan

letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/aritmatika.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
int a, b, c, d, e, f;
    a = 7 + 5;
    b = 8 - 2;
    c = 2 * 3;
    d = 10 + 3 - 7 * 4;
    e = ( ( 10 +3 ) -7) * 4;
    f = -79;
        cout << "Isi variabel a: " << a << endl;
        cout << "Isi variabel b: " << b << endl;
        cout << "Isi variabel c: " << c << endl;
        cout << "Isi variabel d: " << d << endl;
        cout << "Isi variabel e: " << e << endl;
        cout << "Isi variabel f: " << f << endl;
return 0; }
```

Hasilnya seperti berikut:



D. OPERATOR PENUGASAN/ASSIGNMENT

Operator assignment adalah operator untuk memasukkan suatu nilai ke dalam variabel. Dalam bahasa C++, operator assignment menggunakan tanda sama dengan (=). Pembacaan operasi assignment dilakukan dari kanan ke kiri, bukan dari kiri ke kanan.

Perhatikan pada ekspresi berikut:

a = 1000;

Berarti "masukkan nilai 1000 ke dalam variabel a". Dalam bahasa pseudocode, ini biasa ditulis dengan simbol panah ke kiri seperti berikut:

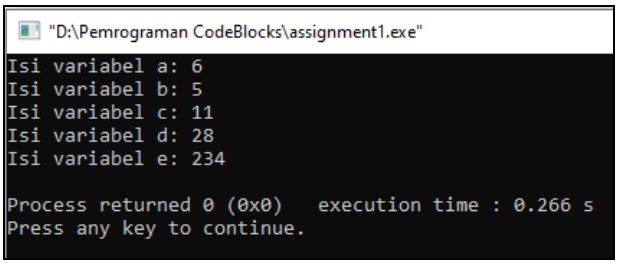
a <- 1000;

Berikut contoh penggunaan operator penugasan menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **assignment1.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/assignment1.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
int a, b, c, d, e;
    a = 5;
    b = 3;
        b = b + 1;
        c = a + b;
        d = c + c + a;
        e = (c + d) * a;
                cout << "Isi variabel a: " << a << endl;
                cout << "Isi variabel b: " << b << endl;
```

```
        cout << "Isi variabel c: " << c << endl;
        cout << "Isi variabel d: " << d << endl;
        cout << "Isi variabel e: " << e << endl;
return 0; }
```

Hasilnya seperti berikut:



Terdapat pula operator **assignment gabungan**, yaitu cara penulisan singkat operator assignment yang digabung dengan operator lain. Dalam bahasa pemrograman C++ (dan juga bahasa turunan C lain seperti PHP, dan JavaScript), operator assignment gabungan ini terdiri dari operator assignment dengan operator lain seperti **aritmatika** dan **bitwise**.

Perhatikan pada ekspresi berikut:

```
a = a + 1;
```

penulisan ekspresi tersebut bisa disingkat dan digabung menjadi:

```
a += 1;
```

Contoh lain ekspresi **b >>= 1** adalah penulisan singkat dari **b = b >> 1**. Tabel berikut merangkum operator assignment dalam bahasa C++:

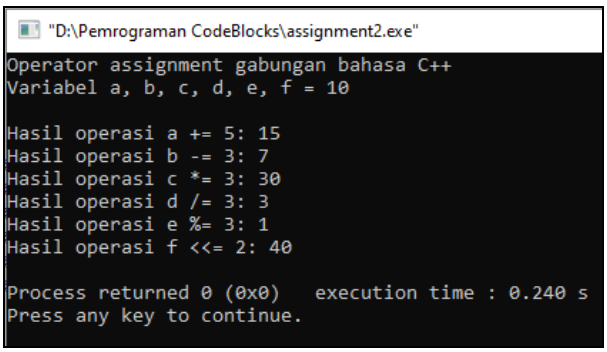
Operator	Contoh	Penjelasan
+=	a += b	a = a + b
-=	a -= b	a = a – b
*=	a *= b	a = a * b
/=	a /= b	a = a / b
%=	a %= b	a = a % b
&=	a &= b	a = a & b
|=	a |= b	a = a | b
^=	a ^= b	a = a ^ b
<<=	a <<= b	a = a << b
>>=	a >>= b	a = a >> b

Berikut contoh penggunaan operator penugasan gabungan menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **assignment2.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/assignment2.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main()
{
int a = 10, b = 10, c = 10, d = 10, e = 10, f = 10;
cout << "Operator assignment gabungan bahasa C++" << endl;
cout << "Variabel a, b, c, d, e, f = 10" << endl;
cout << endl;
    a += 5;
    b -= 3;
    c *= 3;
```

```
    d /= 3;
    e %= 3;
    f <<= 2;
        cout << "Hasil operasi a += 5: "  << a << endl;
        cout << "Hasil operasi b -= 3: "  << b << endl;
        cout << "Hasil operasi c *= 3: "  << c << endl;
        cout << "Hasil operasi d /= 3: "  << d << endl;
        cout << "Hasil operasi e %= 3: "  << e << endl;
        cout << "Hasil operasi f <<= 2: " << f << endl;
return 0;
}
```

Hasilnya seperti berikut:



E. OPERATOR PERBANDINGAN

Operator perbandingan biasanya digunakan untuk membandingkan 2 buah nilai, apakah nilai tersebut sama besar, lebih kecil atau lebih besar. Hasil dari operator perbandingan ini adalah nilai boolean true atau false. Ketika ditampilkan dengan perintah cout, true dan false ini akan ditampilkan compiler C++ sebagai bentuk integer 1 atau 0. Tabel berikut merangkum hasil dari operator perbandingan dalam bahasa C++:

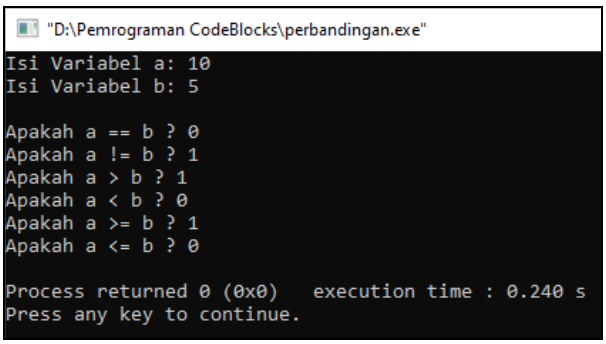
Operator	Penjelasan	Contoh	Hasil
==	Sama dengan	5 == 5	1 (true)
!=	Tidak sama dengan	5 != 5	0 (false)
>	Lebih besar	5 > 6	0 (false)
<	Lebih kecil	5 < 6	1 (true)
>=	Lebih besar atau sama dengan	5 >= 3	1 (true)
<=	Lebih kecil atau sama dengan	5 <= 5	1 (true)

Berikut contoh penggunaan operator perbandingan menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **perbandingan.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/perbandingan.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main()
{
int a = 10;
int b = 5;
bool hasil;

    cout << "Isi Variabel a: " << a << endl;
    cout << "Isi Variabel b: " << b << endl;
    cout << endl;

        hasil = a == b;
        cout << "Apakah a == b ? " << hasil << endl;
        hasil = a != b;
```

```
cout << "Apakah a != b ? " << hasil << endl;
hasil = a > b;
cout << "Apakah a > b ? " << hasil << endl;
hasil = a < b;
cout << "Apakah a < b ? " << hasil << endl;
hasil = a >= b;
cout << "Apakah a >= b ? " << hasil << endl;
hasil = a <= b;
cout << "Apakah a <= b ? " << hasil << endl;
return 0;
}
```

Hasilnya seperti berikut:



F. OPERATOR LOGIKA

Operator logika dalam pemrograman dipakai untuk menghasilkan nilai boolean true atau false dari 2 kondisi atau lebih. Tabel berikut merangkum hasil dari operator logika dalam bahasa C++:

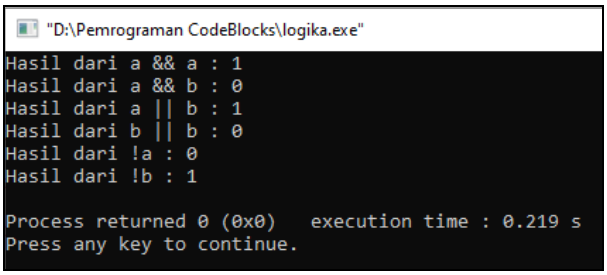
Operator	Nama	Penjelasan	Contoh
&&	And	Menghasilkan true jika kedua operand true	true && false, hasilnya: false
||	Or	Menghasilkan true jika salah satu operand true	true || false, hasilnya: false
!	Not	Menghasilkan true jika operand false	!false, hasilnya: true

Berikut contoh penggunaan operator logika menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **logika.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/logika.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main()
{
bool a = true;
bool b = false;
bool hasil;

    hasil = a && a;
    cout << "Hasil dari a && a : " << hasil << endl;
    hasil = a && b;
    cout << "Hasil dari a && b : " << hasil << endl;
    hasil = a || b;
    cout << "Hasil dari a || b : " << hasil << endl;
    hasil = b || b;
    cout << "Hasil dari b || b : " << hasil << endl;
    hasil = !a;
```

```
    cout << "Hasil dari !a : " << hasil << endl;
    hasil = !b;
    cout << "Hasil dari !b : " << hasil << endl;
  return 0; }
```

Hasilnya seperti berikut:



G. OPERATOR BITWISE

Bitwise adalah operator khusus untuk menangani operasi logika bilangan biner dalam bentuk bit. Bilangan biner merupakan jenis bilangan yang hanya terdiri dari 2 jenis angka saja yaitu 0 dan 1. Jika nilai asal yang dipakai bukan bilangan biner, akan dikonversi secara otomatis oleh compiler C++ menjadi bilangan biner. Misalnya 3 desimal = 0011 dalam bilangan biner.

Dalam penerapannya, operator bitwise tidak terlalu sering kita pakai, kecuali sedang membuat program yang harus memproses bit-bit komputer. Selain itu operator ini cukup rumit dan harus memiliki pemahaman tentang sistem bilangan biner. Tabel berikut merangkum hasil dari operator bitwise dalam bahasa C++:

Operator	Nama	Contoh	Biner	Hasil (biner)	Hasil (desimal)
&	AND	10 & 12	1010 & 1100	1000	8
|	OR	10 | 12	1010 | 1100	1110	14
^	XOR	10 ^ 1	1010 ^ 1100	0110	6
~	NOT	~10	~1010	0101	-11 (Two's complement)
<<	Left shift	10 << 1	1010 << 1	10100	20
>>	Right shift	10 >> 1	1010 >> 1	101	5

Untuk memahami bitwise perhatikan sistem biner berikut ini:

1	1	1	1	1	1	1	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1

Untuk memahami perubahan bilangan desimal ke biner perhatikan contoh berikut:

3 = 0011

8 = 1000

12 = 1100

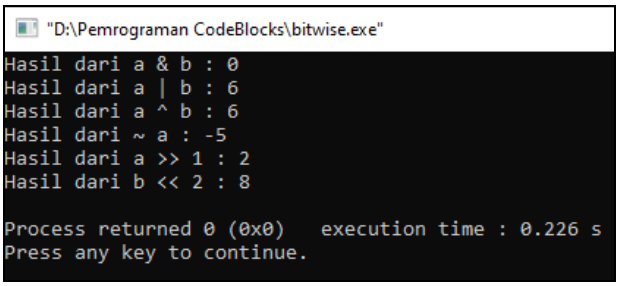
17 = 10001

Berikut contoh penggunaan operator bitwise menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **bitwise.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/bitwise.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main()
{
int a = 4;
```

```
int b = 2;
int hasil;

    hasil = a & b;
    cout << "Hasil dari a & b : " << hasil << endl;
    hasil = a | b;
    cout << "Hasil dari a | b : " << hasil << endl;
    hasil = a ^ b;
    cout << "Hasil dari a ^ b : " << hasil << endl;
    hasil = ~ a;
    cout << "Hasil dari ~ a : " << hasil << endl;
    hasil = a >> 1;
    cout << "Hasil dari a >> 1 : " << hasil << endl;
    hasil = b << 2;
    cout << "Hasil dari b << 2 : " << hasil << endl;
return 0;
}
```

Hasilnya seperti berikut:



Penjelasan dari hasil program menggunakan operator bitwise:

Nilai variabel a = 4, artinya 4 = 0100 dalam biner

Nilai variabel b = 2, artinya 2 = 0010 dalam biner

1. Pada operasi **& (AND)** terhadap kedua variabel (a & b), operasi bitwise "and" ini akan memproses bit per bit dari kedua variabel, jika kedua bit sama-sama 1, maka hasilnya juga 1, selain kondisi tersebut, nilai akhirnya adalah 0. Berikut perhitungan bitwise "and":

   a = 0100

   b = 0010

   hasil = 0000

   a & b = 0000 = 0 (desimal)

2. Pada operasi **| (OR)** terhadap kedua variabel (a | b), hasilnya akan bernilai 0 jika kedua bit bernilai 0, selain itu nilai bit akan di set menjadi 1. Berikut cara perhitungan bitwise "or":

   a = 0100

   b = 0010

   hasil = 0110

   a | b = 0110 = 6 (desimal)

3. Pada operasi **^ (XOR)** terhadap kedua variabel (a ^ b), hasilnya akan bernilai 1 jika salah satu dari kedua variabel bernilai 1 (namun tidak keduanya). Atau dengan kata lain jika kedua bit berlainan, hasilnya 1 tapi kalau sama-sama 0 atau sama-sama 1, hasilnya 0. Berikut cara perhitungan bitwise "xor":

   a = 0100

   b = 0010

   hasil = 0110

   a ^ b = 0110 = 6 (desimal)

4. Pada operasi **~ (NOT)** atau disebut juga komplemen, terhadap variabel a (~ a), hasilnya akan membalikkan nilai bit sebuah variabel dari 0 menjadi 1, dan 1 menjadi nol.. Berikut cara perhitungan bitwise "not":

   a = 0100

hasil = 1011 → biner 1 paling kiri (**1**011) merupakan komplemen negatif dimana nilainya 8. Nilai 8 ini akan dikurangi nilai bit 1 yang ada di sebelah kanannya (10**11**) yang bernilai 3. Dan karena nilainya negatif maka hasilnya 8 – 3 = -5

~ a = -5 (minus 5)

5. Pada operator **shift right (>>)** terhadap variabel a (a >> 1), dimana bahasa C++ akan menggeser posisi bit dalam variabel a ke kanan sebanyak 1 tempat. Berikut proses yang terjadi:

   a = 0100

   hasil = 0010

   a >> 1 = 0010 = 2 (desimal)

6. Pada operator **shift left (<<)** terhadap variabel b (b << 2), dimana bahasa C++ akan menggeser posisi bit dalam variabel b ke kiri sebanyak 2 tempat. Berikut proses yang terjadi:

   b = 0010

   hasil = 1000

   b << 2 = 1000 = 8 (desimal)

H. LATIHAN

Buatlah program menggunakan C++ untuk mengerjakan soal-soal berikut (nama file latihan bebas):

1. Mencari berapa nilai akhir untuk mata kuliah Logika Algoritma Pemrograman bila diketahui:

   Nilai Absen (10%) = 90

   Nilai Tugas (20%) = 100

   Nilai UTS (30%) = 80

   Nilai UAS (40%) = 90

Nilai Akhir = …

2. Diketahui variabel x dan y dengan nilai sebagai berikut, buat programnya dan lihat hasilnya:

```
x = 7;
y = 10;
y = y + 1;          hasilnya = …
z = x * 2;          hasilnya = …
n = y + (z * x);    hasilnya = …
y *= 2              hasilnya = …
n > z;              hasilnya = …
~ y;                hasilnya = …
```

3. Berapakah biner dan carilah hasil dari pernyataan bilangan berikut:

```
15          = …
23          = …
111         = …
51          = …
7 | 5       = …
~ 9         = …
5 >> 2      = …
12 & 6      = …
3 ^ 3       = …
```

6

INPUT/OUTPUT PADA C++

A. SINGLE INPUT

Perintah **cin** merupakan perintah dasar C++ untuk proses input atau menerima data masukan dari user melalui keyboard. Dengan menggunakan perintah **cin** (di eja sebagai **"see-in"**), kita bisa membuat program yang lebih interaktif, yakni meminta masukan data dari user/pengguna. Data ini bisa disimpan ke dalam variabel dan diolah lebih lanjut untuk selanjutnya akan ditampilkan kembali. Kata cin sendiri merupakan singkatan dari **console in**. Format dasar perintah cin adalah sebagai berikut:

cin >> nama_variabel;

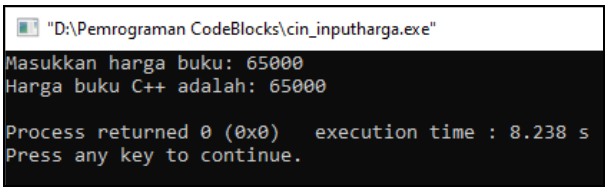
Perintah cin ditulis menggunakan tanda kurung siku kanan dua kali, atau tanda lebih besar dua kali, yaitu bentuk karakter **>>**, yang kemudian diikuti dengan nama variabel yang akan menampung nilai inputan tersebut.

Berikut contoh penggunaan fungsi cin menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **cin_inputharga.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/cin_inputharga.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
int harga;
```

```
cout << "Masukkan harga buku: ";
cin  >> harga;
cout << "Harga buku C++ adalah: " << harga << endl;
return 0; }
```

Hasilnya seperti berikut:



B. MULTIPLE INPUT

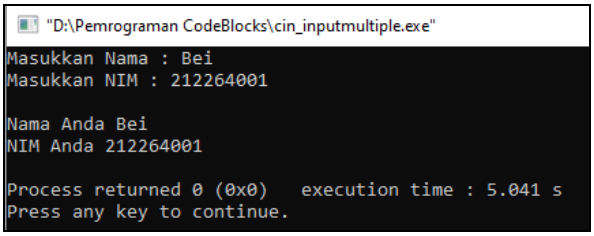
Jika kita ingin menginput lebih dari 1 data, cukup siapkan variabel untuk menampung setiap nilai inputan tersebut. Sebagai contoh kita akan membuat program yang akan meminta user untuk menginput 2 data yaitu nama dan nim. Karena nama berisi teks, maka tipe datanya adalah string sedangkan nim berisi angka, maka tipe datanya int.

Berikut contoh penggunaan fungsi cin dengan multiple input menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **cin_inputmultiple.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/cin_inputmultiple.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
string nama;
int nim;
```

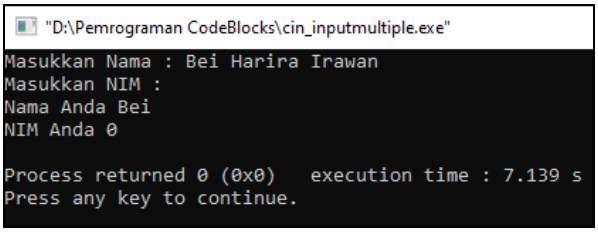
```
        cout << "Masukkan Nama : ";
        cin  >> nama;
        cout << "Masukkan NIM : ";
        cin  >> nim;
        cout << "\n";
        cout << "Nama Anda " << nama << endl;
        cout << "NIM Anda " << nim << endl;
return 0;
}
```

Hasilnya seperti berikut:



Pada hasil output diatas, nilai nama dan nim berhasil di input dan ditampilkan kembali. Namun apabila kita perhatikan terkait proses input nama, jika kita input nama yang mengandung spasi, maka hasilnya tidak sesuai dengan keinginan.

Apabila kita input nama panjang dengan spasi "Bei Harira Irawan" lalu menekan enter, program tidak lagi berhenti pada inputan nim, tetapi langsung menampilkan hasilnya. Nama "Bei Harira Irawan" hanya ditampilkan sebagai "Bei" saja, sedangkan nilai nim menjadi 0. Perhatikan hasilnya seperti berikut:



Masalah ini terjadi karena adanya keterbatasan perintah cin ketika memproses spasi. Sebagai alternatifnya, kita bisa ganti bentuk perintah cin menjadi **Function getline()**.

C. FUNCTION GETLINE()

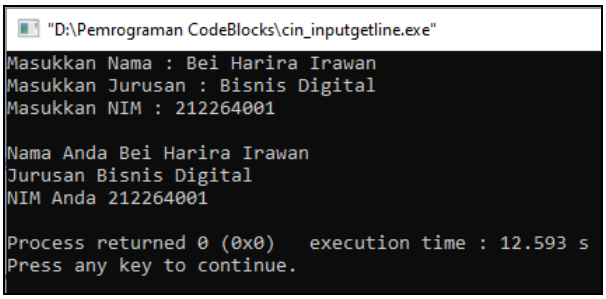
Dalam bahasa C++, perintah cin tidak bisa memproses nilai inputan teks yang diketik mengandung spasi. Untuk mengatasinya kita dapat menggunakan **Function getline()**.

Berikut contoh penggunaan fungsi cin dengan fungsi getline() menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **cin_inputgetline.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/cin_inputgetline.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
string nama, jurusan;
int nim;
        cout << "Masukkan Nama : ";
        getline(cin, nama);
        cout << "Masukkan Jurusan : ";
        getline(cin, jurusan);
```

```
        cout << "Masukkan NIM : ";
        cin  >> nim;
        cout << "\n";
        cout << "Nama Anda " << nama << endl;
        cout << "Jurusan " << jurusan << endl;
        cout << "NIM Anda " << nim << endl;
    return 0; }
```

Hasilnya seperti berikut:



D. LATIHAN

Buatlah program menggunakan C++ untuk mengerjakan soal-soal berikut (nama file latihan bebas):

1. Mencari luas persegi panjang dengan input panjang bernilai 10 dan lebar 5.
2. Mencari jumlah bayar atas pembelian buku dengan output sebagai berikut:

```
TOKO BUKU ADHIRA
Jl. Kapten Sumantri No. 45
-----------------------------------------
Sistem Basis Data       = 65000
Algoritma Dasar C++ = 55000
Total Bayar             =  120000
```

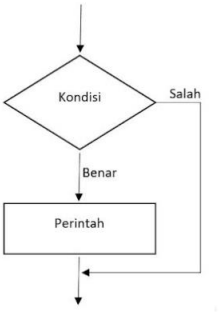
7

PENCABANGAN DENGAN IF DAN SWITCH

A. IF SATU KONDISI (IF …)

Dalam membuat sebuah program ada kalanya dibutuhkan suatu kondisi yang memerlukan pencabangan dan pengulangan. Pencabangan pada sebuah program adalah suatu pemilihan pernyataan (*statement*) yang akan dieksekusi dimana pemilihan tersebut berdasarkan pada suatu kondisi tertentu.

Pada pemrograman C++ umumnya terdapat 2 jenis struktur yang digunakan untuk proses pencabangan yaitu **"IF"** dan **"SWITCH"**. Pencabangan dengan menggunakan pernyataan if merupakan suatu pernyataan yang berguna untuk mengambil keputusan terhadap dua kemungkinan. Pada pernyataan if memiliki dua bentuk yaitu yang mengandung **ELSE** dan tidak mengandung **ELSE**. Pernyataan if satu kondisi mempunyai pengertian, "Jika kondisi bernilai benar, maka perintah akan dikerjakan dan jika kondisi bernilai salah, maka perintah akan diabaikan". Pengertian tersebut dapat dicerminkan melalui gambar berikut ini:



Gambar 7.1 Pernyataan IF Satu Kondisi

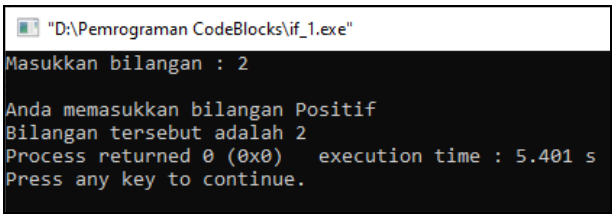
Bentuk sederhana dari pernyataan IF berupa:

```
if(kondisi)
{
    //blok pernyataan yang dijalankan
    //jika kondisi bernilai benar (true)
}
```

Berikut contoh penggunaan fungsi if dengan satu kondisi menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **if_1.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/if_1.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main()
{
int bilangan;

    cout<<"Masukkan bilangan : ";
    cin>>bilangan;

        if(bilangan > 0) {
            cout<< "\n Anda memasukkan bilangan Positif \n";
            cout<< "Bilangan tersebut adalah "<< bilangan;
        }

return 0;
}
```

Hasilnya seperti berikut:



B. IF DUA KONDISI (IF … ELSE …)

Pada dasarnya, kondisi IF ELSE adalah sebuah struktur logika program yang di dapat dengan cara menyambung beberapa perintah IF ELSE menjadi sebuah kesatuan. Jika kondisi pertama tidak terpenuhi atau bernilai false, maka kode program akan lanjut ke kondisi ELSE terakhir atau terdapat kondisi IF yang menghasilkan nilai true.

Berikut format dasar penulisan kondisi IF ELSE dalam bahasa C++:

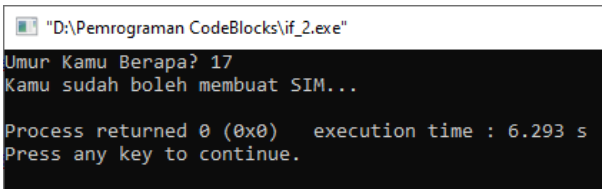
```
if (condition_1) {
  // Kode program yang dijalankan jika condition_1 berisi nilai True
} else {
  // Kode program yang dijalankan jika kondisi tidak terpenuhi
}
```

Berikut contoh penggunaan fungsi if….else menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **if_2.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/if_2.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
#include <iostream>
using namespace std;
int main(){
```

```
    int umur;
    cout << "Umur Kamu Berapa? ";
    cin >> umur;

    if (umur >= 17){
        cout << "Kamu sudah boleh membuat SIM..." << endl;
    } else {
        cout << "Masih dibawah umur..." << endl;
    }
return 0;
}
```

Hasilnya seperti berikut:

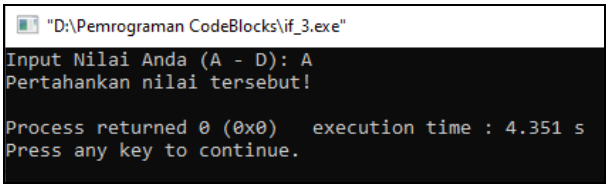


Berikut contoh penggunaan fungsi if…else if menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **if_3.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/if_3.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
char nilai;
    cout << "Input Nilai Anda (A - D): ";
    cin >> nilai;
```

```
        if (nilai == 'A' ) {
            cout << "Pertahankan nilai tersebut!" << endl;
        } else if (nilai == 'B' ) {
            cout << "Harus lebih baik lagi!" << endl;
        } else if (nilai == 'C' ) {
            cout << "Perbanyak lagi belajar!" << endl;
        } else if (nilai == 'D' ) {
            cout << "Kamu harus mengulang!" << endl;
        } else {
            cout << "Maaf, format nilai tidak sesuai" << endl;
        }
    return 0; }
```

Hasilnya seperti berikut:



C. IF BERSARANG (NESTED IF)

Pada kondisi IF bersarang adalah sebuah struktur logika program yang di dapat dengan cara menyambung beberapa perintah IF menjadi sebuah kesatuan. Jika kondisi pertama tidak terpenuhi atau bernilai true, maka kondisi akan di cek lagi menggunakan kondisi IF yang lain. Jika ternyata tidak juga terpenuhi, akan lanjut ke blok kondisi ELSE terakhir.

Berikut format dasar penulisan kondisi IF bersarang dalam bahasa C++:

```
if (condition_1) {
    if (condition_2) {
        // Mengecek kondisi apabila kondisi 1 sebelumnya true
```

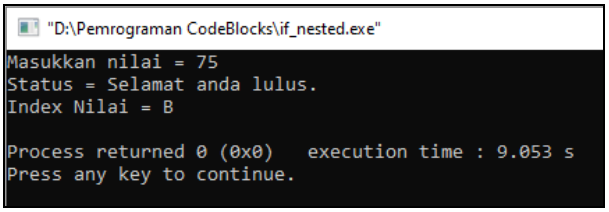
```
        } else  {
                // Bila kondisi 2 tidak terpenuhi
        }
    } else {
      // Kode program yang dijalankan jika semua kondisi tidak terpenuhi
    }
```

Berikut contoh penggunaan fungsi if bersarang menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **if_nested.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/if_nested.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
#include <string>
using namespace std;
int main() {
int nilai;
string index, ket;
    cout << "Masukkan nilai = ";
    cin  >> nilai;

    if (nilai >= 60) {
        ket = "Selamat anda lulus.";
    if (nilai >= 80) {
                index = "A";
        } else if (nilai >= 70) {
                index = "B";
        } else {
                index = "C";
```

```
        }
    }

      else {
        ket = "Maaf, anda belum lulus.";
          if (nilai >= 40) {
              index = "D";
        } else {
              index = "E";
        }
    }

        cout << "Status = " << ket << endl;
        cout << "Index Nilai = " << index << endl;
return 0; }
```

Hasilnya seperti berikut:



D. PERNYATAAN SWITCH (SWITCH … CASE)

Selain menggunakan struktur if, pada pemrograman C++ juga menawarkan penggunaan pencabangan atau pemilihan dengan menggunakan statement switch. Perintah switch memungkinkan kita membuat kondisi yang dapat melakukan sejumlah tindakan berbeda

terhadap sejumlah kemungkinan nilai. Bentuk umum percabangan dalam bentuk switch adalah sebagai berikut :

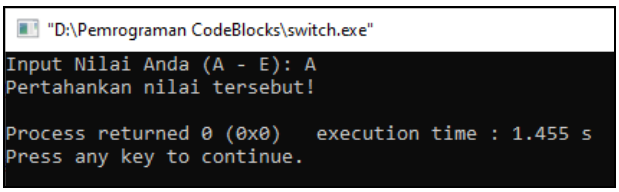
```
switch (ekpresi)
    {
        case nilai_pertama:
        pernyataan_pertama;
        break;
        case nilai_kedua:
        pernyataan_kedua;
        break;
        case nilai_keempat:
        pernyataan_ketiga;
        break;
        ……..
        Default: pernyataan_n;
    }
```

Berikut contoh penggunaan fungsi switch menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **switch.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/switch.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
char nilai;
    cout << "Input Nilai Anda (A - E): ";
    cin >> nilai;
    switch (nilai) {
```

```
            case 'A':
                cout << "Pertahankan nilai tersebut!" << endl;
            break;
            case 'B':
                cout << "Harus lebih baik lagi!" << endl;
            break;
            case 'C':
                cout << "Perbanyak lagi belajar!" << endl;
            break;
            case 'D':
                cout << "Kamu harus mengulang!" << endl;
            break;
            default:
                cout << "Maaf, format nilai tidak sesuai" << endl;
    }
    return 0; }
```

Hasilnya seperti berikut:



E. LATIHAN

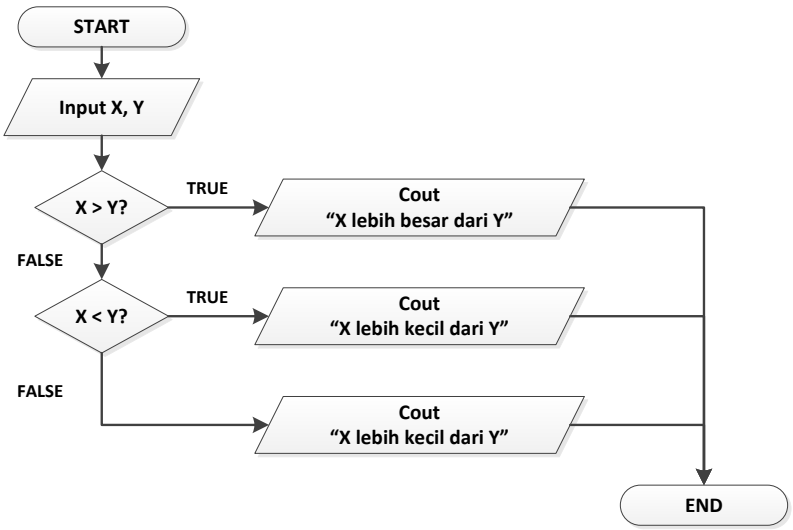
Buatlah program menggunakan C++ untuk mengerjakan soal-soal berikut (nama file latihan bebas):

1. Buat program untuk menentukan jenis kelamin masing-masing (kalimat output bebas).

2. Buat program untuk memilih jenis makanan pada kedai kopi sebagai berikut (kalimat output bebas):

No	Menu	Harga
1	Hot Coffe Americano	18000
2	Hot Latte	20000
3	Cappucino	20000
4	Dolce Coffe	22000

3. Buatlah program dari flowchart berikut:



4. Buatlah program menggunakan nested if untuk menentukan diterima tidaknya bekerja di Bank berdasarkan nilai IPA minimal 80 dan tinggi badan minimal 170 cm, bila memenuhi akan LULUS, bila tidak memenuhi TIDAK LULUS dan bila salah input akan tampil notifikasi INPUT TIDAK SESUAI (kalimat output bebas).

8

LOOPING

Apa yang akan dilakukan saat diminta untuk mencetak kalimat "Logika dan Algoritma Pemrograman" sebanyak 10x ke layar? Mungkin kita akan menggunakan perintah cout sebanyak 10 kali. Namun bagaimana bila diminta untuk 500 baris? Apakah harus menggunakan perintah cout sebanyak 500 kali juga?. Hal ini tentunya sangat tidak efektif dilakukan.

Pengulangan atau looping adalah suatu tindakan untuk melakukan hal yang sama berulang kali. Untuk mendukung penulisan kode untuk proses pengulangan pada C++ dapat dilakukan dengan menggunakan pernyataan-pernyataan berikut:

1. While

   While digunakan untuk mengulang suatu proses yang belum diketahui jumlahnya. Pengecekan kondisi akan dilakukan terlebih dahulu sebelumnya. Jika kondisi masih bernilai true, maka looping akan terus berlanjut untuk di proses. Statement WHILE juga digunakan untuk menyatakan perulangan.

2. Do While

   Sama seperti while, melakukan perulangan walaupun belum diketahui jumlahnya. Instruksi akan dijalankan lebih dahulu, kemudian dilakukan pengecekan kondisi apabila masih bernilai true maka looping akan terus berlanjut.

3. For

   For digunakan untuk mengulang suatu proses yang telah diketahui jumlahnya.

A. PENGULANGAN WHILE

Seperti dijelaskan diatas, while digunakan untuk mengulang suatu proses yang belum diketahui jumlahnya. Pengecekan kondisi akan dilakukan terlebih dahulu. Jika kondisi masih bernilai true, maka looping akan terus berlanjut. Pernyataan untuk menangani proses pengulangan dengan menggunakan pernyataan while dibentuk dengan format:

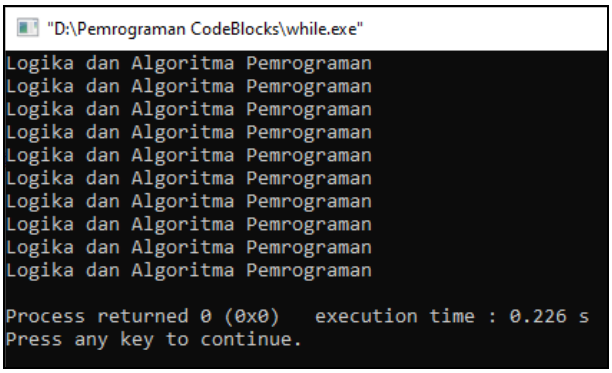
```
while(kondisi)
{
// blok pernyataan
}
```

Dalam hal ini blok pernyataan akan dijalankan secara terus menerus selama kondisi bernilai benar (true). Jika pada keadaan awal, kondisi bernilai salah maka blok pernyataan tidak dapat dijalankan sama sekali.

Berikut contoh penggunaan looping while menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **while.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/while.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
int i = 1;
        while (i <= 10) {
        cout << "Logika dan Algoritma Pemrograman" << endl;
        i++;
        }
return 0; }
```

Hasilnya seperti berikut:



B. PENGULANGAN DO … WHILE

Perulangan DO WHILE merupakan modifikasi dari perulangan statement WHILE, yaitu dengan cara memindahkan posisi pemeriksaan kondisi ke akhir perulangan. Prosesnya adalah lakukan dahulu sebuah perulangan, baru periksa apakah kondisi variabel counter sudah terpenuhi atau belum di akhir perulangan. Berikut format dasar struktur perulangan DO WHILE dalam bahasa C++:

```
start;
do
{
 // kode program
 // kode program
 increment;
}
while (condition)
```

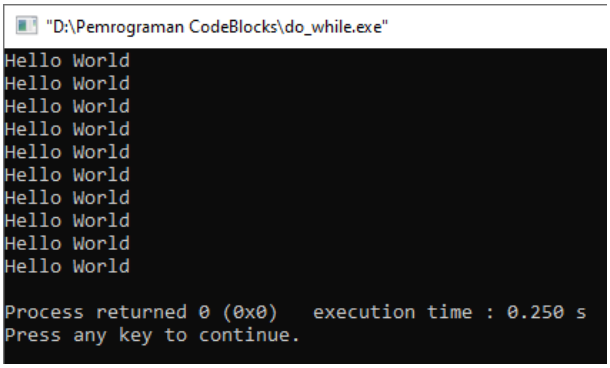
Sama seperti perulangan WHILE, pada bagian start biasanya terdapat perintah inisialisasi variabel counter, misalnya i = 0. Kemudian di dalam blok **do** akan ditulis kode program yang akan di ulang, tambahkan pula

sebuah perintah untuk menaikkan nilai variabel counter, misalnya dengan perintah i++. Di bagian paling bawah, terdapat perintah while (*condition*). Pada perintah inilah kondisi perulangan akan diperiksa. Selama kondisi ini menghasilkan nilai true, maka perulangan akan lanjut ke iterasi atau perulangan berikutnya.

Berikut contoh penggunaan looping do…while menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **do_while.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/do_while.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
int i = 1;
        do {
           cout << "Hello World" << endl;
           i++;
        }
        while (i <= 10);
return 0; }
```

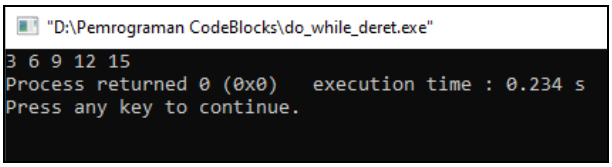
Hasilnya seperti berikut:



Dalam perulangan DO WHILE kita juga bisa mengakses variabel counter. Berikut contoh penggunaan looping do…while dengan mengakses variabel counter menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **do_while_deret.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/do_while_deret.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
int i = 1;
        do {
        cout << i * 3 << " ";
        i++;
        }
        while (i <= 5);
return 0; }
```

Hasilnya seperti berikut:



C. PENGULANGAN FOR

Pengulangan for biasanya menggunakan suatu variabel untuk mengendalikan berapa kali tubuh loop akan dieksekusi dan menentukan kapan loop akan berhenti. Variabel ini disebut juga dengan variabel kontrol. Nilai inisialisasi awal merupakan variabel kontrol, proses inisialisasi nilai awal hanya akan dilakukan sekali saja.

Dalam merancang perulangan for ini kita harus mengetahui 3 komponen yaitu:

1. Kondisi awal perulangan.
2. Kondisi pada saat perulangan.
3. Kondisi yang harus dipenuhi agar perulangan berhenti.

Berikut format dasar struktur perulangan for dalam bahasa C++:

```
for (start; condition; increment)
{
  // kode program
  // kode program
}
```

Start adalah kondisi pada saat awal perulangan. Biasanya kondisi awal ini berisi perintah untuk memberikan nilai kepada variabel counter. Variabel counter sendiri adalah sebuah variabel yang menentukan berapa banyak perulangan dilakukan. Kebanyakan variabel counter ini ditulis menggunakan

variabel **i** (huruf i) sebagai variabel counter walaupun kita boleh juga menggunakan variabel lain selain huruf i.

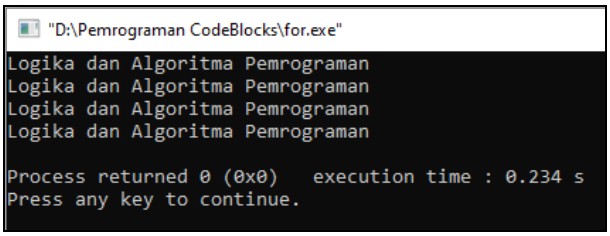
Condition adalah kondisi yang harus dipenuhi agar perulangan dapat berjalan. Selama kondisi ini terus terpenuhi, maka compiler bahasa C++ akan terus melakukan proses perulangan. Misalnya condition ini berisi perintah i < 7, maka selama variabel counter i berisi angka yang kurang dari 7, maka bahasa C++ akan terus melakukan perulangan.

Increment adalah bagian yang dipakai untuk memproses variabel counter agar bisa memenuhi kondisi akhir perulangan. Bagian ini akan selalu di eksekusi di setiap perulangan. Pada bagian increment biasanya berisi operasi increment seperti **i++**, yang sebenarnya sama dengan **i = i + 1**. Maksudnya adalah dalam setiap proses perulangan, naikkan variabel i sebanyak 1 angka. Namun kita juga bisa memberikan nilai lain seperti i = i + 2 sehingga variabel counter akan naik 2 angka setiap perulangan. Sebagai tambahan, terdapat istilah iterasi (*iteration*), yang berarti 1 kali perulangan.

Berikut contoh penggunaan for menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **for.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/for.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
int i;
        for (i = 1; i < 5; i++) {
        cout << "Logika dan Algoritma Pemrograman " << endl;
        }
return 0; }
```

Hasilnya seperti berikut:



D. LATIHAN

Buatlah program menggunakan C++ untuk mengerjakan soal-soal berikut (nama file latihan bebas):

1. Buat program untuk membuat perulangan mundur dari 10 ke 1.
2. Buat program menampilkan bilangan genap dari 1 sampai 20.

9

ARRAY

A. ARRAY 1 DIMENSI

Array adalah deretan variabel yang berjenis sama dan mempunyai nama sama. Anggota atau isi dari array itu sendiri harus satu jenis tipe data, misalkan terdiri dari kumpulan angka bulat saja (integer), kumpulan karakter saja (char), maupun kumpulan angka pecahan saja (double). Untuk mendeklarasikan array dalam bahasa pemrograman C++ dapat dilakukan dengan menggunakan tanda **[ ] (bracket)**. Bentuk umum pendeklarasian array adalah sebagai berikut:

tipe_data nama_array [jumlah_elemen_array];

contoh:

int ArrayLarik[10];

Sebagai contoh, misalkan pada sebuah program kita ingin menyimpan 5 buah nilai siswa. Apabila dilakukan tanpa array, maka kita harus menyiapkan 5 buah variabel sebagai berikut:

int nil1, nil2, nil3, nil4, nil5;

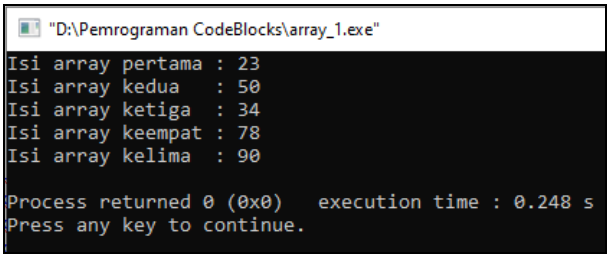
Jika menggunakan array, maka pendefinisian variabel cukup dibuat seperti berikut:

int nil[5];

Berikut contoh penggunaan array 1 dimensi menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **array_1.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/array_1.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
int nilai[5];
        nilai[0] = 23;
        nilai[1] = 50;
        nilai[2] = 34;
        nilai[3] = 78;
        nilai[4] = 90;
                cout << "Isi array pertama : " << nilai[0] <<endl;
                cout << "Isi array kedua   : " << nilai[1] <<endl;
                cout << "Isi array ketiga  : " << nilai[2] <<endl;
                cout << "Isi array keempat : " << nilai[3] <<endl;
                cout << "Isi array kelima  : " << nilai[4] <<endl;
return 0; }
```

Hasilnya seperti berikut:



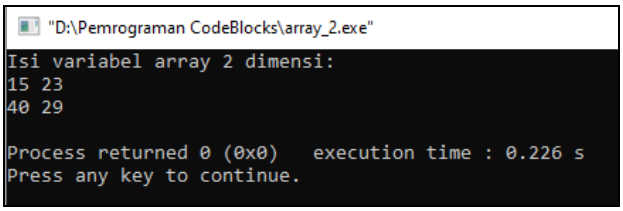
B. ARRAY 2 DIMENSI

Array dua dimensi adalah sebutan untuk array yang penomoran index-nya menggunakan 2 buah angka. Analogi yang sering dipakai seperti titik koordinat dalam diagram kartesius. Diagram kartesius merupakan diagram yang biasa kita pakai untuk membuat grafik.

Berikut contoh penggunaan array 2 dimensi menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **array_2.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/array_2.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
int arr[2][2];
        arr[0][0] = 15;
        arr[0][1] = 23;
        arr[1][0] = 40;
        arr[1][1] = 29;
                cout << "Isi variabel array 2 dimensi:" << endl;
                cout << arr[0][0] <<" "<< arr[0][1] << endl;
                cout << arr[1][0] <<" "<< arr[1][1] << endl;
return 0; }
```

Hasilnya seperti berikut:



C. ARRAY 3 DIMENSI

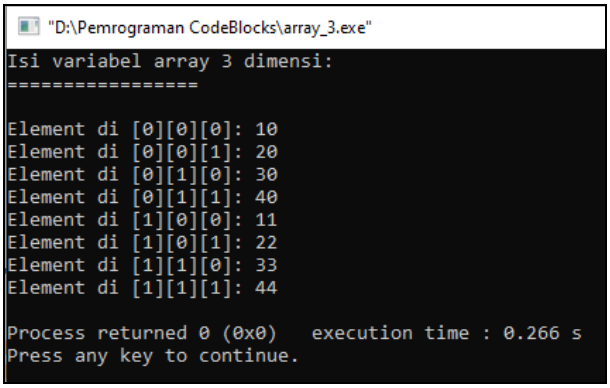
Array tiga dimensi adalah sebutan untuk array yang penomoran index-nya menggunakan 3 buah angka. Analogi yang sering dipakai seperti titik koordinat dalam diagram kartesius 3D.

Berikut contoh penggunaan array 3 dimensi menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **array_3.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/array_3.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main() {
int arr[2][2][2];
        arr[0][0][0] = 10;
        arr[0][0][1] = 20;
        arr[0][1][0] = 30;
        arr[0][1][1] = 40;
        arr[1][0][0] = 11;
        arr[1][0][1] = 22;
        arr[1][1][0] = 33;
        arr[1][1][1] = 44;
cout << "Isi variabel array 3 dimensi:" << endl;
cout << "==================" << endl;
cout << endl;
cout << "Element di [0][0][0]: "<< arr[0][0][0] << endl;
cout << "Element di [0][0][1]: "<< arr[0][0][1] << endl;
cout << "Element di [0][1][0]: "<< arr[0][1][0] << endl;
cout << "Element di [0][1][1]: "<< arr[0][1][1] << endl;
```

```
cout << "Element di [1][0][0]: "<< arr[1][0][0] << endl;
cout << "Element di [1][0][1]: "<< arr[1][0][1] << endl;
cout << "Element di [1][1][0]: "<< arr[1][1][0] << endl;
cout << "Element di [1][1][1]: "<< arr[1][1][1] << endl;
return 0; }
```

Hasilnya seperti berikut:



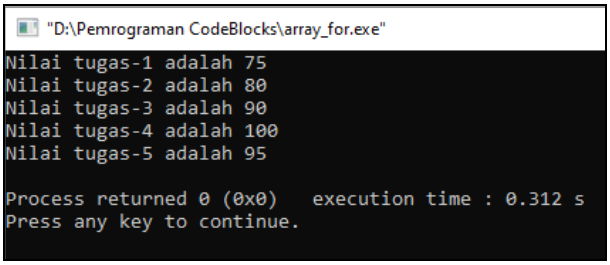
D. ARRAY DENGAN PERULANGAN FOR

Untuk menampilkan array dapat digabungkan dengan menggunakan fungsi perulangan for. Berikut contoh penggunaan array dengan perulangan for menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **array_for.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/array_for.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
int main()
{
int nilaiTugas[5]={75, 80, 90, 100, 95};
```

```
for (int i = 0; i<5; i++){
cout<<"Nilai tugas-"<<i+1<<" adalah "<< nilaiTugas[i]<<endl;
}
return 0;
}
```

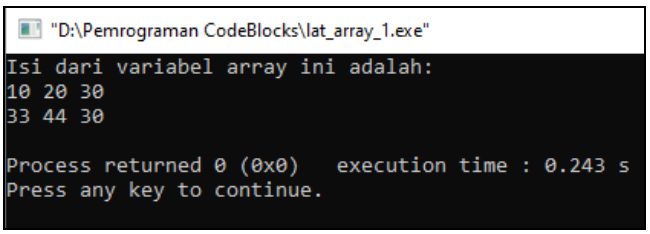
Hasilnya seperti berikut:



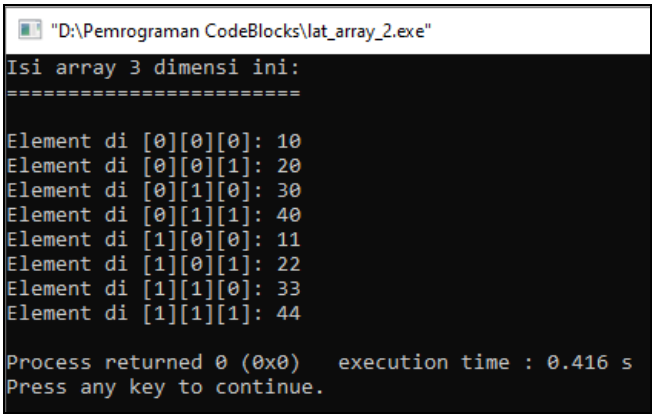
E. LATIHAN

Buatlah program menggunakan C++ untuk mengerjakan soal-soal berikut (nama file latihan bebas):

1. Buat program array untuk membuat array 2 dimensi dengan hasil seperti berikut:



2. Buat program array untuk membuat array 3 dimensi dengan hasil seperti berikut:


"D:\Pemrograman CodeBlocks\lat_array_2.exe"
Isi array 3 dimensi ini:
Element di [0][0][0]: 10
Element di [0][0][1]: 20
Element di [0][1][0]: 30
Element di [0][1][1]: 40
Element di [1][0][0]: 11
Element di [1][0][1]: 22
Element di [1][1][0]: 33
Element di [1][1][1]: 44
Process returned 0 (0x0) execution time : 0.416 s
Press any key to continue.

10

FUNCTION

A. PEMAHAMAN FUNCTION

Dalam membuat program, terkadang ada rancangan program yang dibuat secara berulang-ulang, seperti membaca tabel dari database, menampilkan penjumlahan, menghitung formulasi secara berulang-ulang seperti menghitung gaji dan lainnya. Tugas yang sama ini akan lebih efektif jika dipisahkan dari program utama dan dirancang menjadi sebuah fungsi atau disebut **function**.

Fungsi atau function adalah kode program yang dirancang untuk menyelesaikan sebuah tugas tertentu, dan merupakan bagian dari program utama. Berdasarkan siapa yang membuat, fungsi bisa dibedakan ke dalam 2 kelompok:

1. *Built-In Function*, yaitu fungsi yang sudah ada secara bawaan dari dalam bahasa pemrograman. Bahasa C++ menyediakan banyak fungsi bawaan, belum termasuk yang bisa diakses dari berbagai library atau package pihak ketiga.
2. *User Defined Function*, yaitu fungsi yang dibuat sendiri oleh kita sebagai programmer.

Untuk mempermudah bahasan dan pemahaman luas tentang fungsi ini maka pada buku ini yang akan kita bahas adalah jenis **User Defined Function**. Berikut format dasar cara penulisan fungsi dalam bahasa C++:

```
tipeDataKembalian namaFunction() {
    // Isi function disini...
    // Isi function disini...
```

return nilai; }

Bagian tipeDataKembalian diisi dengan tipe data nilai yang dikembalikan sebuah fungsi. Tipe data ini sudah kita pelajari sebelumnya seperti int, double atau string sebelumnya. Jika suatu fungsi tidak mengembalikan nilai, tipeDataKembalian ditulis sebagai **void** (secara harfiah berarti kosong). Sebuah fungsi yang tidak mengembalikan nilai disebut juga sebagai procedure.

```
void namaFunction() {
    // Isi function disini...
    // Isi function disini...
return nilai; }
```

Void adalah sebuah fungsi (function) yang ada dalam sebuah bahasa pemrograman C, entah itu C++ atau C# dan tidak mengembalikan nilai keluaran (return output) yang didapat dari hasil proses tersebut.

Penulisan namaFunction boleh bebas, tidak ada standar penamaan tertentu untuk fungsi bahasa C++ selama mengikuti aturan penulisan identifier, yakni tidak boleh di awali angka dan tidak boleh mengandung spasi. Pendefinisian User Defined Function harus ditulis di luar function main() seperti format berikut:

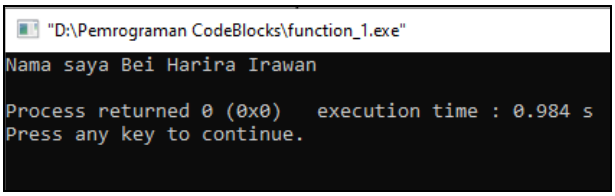
```
#include <iostream>
using namespace std;
        void namaFunction() {
          // Isi function disini...
          // Isi function disini...
          return nilai;
        }
int main() {
          // Jalankan function
```

```
        namaFunction()
return 0; }
```

Berikut contoh penggunaan function dengan 1 fungsi menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **function_1.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/function_1.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
        void kalimat() {
            cout << "Nama saya Bei Harira Irawan" << endl;
        }
int main() {
    kalimat();
return 0; }
```

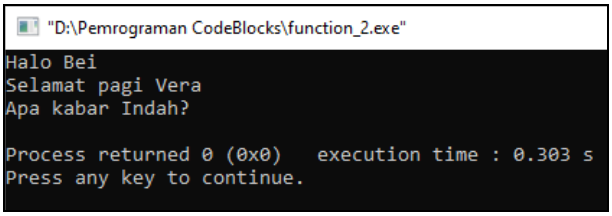
Hasilnya seperti berikut:



Berikut contoh penggunaan function dengan banyak fungsi menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **function_2.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/function_2.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
	void sapaBei() {
		cout << "Halo Bei" << endl;
	}

	void sapaVera() {
		cout << "Selamat pagi Vera" << endl;
	}

	void sapaIndah() {
		cout << "Apa kabar Indah?" << endl;
	}

int main() {
	sapaBei ();
	sapaVera ();
	sapaIndah ();
return 0; }
```

Hasilnya seperti berikut:

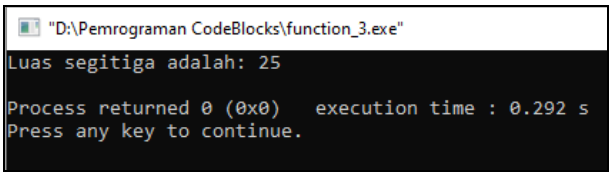


Untuk fungsi yang kompleks, kita bisa menulis variabel di dalam fungsi tersebut. Berikut contoh penggunaan function dengan contoh variabel di

dalamnya menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **function_3.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/function_3.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
    void hitungLuasSegitiga() {
        double alas = 5;
        double tinggi = 7;
        double luas = (alas * tinggi) / 2;
        cout << "Luas segitiga adalah: " << luas << endl;
    }

int main() {
    hitungLuasSegitiga();
return 0; }
```

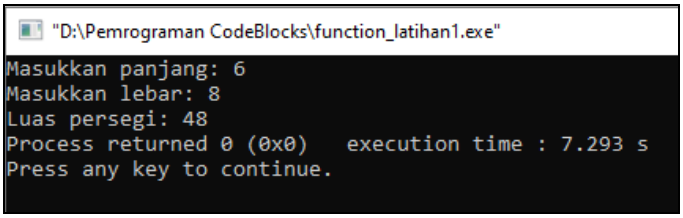
Hasilnya seperti berikut:



B. LATIHAN

Buatlah program menggunakan C++ untuk mengerjakan soal-soal berikut (nama file latihan bebas):

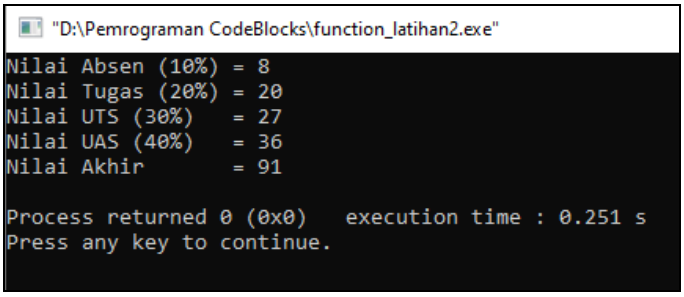
1. Buat program menggunakan function untuk mencari luas persegi panjang dengan memasukkan input panjang dan lebar dari keyboard sehingga dihasilkan output seperti berikut:



2. Buat program menggunakan function untuk mencari nilai akhir mahasiswa apabila diketahui :

   Nilai Absen (10%) = 80
   Nilai Tugas (20%) = 100
   Nilai UTS (30%) = 90
   Nilai UAS (40%) = 90
   Nilai Akhir = …

   Buatlah agar dihasilkan output seperti berikut:



11

PARAMETER

A. PEMAHAMAN PARAMETER

Parameter merupakan sebutan untuk nilai inputan sebuah fungsi pada saat fungsi tersebut didefinisikan, sedangkan argumen adalah sebutan untuk nilai inputan fungsi pada saat fungsi tersebut dipanggil. Sebuah fungsi bisa menerima 1, 2, atau lebih dari 5 parameter atau argumen, namun bisa juga tidak memerlukan sama sekali.

Parameter adalah suatu nilai (berupa variabel) yang dikirimkan ke dalam fungsi, yang kemudian akan ikut diproses di dalam badan fungsi. Dengan menggunakan parameter, suatu fungsi dapat memberikan hasil yang dinamis atau berubah-rubah setiap fungsi tersebut dipanggil.

```
void namaFunction(int param1, int param2, int param3) {
    // Isi function disini...
    // Isi function disini...
return nilai; }
```

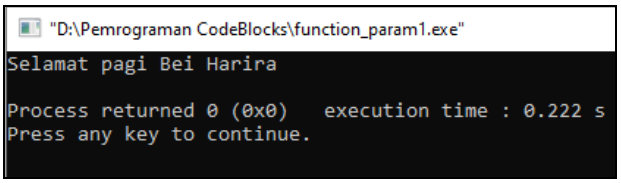
Di dalam tanda kurung setelah namaFunction, adalah tempat untuk penulisan parameter. Parameter dalam bahasa C++ ditulis berpasangan antara tipeData dan nama parameter. Pada contoh diatas, terdapat 3 buah parameter bernama param1, param2 dan param3. Sepanjang isi function, param1, param2 dan param3 bisa diakses sebagaimana variabel biasa.

Berikut contoh penggunaan function dengan parameter di dalamnya menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri

nama **function_param1.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/function_param1.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
    void sapaTeman(string nama) {
    cout << "Selamat pagi " << nama << endl;
    }

int main() {
    sapaTeman("Bei Harira");
return 0; }
```

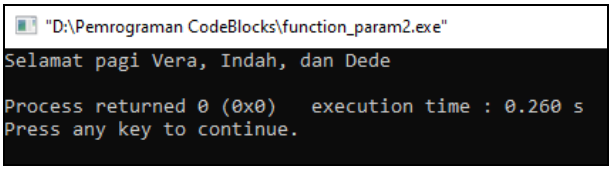
Hasilnya seperti berikut:



Kita bisa menginput lebih dari satu argumen ke dalam fungsi selama pendefinisian fungsi tersebut juga ditulis dengan lebih dari satu parameter. Berikut contoh penggunaan function dengan banyak parameter di dalamnya menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **function_param2.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/function_param2.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
void sapaTeman(string nama1, string nama2, string nama3) {
        cout << "Selamat pagi " << nama1 << ", ";
        cout << nama2 << ", dan " << nama3 << endl;
}

int main() {
        sapaTeman("Vera", "Indah", "Dede");
return 0; }
```

Hasilnya seperti berikut:

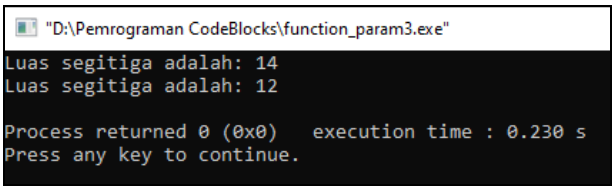


Berikut contoh penggunaan function dengan parameter yang bisa menerima 2 argumen bertipe integer di dalamnya menggunakan bahasa C++. Buat project baru pada CodeBlocks lalu beri nama **function_param3.cpp** dan letakkan pada folder **D:/Pemrograman CodeBlocks** (D:/Pemrograman CodeBlocks/function_param3.cpp) lalu ketikkan perintah berikut:

```
#include <iostream>
using namespace std;
        void hitungLuasSegitiga(int alas, int tinggi) {
                double luas = (alas * tinggi) / 2.0;
                cout << "Luas segitiga adalah: " << luas << endl;
        }
```

```
int main() {
hitungLuasSegitiga(4, 7);
hitungLuasSegitiga(2, 12);
return 0; }
```

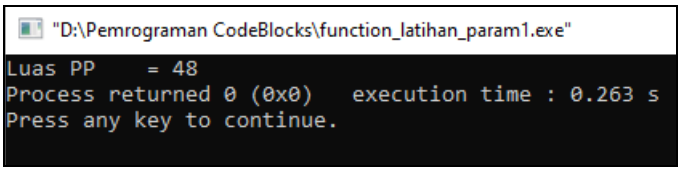
Hasilnya seperti berikut:



B. LATIHAN

Buatlah program menggunakan C++ untuk mengerjakan soal-soal berikut (nama file latihan bebas):

1. Buat program menggunakan fungsi dengan parameter untuk mencari luas persegi panjang dengan panjang = 6 dan lebar = 8, sehingga dihasilkan output seperti berikut:



2. Buat program menggunakan function untuk menghitung total gaji diterima apabila diketahui :

Gaji Pokok	=	3000000
Uang Makan/hari	=	15000
Uang Transport/hari	=	20000
Jumlah Hari Masuk	=	22
Total Gaji	=	…

12

POST TEST PROGRAMMING

A. LATIHAN - 1

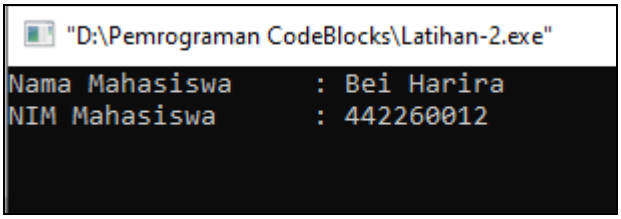
Ketik syntax program berikut dan running lalu lihat hasilnya:

*Latihan-1.cpp

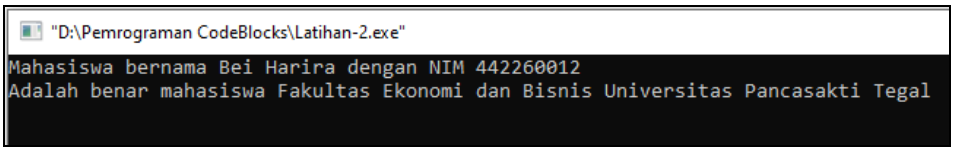
```
1   # include <iostream>
2   using namespace std;
3
4   int main()
5   {
6
7   cout << "Latihan Post Test program C++" << endl;
8   cout << "Belajar pemrograman logika dan algoritma sangat mudah" << endl;
9   return 0;
10  }
```

B. LATIHAN - 2

Buat program dengan model tampilan input sebagai berikut:



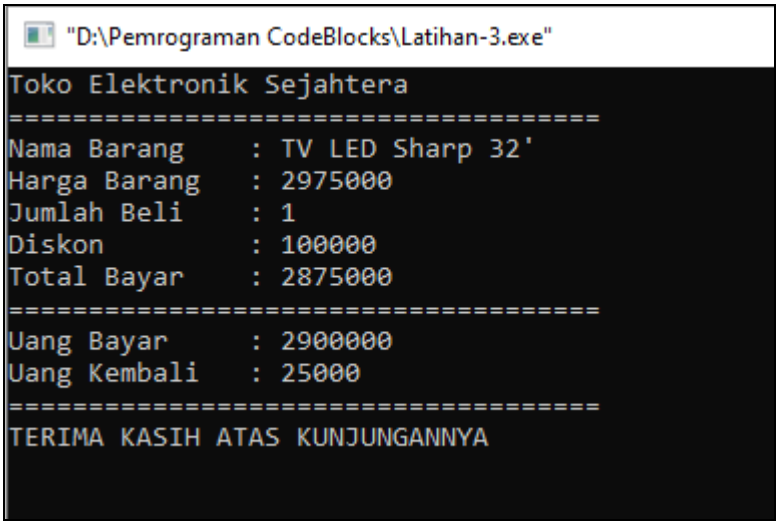
Dengan hasil output sebagai berikut:



C. LATIHAN - 3

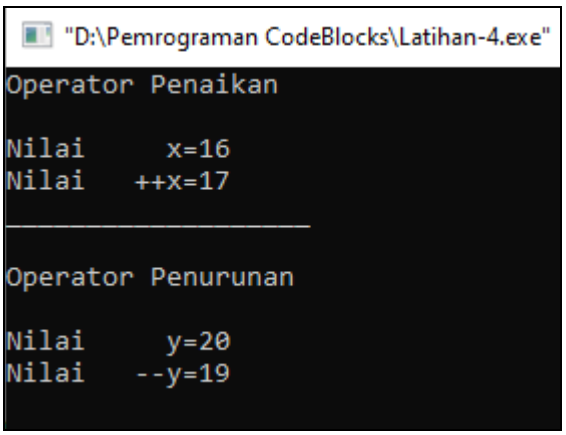
Toko Elektronik Sejahtera menjual barang-barang elektronik. Buat program dengan input data nama barang, harga barang, jumlah beli, diskon dan uang

bayar. Carilah total bayar dan uang kembali secara otomatis. Buat agar output sebagai berikut:



D. LATIHAN - 4

Diketahui nilai dari variabel x=21 dan nilai dari variabel y=17. Hitunglah nilai x, jika diberikan nilai x-=5 dan hitunglah nilai y, jika diberikan nilai y+=3. Sehingga akan diperoleh nilai penaikan dan penurunan dari nilai x dan y dengan output sebagai berikut:



E. LATIHAN - 5

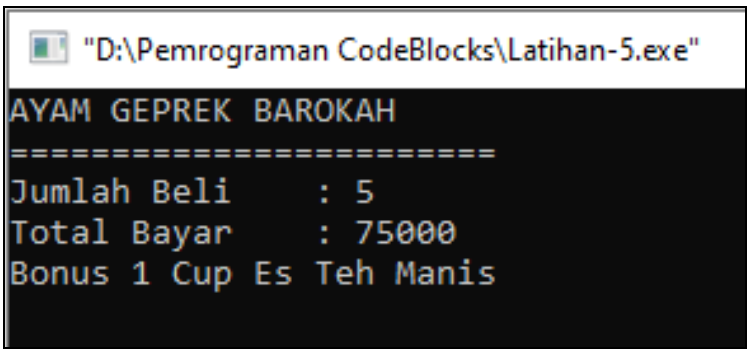
Ketik syntax program berikut dan running lalu lihat hasilnya:

*Latihan-5.cpp

```
1   # include <conio.h>
2   # include <iostream>
3   # include <stdio.h>
4   # include <stdlib.h>
5   using namespace std;
6   main()
7   {
8   int usia;
9   system("cls");
10  cout << "Berapakah usiamu? "; cin >> usia;
11  if (usia >= 17){
12          cout << "Kamu sudah boleh membuat SIM motor..." << endl;
13      }
14      else{
15          cout << "Kamu belum cukup umur untuk mengendarai motor" << endl;
16      }
17  getch();
18  }
19
```

F. LATIHAN - 6

Buat program sederhana untuk pembelian paket Ayam Geprek seharga Rp. 15.000,-/paket. Jika anda membeli dengan jumlah 3 paket atau lebih, maka akan tampil pesan "Bonus 1 Cup Es Teh Manis". Tapi jika anda membeli kurang dari 3 paket, maka akan tampil "Maaf anda tidak dapat bonus". Buat agar tampilan output sebagai berikut:



G. LATIHAN - 7

Ketik syntax program berikut dan running lalu lihat hasilnya:

Latihan-6.cpp

```
# include <conio.h>
# include <iostream>
# include <stdio.h>
# include <stdlib.h>
using namespace std;
main()
{
int kode;
system("cls");
cout << "Masukkan kode (1, 2 atau 3) : ";
cin >> kode;
switch(kode)
{
    case 1:
    cout << "Hari ini Bahagia";
    case 2:
    cout << " dan Gembira";
    break;
    default:
    cout << "Sedih";
    break;
}
getch();
}
```

H. LATIHAN - 8

Ketik syntax program berikut dan running lalu lihat hasilnya:

Latihan-8.cpp

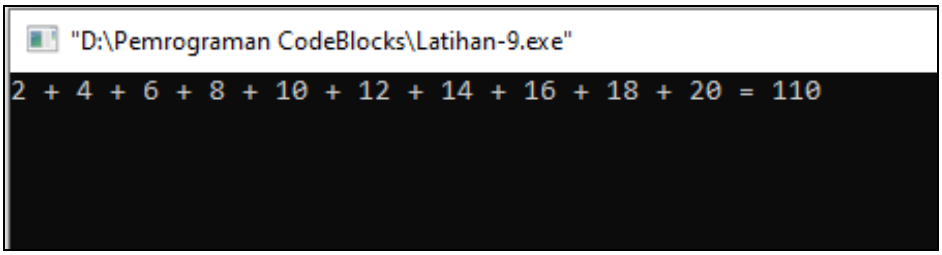
```
# include <conio.h>
# include <iostream>
# include <stdio.h>
# include <stdlib.h>
using namespace std;
main()
{
int a;
system("cls");
for (a = 1; a <= 10; ++a)
    cout << a;
getch();
}
```

I. LATIHAN - 9

Buat program untuk menghasilkan output penjumlahan berikut:

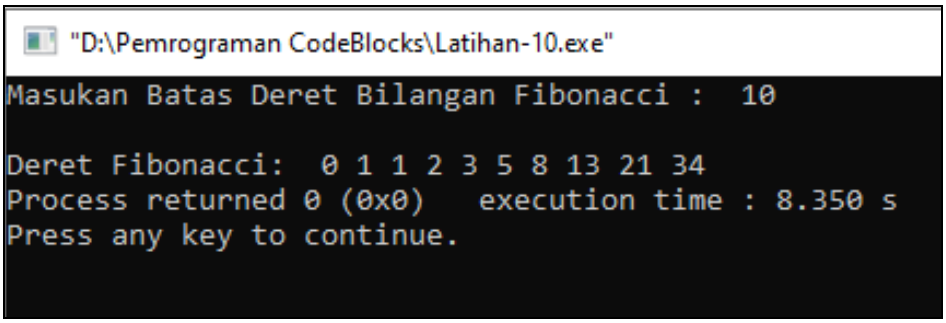
2 + 4 + 6 + 8 + 10 + 12 + 14 + 16 + 18 + 20 = 110

Dengan output seperti berikut:



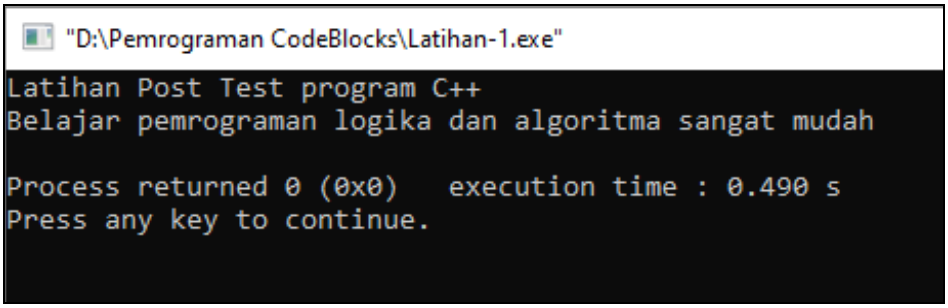
J. LATIHAN - 10

Buat program untuk menghasilkan deret bilangan Fibonacci dengan output seperti berikut:



JAWABAN

A. LATIHAN - 1



B. LATIHAN - 2

*Latihan-2.cpp

```
1   # include <conio.h>
2   # include <iostream>
3   # include <stdio.h>
4   # include <stdlib.h>
5   using namespace std;
6   int main()
7   {
8   char nm[30], nim[15];
9   system("cls");
10  cout << "Nama Mahasiswa      : "; gets(nm);
11  cout << "NIM Mahasiswa       : "; cin >> nim;
12  cout << "____________________" << endl;
13  cout << "\n";
14  system("cls");
15  cout << "Mahasiswa bernama " << nm << " dengan NIM " << nim << endl;
16  cout << "Adalah benar mahasiswa Fakultas Ekonomi dan Bisnis Universitas Pancasakti Tegal" << endl;
17  getch();
18  }
```

C. LATIHAN - 3

*Latihan-3.cpp

```
1  # include <conio.h>
2  # include <iostream>
3  # include <stdio.h>
4  # include <stdlib.h>
5  using namespace std;
6  main()
7  {
8  string nm_brg;
9  int hrg,jml,diskon,tobar,ubar,ukem;
10 system("cls");
11 cout<<"Toko Elektronik Sejahtera ";
12 cout<<"\n====================================";
13 cout<<"\nNama Barang     : "; getline(cin, nm_brg);
14 cout<<"Harga Barang    : ";cin>>hrg;
15 cout<<"Jumlah Beli     : ";cin>>jml;
16 cout<<"Diskon          : ";cin>>diskon;
17 tobar = jml * hrg - diskon;
18 cout<<"Total Bayar     : "<<tobar<<endl;
19 cout<<"====================================";
20 cout<<"\nUang Bayar      : ";cin>>ubar;
21 ukem=ubar-tobar;
22 cout<<"Uang Kembali    : "<<ukem<<endl;
23 cout<<"====================================";
24 cout<<"\nTERIMA KASIH ATAS KUNJUNGANNYA ";
25 getch();
26 }
27
```

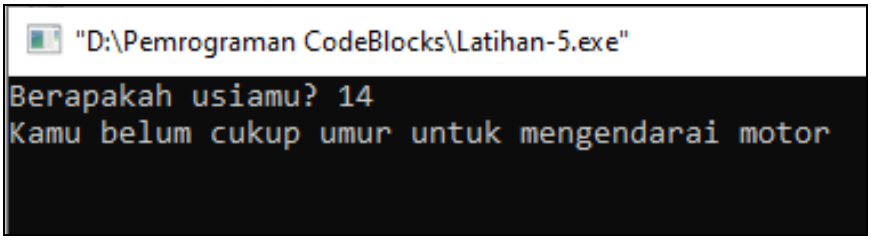
D. LATIHAN - 4

Latihan-4.cpp

```
1  # include <conio.h>
2  # include <iostream>
3  # include <stdio.h>
4  # include <stdlib.h>
5  using namespace std;
6  main()
7  {
8  int x = 21, y = 17;
9  system("cls");
10 cout<<"Operator Penaikan" << endl;
11 x -= 5;
12 printf("\nNilai     x=%d ",x);
13 printf("\nNilai   ++x=%d ",++x);
14 cout<<"\n_____________________" << endl;
15 cout<<"\nOperator Penurunan" << endl;
16 y += 3;
17 printf("\nNilai     y=%d ",y);
18 printf("\nNilai   --y=%d ",--y);
19 getch();
20 }
21
```

E. LATIHAN - 5

```
"D:\Pemrograman CodeBlocks\Latihan-5.exe"
Berapakah usiamu? 19
Kamu sudah boleh membuat SIM motor...
```

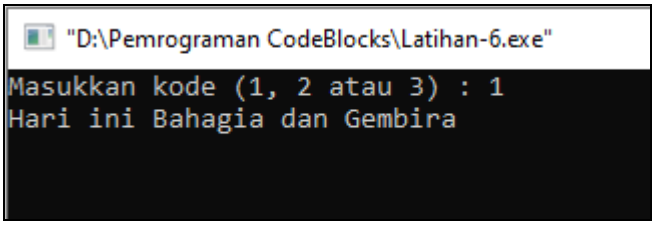


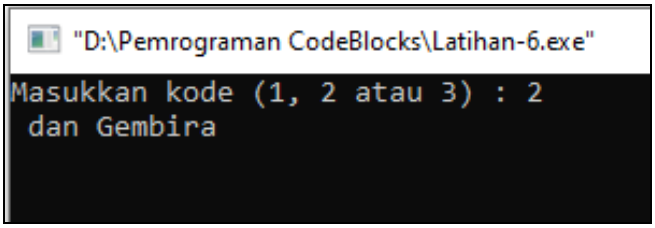
F. LATIHAN - 6

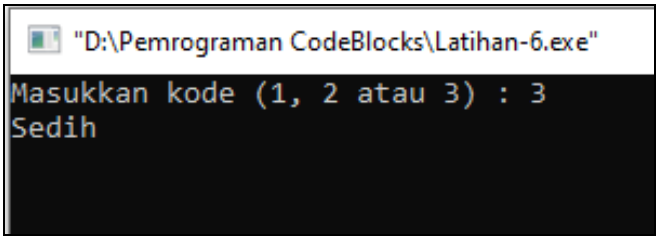
Latihan-5.cpp

```
1   # include <conio.h>
2   # include <iostream>
3   # include <stdio.h>
4   # include <stdlib.h>
5   using namespace std;
6   main()
7   {
8   int total, jml;
9   int hrg = 15000;
10  system("cls");
11  cout << "AYAM GEPREK BAROKAH" << endl;
12  cout << "=======================" << endl;
13  cout << "Jumlah Beli     : "; cin >> jml;
14  total = jml * hrg;
15  cout << "Total Bayar     : " << total << endl;
16  if(jml >= 3)
17      cout << "Bonus 1 Cup Es Teh Manis" << endl;
18  else
19      cout << "Maaf anda tidak dapat bonus" << endl;
20  getch();
21  }
22
```

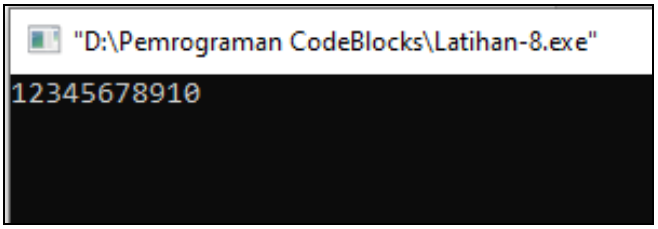
G. LATIHAN - 7




"D:\Pemrograman CodeBlocks\Latihan-6.exe"
Masukkan kode (1, 2 atau 3) : 2
dan Gembira


"D:\Pemrograman CodeBlocks\Latihan-6.exe"
Masukkan kode (1, 2 atau 3) : 3
Sedih

H. LATIHAN - 8


"D:\Pemrograman CodeBlocks\Latihan-8.exe"
12345678910

I. LATIHAN - 9

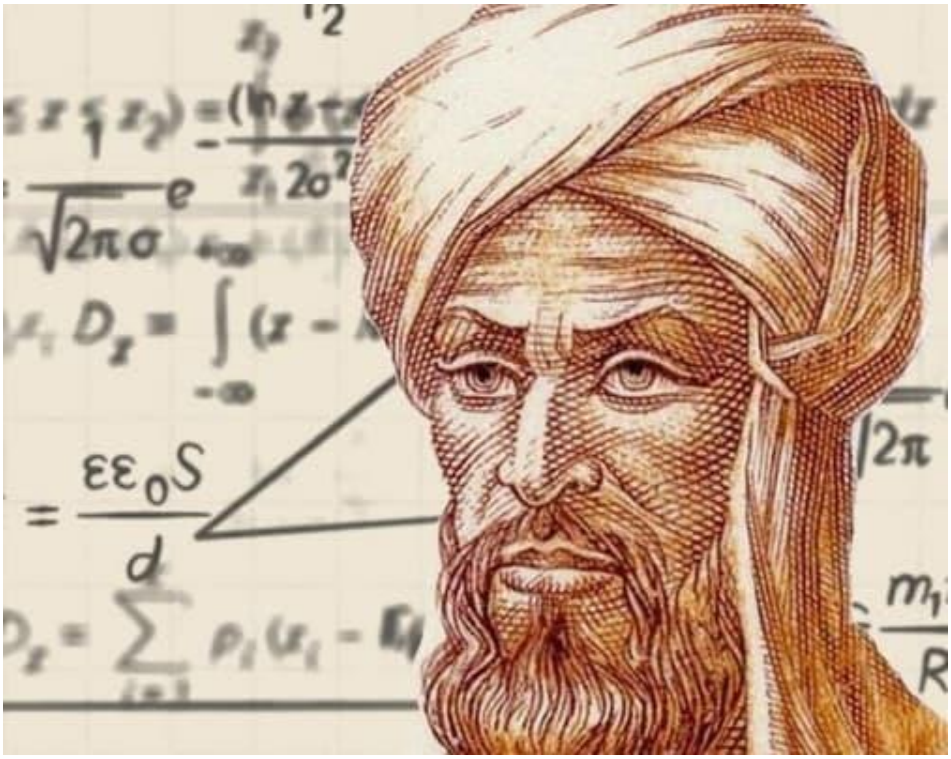
Latihan-9.cpp

```
1   # include <conio.h>
2   # include <iostream>
3   # include <stdio.h>
4   # include <stdlib.h>
5   using namespace std;
6   main()
7   {
8   int a, b = 0;
9   system("cls");
10  for (a = 2; a < 20; a+=2)
11  {
12  cout << a;
13  if(a <= 20)
14  cout << " + ";
15  b = b + a;
16  }
17  b = b + a;
18  cout << a;
19  cout << " = " << b;
20  getch();
21  }
22
```

J. LATIHAN - 10

Latihan-10.cpp

```cpp
#include <iostream>
using namespace std;
int main()
{
int n, f1 = 0, f2 = 1, berikutnya = 0;
cout << "Masukan Batas Deret Bilangan Fibonacci :  ";
cin >> n;
cout<<endl;
cout << "Deret Fibonacci: ";
    for (int i = 1; i <= n; ++i)
    {
        if(i == 1)
        {
            cout << " " << f1<<" ";
            continue;
        }
        if(i == 2)
        {
            cout << f2 << " ";
            continue;
        }
        berikutnya = f1 + f2;
        f1 = f2;
        f2 = berikutnya;
        cout << berikutnya << " ";
    }
    return 0;
}
```



Muhammad ibn Musa Al Khwarizmi

REFERENSI

A. Kadir, "Buku Pintar C++ untuk Pemula." Andi Offset, Yogyakarta.

A. Pranata, "Pemrograman Borland C++." Andi Offset, Yogyakarta.

Budi Raharjo, Pemrograman C++, Informatika, Bandung.

Sjukani, Moh. Algoritma dan Struktur Data 1 dengan C, C++ dan Java Edisi IV. Jakarta: Mitra Wacana Media.

https://teknojurnal.com/logika-pemrograman-dasar-pemula

https://www.duniailkom.com