

MAKALAH
MODUL CODE BLOCKS IDE DAN
PENGENALAN BAHASA C++ (BAGIAN PERTAMA)
“ ”



Oleh
NOVRI ANTO
21104065

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
PURWOKERTO
2023/2024

A. Tujuan

Laporan ini bertujuan untuk memberikan informasi rinci terkait apa yang telah saya pelajari dari modul 1, beserta latihan-latihan dari guided dan unguided

B. Landasan Teori

1. Sekilas Tentang C++

Bahasa pemrograman C++ dikembangkan oleh Bjarne Stroustrup di AT&T Bell Laboratories pada awal tahun 1980-an, berdasarkan standar ANSI C (American National Standard Institute). Awalnya, prototipe C++ muncul sebagai versi C yang ditingkatkan dengan fitur kelas, dan disebut sebagai "C dengan kelas" (C with class). Pada tahun 1983-1984, "C dengan kelas" disempurnakan dengan penambahan fitur operator overloading dan fungsi, yang kemudian dikenal sebagai C++. Simbol ++ adalah operator dalam C untuk peningkatan, yang menunjukkan bahwa bahasa baru ini adalah versi yang lebih maju dari C.

Borland International merilis compiler Borland C++ dan Turbo C++, yang keduanya dapat digunakan untuk mengkompilasi kode C++. Perbedaannya, Borland C++ dapat digunakan di lingkungan DOS dan juga untuk pemrograman Windows. Selain Borland International, beberapa perusahaan lain juga merilis compiler C++, seperti Topsispeed C++, Zortech C++, dan Code Blocks. Dalam praktikum ini, kita akan menggunakan bahasa C++.

2. Dasar Pemrograman

a. Struktur Program C++

Secara umum, pembagian struktur bahasa pemrograman C++ adalah sebagai berikut:

C++	Keterangan
<pre>#include <iostream> #include <conio.h> #include "newlibrary.h"</pre>	Pendeklarasian library yang akan digunakan di dalam program
<pre>#define PHI 3.14 const int constant1; const float constant2 = 0.5;</pre>	Pendefinisian konstanta
<pre>struct new_record_type { int element1; float element2; }</pre>	Pendefinisian tipe data bentukan / record type / struktur
<pre>int var1; float var2[2];</pre>	Pendeklarasian variabel
<pre>int function_A(){ // } void procedure_B(){ // }</pre>	Pendeklarasian fungsi dan prosedur
<pre>int main(){ // blok program return 0; }</pre>	Fungsi utama

b. Pengenal (Identifier)

Identifier merupakan nama yang biasa digunakan untuk variabel, konstanta, fungsi atau objek lain yang didefinisikan oleh program.

Aturan yang digunakan untuk menentukan identifier:

- Harus diawali dengan huruf (A....Z, a....z) atau garis bawah (_).
- Karakter selanjutnya bisa berupa huruf, digit atau karakter garis bawah (_) atau dollar (\$).
- Panjang maksimal identifier adalah 32 karakter, jika lebih maka yang dianggap adalah 32 karakter awal.
- Tidak boleh mengandung spasi.
- Tidak boleh menggunakan operator aritmatika (+ - / * %).
- Bahasa C bersifat case sensitive, jadi huruf besar dan huruf kecil dianggap berbeda.

Contoh:

panjang (berbeda dengan: Panjang)

nilai4

luas_total

harga_beli\$

c. Tipe Data Dasar

Data merupakan suatu nilai yang dapat dinyatakan dalam bentuk konstanta atau variabel. Data berdasarkan jenisnya dibagi dalam 5 kelompok, yang dinamakan sebagai tipe data dasar. Kelima kelompok tersebut:

Type data	Contoh	Ukuran	Jangkauan nilai
Char	char nama[20];	1 Byte	-128 s.d. +127
Int	int nilai; int jumlah = 0;	2 Byte	-32768 s.d +32767
Long	long selisih;	4 Byte	-2.147.438.648 s.d +2.147.438.647
Float	float jumlah;	4 Byte	3.4e-38 s.d 3.4e+38
Double	double hasil;	8 Byte	1.7e-308 s.d 1.7e+308

d. Variabel

Variabel dalam program digunakan untuk menyimpan nilai, nilai variabel bisa berubah – ubah selama program berjalan. Aturan penamaan variabel sesuai dengan aturan penamaan identifier. Bentuk umum pendeklarasian suatu variabel dalam bahasa pemrograman C dapat ditulis sebagai berikut.

tipe_data nama_variabel;

Contoh:

int x, y;

char ch;

Kita juga dapat langsung memberikan nilai awal (inisialisasi) pada suatu variabel pada saat variable tersebut di deklarasikan.

Contoh:

int x=20, y=6;

char nama[30] = "Budi";

e. Konstanta

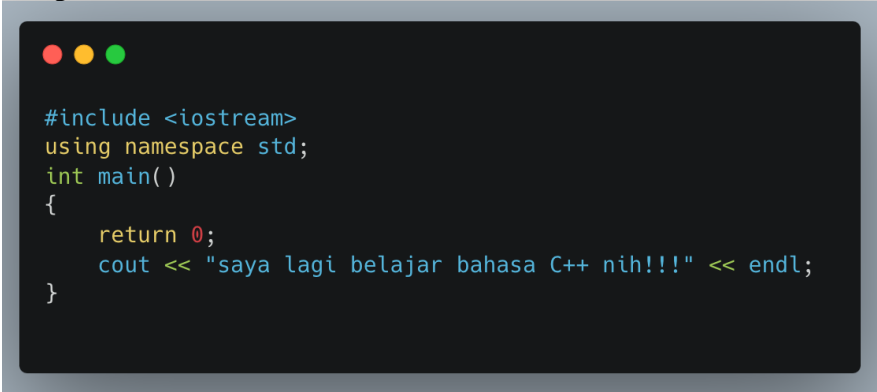
Konstanta menyatakan nilai yang selalu tetap. Seperti halnya dengan variabel, konstanta juga mempunyai tipe. Untuk mendeklarasikan suatu nilai yang sifatnya konstan kita cukup menambahkan kata `const` di depan tipe data dan variabel.

Contoh:

```
const float phi = 3.14;
```

```
const int n = 20;
```

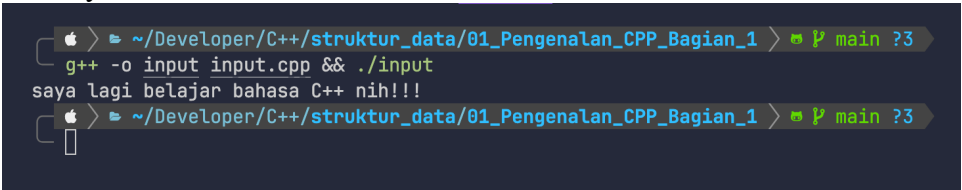
3. Output



```
#include <iostream>
using namespace std;
int main()
{
    return 0;
    cout << "saya lagi belajar bahasa C++ nih!!!" << endl;
}
```

Gambar 1 Output di C++ menggunakan `std::cout`


Hasilnya:



```
> ~/Developer/C++/struktur_data/01_Pengenalan_CPP_Bagian_1 > main ?3
g++ -o input input.cpp && ./input
saya lagi belajar bahasa C++ nih!!!
> ~/Developer/C++/struktur_data/01_Pengenalan_CPP_Bagian_1 > main ?3
```

4. Input

`cin()` merupakan salah satu fungsi yang digunakan untuk meminta inputan keyboard dari user. Bentuk umumnya pendeklarasiannya adalah sebagai berikut:



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

int main()
{
    int inp;
    cin >> inp;
    cout << "nilai = " << inp;
    return 0;
}
```

Hasilnya:

```

> ~/Developer/C++/struktur_data/01_
g++ -o input input.cpp && ./input
20
nilai = 20%
> ~/Developer/C++/struktur_data/01_

```

5. Operator

Operator adalah simbol yang digunakan untuk melaksanakan operasi atau manipulasi tertentu. Bahasa C dikenal kaya akan berbagai jenis operator, termasuk: Operator Aritmatika, Operator Penugasan (assignment operator), Operator Logika, Operator Unary, Operator Bitwise, Operator Kondisional, dan lain sebagainya.

Tabel 1-4 Operator, Arah Proses, dan Jenjangnya

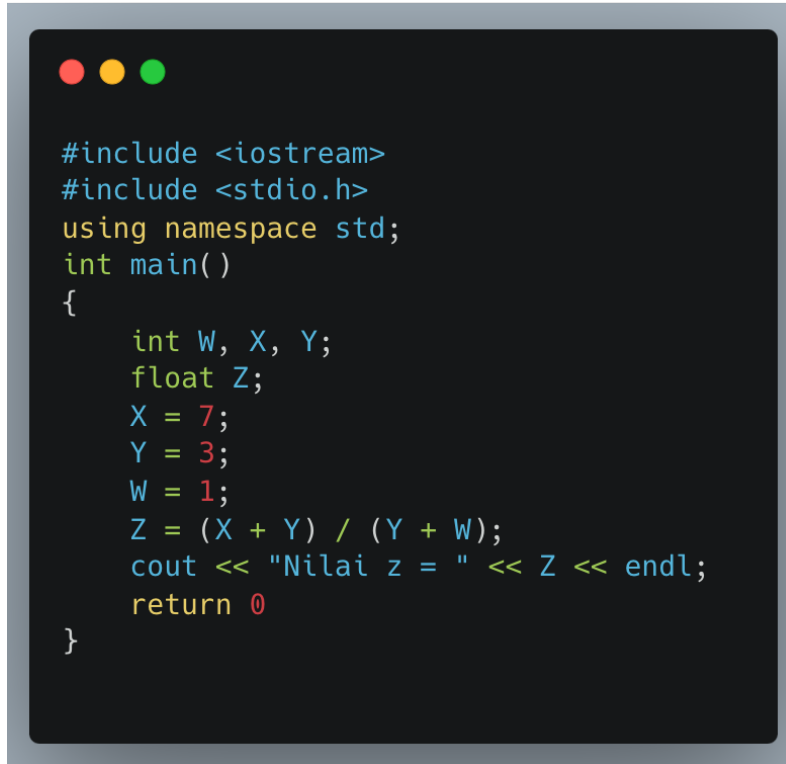
Kategori (Arti)	Operator	Kategori (Arti)	Operator
Panggilan fungsi, subscript <i>array</i> , dan elemen struktur data	() [] ->	Operator Hubungan (sama dengan, tidak sama dengan)	== !=
Operator Unary (NOT, komplemen, negasi, inkremen, dekremen , <i>address</i> , <i>indirection</i>)	!	Operator Bitwise AND	&
	~	Operator Bitwise XOR	^
	-	Operator Bitwise OR	
	++	Operator Logika AND	&&
	--	Operator Logika OR	
	& *	Operator Kondisional	?:
Operator Aritmatika(Perkalian, pembagian, Sisa Pembagian/mod)	*	Operator Pengerjaan Aritmatika (<i>assignment</i> , <i>assignment</i> perkalian, <i>assignment</i> pembagian, <i>assignment</i> mod, <i>assignment</i> penjumlahan, <i>assignment</i> pengurangan)	=
	/		*=
	%		/=
			%=
Operator Aritmatika (Pertambahan, Pengurangan)	+		+=
	-		-=

Operator Bitwise Pergeseran Bit (shift kiri, shift kanan)	<< >>	Operator Pengerjaan Bitwise (<i>assignment</i> AND bitwise, <i>assignment</i> OR bitwise, <i>assignment</i> XOR bitwise, <i>assignment</i> shift kanan)	&= ^= = <<= >>=
Operasi Hubungan (kurang dari, kurang dari atau sama dengan lebih dari, lebih dari atau sama dengan)	< <= > >=	Operator Koma	,

a. Operator Aritmatika

Misalkan terdapat ungkapan sebagai berikut : $A + B / C + D$. Untuk mengubah jenjang dapat digunakan

tanda kurung '()' (sebagai operator jenjang tertinggi) sebagai berikut: $(A + B)/(C + D)$.



```
#include <iostream>
#include <stdio.h>
using namespace std;
int main()
{
    int W, X, Y;
    float Z;
    X = 7;
    Y = 3;
    W = 1;
    Z = (X + Y) / (Y + W);
    cout << "Nilai z = " << Z << endl;
    return 0
}
```

b. Operator Pengerjaan (*Assignment*)

Operator ini digunakan untuk memindahkan nilai dari suatu ungkapan ke suatu pengenalan. Di samping operator pengerjaan = yang umumnya digunakan di bahasa-bahasa pemrograman, bahasa C menyediakan beberapa operator pengerjaan lain. Misalnya: $A += 7$, ekuivalen dengan, $A = A + 7$

c. Operator Logika

Operasi logika membandingkan dua buah nilai logika. Nilai logika adalah nilai benar atau salah.

Misalnya:

1. $(kar > 'A') \ \&\& \ (kar < 'Z')$

Hasil operasi $\&\&$ bernilai benar hanya jika $kar > 'A'$ dan $kar < 'Z'$

2. $(pilihan == 'Y') \ || \ (pilihan == 'y')$

Hasil operasi logika $||$ bernilai benar jika pilihan berupa 'Y' atau 'y'.

3. $!operand$

Hasil operand $!$ akan bernilai benar jika operand bernilai salah dan sebaliknya.

6. Kondisional

Untuk menyelesaikan suatu masalah diperlukan pengambilan keputusan, Bahasa C menyediakan beberapa jenis pernyataan berupa operator kondisi sebagai berikut:

1. Pernyataan if
2. Pernyataan if-else
3. Pernyataan switch

Pernyataan pengambilan keputusan di atas memerlukan suatu kondisi sebagai basis pada pengambilan keputusan. Kondisi umum yang dipakai keadaan benar atau salah.

a. Bentuk 1 (if else)

```
if (kondisi)
    pernyataan1;
else
    pernyataan2;
```

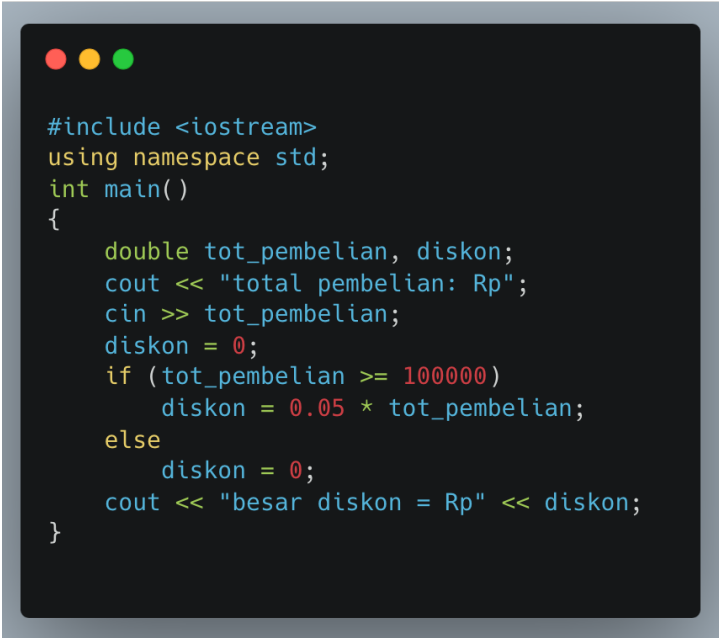
Arti dari pernyataan if-else di atas adalah:

1. Jika kondisi benar, maka *pernyataan1* dijalankan.
2. Jika kondisi salah, maka *pernyataan 2* yang akan dijalankan.

pernyataan1 dan *pernyataan2* dapat berupa sebuah pernyataan tunggal, pernyataan majemuk ataupun pernyataan kosong. Bentuk ini dapat disederhanakan dengan kondisional ekspresi dengan bentuk umumnya sebagai berikut.

$expr1 ? expr2 : expr3$

Jika *expr1* benar maka *expr2* yang dijalankan, sedangkan jika salah maka *expr3* yang dijalankan. Untuk lebih jelasnya perhatikan contoh berikut.



```
#include <iostream>
using namespace std;
int main()
{
    double tot_pembelian, diskon;
    cout << "total pembelian: Rp";
    cin >> tot_pembelian;
    diskon = 0;
    if (tot_pembelian >= 100000)
        diskon = 0.05 * tot_pembelian;
    else
        diskon = 0;
    cout << "besar diskon = Rp" << diskon;
}
```

```

#include <iostream>
using namespace std;
int main()
{
    double tot_pembelian, diskon;
    cout << "total pembelian: Rp";
    cin >> tot_pembelian;
    diskon = (tot_pembelian >= 100000) ? 0.05 * tot_pembelian : 0;
    cout << "besar diskon = Rp" << diskon;
}

```

b. Bentuk 2 (switch)

Bentuk ketiga menggunakan pernyataan switch, merupakan pernyataan yang dirancang khusus untuk menangani pengambilan keputusan yang melibatkan banyak alternatif. Pengujian pada switch akan dimulai dari *kondisi1*, kalau nilai *kondisi1* cocok maka *pernyataan1* dilakukan, bila tidak cocok akan diteruskan pada pengecekan *pernyataan2*. Bila tidak ditemukan kondisi yang cocok maka statement pada default akan dilakukan. Contoh penggunaan switch:


```

#include <stdio.h>
#include <iostream>

using namespace std;

int main()
{
    int kode_hari;
    puts("Menentukan hari\n");
    puts("1=Senin 3=Rabu 5=Jumat 7=Minggu ");
    puts("2=Selasa 4=Kamis 6=Sabtu ");
    cin >> kode_hari;
    switch (kode_hari)
    {
        case 1:
            puts("Hari Senin");
            break;
        case 2:
            puts("Hari Selasa");
            break;
        case 3:
            puts("Hari Rabu");
            break;
        case 4:
            puts("Hari Kamis");
            break;
        case 5:
            puts("Hari Jumat");
            break;
        case 6:
            puts("Hari Sabtu");
            break;
        case 7:
            puts("Hari Minggu");
            break;
        default:
            puts("Kode masukan salah!!!");
    }
    return 0;
}

```

C. Guided

a. Nomor 1

```

#include <iostream>
#include <stdio.h>
using namespace std;

int main()
{
    // Nomor 1
    float bilA;

    float bilB;

    cout << "Bilangan A: " << endl;
    cin >> bilA;

    cout << "Bilangan B: " << endl;
    cin >> bilB;

    // Hasil penjumlahan
    cout << "Hasil penjumlahan: " << bilA + bilB << endl;

    // Hasil pengurangan
    cout << "Hasil pengurangan: " << bilA - bilB << endl;

    // Hasil perkalian
    cout << "Hasil perkalian: " << bilA * bilB << endl;

    // Hasil pembagian
    cout << "Hasil pembagian: " << bilA / bilB << endl;

    return 0;
}

```

Output:

bilA = 10;

bilB = 20;

```

🍏 > ~/Developer/C++/struktur_data/01_Pengenala
g++ -o guided_01 guided_01.cpp && ./guided_01
Bilangan A:
10
Bilangan B:
20
Hasil penjumlahan: 30
Hasil pengurangan: -10
Hasil perkalian: 200
Hasil pembagian: 0.5
🍏 > ~/Developer/C++/struktur_data/01_Pengenala

```

b. Nomor 2

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

int main()
{
    // Nomor 2
    string angka[] = {"Nol", "Satu", "Dua", "Tiga", "Empat", "Lima",
        "Enam", "Tujuh", "Delapan", "Sembilan", "Sepuluh",
        "Sebelas", "Dua Belas", "Tiga Belas", "Empat
        Belas", "Lima Belas", "Enam Belas", "Tujuh Belas", "Delapan Belas",
        "Sembilan Belas", "Dua Puluh",
        "Tiga Puluh", "Empat Puluh", "Lima Puluh", "Enam
        Puluh", "Tujuh Puluh", "Delapan Puluh", "Sembilan Puluh"};

    string seratus = "Seratus";

    int num;
    cout << "Masukkan angka (0-100): ";
    cin >> num;

    if (num < 0 || num > 100)
    {
        cout << "Angka harus antara 0 dan 100." << endl;
    }
    else if (num <= 20)
    {
        cout << angka[num] << endl;
    }
    else if (num < 100)
    {
        int puluhan = num / 10;
        int satuan = num % 10;
        if (satuan == 0)
        {
            cout << angka[18 + puluhan] << endl;
        }
        else
        {
            cout << angka[18 + puluhan] << " " << angka[satuan] << endl;
        }
    }
    else
    {
        cout << seratus << endl;
    }

    return 0;
}
```

Hasilnya:

```

[  🍏 > ~/Developer/C++/struktur_d
./guided_02
Masukkan angka (0-100): 23
Dua Puluh Tiga

[  🍏 > ~/Developer/C++/struktur_d
./guided_02
Masukkan angka (0-100): 45
Empat Puluh Lima

[  🍏 > ~/Developer/C++/struktur_d
./guided_02
Masukkan angka (0-100): 100
Seratus

[  🍏 > ~/Developer/C++/struktur_d
./guided_02
Masukkan angka (0-100): 99
Sembilan Puluh Sembilan

[  🍏 > ~/Developer/C++/struktur_d

```

c. Nomor 3

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

int main()
{
    int n;
    cout << "Enter a value for n: ";
    cin >> n;
    for (int i = n; i > 0; i--)
    {
        for (int s = 0; s < n - i; s++)
        {
            cout << " ";
        }
        for (int j = i; j > 0; j--)
        {
            cout << j;
        }
        cout << "*";
        for (int k = 1; k <= i; k++)
        {
            cout << k;
        }
        cout << endl;
    }
    for (int s = 0; s < n; s++)
    {
        cout << " ";
    }
    cout << "*" << endl;
    return 0;
}
```

Hasilnya:

```
~/Developer/C++/struktur_data/01_Pengenalan
g++ -o guided_03 guided_03.cpp && ./guided_03
Enter a value for n: 5
54321*12345
 4321*1234
   321*123
    21*12
     1*1
      *

```

D. Unguided

1. (Input/Output) Tuliskan kode berikut dan jalankan. a) Masukkan nama lengkap anda dan nim anda. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban. b) Masukkan nama pertama anda dan nim anda. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.

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

int main()
{
    string nama, nim;
    cout << "Siapa nama anda? ";
    cin >> nama;
    cout << "Berapa nim anda? ";
    cin >> nim;
    cout << "Nama saya:" << nama << endl;
    cout << "NIM saya:" << nim << endl;
    return 0;
}
```

Hasilnya:

```
Apple > ~/Developer/C++/struktur_data/01_Pengenalan_CPP_
g++ -o unguided_01 unguided_01.cpp && ./unguided_01
Siapa nama anda? Novri
Berapa nim anda? 21104065
Nama saya:Novri
NIM saya:21104065
Apple > ~/Developer/C++/struktur_data/01_Pengenalan_CPP_
```

2. (Operasi aritmatika) Tuliskan kode berikut dan jalankan. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.

```
%23include <iostream>
using namespace std;

int main()
{
    int bil1 = 3, bil2 = 4, hasil1;
    float bil3 = 3.0, bil4 = 4.0, hasil2;
    hasil1 = bil1 %2B bil2;
    cout << hasil1 << endl;
    hasil1 = bil1 - bil2;
    cout << hasil1 << endl;
    hasil1 = bil1 * bil2;
    cout << hasil1 << endl;
    hasil1 = bil1 / bil2;
    cout << hasil1 << endl;
    hasil1 = bil2 / bil1;
    cout << hasil1 << endl;
    hasil1 = bil1 % bil2;
    cout << hasil1 << endl;
    hasil1 = bil2 % bil1;
    cout << hasil1 << endl;
    hasil2 = bil3 / bil4;
    cout << hasil2 << endl;
    return 0;
}
```

Hasilnya:

```
Apple > ~/Developer/C++/struktur_data/01_Pengenalan_CPP_
g++ -o unguided_02 unguided_02.cpp && ./unguided_02
7
-1
12
0
1
3
1
0.75
Apple > ~/Developer/C++/struktur_data/01_Pengenalan_CPP_
```

3. (Operasi perbandingan) Tuliskan kode berikut dan jalankan. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.

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

int main()
{
    int bill = 2, bil2 = 3, hasil;
    hasil = bill > bil2;
    cout << hasil << endl;
    hasil = bill >= bil2;
    cout << hasil << endl;
    hasil = bill < bil2;
    cout << hasil << endl;
    hasil = bill <= bil2;
    cout << hasil << endl;
    hasil = bill == bil2;
    cout << hasil << endl;
    hasil = bill != bil2;
    cout << hasil << endl;
    return 0;
}
```

Hasilnya:


```
Apple > ~/Developer/C++/struktur_data/01_Pengenalan_CPP
g++ -o unguided_03 unguided_03.cpp && ./unguided_03
0
0
1
1
0
1
Apple > ~/Developer/C++/struktur_data/01_Pengenalan_CPP
```

4. (Operasi logika) Tuliskan kode berikut dan jalankan. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.

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

int main()
{
    int bil1 = 2, bil2 = 3, hasil;
    hasil = bil1 <= bil2 and bil1 < bil2;
    cout << hasil << endl;
    hasil = bil1 >= bil2 or bil1 < bil2;
    cout << hasil << endl;
    hasil = not(bil1 >= bil2) or bil1 < bil2;
    cout << hasil << endl;
    return 0;
}
```

Hasilnya:

```
Apple > ~/Developer/C++/struktur_data/01_Pengenalan_CPP_Bagi
g++ -o unguided_04 unguided_04.cpp && ./unguided_04
1
1
1
Apple > ~/Developer/C++/struktur_data/01_Pengenalan_CPP_Bagi
```

5. (Percabangan if-else) Tuliskan kode berikut dan jalankan. Masukkan input 80, 81, dan 79. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.

```

#include <iostream>
using namespace std;

int main()
{
    int nilai;
    cin >> nilai;

    if (nilai > 80)
    {
        cout << "A" << endl;
    }
    else
    {
        cout << "Bukan A" << endl;
    }

    return 0;
}

```

Hasilnya:

```

[Apple] > ~/Developer/C++/struktur_data/01_Pengenalan_CPP_
g++ -o unguided_05 unguided_05.cpp && ./unguided_05
80
Bukan A
[Apple] > ~/Developer/C++/struktur_data/01_Pengenalan_CPP_
g++ -o unguided_05 unguided_05.cpp && ./unguided_05
81
A
[Apple] > ~/Developer/C++/struktur_data/01_Pengenalan_CPP_
g++ -o unguided_05 unguided_05.cpp && ./unguided_05
79
Bukan A
[Apple] >

```

6. (Perulangan for-to-do) Tuliskan kode berikut dan jalankan. Masukkan 1 dan 10. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.

```

#include <iostream>
using namespace std;

int main()
{
    int a, b, bilangan;

    cout << "Masukan batas bawah: ";
    cin >> a;

    cout << "Masukan batas atas: ";
    cin >> b;

    for (bilangan = a; bilangan <= b; bilangan++)
    {
        cout << "Bilangan " << bilangan << endl;
        return 0;
    }
}

```

Hasilnya:

```

Apple > ~/Developer/C++/struktur_data/01_Pengenalan_CPP_
g++ -o unguided_06 unguided_06.cpp && ./unguided_06
Masukan batas bawah: 1
Masukan batas atas: 10
Bilangan 1
Apple > ~/Developer/C++/struktur_data/01_Pengenalan_CPP_

```

7. (Perulangan while-do) Tuliskan kode berikut dan jalankan. Masukkan pada input bilangan 10. Screenshot kode dan hasilnya, lalu tempelkan pada jawaban.

```

#include <iostream>
using namespace std;

int main()
{
    int bilangan, asli, jumlah;

    cout << "Masukkan bilangan asli: ";
    cin >> asli;
    bilangan = 1;
    jumlah = 0;

    while (bilangan <= asli)
    {
        if (bilangan % 2 == 0)
        {
            jumlah += bilangan;
        }
        bilangan++;
    }

    cout << "Jumlah bilangan genap: " << jumlah << endl;
    return 0;
}

```

Hasilnya:

```

> ~/Developer/C++/struktur_data/01_Pengenalan_CPP_
g++ -o unguided_07 unguided_07.cpp && ./unguided_07
Masukkan bilangan asli: 10
Jumlah bilangan genap: 30
> ~/Developer/C++/struktur_data/01_Pengenalan_CPP_

```

E. Kesimpulan

C++ adalah bahasa pemrograman yang kuat dan serbaguna, sering digunakan untuk pengembangan perangkat lunak, game, aplikasi sistem, dan banyak lagi. Berikut adalah beberapa poin penting dari pengenalan C++:

- Sintaks Dasar:** C++ memiliki sintaks yang mirip dengan bahasa C, tetapi dengan tambahan fitur seperti pemrograman berorientasi objek. Struktur dasar program C++ mencakup fungsi `main()`, yang merupakan titik awal eksekusi program.
- Variabel dan Tipe Data:** C++ mendukung berbagai tipe data seperti `int`, `float`, `double`, `char`, dan `bool`. Variabel digunakan untuk menyimpan data yang dapat diubah selama eksekusi program.

- c. Kontrol Aliran: C++ menyediakan berbagai struktur kontrol aliran seperti if, else, for, while, dan do-while untuk mengatur alur eksekusi program berdasarkan kondisi tertentu.
- d. Fungsi: Fungsi adalah blok kode yang dapat dipanggil untuk menjalankan tugas tertentu. Fungsi membantu dalam modularisasi kode dan meningkatkan keterbacaan serta pemeliharaan program.
- e. Pemrograman Berorientasi Objek (OOP): C++ mendukung OOP, yang memungkinkan penggunaan konsep seperti kelas dan objek, enkapsulasi, pewarisan, dan polimorfisme. Ini membantu dalam menciptakan kode yang lebih terstruktur dan mudah dikelola.