UNIVERSITAS BINAWAN

Terbatas

Hanya untuk Mahasiswa

Universitas Binawan

MODUL ALGORITMA DAN PEMPROGRAMAN DASAR

Nur Fitriyani Sahamony.S.Pd.,M.Si

DAFTAR ISI

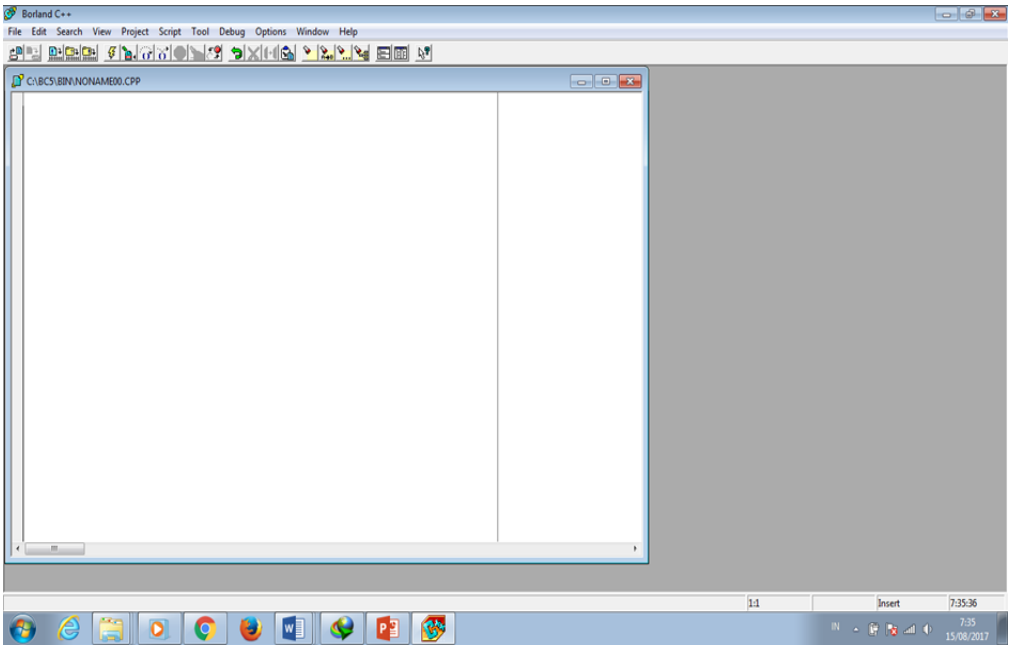
I. PENDAHULUAN

Algoritma merupakan urutan langkah-langkah logis untuk menyelesaikan sebuah masalah yang disusun secara sistematis. Setiap aksi harus dapat dikerjakan dan mempunyai efek tertentu. Algoritma dapat dituliskan dengan beberapa cara, yaitu dengan penggunaan bahasa alami yang digunakan sehari-hari, simbol grafik bagan alir, dan penggunaan bahasa pemograman seperti bahasa C atau C++.

Bahasa pendahulu C++ adalah C. Pencipta C adalah *Brian W. Kerninghan* dan *Dennis M. Ritchie* pada sekitar tahun 1972, dan sekitar satu dekade setelahnya diciptakanlah C++, oleh *Bjarne Stroustrup* dari Laboratorium Bell, AT&T, pada tahun 1983. C++ cukup kompatibel dengan bahasa pendahulunya C. Pada mulanya C++ disebut " a better C ". Nama C++ sendiri diberikan oleh *Rick Mascitti* pada tahun 1983, yang berasal dari operator increment pada bahasa C. Keistimewaan yang sangat berari dari C++ ini adalah karena bahasa ini mendukung pemrograman yang berorientasi objek (OOP / Object Oriented Programming).

Borland C++ adalah perangkat lunak untuk menyusun aplikasi yang berdasarkan pada bahasa pemrograman C dan bekerja dalam lingkungan sistem operasi windows. Langkah – langkah menuliskan program dalam Borland C++ sebagai berikut :

1. Bukalah software Borland C++, dan akan terlihat tampilan awal Borland C++ sebagai berikut :



2. Tulis source code program bahasa C++

   Source code C++ dapat ditulis pada text edit Borland C++.

3. Kompile file dengan /CTRL+ F9/ pilih submenu Run pada menu Debug

   Kompile file dijalankan Untuk mengubah source code menjadi sebuah program, kita gunakan compiler. Setelah source code tercompile, terbentuklah sebuah file objek dengan ekstension " .obj ". File " .obj " ini belum merupakan sebuah program executable.

4. Jalankan Program dengan /CTRL+F9 /pilih submenu Run pada menu Debug

   Setelah kita kompile file yang berisi source code, maka sebagai hasil kompilasi tersebut kita akan mendapatkan suatu file yang bisa dijalankan (executable file).

5. Untuk menyimpan pilih menu "Save As"

II. PENGENALAN C++

Program C++ memiliki bentuk umum seperti di bawah ini:

```
# prepocessor directive
void main()
{
// Batang Tubuh Program Utama
}
```

Penjelasan :

1. Include

   Adalah salah satu pengarah *prepocessor directive* yang tersedia pada C++. Preprocessor selalu dijalankan terlebih dahulu pada saat proses kompilasi terjadi.

   Bentuk umumnya :

   # include <nama_file>

   tidak diakhiri dengan tanda semicolon, karena bentuk tersebut bukanlah suatu bentuk pernyataan, tetapi merupakan *prepocessor directive*. Baris tersebut menginstrusikan kepada kompiler yang menyisipkan file lain dalam hal ini file yang berakhiran .h (file header) yaitu file yang berisi sebagai deklarasi, misalkan:

   - # include <iostream.h> : diperlukan pada program yang melibatkan objek cout
   - # include <conio.h> : diperlukan bila melibatkan clrscr(), yaitu perintah untuk membersihkan layar.
   - # include <iomanip.h> : diperlukan bila melibatkab setw() yang bermanfaat untuk mengatur lebar dari suatu tampilan data
   - # include <math.h> : diperlukan pada program yang menggunakan operasi sqrt() yang bermanfaat untuk operasi matematika kuadrat

2. Fungsi main ()

   Fungsi ini menjadi awal dan akhir eksekusi program C++. main adalah nama judul fungsi. Melihat bentuk seperti itu dapat diambil kesimpulan bahwa batang tubuh program utama berada didalam fungsi main( ). Berarti dalam setiap pembuatan program utama,

maka dapat dipastikan seorang pemrogram menggunakan minimal sebuah fungsi. Pembahasan lebih lanjut mengenai fungsi akan diterangkan kemudian.

3. Komentar

   Komentar tidak pernah dicompile oleh compiler. Dalam C++ terdapat 2 jenis komentar, yaitu :

   Jenis 1 : /* Komentar anda diletakkan di dalam ini

   Bisa mengapit lebih dari satu baris */

   Jenis 2 : // Komentar anda diletakkan disini (hanya bisa perbaris)

4. Tanda Semicolon

   Tanda semicolon " ; " digunakan untuk mengakhiri sebuah pernyataan. Setiap pernyataan harus diakhiri dengan sebuah tanda semicolon.

5. Mengenal cout(dibaca : C out)

   Pernyataan cout merupakan sebuah objek di dalam C++, yang digunakan untuk mengarahkan data ke dalam standar output (cetak pada layar)

   Contoh :

```
# include <iostream.h>

void main ()
{
   cout << " HAi, selamat menggunakan C++ ";
}
```

   Tanda " << " merupakan sebuah operator yang disebut operator "penyisipan/peletakan"

III. VARIABEL, TIPE DATA

1. VARIABEL

Variabel adalah suatu pengenal (identifier) yang digunakan untuk mewakili suatu nilai tertentu di dalam proses program. Berbeda dengan konstanta yang nilainya selalu tetap, nilai dari suatu variable bisa diubah-ubah sesuai kebutuhan. Untuk memperoleh nilai dari suatu variable digunakan pernyataan penugasan (*assignment statement*), yang mempunyai sintaks sebagai berikut :

```
Variabel = ekspresi ;
```

Nama dari suatu variable dapat ditentukan sendiri oleh pemrogram dengan aturan sebagai berikut :

1. Terdiri dari gabungan huruf dan angka dengan karakter pertama harus berupa huruf. Bahasa C ++ bersifat ***case-sensitive*** artinya huruf besar dan kecil dianggap berbeda. Jadi antara **nim, NIM** dan **Nim** dianggap berbeda.
2. Tidak boleh mengandung spasi.
3. Tidak boleh mengandung simbol-simbol khusus, kecuali garis bawah (underscore). Yang termasuk symbol khusus yang tidak diperbolehkan antara lain : $, ?, %, #, !, &, *, (,), -, +, =dsb.
4. Panjangnya bebas, tetapi hanya 32 karakter pertama yang terpakai.

Contoh penamaan variabel yang benar :

NIM, a, x, nama_mhs, f3098, f4, nilai, budi, dsb.

Contoh penamaan variable yang salah :

%nilai_mahasiswa, 80mahasiswa, rata-rata, ada spasi, penting!, dsb

2. DEKLARASI

Deklarasi diperlukan bila kita akan menggunakan pengenal (identifier) dalam program. Identifier dapat berupa variable, konstanta dan fungsi.

1. Deklarasi Variabel

Bentuk umumnya :

```
Nama_tipe nama_variabel ;
```

Contoh :

int x; // *Deklarasi x bertipe integer*

char y, huruf, nim[10]; // *Deklarasi variable bertipe char* float nilai; // *Deklarasi variable bertipe float*

double beta; // *Deklarasi variable bertipe double*

int array[5][4]; // *Deklarasi array bebertipe integer*

Contoh :

```
#include <iostream.h>
void main()
{
int n;
n=66;  // sama juga jika ditulis int n=66;
   cout<<n<<endl;    // n sebagai variabel
   cout<<'n'<<endl;    // end sebagai karakter
}
```

Outputnya :

```
66
n
```

2. Deklarasi Konstanta

a. Menggunakan keyword const

Contoh : const float PI = 3.14152965;

Berbeda dengan variable, konstanta bernama tidak dapat diubah jika telah diinisialisasi

b. Menggunakan #define

Contoh : #define PI 3.14152965

Keuntungan menggunakan #define apabila dibandingkan dengan **const** adalah kecepatan kompilasi, karena sebelum kompilasi dilaksanakan, kompiler pertama kali mencari symbol #define (oleh sebab itu mengapa # dikatakan prepocessor directive) dan mengganti semua Phi dengan nilai 3.14152965.

3. TIPE DATA

Tipe data dapat dikelompokkan menjadi atas dua macam :

1. Tipe Dasar.

   Adalah tipe yang dapat langsung dipakai.

Tipe Dasar	Ukuran Memori (byte)	Jangkauan Nilai	Jumlah Digit Presisi
Char	1	-128 hingga +127	-
Int	2	-32768 hingga +32767	-
Long	4	-2.147.438.648 hingga 2.147.438.647	-
Float	4	3,4E-38 hingga 3,4E38	6-7
Double	8	1.7E-308 hingga 1.7E308	15-16
long double	10	3.4E-4932 hingga 1.1E4932	19

   NB : Untuk mengetahui ukuran memori dari suatu tipe digunakan fungsi sizeof(tipe)

2. Tipe Bentukkan.

   Merupakan tipe yang dibentuk dari tipe dasar. Seperti Tipe Struktur.

KARAKTER & STRING LITERAL

String adalah gabungan dari karakter

Contoh : " Belajar "→ Literal String

"B"→Karakter Panjang

String

strlen() → nama fungsi untuk menghitung panjang string

Fungsi strlen() dideklarasikan dalam file string.h. Jadi bila anda ingin menggunakan fungsi strlen(), maka prepocessor directive #include<string.h> harus dimasukkan dalam program diatas main().

Contoh :

```
#include <iostream.h>
#include <string.h>
void main()
{
  cout<<strlen("Selamat Pagi.\n")<<endl;
  cout<<strlen("Selamat Pagi.")<<endl;
  cout<<strlen("Selamat")<<endl;
  cout<<strlen("S")<<endl;
  cout<<strlen("");
}
```

Outputnya:

```
14
13
7
1
0
```

Dalam C++, selain \n terdapat juga beberapa karakter khusus yang biasa disebut *escape sequence characters*, yaitu

Karakter	**Keterangan**
\0	Karakteeer ber-ASCII nol (karakter null)
\a	Karakter bell
\b	Karakter backspace
\f	Karakter ganti halaman (formfeed)
\n	Karakter baris baru (newline)
\r	Karakter carriage return (ke awal baris)
\t	Karakter tab horizontal
\v	Karakter tab vertika
\\	Karakter \
\’	Karakter ‘
\”	Karakter “
\?	Karakter ?
\ooo	Karakter yang nilai oktalnya adalah ooo (3 digit octal)
\xhh	Karakter yang nilai heksadesimalnya adalah hh (2 digit heksadesimal)

KEYWORD & IDENTIFIER

Dalam bahasa pemrograman, suatu program dibuat dari elemen-elemen sintaks individual yang disebut token, yang memuat nama variable, konstanta, keyword, operator dan tanda baca.

Contoh :

```
# include <iostream.h>
void main()
{
  int n=66;
  cout<<n<<endl;    // n sebagai variabel
}
```

Output :

```
66
```

Token int, return dan endl adalah suatu keyword

Token = dan << adalah operator

Token(,), {, ;, dan } adalah tanda baca

Baris pertama berisi suatu prepocessor directive yang bukan bagian sebenarnya dari program

TIPE STRUKTUR

Suatu tipe data yang merupakan kumpulan dari tipe data lainnya. Struktur terdiri dari data yang disebut field. Field – field tersebut digabungkan menjadi satu tujuan untuk kemudahan dalam operasi. Bentuk umumnya :

```
tyedef struct{ tipe nama_field1;
               tipe nama_field2;
               tipe nama_field3;
               . . . .
              }nama_variabel;
```

Contoh :

```
# include <iostream.h>

typedef struct { int tahun;
                 int bulan;
                 int tanggal;
               } data_tunggal;

data_tunggal tanggal_lahir;
void main ()
{
   tanggal_lahir.tanggal=13;
   tanggal_lahir.bulan=1;
   tanggal_lahir.tahun=1982;

   cout << tanggal_lahir.tanggal << '/'
        << tanggal_lahir.bulan << '/'
        << tanggal_lahir.tahun << endl;
}
```

Latihan :

1. Buatlah program dengan menggunakan define untuk menghitung volume Tabung (Rumus Volume Tabung : phi x jari-jari x jari-jari x tinggi) dan Luas Tabung (Rumus Luas tabung : 2 x phi x jari-jari x tinggi)dimana jari-jari 14 dan tinggi 18.
2. Buatlah program untuk mencatat data mahasiswa yang terdiri dari field nama, nim dan nilai.

IV. OPERATOR

Operator adalah symbol yang biasa dilibatkan dalam program untuk melakukan sesuatu operasi atau manipulasi.

1. OPERATOR PENUGASAN

Operator Penugasan (*Assignment operator*) dalam bahasa C++ berupa tanda sama dengan ("=").

Contoh :

```
nilai = 80;
A = x * y;
```

Penjelasan :

variable "nilai" diisi dengan 80 dan

variable "A" diisi dengan hasil perkalian antara x dan y.

2. OPERATOR ARITMATIKA

Operator	Deskripsi	Contoh
+	Penjumlahan (Add)	m + n
-	Pengurangan (Substract)	m – n
*	Perkalian (Multiply)	m * n
/	Pembagian (Divide)	m / n
%	Sisa Pembagian Integer (Modulus)	m % n
-	Negasi (Negate)	-m

Operator % (modulus) digunakan untuk mencari sisa pembagian antara dua bilangan.

Misalnya : 9 % 2 = 1, 9 % 3 = 0

Contoh :

```
#include <iostream.h>
void main()
{
    int m = 82, n = 26;
    cout<<m<<" + "<<n<<" = "<<m+n<<endl;
    cout<<m<<" - "<<n<<" = "<<m-n<<endl;
    cout<<m<<" * "<<n<<" = "<<m*n<<endl;
    cout<<m<<" / "<<n<<" = "<<m/n<<endl;
    cout<<m<<" % "<<n<<" = "<<m%n<<endl;
    cout<<"-"<<m<<" = "<<-m<<endl;
}
```

Output :

```
82 + 26 = 108
82 - 26 = 56
82 * 26 = 2132
82 / 26 = 3
82 % 26 = 4
-82 = -82
```

Karena tipe datanya adalah int, maka 82/26=3, supaya dapat merepresentasikan nilai yang sebenarnya, gunakan tipe data float.

3. OPERATOR HUBUNGAN (PERBANDINGAN)

Operator Hubungan digunakan untuk membandingkan hubungan antara dua buah operand (sebuah nilai atau variable). Operator hubungan dalam bahasa C++

Operator	Arti
= =	Sama dengan (bukan assignment)
!=	Tidak sama dengan
>	Lebih besar
<	Lebih kecil
>=	Lebih besar atau sama dengan
<=	Lebih kecil atau sama dengan

4. OPERATOR NAIK DAN TURUN (INCREMENT DAN DECREMENT)

Operator increment → ++

Operator decrement → --

Contoh :

```
#include <iostream.h>
void main()
{
   int m = 44, n = 66;

   cout<<"m = "<<m<<", n = "<<n<<endl;
   ++m; --n;
   cout<<"m = "<<m<<", n = "<<n<<endl;
   m++; n--;
   cout<<"m = "<<m<<", n = "<<n<<endl;
}
```

Outputnya :

```
m = 44, n = 66
m = 45, n = 65
m = 46, n = 64
```

5. OPERATOR BITWISE

Operator	Deskripsi	Contoh
<<	Geser n bit ke kiri (left shift)	m << n
>>	Geser n bit ke kanan (right shift)	m >> n
&	Bitwise AND	m & n
|	Bitwise OR	m | n
^	Bitwise XOR	m ^ n
~	Bitwise NOT	~m

Contoh :

```
# include <iostream.h>
void main()
{
   int m = 82, n = 26;
   cout<<m<<" << 2"<<" = "<<(m<<2)<<endl;
   cout<<m<<" >> 2"<<" = "<<(m>>2)<<endl;
   cout<<m<<" & "<<n<<" = "<<(m&n)<<endl;
   cout<<m<<" | "<<n<<" = "<<(m|n)<<endl;
   cout<<m<<" ^ "<<n<<" = "<<(m^n)<<endl;
   cout<<"~82 = "<<~m<<endl;
}
```

Output :

```
82 << 2 = 328
82 >> 2 = 20
82 & 26 = 18
82 | 26 = 90
82 ^ 26 = 72
~82 = -83
```

6. OPERATOR LOGIKA

Operator logika digunakan untuk menghubungkan dua atau lebih ungkapan menjadi sebuah ungkapan berkondisi.

Operator	Deskripsi	Contoh
&&	logic AND	m && n
||	logic OR	m ||n
!	logic NOT	!m

Contoh :

```
#include <iostream.h>
void main()
{
   int m = 166;
   cout<<"(m>=0 && m<=150) -> "<<(m>=0 && m<=150)<<endl;
   cout<<"(m>=0 || m<=150) -> "<<(m>=0 || m<=150)<<endl;
}
```

Outputnya :

```
(m>=0 && m<=150) -> 0
(m>=0 || m<=150) -> 1
```

7. OPERATOR KONDISI

Operator kondisi digunakan untuk memperoleh nilai dari dua kemungkinan *ungkapan1* dan *ungkapan2* : *ungkapan3*

Bila nilai *ungkapan1* benar, maka nilainya sama dengan *ungkapan2*, bila tidak maka nilainya sama dengan *ungkapan3*

Contoh :

```
#include <iostream.h>
void main()
{
   int m = 26, n = 82;
   int min = m < n ? m : n;
   cout<<"Bilangan terkecil adalah "<<min<<endl;
}
```

Outputnya :

```
Bilangan terkecil adalah 26
```

V. PEMILIHAN

Suatu Struktur dasar algoritma yang memiliki satu atau lebih kondisi tertentu dimana sebuah instruksi dilaksanakan jika sebuah kondisi/persyaratan terpenuhi. Ada beberapa bentuk struktur dasar pemilihan ini :

1. **Pernyataan if**

   Sebuah pernyataan yang dapat dipakai muntuk mengambil keputusan berdasarkan suatu kondisi. Bentuk pernyataan ini ada dua macam :

   - if saja dan
   - else

   Bentuk Umumnya Satu Kasus:

   ```
   if (kondisi)
       pernyataan ;
   ```

   Pernyataan dilaksanakan jika dan hanya jika kondisi yang diinginkan terpenuhi, jika tidak program tidak memberikan hasil apa-apa. Contoh :

```
// Contoh Penggunaan if

# include <iostream.h>

void main ()
{
   int usia;

   cout << "Berapa usia Anda : ";
   cin >> usia;

   if (usia < 17)
      cout << "Anda tidak diperkenankan menonton" << endl;
}
```

   Bentuk Umumnya Dua Kasus :

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

   Pernyataan1 dilaksanakan jika dan hanya jika kondisi yang diinginkan terpenuhi, jika tidak, lakukan pernyataan2.

   Jika Anda tidak mempergunakan pernyataan else program tidak akan error, namun jika anda mempergunakan pernyataan else tanpa didahului pernyataan if, maka program akan error.

   Bentuk Umumnya Banyak Kasus :

```
if (kondisi)
{
        pernyataan1;
        pernyataan1a;
        pernyataan1b;
}
else
{
pernyataan2;
pernyataan2a;
pernyataan2b;
}
```

Contoh :

```
# include <iostream.h>

void main ()
{
   int usia;

   cout << "Berapa usia Anda ? ";
   cin >> usia;

   if (usia < 17)
      cout << "Anda tidak diperkenankan menonton" << endl;
   else
      cout << "Selamat Menonton" << endl;
}
```

2. Pernyataan Switch

Pernyataan swich adalah pernyataan yang digunakan untuk menjalankan salah satu pernyataan dari beberapa kemungkinan pernyataan, berdasarkan nilai dari sebuah ungkapan dan nilai penyeleksian.

Pernyataan if…else if jamak dapat dibangun dengan pernyataan switch.

Bentuk Umumnya :

```
switch (ekspresi)

{
case konstanta1 :
```

```
pernyataan1 ;
break ;
case konstanta2 :
pernyataan2 ;
break ;
case konstanta3 :
pernyataan3 ;
break ;
:
:
case konstantaN :
pernyataanN ;
break ;
default :
pernyataanlain;

}
```

Latihan :

1. Buatlah program untuk mencari apakah bilangan tersebut ganjil atau genap, dimana bilangan merupakan piranti masukkan
2. Buatlah program untuk menseleksi suatu bilangan dengan ketentuan sebagai berikut :
   0<=nilai <30 : Nilai rendah
   30<=nilai < 60 : Nilai sedang
   60<=nilai<=100 : Nilai tinggi

VI. PENGULANGAN

Sebuah / kelompok instruksi diulang untuk jumlah pengulangan tertentu. Baik yang terdifinisikan sebelumnya ataupun tidak.

Struktur pengulangan terdiri atas dua bagian :

1. Kondisi pengulangan yaitu ekspresi boolean yang harus dipenuhi untuk melaksanakan pengulangan
2. Isi atau badan pengulangan yaitu satu atau lebih pernyataan (aksi) yang akan diulang.

Perintah atau notasi dalam struktur pengulangan adalah :

1. Pernyataan while

   Pernyataan while merupakan salah satu pernyataan yang berguna untuk memproses suatu pernyataan atau beberapa pernyataan beberapa kali. Pernyataan while memungkinkan statemen-statemen yang ada didalamnya tidak diakukan sama sekali.

   Bentuk umum :

   ```
   while (kondisi)
   {
   Pernyataan ;
   }
   ```

2. Pernyataan do..while

   Pernyataan do…while mirip seperti pernyataan while, hanya saja pada do…while pernyataan yang terdapat didalamnya minimal akan sekali dieksekusi.

   Bentuk Umumnya :

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

3. Pernyataan for

   Pernyataan for digunakan untuk menghasilkan pengulangan(looping) beberapa kali tanpa penggunaan kondisi apapun. Pada umumnya looping yang dilakukan oleh for telah diketahui batas awal, syarat looping dan perubahannya.

   Bentuk Umumnya :

   ```
   for (inisialisasi ; kondisi ; perubahan)
   {
   ```

Statement;

}

4. Pernyataan continue dan break

   Pernyataan break akan selalu terlihat digunakan bila menggunakan pernyataan switch. Pernyataan ini juga digunakan dalam loop. Bila pernyataan ini dieksekusi, maka akan mengakhiri loop dan akan menghentikan itrasi pada saat tersebut.

5. Pernyataan go to

   Pernyataan go to, diperlukan untuk melakukan suatu lompatan ke suatu pernyataan berlabel yang ditandai dengan tanda " : ".

   Bentuk Umumnya :

```
goto bawah;

pernyataan1;
pernyataan2;
bawah : pernyataan 3;
```

Latihan

Buatlah program yang menampilkan 5 buah bilangan, yaitu mulai dari bilanagn ke 5 sampai bilangan ke 1 dengan nilai awal bilangan 8. Tampilan bilangan tersebut adalah menurun dan contohnya adalah : bilangan ke 5, i=3 (diperoleh dari 8-5) dan seterusnya sampai bilangan 1, i=7 (diperoleh dari 8-1=7)