BAB 3
PERINTAH INPUT – OUTPUT

1. Perintah Output

Perintah Output adalah perintah yang digunakan untuk mengeluarkan hasil proses komputer sehingga bisa dibaca oleh si pemakai (*user*). Adapun media yang digunakan untuk menampilkan hasil output tadi bisa berupa monitor atau dicetak ke kertas melalui printer. Salah satu fungsi untuk menampilkan output dalam Turbo C++ adalah fungsi ***printf()***.

Fungsi ***printf()*** digunakan untuk mencetak data baik berupa teks, numerik, konstanta maupun variabel. Contoh penggunaan ***printf()*** bisa dilihat pada contoh program berikut :

```
/*
   Contoh macam-macam penggunaan perintah printf()
*/
#include <stdio.h>
#include <conio.h>

void main()
{
   int jml=2;
   float hrg=4000,ttl=jml*hrg;

   clrscr();
   printf("Selamat Datang di Turbo C++ \n");
   printf("Satu unit %s seharga %f maka jika \n","Komputer",hrg);
   printf("Anda membeli sebanyak %d, maka harganya %f \n",jml,ttl);
   getch();
}
```

Program 3.1 Contoh Pemakaian *printf()*

Bisa dilihat dari contoh diatas, bahwa perintah *printf()* bisa diikuti dengan tanda %s, %f, %d dan lain-lain. Tanda % tersebut disebut dengan **penentu format** *(format specifier)*.

Perintah lain untuk menampilkan output adalah dengan ***puts()*** dan ***putchar()***

- Fungsi ***printf()*** digunakan untuk menampilkan semua jenis data (numeric dan karakter)
- Fungsi ***puts()*** digunakan untuk menampilkan data string dan secara otomatis akan diakhiri dengan perpindahan baris.
- Fungsi ***putchar()*** digunakan untuk menampilkan sebuah karakter.

2. Penentu Format (*Format Specifier*)

Penggunaan *penentu format* sangat berkaitan dengan tipe data yang akan dicetak, artinya setiap tipe data mempunyai *penentu format* masing-masing. Tabel berikut merupakan tabel penentu format untuk masing-masing tipe data.

Tabel 3.1 Tabel Penentu Format *printf()*

No	Tipe Data	Penentu Format untuk printf()
1	Integer	%d
2	Floating Point	
	✓ Bentuk Desimal	%f
	✓ Bentuk Berpangkat	%e
	✓ Yang lebih pendek antara Desimal dan Berpangkat	%g
3	Double Precision	%lf
4	Character	%c

5	String	%s
6	Unsigned Integer	%u
7	Long Integer	%ld
8	Long unsigned integer	%lu
9	Unsigned Hexadecimal Integer	%x
10	Unsigned Octal Integer	%o

Yang harus diperhatikan disini adalah bahwa urutan dari letak penentu format harus sesuai dengan urutan konstanta atau variabel yang akan mengisinya, tentunya untuk penentu format yang lebih dari satu. Hal ini dikarenakan, penentu format untuk masing-masing tipe data berbeda-beda.

3. Penentu Lebar Field (*Field Width Specifier*)

Bila kita mencetak data yang bertipe *float*, sering kali terlihat tampilan yang kurang bagus, misalnya angka desimal yang tercetak terlalu banyak. Sebagai contoh :

```
printf("Nilai Rata-Rata Anda = %f ",80.25);
```

maka pada saat program diatas dijalankan, output yang tampil sebagai berikut :

```
Nilai Rata-Rata Anda = 80.250000
```

Jumlah angka desimal, sebenarnya bisa kita atur, demikian juga dengan lebar data (lebar *field*). Untuk mengatur format *float* tersebut dengan bentuk sebagai berikut :

%a.bf	atau	%-a.bf

Ket :
a : lebar field
b : jumlah desimal
- : untuk mengatur rata kiri

Maka agar tampilan program diatas lebih bagus, maka harus diberikan format sebagai berikut :

```
printf("Nilai Rata-Rata Anda = %5.2f \n",80.25);
```

maka pada saat program diatas dijalankan, output yang tampil sebagai berikut :

```
Nilai Rata-Rata Anda = 80.25
```

Kebanyakan dari contoh-contoh program diatas banyak terdapat pernyataan \n disetiap akhir string yang dicetak. Tanda tersebut (\n) disebut dengan *Escape Sequence*. \n artinya data yang dicetak setelahnya akan di tampilkan di baris baru dengan kata lain \n berfungsi untuk berpindah baris pencetakan.

4. Escape Sequences

Disebut *Escape Sequence* karena notasi '\' dianggap sebagai karakter escape (menghindar) dalam arti bahwa karakter yang terdapat setelah tanda '\' dianggap bukan teks biasa, jadi karakter ini dilarikan dari pengertian sebagai teks biasa. Beberapa *escape sequence* bisa dilihat pada tabel berikut.

Tabel 3.2 *Escape Sequence*

No	*Escape Sequence*	Pengertian
1	\b	Backspace (mundur satu spasi)
2	\f	Formfeed (ganti halaman)
3	\n	Ganti Baris Baru (new line)
4	\r	*Carriage Return*
5	\t	Tab (*default* = 8 karakter*)*
6	\’	Tanda Kutip Tunggal (‘)
7	\”	Tanda Kutip Ganda (“)
8	\\	Backslash
9	\xaa	Kode ASCII dalam Hexadesimal (aa : angka ascii)
10	\aaa	Kode ASCII dalam Oktal (aaa : angka ascii)
11	\a	Bunyi bell (*alert*)

Program berikut merupakan contoh dari penggunaan *Escape Sequence*.

```
/*
   Contoh Penggunaan Escape Sequence & Penentu Format
   \t = tabulasi, \n = pindah baris
   %6.2f = 6 digit 2 desimal untuk format float
*/
#include <stdio.h>
#include <conio.h>

void main()
{
   float bil1=3.14,bil2=8.50,bil3=88.0;
   float bil4=13.7,bil5=70.80,bil6=100.45;
   clrscr();
   printf("\'NKR\' Singkatan dari \"Naya Kartika Ramadhani\" \n\n");
   printf("\n Tekan Enter...!\n\n");getch();
   printf("Escape Sequence tab(\\t) \n");
   printf("------------------------\n");
   printf("%6.2f \t%6.2f \t%6.2f \n",bil1,bil2,bil3);
   printf("%6.2f \t%6.2f \t%6.2f \n",bil4,bil5,bil6);
   printf("------------------------\n");
   getch();
}
```

Program 3.2 Pemakaian *Escape Sequence*

Jika program 3.2 dijalankan, maka akan menghasilkan output sebagai berikut :

```
'NKR' Singkatan dari "Naya Kartika Ramadhani"


Tekan Enter…!

Escape Sequence tab(\t)
------------------------------------
 3.14      8.50     88.00
13.70     70.80    100.45
------------------------------------
```

5. Mencetak Kode ASCII

Pada saat tertentu, kita memerlukan mencetak sebuah karakter ASCII, terutama pada saat kita membutuhkan sebuah tampilan yang lebih menarik, misalkan kita ingin membuat sebuah kotak. Di Turbo C++ fasilitas untuk mencetak kode ASCII bisa menggunakan *Escape Sequence* \x dan diikuti kode Hexadesimal dari kode karakter yang bersangkutan. Program dibawah ini mencontohkan cara membuat kotak dengan kode ASCII.

```
/*
   Contoh Penggunaan Escape Sequence
   untuk mencetak kode ASCII
*/
#include <stdio.h>
#include <conio.h>

void main()
{
  clrscr();
  printf("\xDA\xC4\xC4\xC4\xBF\n"); /* KiriAtas Datar3x KananAtas */
  printf("\xB3   \xB3\n");          /* GarisVertikal spasi3x GarisVertikal */
  printf("\xC0\xC4\xC4\xC4\xD9\n"); /* KiriBawah Datar3x KananBawah */
  getch();
}
```

Program 3.3 Mencetak Kode ASCII

Hasil dari program 3.3 adalah sebagai berikut :

6. Menampilkan data ke printer

Untuk menampilkan data ke printer dapat menggunakan fungsi fprintf(), fputs() dan fputc().

- Fungsi fprintf() digunakan untuk mencetak semua jenis tipe data ke printer dan secara otomatis memberikan efek perpindahan baris.
- Fungsi fputs() digunakan untuk mencetak tipe data string ke printer
- Fungsi fputc() digunakan untuk mencetak tipe data karakter ke printer

```
Contoh program :
#include "stdio.h"
#include "conio.h"
void main()
{
      fprintf(stdprn, "Hallo, Saya akan tercetak di printer");
      fputs(stdprn, "Saya juga akan tercetak di printer");
}
```

7. Perintah Input

Setiap bahasa pemrograman tidak akan bisa digunakan secara fleksibel jika tidak memiliki perintah input. Perintah input adalah sebuah perintah dalam bahasa program yang mampu meneruskan nilai dari operator untuk diproses oleh komputer. Perintah input memerlukan perangkat keras input, biasanya adalah keyboard. Dalam Turbo C++, terdapat tiga perintah input yaitu *scanf(), getche(), getch()* dan *gets()*.

a. Fungsi scanf()

Bentuk umum dari fungsi *scanf()* adalah sebagai berikut :

```
scanf("penentu format",&namavariabel);
```

Penggunaan *scanf()* biasanya dikombinasikan dengan perintah *printf()*. Perintah *printf()* disini berfungsi sekedar menampilkan keterangan tentang apa yang harus diinputkan, sehingga operator bisa langsung mengerti harus memasukkan data apa.

Untuk penentu format pada *scanf()*, dapat dilihat pada tabel berikut :

Tabel 3.3 Tabel Penentu Format *scanf()*

No	Tipe Data	Penentu Format untuk printf()
1	Integer	%d
2	Floating Point	
	✓ Bentuk Desimal	%e atau %f
	✓ Bentuk Berpangkat	%e atau %f
3	Double Precision	%lf
4	Character	%c
5	String	%s
6	Unsigned Integer	%u
7	Long Integer	%ld
8	Long unsigned integer	%lu
9	Unsigned Hexadecimal Integer	%x
10	Unsigned Octal Integer	%o

Selain itu, penggunaan *scanf()* juga harus menyertakan tanda '&' pada awal nama variabel. Tanda '&' disini befungsi sebagai operator alamat (*address operator*).

Contoh penggunaan *scanf()* sebagai berikut :

```
/*
  Perintah input : berfungsi untuk memasukkan data dari media keyboard
  Syntax :
     scanf("penentu format",&namavariabel);

*/
#include <stdio.h>
#include <conio.h>
#include <string.h>

void main()
{   float pjg,lbr,ls ;
    char nama[25];

    clrscr();
    printf("Masukkan Nama Anda       : ");scanf("%s",nama);fflush(stdin);
    printf("Masukkan Panjang Persegi : ");scanf("%f",&pjg);
    printf("Masukkan Lebar Persegi   : ");scanf("%f",&lbr);
    ls = pjg * lbr;
    printf("Saudara %s Luas Persegi Panjang Anda : %5.2f",nama,ls);
    getch();
}
```

Program 3.4 Pemakaian fungsi *scanf()*

Jika terdapat beberapa proses input (memasukkan data) sekaligus, maka sebaiknya ditambahkan fungsi **fflush(stdin);** setelah fungsi scanf(). Fungsi fflush(stdin) berfungsi menghapus buffer di dalam alat I/O.

Scanf() juga bisa digunakan untuk menginputkan beberapa data sekaligus dalam satu baris asalkan jumlah dan tipe penentu format sesuai dengan variabel yang akan diinputkan. Data yang akan dimasukkan dapat dipisahkan dengan s*pasi*, *tab* atau tanda pemisah lain seperti *koma* (,), *garis hubung*(-), atau *titik dua*(:). Pemisah data dalam input yang digunakan harus sama dengan pemisah data dalam *scanf()*.

Contoh 1 :

```
printf("Masukkan 3 bilangan : ");
scanf("%d %f %d",&bil1,&bil2,&bil3);
```

Hasil :

```
Masukkan 3 bilangan : 10 2.5 150
```

Contoh 2 :

```
printf("Masukkan Jam Masuk (hh:mm:ss): ");
scanf("%d:%d:%d",&hh,&mm,&ss);
```

Hasil :

```
Masukkan Jam Masuk (hh:mm:ss): 08:35:12
```

Contoh 3 :

```
printf("Masukkan Tgl. Lahir Anda (dd/mm/yy) :");
scanf("%d/%d/%d",&dd,&mm,&yy);
```

Hasil :

```
Masukkan Tgl. Lahir Anda (dd/mm/yy) : 29/09/08
```

Hal-hal yang perlu diperhatikan dalam pemakaian fungsi scanf() :

a. Fungsi scanf() memakai penentu format
b. Fungsi scanf() memberi pergantian baris secara otomatis
c. Fungsi scanf() tidak memerlukan penentu lebar field
d. Variabelnya harus menggunakan operator alamat &

b. Fungsi getche() dan Fungsi getch()

Fungsi input *getche()* memiliki sifat yang sedikit berbeda dari *scanf()*. Perbedaan tersebut antara lain :

1. Bila dalam *scanf()* jumlah karakter data yang diinputkan boleh bebas, maka dalam *getche()* hanya sebuah karakter yang bisa diterima.
2. Bila *scanf()* membutuhkan tombol RETURN/ENTER untuk mengakhiri input, maka dalam *getche()* tidak membutuhkannya. Input dianggap selesai begitu kita memasukkan satu karakter dan secara otomatis akan melanjutkan ke baris perintah berikutnya.

getche() merupakan singkatan dari *get character and echo* yang artinya 'menerima sebuah karakter kemudian tampilkan'. Input yang diterima *getche()* akan disimpan ke dalam variabel karakter yang sebelumnya harus sudah dideklarasikan. Contoh penggunaannya seperti statemen dibawah ini :

```
x=getche();
```

Jadi variabel x akan menyimpan data yang diinputkan melalui *getche()*.

Fungsi input lain yang mirip dengan *getche()* adalah *getch()*. Satu-satunya perbedaan antara *getche()* dan *getch()* adalah *getche()*akan menampilkan karakter yang kita ketikkan, sedangkan *getch()* tidak akan menampilkan, melainkan hanya menyimpannya dalam memori saja, jadi apa yang kita ketikkan tidak akan muncul dilayar sebelum kita memberikan perintah untuk mencetak nilai tersebut.

a. Fungsi getch() dan getche() digunakan untuk membaca data karakter.
b. Karakter yang dimasukkan tidak perlu diakhiri dengan penekanan tombol enter.
c. Tidak memberikan efek pergantian baris secara otomatis
d. Jika menggunakan fungsi getch() karakter yang dimasukkan tidak akan ditampilkan pada layar sehingga sering digunakan untuk meminta inputan berupa password.
e. Sedangkan pada getche() karakter yang dimasukkan akan ditampilkan pada layar.

Contoh:

```
/*
   Contoh penggunaan scanf() untuk input beberapa data sekaligus
   Program Konversi Jam:Menit menjadi menit.
*/
#include <stdio.h>
#include <conio.h>

void main()
{
   int jam,menit,hasil;
   clrscr();
   /* Cetak Judul */
   printf("\n\xDB\xDB\xDB Program Konversi Jam \xDB\xDB\xDB");

   /* inputkan jam:menit */
   printf("\n\nInputkan Jam dan Menit (jam:menit) : ");
   scanf("%d:%d",&jam,&menit);

   /* Hitung konversi */
   hasil=jam*60+menit;

   /* Cetak Hasil */
   printf("\n\n%d jam %d menit = %d menit",jam,menit,hasil);
   getche();
   }
```

Program 3.5 Pemakaian fungsi *scanf()* untuk input beberapa data

c. Fungsi gets()

a. Fungsi gets() digunakan untuk memasukkan data bertipe karakter dan tidak dapat digunakan untuk memasukkan data numerik.

b. Harus diakhiri dengan penekanan tombol enter

c. Cursor secara otomatis akan pindah baris

d. Tidak memerlukan penentu format

Contoh pemakaian gets

```
/* Program inputan tipe data karakter/string */
#include <stdio.h>
#include <conio.h>
void main()
{
   char nama[20],alamat[50],tmp_lhr[15],tgl_lhr[15];
   clrscr();
   printf("Masukkan nama Anda          : ");gets(nama);
   printf("Masukkan alamat Anda        : ");gets(alamat);
   printf("Masukkan tempat Lahir Anda  : ");gets(tmp_lhr);
   printf("Masukkan Tanggal Lahir Anda : ");gets(tgl_lhr);

   printf("Hello, Nama Anda adalah %s\n", nama);
   printf("Anda saat ini tinggal di %s\n", alamat);
   printf("Sedangkan tempat lahir Anda di %s\n", tmp_lhr);
   printf("Dan Anda dilahirkan pada tanggal %s\n", tgl_lhr);
   getch();
}
```

Program 3.6 Pemakaian fungsi *gets()* untuk input data string

d. getchar()

a. Fungsi getchar() digunakan untuk membaca data yang bertipe karakter
b. Harus diakhiri dengan penekanan tombol enter
c. Karakter yang dimasukkan terlihat pada layar
d. Pergantian baris secara otomatis

Soal 3.A

1. Tuliskan perintah output dalam Turbo C++, berikan contohnya!
2. Dalam menuliskan output terdapat bagian yang merupakan penentu format. Jelaskan apa yang dimaksud dengan penentu format!
3. Bagaimana format penulisan perintah *printf* dengan menggunakan penentu format, jika diketahui nilai dari variabel-variabel sebagai berikut :

   Hidup = 5, mati = 1, sisa=4

   Dan ingin menampilkan kalimat berdasarkan nilai diatas seperti dibawah ini :

   Ayam 5 mati 1 tinggal 4
4. Tuliskan simbol penentu format dari integer, float, long integer, dan long unsign integer dan berikan contohnya dalam penggalan program !
5. Bagaimana menentukan lebar field dari sebuah tipe data float jika kita menginginkan sebuah variabel float dicetak dengan 9 digit dan 2 desimal? Berikan contohnya!
6. Jelaskan apa yang dimaksud dengan *Escape Sequence*?
7. Jelaskan apa fungsi dari *Escape Sequence* berikut ini dan tuliskan contoh penggunannya!
   a. \n
   b. \”
   c. \t
   d. \xaa

   e. Bagaimana menghasikan tampilan seperti tampilan berikut, dengan menggunakan tanda escape squence

   Amir bilang “Saya ijin makan \ minum sebentar”
8. Sebutkan 3 perintah input dalam bahasa C dan jelaskan secara singkat!
9. Jelaskan apa perbedaan dari ketiga perintah diatas?

10. Tuliskan bentuk dasar dari perintah *scanf()*!
11. Apa fungsi tanda '&' pada perintah input *scanf()*?
12. Pada perintah *scanf()* terdapat penentu format. Jelaskan apa perbedaan penentu format pada *printf()* dan pada *scanf()*?
13. Bagaimanakah caranya untuk menginputkan beberapa data sekaligus dalam satu baris?
14. Tuliskan bagaimana programnya apabila kita ingin memasukkan data dengan tampilan sebagai berikut :

```
Masukkan Tempat/Tgl. Lahir Anda :
```

    Dimana inputan tersebut dalam satu baris, dan data yang harus diinputkan adalah *tempat_lahir, tgl_lahir/bln_lahir/thn_lahir*.
15. Tuliskan kepanjangan dari *getche*?

Soal 3.B

1. Buat program untuk menghitung jumlah dan selisih dua buah bilangan integer, kemudian tampilkan jumlah dan selisih tadi dengan menggunakan format lebar field = 8.

    Data input : bilangan 1, bilangan 2

    Data output : jumlah, selisih

    Tampilan yang diinginkan sebagai berikut :

```
Masukkan Bilangan 1           : ……
Masukkan Bilangan 2           : ……
Jumlah antara … dan … adalah  : ……
Selisih antara … dan … adalah : ……
```

2. Buat program untuk menghitung luas dan keliling persegi panjang, yang mana rumusnya adalah :

   luas = panjang * lebar

   kell=2*(panjang + lebar)

   Semua variabel menggunakan tipe *float.* panjang dan lebar diinputkan dengan *scanf()* dan luas serta keliling yang sudah dihitung dicetak dengan format lebar field = 10 dan jumlah angka di belakang titik desimal = 2.

   Tampilan yang diinginkan sebagai berikut :

```
----------------------------------------
      PROGRAM HITUNG PERSEGIPANJANG
----------------------------------------
Masukkan Panjang Persegipanjang  :<input>
Masukkan Lebar Persegipanjang    :<input>
----------------------------------------
Luas Persegipanjang              :<output>
Keliling Persegipanjang          :<output>
----------------------------------------
```

8. Operator

a. Operator Penugasan

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

Contoh :

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

Artinya : variable "nilai" diisi dengan 80 dan variable "A" diisi dengan hasil perkalian antara x dan y.

b. Operator Aritmatika

Bahasa C menyediakan lima operator aritmatika, yaitu :

1.* : untuk perkalian
2./ : untuk pembagian
3.% : untuk sisa pembagian (*modulus*)
4.+ : untuk pertambahan
5.- : untuk pengurangan

Catatan :

Operator % digunakan untuk mencari sisa pembagian antara dua bilangan.

Misalnya :

1.9 % 2 = 1 9 % 3 = 0 9 % 5 = 4 9 % 6 = 3

Contoh Program :

```
#include <stdio.h>
#include <conio.h>
void main()
{
     clrscr(); // untuk membersihkan layar
     printf("Nilai dari 9 + 4 = %i", 9 + 4);
     printf("Nilai dari 9 - 4 = %i", 9 - 4);
     printf("Nilai dari 9 * 4 = %i", 9 * 4);
     printf("Nilai dari 9 / 4 = %i", 9 / 4);
     printf("Nilai dari 9 \% 4 = %i", 9 % 4);
     getch();
}
```

Program 3.7 Pemakaian Operator Arithmatika

c. **Operator Hubungan (Perbandingan)**

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

Tabel 3.4 Tabel Penentu Format *scanf()*

Operator	Arti	Contoh	
<	Kurang dari	x < y	Apakah x kurang dari y
<=	Kurang dari sama dengan	x <= y	Apakah x kurang dari sama dengan y
>	Lebih dari	x > y	Apakah x lebih dari y
>=	Lebih dari sama dengan	x >= y	Apakah x lebih dari sama dengan y
==	Sama dengan	x == y	Apakah x sama dengan y
!=	Tidah sama dengan	x != y	Apakah x tidak sama dengan y

d. Operator Logika

Jika operator hubungan membandingkan hubungan antara dua buah operand, maka operator logika digunakan untuk membandingkan logika hasil dari operator-operator hubungan. Operator logika ada tiga macam, yaitu :

1. && : Logika AND (DAN)
2. || : Logika OR (ATAU)
3. ! : Logika NOT (INGKARAN)

e. Operator Bitwise

Operator bitwise digunakan untuk memanipulasi bit-bit dari nilai data yang ada di memori. Operator bitwise dalam bahasa C :

1. << : Pergeseran bit ke kiri
2. >> : Pergeseran bit ke kanan
3. & : Bitwise AND
4. ^ : Bitwise XOR (exclusive OR)
5. | : Bitwise OR
6. ~ : Bitwise NOT

f. Operator Unary

Operator Unary merupakan operator yang hanya membutuhkan satu operand saja. Dalam bahasa C terdapat beberapa operator unary, yaitu :

Tabel 3.5 Tabel Operator Unary

Operator	Arti/Maksud	Letak	Contoh	Equivalen
-	Unary minus	Sebelum operator	A + -B * C	A + (-B) * C
++	Peningkatan dengan penambahan nilai 1	Sebelum dan sesudah	A++	A = A + 1
--	Penurunan dengan pengurangan nilai 1	Sebelum dan sesudah	A--	A = A – 1
sizeof	Ukuran dari operand dalam byte	Sebelum	sizeof(I)	-
!	Unary NOT	Sebelum	!A	-
~	Bitwise NOT	Sebelum	~A	-
&	Menghasilkan alamat memori operand	Sebelum	&A	-
*	Menghasilkan nilai dari pointer	Sebelum	*A	-

Catatan Penting ! :

Operator peningkatan ++ dan penurunan -- jika diletakkan sebelum atau sesudah operand terdapat perbedaan. Perhatikan contoh berikut :

```
#include <stdio.h>
#include <conio.h>
void main()
{
   int x, nilai;
   clrscr(); x = 5;
   nilai = ++x; /* berarti x = x + 1; nilai = x; */
   printf("nilai = %d, x = %d\n", nilai, x);
   nilai = x++; /* berarti nilai = x; nilai = x + 1; */
   printf("nilai = %d, x = %d\n", nilai, x);
   getch();
}
```

Program 3.8. Penggunaan Operator Unary ++

```
#include <stdio.h>
#include <conio.h>
{
   int b, nilai;
   clrscr(); // untuk membersihkan layar
   nilai = --b; /* berarti b = b - 1; nilai = b; */
   printf("nilai = %d, b = %d\n", nilai, b);
   nilai = b--; /* berarti nilai = b; b = b + 1; */
   printf("nilai = %d, b = %d\n", nilai, b);
   getch();
}
```

Program 3.9. Penggunaan Operator Unary --