Membuat Program dengan Menggunakan Bahasa "C"

Dian Wirdasari

Abstrak

Struktur dari program C merupakan kumpulan dari sebuah atau lebih fungsi-fungsi. Fungsi pertama yang harus ada di program C yaitu bernama **main ()**. Fungsi main() ini adalah fungsi pertama yang akan diproses pada saat program di-kompile dan dijalankan. Suatu fungsi di program C dibuka dengan kurung kurawal buka ({) dan ditutup dengan kurung kurawal tutup (}). Di antara kurung kurawal dituliskan statemen-statemen program C. Fungsi-fungsi lain selain fungsi utama dapat ditulis setelah atau sebelum fungsi utama dengan deskripsi prototype fungsi pada bagian awal program. Dapat juga dituliskan pada file lain yang apabila ingin dipakai maka harus menuliskan header filenya, dengan *preprocessor directive* **#include**. File ini disebut file pustaka (*library file*).

Kata Kunci: program, bahasa C

A. PENDAHULUAN

Akar dari bahasa C adalah bahasa BCPL yang dikembangkan oleh Martin Richard pada tahun 1967. Bahasa ini kemudian dikembangkan oleh Ken Thompson menjadi bahasa B pada tahun 1970. Perkembangan selanjutnya menjadi bahasa C oleh Dennis Richie sekitar 1970-an di Bell Telephone Laboratories (sekarang adalah AT&T Bell Laboratories).

Bahasa C pertama kali digunakan di computer Digital Equipment Corporation PDP-11 yang menggunakan system operasi UNIX, (±90% sistem operasi UNIX ditulis dalam bahasa C) dan sampai sekarang bahasa ini telah dipergunakan secara praktis pada hampir semua sistem operasi. Selain itu, banyak bahasa pemrograman populer seperti PHP dan Java menggunakan sintaks dasar yang mirip bahasa C.

Pada tahun 1983, American National Standards Institute (ANSI) membentuk suatu komite, X3J11, untuk mengembangkan suatu spesifikasi standard untuk C dan berhasil diselesaikan pada tahun 1989. ANSI C didukung oleh kebanyakan compiler. Banyak kode C yang ditulis sekarang didasarkan pada ANSI C. Semua program yang ditulis dengan standard C dijamin akan berfungsi dengan baik pada platform lain yang memiliki C. Tetapi banyak juga program C yang hanya dapat di kompilasi pada platform tertentu dengan compiler tertentu sehubungan dengan library non standard, misalnya untuk graphic.

Pada tahun 1986, dikembangkan superset C (kompatibel dengan C, namun dilengkapi dengan kemampuan pemrograman berorientasi objek) oleh Bjarne Stroustrup yaitu bahasa C++ *(C with Class)* dan sekarang merupakan bahasa yang banyak dipergunakan pada sistem operasi Microsoft Windows; sedangkan C tetap merupakan bahasa yang populer di Unix.

Setelah proses standarisasi oleh ANSI, sepesikasi bahasa C masih relatif statis untuk beberapa saat, sedangkan C++ terus berevolusi. Revisi standard tahun 1990, mengawali publikasi sebagai ISO 9899:1999 pada tahun 1999. Standard ini disebut sebagai "C99" telah diadopsi sebagai ANSI standard pada tahun 2000.

Kemampuan baru C99 meliputi:

1. fungsi inline function
2. membebaskan pembatasan terhadap tempat deklarasi variabel (seperti pada C++)

3. menambah beberapa type data baru, termasuk long long int (untuk mengurangi kesulitan transisi 32-bit ke 64-bit), type data boolean, dan suatu yang baru untuk bilangan complex.
4. array variable-length
5. dukungan resmi terhadap one-line comment yang dimulai dengan //, dipinjam dari C++
6. beberapa fungsi library baru, seperti snprintf()
7. beberapa header file baru, seperti stdint.h

Dukungan terhadap C99 cukup beragam, dimana GCC dan beberapa compiler lainnya mendukung fasilitas C99, tetapi compiler yang dibuat oleh Microsoft dan Borland tidak.

B. KELEBIHAN DAN KEKURANGAN BAHASA C

- **Kelebihan Bahasa C**
  1. Bahasa C tersedia hampir di semua jenis komputer.
  2. Kode bahasa C bersifat portable untuk semua jenis komputer. Suatu program yang ditulis dengan versi bahasa C tertentu akan dapat dikompilasi dengan versi bahasa C yang lain hanya dengan sedikit modifikasi.
  3. C adalah bahasa pemrograman yang fleksibel. Dengan bahasa C, kita dapat menulis dan mengembangkan berbagai jenis program mulai dari operating system, word processor, graphic processor, spreadsheets, ataupun kompiler untuk suatu bahasa pemrogaman.
  4. Bahasa C hanya menyediakan sedikit kata-kata kunci, hanya terdapat 32 kata kunci. Yaitu: **auto break case char const continue default do double else enum extern float for goto if int long register return short signed sizeof static struct switch typedef union unsigned void volatile while**
  5. Proses executable program bahasa C lebih cepat.
  6. Dukungan pustaka yang banyak.
  7. C adalah bahasa yang terstruktur.
  8. Bahasa C termasuk bahasa tingkat menengah.
  9. Dibandingkan dengan assembly, kode bahasa C lebih mudah dibaca dan ditulis.

- **Kekurangan Bahasa C**
  1. Banyaknya Operator serta fleksibilitas penulisan program kadang-kadang membingungkan pemakai.
  2. Para pemrogram C tingkat pemula umumnya belum pernah mengenal pointer dan tidak terbiasa menggunakannya. Keampuhan C justru terletak pada pointer.

C. EDITOR BAHASA C

Editor bahasa C yang digunakan adalah Turbo C++ versi 3.0.

Sekilas Mengenai Editor Turbo C

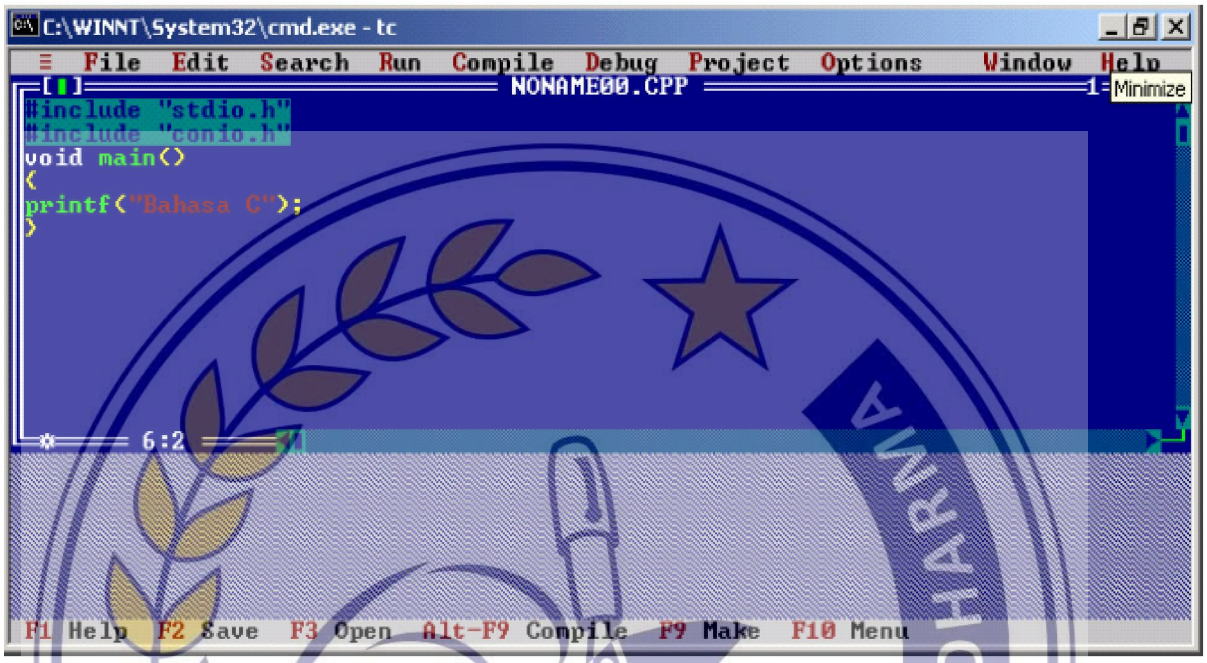
- Untuk mengkompilasi Program, langkah-langkahnya sbb :
  - Pilih menu Compile dengan menekan **Alt + C**
  - Pilih Submenu **Compile**
  - Enter

  Akan ditampilkan hasil kompilasi Program, tekan sembarang tombol. Atau dapat dilakukan dengan menekan tombol **Alt + F9** bersamaan.
- Untuk menjalankan program:
  - Pilih menu Run dengan menekan **Alt + R**
  - Pilih submenu **Run** dan tekan Enter

Atau dapat dilakukan dengan menekan tombol **Ctrl + F9** bersamaan.

- Hasil tampilan dapat dilihat dengan menampilkan user screen, dengan cara:
  - Pilih menu Window dengan menekan **Alt + W**
  - Pilih submenu **User screen** dan tekan Enter

  Atau dengan menekan tombol **Alt + F5** bersamaan.
- **Tampilan Turbo C Versi 3.0**:



Gambar 1. Tampilan Turbo C

Tampilan menu editor Turbo C (Turbo C++)

- **File** , terdiri dari :
  (1) **New**, untuk memulai program baru
  (2) **Open**, untuk mengambil atau membuka program
  (3) **Save**, untuk menyimpan file/program
  (4) **Save as**, untuk menyimpan file/program
  (5) **Save all,** untuk menyimpan seluruh file/program
  (6) **Change dir**, untuk mengubah directory
  (7) **Print,** untuk mencetak program
  (8) **DOS Shell**, untuk menuju ke DOS Shell
  (9) **Quit**, untuk keluar dari Turbo C
- **Edit**, terdiri dari :
  (1) **Undo**, untuk membatalkan pengeditan terakhir
  (2) **Redo,** untuk kembali ke pengeditan terakhir yang telah di undo.
  (3) **Cut,** untuk memotong bagian tertentu dari program.
  (4) **Copy**, untuk menduplikasi bagian program
  (5) **Paste**
  (6) **Clear,** untuk menghapus bagian tertentu dari program
  (7) **Copy example**
  (8) **Show Clipboard**
- **Search,** terdiri dari :
  (1) **Find…**

(2) **Replace…**
(3) **Search again**
(4) dst

- **Run**, terdiri dari :
  (1) **Run…**, untuk menjalankan program
  (2) **Program reset**
  (3) **Go to cursor**
  (4) dst
- **Compile**, terdiri dari :
  (1) **Compile**, untuk mengkompilasi program
  (2) **Make**
  (3) **Link**
  (4) **dst**
- **Debug**, terdiri dari
  (1) **Inspect**
  (2) **Evaluate/modify**
  (3) dst
- **Project**, terdiri dari :
  (1) **Open project**
  (2) **Close project**
  (3) dst
- **Options**, terdiri dari :
  (1) **Application**
  (2) **Compiler**
  (3) **Transfer**
  (4) dst
- **Window**, terdiri dari :
  (1) **Size/Move**
  (2) **Zoom**
  (3) **Tile**
  (4) dst
- **Help**, terdiri dari
  (1) **Contens**
  (2) **Index**
  (3) **Topic search**
  (4) dst

D. UNSUR-UNSUR BAHASA C

1. Struktur Dasar Program C

Struktur dari program C merupakan kumpulan dari sebuah atau lebih fungsi-fungsi. Fungsi pertama yang harus ada di program C yaitu bernama **main ()**. Fungsi main() ini adalah fungsi pertama yang akan diproses pada saat program di-kompile dan dijalankan, sehingga disebut sebagai fungsi yang mengontrol fungsi-fungsi lain.

Suatu fungsi di program C dibuka dengan kurung kurawal buka ({) dan ditutup dengan kurung kurawal tutup (}). Di antara kurung kurawal dapat dituliskan statemen-statemen program C. Fungsi-fungsi lain selain fungsi utama dapat dituliskan setelah atau sebelum fungsi utama dengan deskripsi prototype fungsi pada bagian awal program. Dapat juga dituliskan pada file lain yang

apabila ingin dipakai maka harus menuliskan header filenya, dengan *preprocessor directive* **#include**. File ini disebut file pustaka (*library file*).

Struktur Dasar Program C

```
#include <stdio.h>   ──────►  Preprocessor directive
Fungsi_lain();       ──────►  Prototype fungsi lain
main()
{
 statemen-statemen;          } Fungsi utama
}

Fungsi_fungsi_lain()
{
 statemen-statemen;          } Fungsi fungsi lain yang ditulis
}                              pemrogram komputer
```

Contoh 1

```
#include <stdio.h>
main()
{
  printf ("Selamat Datang Di Pemrograman C\n");
  printf ("Semoga Sukses!!!!");
}
```

Compile program tersebut dengan menekan **alt+F9,** ketika komentar Success ditampilkan, kemudian tekan **ctrl+F9** untuk me-run program, tekan **alt+F5** untuk masuk mode user screen dan melihat hasil yang ditampilkan dari source code di atas. Akan terlihat yang tampil adalah :

```
Selamat Datang Di Pemrograman C
Semoga Sukses!!!!
```

Dari contoh 1, terdapat perintah printf(""), perintah ini berfungsi untuk menampilkan apa yg terdapat di antara tanda petik dua (") untuk tampil di layar monitor.

2. Memecah Baris Statemen

Suatu statement yang panjang dalam program C dapat ditulis dalam beberapa baris dengan menggunakan tanda '\'. Akhir dari suatu baris yang menggunakan tanda tersebut menunjukkan bahwa baris berikutnya adalah baris sambungannya.

Contoh 2

```
#include <stdio.h>
main()
{
  printf ("Sebuah kalimat yang panjang dalam bahasa C dapat \
ditulis dalam dua baris penulisan \n");
}
```

3. File Judul

File judul (header file) merupakan file yang berisi *prototype* (judul, nama, dan syntak) dari sekumpulan fungsi-fungsi pustaka tertentu. Jadi, file ini hanya berisi prototype dari fungsi-

fungsi pustaka, sedangkan fungsi-fungsi pustakanya sendiri disimpan di file pustaka (*library file* dengan nama extension filenya adalah .LIB). Misalnya prototype dari fungsi-fungsi pustaka **printf()** dan **scanf()** terdapat di file judul **stdio.h**, sehingga jika fungsi-fungsi ini akan digunakan di program, maka nama file judulnya harus dilibatkan dengan menggunakan preprocessor **#include**.

File judul **stdio.h** berisi prototype fungsi-fungsi pustaka untuk operasi input/output standar. Contoh lain dari file judul adalah **math.h** berisi prototype fungsi-fungsi pustaka untuk operasi matematika. Ada dua cara untuk melibatkan file judul dalam suatu program C, yaitu:

`#include <stdio.h>` atau **`#include "stdio.h"`**

Nama dari file judul yang ditulis dalam tanda petik dua, compiler C akan mencari file ini dalam *default directory* dan kemudian ke directory file-file pustaka. Jika yang digunakan adalah "<>", maka compiler C hanya akan mencari di directory file-file pustaka saja.

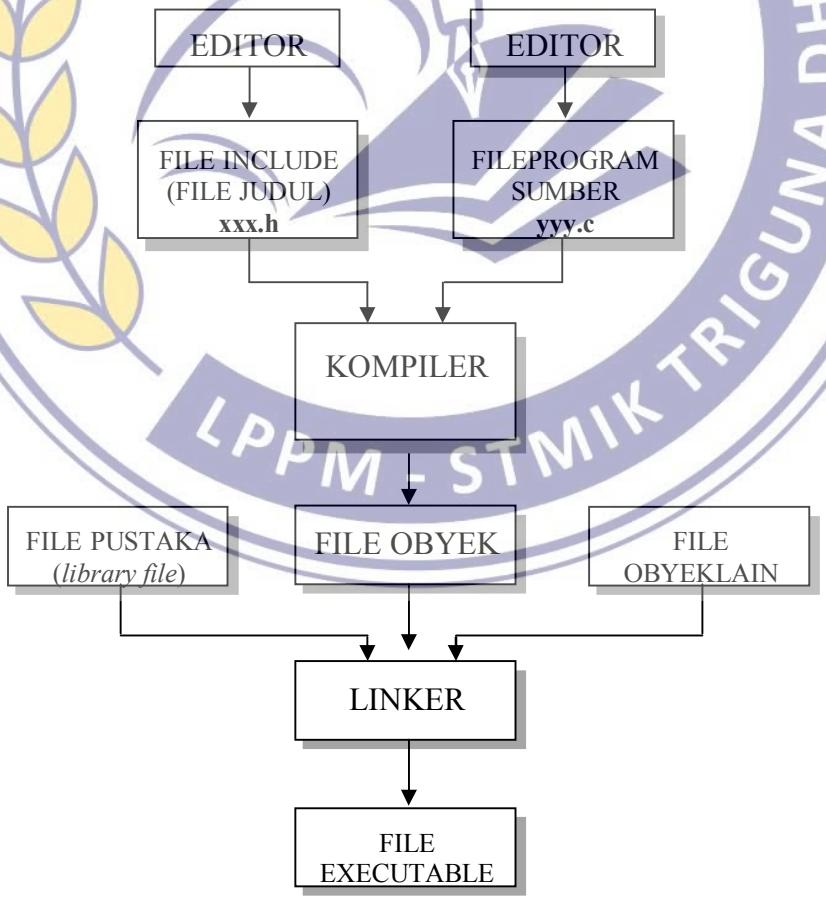
4. Komentar

- Dituliskan di antara tanda /* dan */. Untuk beberapa baris komentar.
- Pada beberapa kompilator, menggunakan tanda // yang hanya bisa diterapkan dalam satu baris saja

5. Pemrosesan Program Sumber Dalam Bahasa C

Sebelum program C dapat dijalankan, harus di compile dan di linking terlebih dahulu.

- Compile adalah proses menterjemahkan seluruh program ke dalam bahasa mesin sekaligus. Compile dapat berhasil jika dalam program tidak ada kesalahan sama sekali.
- Linking adalah proses untuk menggabungkan beberapa file program hasil kompilasi.



Gambar 2. Proses Program Sumber dalam Bahasa C

E. KONSEP TIPE DATA

1. Jenis Statement Dalam Bahasa C

Statement dalam bahasa C selalu diakhiri dengan tanda titik koma (';'). Statement dapat digolongkan menjadi dua yaitu statement yang tidak dieksekusi dan yang dieksekusi.

Statement non-executable:

Statement non-executable adalah statement yang dibuat tidak untuk dieksekusi, melainkan sekedar komentar, atau statement untuk melakukan deklarasi nama (yang mungkin sekaligus melakukan inisialisasi nilai).

Deklarasi

Bagian deklarasi mewakili semua nama (identifier) yang didefinisikan dan akan dipakai. Nama yang harus dideklarasikan sebelum dipakai dalam lingkup yang sesuai adalah :

- Deklarasi nama konstanta dan nilainya
- Deklarasi struktur dan union
- Deklarasi nama type yang didefinisikan
- Deklarasi nama variabel dan type yang sudah didefinisikan. Deklarasi nama variabel dapat diikuti dengan inisialisasi nilainya atau tidak.
- Deklarasi fungsi (prototype)

Statement *executable*:

Statement *executable* adalah instruksi yang akan dikerjakan oleh komputer, meliputi pemberian harga, kondisional, pengulangan atau kalimat percabangan sebagai berikut:

- Assignment (dengan operator =)
- Kondisional
  - if (<kondisi>) { };
  - if () { } else { };
  - switch
- Pengulangan
  - while
  - do while
  - for
- Pencabangan
  - goto
  - continue
  - break
  - return

2. Identifier

Nama ***(identifier)*** dipakai untuk mengenali suatu objek dalam sebuah program.

Struktur Blok dan nama

Sebuah "Blok" dalam bahasa C dituliskan di antara tanda kurung kurawal buka "{" dan tutup "}". Bahasa C tidak mengenal deklarasi blok bertingkat *(nested)* seperti Pascal atau Ada.

Deklarasi nama (fungsi, variabel, tipe, konstanta) yang dilakukan di luar fungsi disebut *deklarasi eksternal*. Deklarasi di dalam fungsi disebut *deklarasi internal*. Variabel dengan deklarasi internal, lokal terhadap blok tempat ia dideklarasikan. Nama variabel dengan deklarasi eksternal berlaku global dalam file tempat ia dideklarasikan.

Aturan nama

Bahasa C memiliki peraturan tersendiri dalam penamaan. Yaitu:

1. Panjang identifier maksimal 31 karakter.
2. Karakter pertama diawali dengan karakter huruf, tidak boleh angka dan tidak boleh menggunakan special karakter seperti **+, =, >, ?,** dll.
3. Karakter kedua dan seterusnya boleh menggunakan huruf, angka, dan garis bawah "_" (underscore).
4. Dalam C berlaku **case sensitive,** contoh: **cool** itu tidak sama dengan **cOOl**, berbeda dengan **CooL**.
5. Tidak boleh menggunakan kata kunci yang digunakan di C.

Contoh penamaan variabel yang benar: NIM, a, x, nama_mhs, f3098, f4, nilai, budi, dsb. Sedangkan contoh penamaan variable yang salah: %nilai_mahasiswa, 80mahasiswa, rata-rata, ada spasi, penting!, dsb.

Aturan akses nama

- Berdasarkan deklarasinya, dibedakan atas nama global (deklarasi global) dan nama lokal (deklarasi lokal)
- Nama global dapat diakses oleh semua fungsi dalam file yang sama (supaya nama ini dapat diakses oleh fungsi di file lain, nama ini harus dideklarasikan lagi di file tersebut)
- Nama yang dideklarasi pada suatu fungsi hanya dapat diakses dalam fungsi tersebut
- Jika ada nama yang sama, yang diacu adalah nama lokal.

3. Konstanta

Ada beberapa macam penulisan nilai konstanta dalam teks program (sesuai dengan type yang mewakili konstanta tsb) yaitu : integer, karakter, floating, enumerasi, dan string.

Konstanta Integer

- Konstanta integer terdiri dari deretan angka

Konstanta Karakter

- Konstanta karakter terdiri dari deretan satu/lebih karakter yang diapit petik tunggal, contoh 'r'.
- Karakter yang tidak kelihatan di layar atau beberapa karakter khusus, tidak dapat dituliskan langsung sehingga harus menggunakan *escape sequence* (semua escape sequence selalu diawali dengan backslash "\")

Tabel Konstanta Karakter Escape

Esc. Seq.	Nama	Esc. Seq.	Nama
\a	Alert (bell)	\v	Vertical tab
\b	Backspace	\'	Single quotation mark
\f	Form feed (ganti halaman)	\"	Double quotation mark
\n	Newline (baris baru)	\\	Backslash
\r	Carriage return	\ddd	ASCII character (in octal notation)
\t	Horizontal tab	\xdd	ASCII character (in hex notation)
\?	Question mark		

Konstanta Floating

- Konstanta floating terdiri atas bagian integer, titik desimal, bagian pecahan, dan bagian eksponen yang diawali huruf 'e' atau 'E'. Titik desimal atau bagian eksponen dapat tidak ada, namun salah satu harus tetap ada.
- Contoh: 3.141592654, 6.02217e23, 3E8

Konstanta Enumerasi

- Dideklarasikan sebagai enumerator, representasi internalnya adalah konstanta dengan tipe int.

Konstanta String *(String Literal)*

- Konstanta *string* adalah deretan karakter yang dibatasi dengan petik ganda, contoh "IF-223".
- Bertipe *"array of character"* dengan kelas penyimpanan statik, terinisialisasi dengan karakter yang diberikan (berakhiran '\0'). Efek perubahan pada konstanta string tak terdefinisi.
- Bedakan antara konstanta string (misal "I") dan konstanta karakter (misal 'I'). Konstanta string "I" adalah array dengan dua elemen (karakter I dan '\0'). Konstanta karakter 'I' mempunyai nilai integer sesuai dengan kode set karakter yang dipakai.

4. Tipe Data Dasar

Program C menyediakan lima macam tipe data dasar, yaitu tipe data integer (nilai numerik bulat yang dideklarasikan int), floating point (nilai numerik pecahan ketepatan tunggal yang dideklarasikan dengan float), double-precision (nilai numerik pecahan ketepatan ganda yang dideklarasikan dengan double), karakter (dideklarasikan dengan char), dan kosong (dideklarasikan dengan void).

Tabel Tipe-tipe Data Dasar

Type Data	Lebar	Jangkauan Nilai	
		Dari	Sampai dengan
int	16 bit	-32768	32767
signed int	16 bit	-32768	32767
short int	16 bit	-32768	32767
signed short int	16 bit	-32768	32767
unsigned int	16 bit	0	65535
unsigned short int	16 bit	0	65535
long int	32 bit	-2147483648	2147483649
signed long int	32 bit	-2147483648	2147483649
unsigned long int	32 bit	0	4294967296
float	32 bit	3.4E-38	3.4E+38(7 digit)
double	64 bit	1.7E-308	1.7E+308(15 digit)
long double	80 bit	3.4E-4932	1.1E+4932(19 digit)
char	8 bit	-128	127
signed char	8 bit	-128	127
unsigned char	8 bit	0	255

Catatan:

- Objek bertipe void tidak dapat didefinisikan, karena ukuran tidak diketahui. Tipe ini dipakai

untuk return value fungsi atau untuk mendeklarasikan/ mendefinisikan pointer ke objek yang tipenya tidak diketahui.

- C tidak melakukan pemeriksaan saat *run time* untuk memastikan bahwa harga yang di-assign ke variabel sesuai dengan range yang ada.

5. Deklarasi Tipe Variabel

Variabel adalah suatu pengenal yang digunakan untuk menyimpan suatu nilai dan nilai dari variabel dapat berubah-ubah selama proses dari program.

Format deklarasi tipe variabel:

Tipe NamaVar1, **NamaVar2**, ...;

Berikut ini adalah contoh pendeklarasian variabel:

int x; /*mendeklarasikan variabel x tipe integer*/
float a,b; /*mendeklarasikan variabel a dan b tipe float*/
char y, huruf, nim[10]; /*mendeklarasikan variabel y dan huruf tipe karakter, serta variabel nim tipe array karakter dengan banyak elemen 10*/
int array[5][4]; /* Deklarasi variabel array bertipe array integer dua dimensi */

Contoh 3

```
#include <stdio.h>
main()
{ /* Program deklarasi variabel*/
int i;
float j;
char huruf;
i = 5; j = 123.45; huruf = 'D';
printf ("Ini nilai i : %d \n", i);
printf ("Ini nilai j : %f \n", j);
printf ("Ini nilai huruf : %c \n", huruf);
}
```

Jika program pada Contoh 3 dijalankan, akan terlihat tampilan sebagai berikut:

```
Ini nilai i : 5
Ini nilai j : 123.449997
Ini nilai huruf : D
```

6. Deklarasi Konstanta Bernama

Dalam bahasa C, konstanta bernama dapat dibuat dengan dua cara, yaitu dengan memakai deklarasi const, atau dengan memanfaatkan fasilitas makro.

Format deklarasi konstanta bernama:

const [**tipe**] **nama_konstanta** = **nilai**;

Contoh:

const double pi = 3.141592654;
const float rad = 57.29577951;
const umur = 40; /* sama dengan const int umur = 40 */

Contoh 4

```
#include <stdio.h>
#include <conio.h>
main(){
/* Program mencari luas lingkaran*/
float f_jari,f_luas; /* Deklarasi variable */
const float phi=3.14; /* Deklarasi konstanta phi */
clrscr();
printf ("Input nilai jari-jari = ");
scanf ("%f", &f_jari);
f_luas = phi*f_jari*f_jari;
printf ("Luasnya adalah = %8.3f \n", f_luas);
}
```

Jika program pada Contoh 4 dijalankan, akan terlihat tampilan sebagai berikut:

```
Input nilai jari-jari = 50
Luasnya adalah = 7850.000
```

Definisi konstanta dengan kata kunci ”const” baru ada dalam ANSI C. Versi lamanya memakai preprocessor untuk mendefinisikan konstanta. Bahasa C menyediakan preprocessor directive #define untuk mendefinisikan suatu konstanta, makro ataupun nama. Preprocessor #define dapat diletakkan di dalam program yang sama atau diletakkan dalam file yang terpisah dengan programnya. Kemudian nama filenya dapat dilibatkan di dalam program dengan menggunakan preprocessor directive #include.

Aturan penulisan konstanta menggunakan #define :

#define nama_konstanta nilai

Contoh penggunaan #define sebagai berikut:

```
#define pi 3.141592654
#define umur 40
#define pecahan float
```

Contoh 5

```
#include <stdio.h>
#include <conio.h>

#define utama       main
#define mulai       {
#define selesai     }
#define pecahan     float
#define cetak       printf
#define masukkan    scanf

utama()
mulai
pecahan rupiah,dollar,uang;
   clrscr();
   cetak("$1 berapa rupiah = Rp ");
   masukkan("%f",&rupiah);
   cetak("Uang Anda ada berapa dollar = $ ");
   masukkan("%f",&dollar);
   uang = dollar * rupiah;
```

```
   cetak("Uang Anda sebanyak = Rp %8.3f\n",uang);
selesai
```

Jika program pada Contoh 5 dijalankan, akan terlihat tampilan sebagai berikut:

```
$1 berapa rupiah = Rp 10000
Uang Anda ada berapa dollar = $ 3.5
Uang Anda sebanyak = Rp 35000.000
```

F. OPERATOR

1. 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.

2. Operator Aritmatika

Bahasa C menyediakan lima operator aritmatika, yaitu:

- * : untuk perkalian
- / : untuk pembagian
- % : untuk sisa pembagian (modulus)
- + : untuk pertambahan
- - : untuk pengurangan

Contoh 6

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

Jika program pada Contoh 6 dijalankan, akan terlihat tampilan sebagai berikut:

```
Nilai dari 9 + 4 = 13
Nilai dari 9 - 4 = 5
Nilai dari 9 * 4 = 36
Nilai dari 9 / 4 = 2
```

3. Operator Hubungan (Relational)

Operator Hubungan (relational operator) digunakan untuk membandingkan hubungan antara

dua buah operand (sebuah nilai atau variable) yang akan menghasilkan nilai **true** atau **false**. Operator hubungan dalam bahasa C adalah:

Op.	Arti	Contoh	
<	Kurang dari	x < y	Apakah x kurang dari y
<=	Kurang dari atau sama dengan	x <= y	Apakah x kurang dari atau sama dengan y
>	Lebih dari	x > y	Apakah x lebih dari y
>=	Lebih dari atau sama	x >= y	Apakah x lebih atau sama dengan y
==	Sama dengan	x == y	Apakah x sama dengan y
!=	Tidak sama dengan	x != y	Apakah x tidak sama dengan y

Contoh :
(7 == 5) akan menghasilkan **false**.
(5 > 4) akan menghasilkan **true**.
(3 != 2) akan menghasilkan **true**.
(6 >= 6) akan menghasilkan **true**.
(5 < 5) akan menghasilkan **false**.

4. Operator Logika

Digunakan untuk membandingkan logika hasil dari operator-operator hubungan. Operator logika ada tiga macam, yaitu: && (AND), ||(OR), dan ! (NOT). Operator **!** (NOT) hanya memiliki satu operand yang berada dikanannya. Contoh:
!(5 == 5) mengembalikan **false.**
!(6 <= 4) mengembalikan **true**.
!true mengembalikan **false**.
Operator logika **&&** (AND) || (OR), dengan hasil operasinya sebagai berikut:

Operand1 a	Operand2 b	Hasil a && b	Hasil a || b
true	true	**true**	**true**
true	false	**false**	**true**
false	true	**false**	**true**
false	false	**false**	**false**

5. Operator Bitwise

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

- << : Pergeseran bit ke kiri
- >> : Pergeseran bit ke kanan
- & : Bitwise AND
- ^ : Bitwise XOR (exclusive OR)
- | : Bitwise OR
- ~ : Bitwise NOT

Contoh 7

```
/* pemakaian beberapa operator terhadap bit */
```

```
#include <stdio.h>
#include <conio.h>
main(){
int n = 10; /* 1010 */
int x = 1;  /* 0001 */
int y = 2;  /* 0010 */
clrscr();
printf ("n = %d \n", n);
printf ("x = %d \n", x);
printf ("y = %d \n", y);
printf ("n & 8 = %d \n", n & 8); /* 1010 AND 1000 = 1000 */
printf ("x & ~ 8 = %d \n", x & ~8); /* 0001 AND 0111 = 0001 */
printf ("y << 2 = %d \n", y << 2); /* 0010 ==> 1000 = 8 */
printf ("y >> 3 = %d \n", y >>3); /* 0010 ==> 0000 = 0 */
getch();
}
```

Jika program dijalankan, akan terlihat sebagai berikut:

```
n = 10
x = 1
y = 2
n & 8 = 8
x & ~ 8 = 1
y << 2 = 8
y >> 3 = 0
```

6. Operator Unary

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

Operator	Arti	Contoh	Equivalen
-	Unary minus	A+-B*C	A+(-B)*C
++	Increment	A++	A=A+1
--	Decrement	A--	A=A-1
Sizeof	Ukuran dari operand dalam byte	Sizeof(1)	-
!	Unary NOT	!A	-
&	Menghasilkan alamat memory operand	&A	-
*	Menghasilkan nilai dari pointer	*A	-

Catatan:

Operator increment ++ dan decrement -- jika diletakkan sebelum atau sesudah operand terdapat perbedaan.

Perhatikan contoh berikut:

Contoh 8

```
#include <stdio.h>
#include <conio.h>
main(){
```

```
int i, hasil;
i = 0;
clrscr();
hasil = ++i; /* i=i+1; hasil=i; */
printf("hasil = %d, nilai i = %d\n", hasil, i);
hasil = i++; /* hasil=i; i=i+1; */
printf("hasil = %d, nilai i = %d\n", hasil, i);
getch();
}
```

Jika program dijalankan terlihat tampilan sebagai berikut:

```
hasil = 1, nilai i = 1
hasil = 1, nilai i = 2
```

Contoh 9

```
#include <stdio.h>
#include <conio.h>
/* pemakaian beberapa operator terhadap RELATIONAL DAN bit */
main (){
char i, j;
i = 3; /* 00000011 dalam biner */
j = 4; /* 00000100 dalam biner */
clrscr();
printf ("i = %d \n", i);
printf ("j = %d \n", j);

/* utk op logika, semua data akan bernilai 1 (true) kecuali data 0 (nol)
akan tetap bernilai 0 (false) */

/* i=3(true) && j=4(true) => 1 (true) */
printf (" i && j = %d \n", i && j);

/* i=3(00000011) & j=4(00000100) => 0: 00000000 dalam biner */
printf (" i & j = %d \n", i & j);

/* i=3(true) || j=4(true) => 1 (true) */
printf (" i|| j = %d \n", i || j);
/* i=3(00000011) | j=4(00000100) => 7: 00000111 dalam biner */
printf (" i| j = %d \n", i | j);
printf (" i^j = %d \n", i ^ j); /* 7: 00000111 biner */
getch();
}
```

Jika program pada Contoh 9 dijalankan, akan terlihat tampilan sebagai berikut:

```
i = 3
j = 4
i && j = 1
i & j = 0
i || j = 1
i | j = 7
i ^ j = 7
```

7. Operator Kombinasi

Digunakan untuk memendekkan penulisan operasi penugasan. Contoh:
x = x + 5;
y = y * 9;
dapat dipendekkan menjadi:
x += 5;
y *= 9;

Operator Kombinasi	Arti padanannya
x += 2;	x = x + 2
x -= 2;	x = x – 2
x *= 2;	x = x * 2
x /= 2;	x = x / 2
x %= 2;	x = x % 2
x <<= 2;	x = x << 2
x >>= 2;	x = x >> 2
x &= 2;	x = x & 2
x |= 2;	x = x | 2
x ^= 2;	x = x ^ 2

8. Prioritas Operator

Prioritas	Operator
Tertinggi	! + - & (Operator unary)
	* / % (Operator aritmatika)
	+ - (Operator aritmatika)
	> >= < <= (Operator hubungan)
	== != (Operator hubungan)
	&& (Operator logika)
	|| (Operator logika)
Terendah	= += -= *= /= %=

G. DAFTAR PUSTAKA

Budi Sutedjo, S.Kom., MM dan Michael AN, S.Kom., 2004, **Algoritma dan Teknik Pemrograman Konsep, Implementasi dan Aplikasi**, Yogyakarta: ANDI.

Fathul Wahid, 2004, **Dasar-Dasar Algoritma dan Pemrograman**, Yogyakarta: ANDI.

Heri Sismoro, 2005, **Pengantar Logika Informatika, Algoritma dan Pemrograman Komputer**, Yogyakarta: ANDI.

Jogiyanto Hartono, 2000, **Konsep Dasar Pemrograman Bahasa C**, Yogyakarta: ANDI.

..... , 2004, **Pengenalan Komputer: Dasar Ilmu Komputer, Pemrograman, Sistem Informasi dan Inteligensi Buatan**, Yogyakarta: ANDI.

Yulikuspartono, S.Kom., 2004, **Pengantar Logika dan Algoritma**, Yogyakarta: ANDI.