BAB III

DASAR PEMROGRAMAN C

3.1 PENGENAL / *IDENTIFIER*

Pengenal (*identifier*) merupakan nama yang biasa digunakan untuk variable, kontanta, fungsi atau obyek yang lain, yang didefenisikan oleh pembuat program. Aturan yang berlaku dalam membuat suatu pengenal yaitu;

a) Pengenal haruslah diawali dengan huruf (A..Z, a..z) atau karakter garis bawah(**underscore**) **" _ "**.
b) Selanjutnya dapat berupa huruf, angka (0..9)
c) Penulisan huruf besar(uppercase) dan huruf kecil(lowercase) dibedakan (mis. **nama ≠ Nama**)
d) Tidak boleh berupa karakter khusus (mis. @, %, &, ^, !, dll.) dan operator aritmatika(mis. +,-,*,/,%)
e) Tidak boleh mengandung **Spasi**
f) Panjang pengenal hanya terdiri dari 32 karakter
g) Bukan Kata Kunci (***reseved word***) milik Bahasa C

Contoh :

nim
nilai_uts
rata2
Jumlah

Karakter "garis-bawah" biasa dipakai untuk mempermudah pembacaan terhadap suatu pengenal, misalkan **nilai_uts** akan lebih mudah dibaca dan diingat dibandingkan dengan **nilaiuts**. Untuk menghindari kesalahan, pengenal tidak bolehmengunakan nama yang tergolong sebagai kata kunci, misalkan; main, tidak boleh digunakan sebagai pengenal.

3.2 KATA KUNCI

Bahasa C mempunyai 32 buah kata yang dicadangkan (reserved words), di antaranya:

KATA KUNCI			
auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

3.3 TIPE DATA DASAR

Pengertian data menyiratkan suatu nilai yang bisa dinyatakan dalam bentuk konstanta atau variabel. Konstanta menyatakan nilai yang tetap, sedangkan variabel menyatakan nilai yang dapat diubah-ubah selama eksekusi

berlangsung. Tipe data, dapat dibagi menjadi lima kelompok, yang disebut tipe data dasar :

Tipe data bilangan bulat >> int
Tipe data bilangan real presisi-tunggal >> float
Tipe data bilangan real presisi-ganda >> double
Tipe data karakter >> char
Tak bertipe >> void

Tipe	Total bit	Range	Keterangan
Char	8	-128 s/d 127	Karakter
Int	16	-32768 s/d 32767	Integer
Float	32	3.4E-38 s/d 3.4e+38	Bilangan
Double	64	1.7e308 s/d 1.7e+308	Bilangan
Void	0	-	Tak Bertipe

3.4 VARIABEL

Variabel merupakan salah satu bentuk pengenal(***identifier***) digunakan dalam program untuk meyimpan suatu nilai, dan nilai yang ada padanya dapat diubah-ubah selama eksekusi program berlagsung.

Mendeklarasikan Variabel

Variabel yang akan yang digunakan dalam program haruslah dideklasikan terlebih dahulu. Pengertian dideklasikan disini berarti memesan memori dan menentukan jenis data yang bisa disimpan didalamnya. Bentuk pendeklarasian variabel :

Tipe-Data variabel(pengenal)

Pada pendeklarasian variabel yang jumlahnya lebih dari satu, dapat dipisahkan dengan koma.

Contoh 1 :

```
Int nilai1;
Int nilai2;
Int total;
```

Contoh 2:

```
Int nilai1,nilai2,total;
```

Memberikan Nilai Ke Variabel

Untuk memberikan nilai ke variabel yang telah dideklarisikan, bentuk pernyataan yang digunakan :

variabel = nilai;

contoh :

```
nilai1 = 10;
harga_barang = 1000;
total_harga = harga_barang * jumlah;
```

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

main()
{
int nilai;

  nilai=100;
  printf("Nilai Yang Tercetak (1) = %d n",nilai);
  nilai=75;
  printf("Nilai Yang Tercetak (2) = %d n",nilai);

getch();
}
```

Program 3.1 Penggunan Variabel

3.5 PEMODIFIKASIAN TIPE

Ada beberapa pemodifikasi tipe(***type modifier***) yang dapat dikenakan di awal tipe data dasar (kecuali void). Pemodifikasi tipe tersebut yaitu pemodifikasian tipe signed dan unsigned.

NAMA	KETERANGAN	UKURAN	JANGKAUAN
char	Abjad/karakter atau untuk bilangan bulat kecil	1 byte	signed: -128 to 127 unsigned: 0 to 255
short int (short)	Bilangan bulat dengan jangkauan pendek	2 byte	signed: -32768 to 32767 unsigned: 0 to 65535
int	Bilangan bulat	4 byte	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
long int (long)	Integer dengan jangkauan panjang	4 byte	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
bool	Boolean, dapat bernilai benar atau salah (true or false)	i byte	true or false
float	Angka dengan titik mengambang (bilangan cacah)	4 byte	3.4e +/- 38 (7 digit)
double	Bilangan cacah dengan ketelitian ganda	8 byte	1.7e +/- 308 (15 digits)
long double	Bilangan cacah dengan ketelitian ganda panjang	8 byte	1.7e +/- 308 (15 digits)
wchar_t	Karakter lebar, biasa dipakai untuk Unicode karakter	2 byte	1 karakter lebar

3.6 KONSTANTA

Konstanta adalah suatu pengenal yang sifatnya bernilai tetap, artinya ketika program tersebut dieksekusi, konstanta terebut nilainya tidak dapat lagi dirubah. Konstanta dideklarasikan diluar main() dengan menggunakan fasilitas makro #define. Contoh :

```
#define phi 3.14
#define usia 30
```

3.7 OPERATOR

Operator merupakan simbol atau karakter yang biasa dilibatkan dalam program untuk melakukan sesuatu operasi atau manilpulasi, seperti menjumlahkan dua buah nilai, memberikan nilai ke suatu variabel, memebandingkan kesamaan dua buah nilai.

Sebagian operator Bahasa C tergolong sebagaii operator ***binary***, yaitu operator yang dikenakan terhadap dua buah nilai(operand).

Operator Aritmatika

Operator untuk operasi aritmatika yang tergolong sebagai operator binary, yaitu :

OPERATOR	KETERANGAN
*	Perkalian
/	Pembagian
+	Penjumlahan
-	Pengurangan
%	Modulus / Sisa bagi

contoh :

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

main()
{
int nilai1,nilai2,hasil;

  nilai1=30;
```

```
  nilai2=12;

  hasil=nilai1*nilai2;
  printf("Hasil Perkalian = %d \n",hasil);

  hasil=nilai1+nilai2;
  printf("Hasil Penjumlahan = %d \n",hasil);

getch();
}
```

Program 3.2 Penggunan Operator Aritmatika

Operator Penaikan dan Penurunan

Masih berkaitan dengan operasi aritmatika, Bahasa C menyediakan operator yang disebut sebagai operator penaikan dan operator penurunan.

OPERATOR	KETERANGAN
++	Incremen (plus 1)
--	Decrement (minus 1)

Operator penaikan digunakan menaikkan nilai variabell sebesar satu, sedangkan operator penurunan dipakai untuk menurunkan nilai variabel sebesar satu. Penempatan operator terhadap variabel dapat dilakukan dimuka atau dibelakangnya, sebagai contoh :

```
x = x + 1;
x = y - 1;
```

Bisa ditulis menjadi:

```
++x;
--y;
```

Atau:

```
X++;
y--;
```

Pada contoh diatas penempatan operator penaikan atau penurunan didepan atau dibelaj=kang tidak ada bedanya. Namun sesungguhnya perbedaannya ada, hanya saja pada contoh ini tidak tampak. Perbedaan akan terlihat dengan jelas dengan melihat dua contoh berikut:

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

main()
{
  int r=10;

  printf("Sebelum Penaikan %d \n",r);

  printf("Setelah Penaikan %d \n",++r);

getch();
}
```

Program 3.3 Penggunan Operator Penaikan

Pada program 3.3, statement printf() yang pertama, menampilkan nilai **r** awal yang diberi nilai 10. Kemudian printf() yang kedua, nilai **r** disandingkan dengan operator penaikan menjadi **++r** , sehngga nilai r berubah menjadi 11. Operator penaikan ++r, identik dengan r=r+1;

Prioritas Operator Aritmatika

Masing-masing operator memiliki prioritas tertentu yang digunakan sebagai acuan dalam menyelesaikan pernyataan aritmatika. Operator yang memiliki prioritas tertinggi, akan diutamakan atau didahulukan dalam pengerjaannya. Daftar prioritas operator sebagai berikut :

PRIORITAS	OPERATOR	KETERANGAN
Tertinggi	++ ; --	Penaikan ; Penurunan
	* ; / ; %	Kali ; Bagi ; Modulus
Terendah	+ ; -	Tambah ; Kurang

Jika operator memiliki prioritas yang sama (pada tabel terletak pada baris yang sama), operator yang terletak disebelah kiri yang akan diutamakan untuk dikerjakan terlebih dahulu.
Contoh :

```
x = 2 + 3 * 2;
```

Pernyataan ini memberikan nilai 8 ke x, disebabkan 3 * 2 yang dikerjakan terlebih dahulu (karena * mempunyai prioritas yang lebih tinggi daripada +). Pernyataan diatas identik dengan:

```
z = 2 + (3 * 2);
```

```
x = 2 * 3 % 2;
```

Operator * dan % mempunyai prioritas yang sama, namun karena yang terletak disebelah kiri adalah * maka 2 * 3 akan dikerjakan terlebih dahulu. Dengan demikian pernyataan ini identik dengan:

```
x = (2 * 3) % 2;
```

Untuk mengubah urutan pengerjaan, tanda kurung bisa digunakan. Contoh :

```
x = (2 + 3) * 2;
```

Akan memberikan nilai 10 ke x, sebab 2 + 3 dikerjakan terlebih dahulu dan hasilnya baru dikalikan dengan 2.

Operator Manipulasi Bit

Untuk keperluan manipulasi data dalam bentuk bit, Bahasa C menyediakan enam buah operator seperti berikut :

OPERATOR	OPERASI
<<	Geser bit ke kiri
>>	Geser bit ke kanan
&	Dan (AND)
I	Atau (OR)
^	XOR
~	NOT (komplemen)

Seluruh operator manipulasi bit hanya bisa dikenakan pada operand bertipe integer atau karakter.

Prioritas operator manipulasi bit

Tertinggi	~
	>> <<
	&
	^
Terendah	I

Catatan: Operator manipulasi bit mempunyai prioritas lebih rendah dibandingkan operator aritmatika.

Bit terkanan dalam penyajian bilangan biner disebut sebagai bit 0.

7	6	5	4	3	2	1	0
1	0	0	1	1	1	1	0

Operator geser kiri (<<) dan geser kanan (>>)

Bentuk umum pemakaian operator << dan >> :

Nilai << jumlah bit digeser ke kiri
Nilai >> jumlah bit digeser ke kanan

Contoh program yang melibatkan operator << dan >> :

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

main()
{
  unsigned x = 93;

  printf("Nilai x semula = %d\n", x);

  x = x << 1;      /* geser ke kiri 1 bit */
  printf("Nilai x kini       = %d\n", x);

getch();
}
```

Program 3.4 Penggunan Operator Manipulasi Bit <<

Contoh Eksekusi:

Nilai x semula = 93
Nilai x kini = 186

Jika x bernilai 93 kemudian digeser ke kiri 1 bit ternyata hasilnya adalah 186.

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

main()
{
  unsigned x = 93;
```

```
  printf("Nilai x semula = %d\n", x);

  x = x >>1;      /* geser ke kanan 1 bit */
  printf("Nilai x kini       = %d\n", x);

getch();
}
```

Program 3.5 Penggunan Operator Manipulasi Bit >>

Contoh Eksekusi:

Nilai x semula = 93
Nilai x kini = 46

Bila x = 93 dan digeser ke kanan 1 bit ternyata hasilnya adalah 46. Penjelasannya adalah sebagai berikut:

Operator Atau (|), XOR (^), Dan (&)

Bentuk pemakaian operator |, ^ dan & :

operand_1 operator operand_2

Operasi bit dilakukan antara ***operand1*** dan ***operand2*** untuk posisi bit yang sama (bit 0 operand1 dengan bit 0 operand2, bit ke-1 operand1 dengan bit operand2, dan seterusnya).

Operator Atau / OR (|) mempunyai sifat, bit hasil bernilai 1 jika ada bit operand yang bernilai 1. Operator Dan / AND (&) mempunyai sifat, bit hasil bernilai 1 hanya jika kedua bit operand bernilai 1. Operator XOR (^) mempunyai sifat, bit hasil bernilai 1 jika hanya satu operand yang bernilai 1. Tabel berikut memberikan penjeasan seluruh kemungkinan nilai bit operand beserta hasil operasinya.

Tabel 3-9. Kemungkinan operasi AND, OR, XOR

bit operand1	bit operand2	Hasil		
		|	&	^
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	0

Contoh dalam bentuk program:

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

main()
{
  unsigned nilai = 81;
  unsigned cadar = 99;
  unsigned a, b, c;

  a = nilai | cadar;          /*  operasi atau   */
  b = nilai & cadar;        /*  operasi dan    */
  c = nilai ^ cadar;        /*  operasi XOR    */
  printf ("a = %u, b = %u, c = %u\n",  a, b, c);

getch();
}
```

Program 3.6 Penggunan Operator "|", " ^", dan "&"

Contoh Eksekusi:
a = 115, b = 65, c = 50

Operator Komplemen (~)

Bentuk pemakaian

~operand

Operator ini mempunyai sifat membalik (menginversi) nilai setiap bit. Jika bit operand bernilai 1 hasilnya 0, dan bila bit operand bernilai 0 hasilnya 1.

Contoh pemakaian operator komplemen;

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

main()
{
unsigned nilai = 81;
unsigned a;

a = ~nilai;      /* komplemen dari nilai */

printf("a = %u\n", a);

getch();
}
```

Program 3.7 Penggunan Komplemen

Contoh Eksekusi:
a = 65454

Operator Sizeof

Bahasa C mempunyai operator yang akan menghasilkan ukuran dari suatu variabel atau suatu tipe pada saat kompilasi. Oerator tersebut bernama **sizeof,** dengan operand (berupa tipe atau variabel) ditempatkan di dalam tanda kurung.
Contoh :

sizeof(char) 1
sizeof(int) 2

Operator ini sangat bermanfaat untuk menghitung besarnya ukuran sebarang variabel atau tipe, terutama untuk variabel atau tipe yang kompleks (seperti struktur, yang akan dibahas pada bab XI).

Program berikut memberikan contoh tentang pemakaina sizeof.

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

main()
{
int i;
float f;

printf("ukuran variabel int=%d\n", sizeof(i));
printf("ukuran variabel float=%d\n", sizeof(f));
printf("ukuran tipe char=%d\n", sizeof(char));
printf("ukuran tipe double=%d\n",sizeof(double));

getch();
}
```

Program 3.8 Penggunan sizeof

Contoh Eksekusi:

```
Ukuran variabel int    = 2
Ukuran variabel float = 4
Ukuran tipe char       = 1
Ukuran tipe double     = 8
```

Prioritas Operator

Seluruh operator dalam C beserta prioritasnya ditunjukkan pada tabel 3-10. Beberapa diantaranya tidak dibahas pada bab ini, melainkan pada bab-bab di belakang.

Maksud pengerjaan dari kiri ke kanan atau kanan ke kiri dijeaskan melalui contoh berikut ini:

a = b = c;

Pada pernyataan di atas, operator yang dilibatkan (=) mempunyai sifat pengerjaan dimulai dari kanan. Berarti b = c akan dikerjakan terlebih dahulu, barulah kemudian mengerjakan a = b.

x = 2 * 3 * 4;

Pada pernyataan di atas, 2 * 3 akan dikerjakan terlebih dahulu, barulah kemudian mengerjakan perkalian hasil 6 dengan 4. Adapun karena prioritas = lebih rendah dari *, maka 2 * 3 * 4 dikerjakan lebih dahulu. Selanjutnya hasilnya baru diberikan ke x.

LATIHAN - 3

Lat 3.1 :
Perhatikan program berikut, **dan pada bagian man kesalah program terjadi** ?

```
#include <stdio.h>
#include <stdio.h>
main()
{
  int bil1,bil2,tot@l;

  bil1=200;
  bil2=300;
  total=Bil1+bil2;
  printf("Tot@l = %i", total);

getch();
}
```

Lat 3.2 :
Tuliskan Kode Program yang dapat menampilkan tampilan sebagai berikut ?

```
#================================================#
# Program Mengitung Luas dan Keliling Lingkaran #
#================================================#

Ketik Jari-jari lingkaran : _______

# HASIL  #
Keliling = __
Luas     = __
```

Lat 3.3 :
Berdasarkan pada latihan Lat 3.2, buatlah program untuk menghitung :

a. Luas Persegi Panjang
b. Keliling Persegi Panjang
c. Luas Segitiga
d. Keliling Segitiga

Lat 3.4:
Buatlah program yang dapat menyelesaikan soal-soal fisika anda ketika anda berada di SMU !

Lat 3.5:
Tuliskan Kode Program yang dapat menampilkan tampilan sebagai berikut ?

```
#=======================================#
#      Program Penjualan Minimarket     #
#=======================================#

Nama Barang    : ____
Harga Barang   : ____
Jumlah Barang  : ____

# HASIL  #
Nama Barang    : ____
Harga Barang   : ____
Jumlah Barang  : ____
Total Bayar    : ____
```