Bab 2. Dasar-Dasar Pemrograman C

Konsep Pemrograman

Politeknik Elektronika Negeri Surabaya

2006

Overview

- Tipe Data Standar *(Standart Data Type)*
- Aturan Pendefinisian Identifier
- Variabel
  - Mendeklarasikan Variabel
  - Inisialisasi Variabel
- Konstanta
- Operator
  - Operator Aritmatika
  - Operator Penurunan dan Penaikan
  - Prioritas Operator Aritmatika
  - Operator Penugasan
  - Operator Kombinasi (Pemendekan)
- Operasi I/O
  - Fungsi Standart
  - Fungsi Standart Untuk Operasi I/O
  - Fungsi Standart untuk Operasi Output
    - `printf()`
    - `putchar()`
  - Fungsi Standart untuk Operasi Input
    - `scanf()`
    - `getchar()`

Tipe Data Standar

- Data merupakan suatu nilai yang bisa dinyatakan dalam bentuk konstanta atau variabel.
  - Konstanta menyatakan nilai yang tetap.
  - Variabel menyatakan nilai yang dapat diubah-ubah selama eksekusi berlangsung.
- Berdasarkan jenis/tipenya, data dapat dibagi menjadi lima kelompok, yang dinamakan sebagai tipe data dasar.
  - Bilangan bulat (integer) → int (short int, long int, signed int, unsigned int)
  - Bilangan real presisi-tunggal → float
  - Bilangan real presisi-ganda → double
  - Karakter → char
  - Tak-bertipe (*void*), keterangan lebih lanjut tentang void dijelaskan dalam Bab V.

Tabel Tipe Data

Tabel 2-1. Ukuran memori untuk tipe data

Tipe	Total bit	Kawasan	Keterangan
char	8	-128 s/d 127	karakter
int	32	-2147483648 s/d 2147483647	bilangan integer
float	32	1.7E-38 s/d 3.4E+38	bilangan real presisi-tunggal
double	64	2.2E-308 s/d 1.7E+308	bilangan real presisi-ganda

- Untuk tipe data *short int*, *long int*, *signed int* dan *unsigned int*, maka ukuran memori yang diperlukan serta range-nya sebagai berikut :

Tabel 2-2 Ukuran memori untuk tipe data int

Tipe	Total bit	Kawasan	Keterangan
short int	16	-32768 s/d 32767	short integer
long int	32	-2147483648 s/d 2147483647	long integer
signed int	32	-2147483648 s/d 2147483647	biasa disingkat dengan int
unsigned int	32	0 s/d 4294967295	bilangan int tak bertanda

- Ukuran dan kawasan dari masing-masing tipe data adalah bergantung pada jenis mesin yang digunakan (misalnya mesin 16 bit bisa jadi memberikan hasil berbeda dengan mesin 32 bit).

Aturan Pendefinisian Identifier

→ Variabel, konstanta & nama fungsi

- Identifier harus diawali dengan huruf (A..Z, a..z) atau karakter garis bawah (_).
- Selanjutnya dapat berupa huruf, digit (0..9) atau karakter garis bawah atau tanda dollar ($).
- Panjang pengenal boleh lebih dari 31 karakter, tetapi hanya 31 karakter pertama yang akan dianggap berarti.
- Pengenal tidak boleh menggunakan nama yang tergolong sebagai kata-kata cadangan (*reserved words)* seperti int, if, while dan sebagainya.

Mendeklarasikan Variabel

- Variabel digunakan dalam program untuk menyimpan suatu nilai, dan nilai yang ada padanya dapat diubah-ubah selama eksekusi program berlangsung.
- Variabel yang akan digunakan dalam program haruslah dideklarasikan terlebih dahulu.
- Pengertian deklarasi di sini berarti **memesan memori dan menentukan jenis/tipe** data yang bisa disimpan di dalamnya.
- Bentuk umum deklarasi variabel:

```
tipe daftar-variabel;
```

- Pada pendeklarasian varibel, daftar-variabel dapat berupa sebuah variabel atau beberapa variabel yang dipisahkan dengan koma. Contoh:

```
int bil;
float luas, radius;
```

Inisialisasi Variabel

- Adakalanya dalam penulisan program, setelah dideklarasikan, variabel langsung diberi nilai awal (inisialisasi)

```
int bil, total;

bil = 10;
total = 0;
```

- Dua pernyataan di atas sebenarnya dapat disingkat melalui pendeklarasian yang disertai penugasan nilai, sebagai berikut :

```
int bil = 10, total=0;
```

- Cara seperti ini banyak dipakai dalam program C, di samping menghemat penulisan pernyataan, juga lebih memberikan kejelasan, khususnya untuk variabel yang perlu diberi nilai awal (diinisialisasi) seperti `total` yang dijadikan sebagai variabel penampung.

Konstanta

- Konstanta menyatakan nilai yang tetap.
- Berbeda dengan variabel, suatu konstanta tidak dideklarasikan.
- Namun seperti halnya variabel, konstanta juga memiliki tipe.
- Penulisan konstanta mempunyai aturan tersendiri, sesuai dengan tipe masing-masing.
- Pendefinisian konstanta menggunakan *preprocesor directive* `#define` , dengan tanpa diakhiri dengan titik koma

Konstanta

- Konstanta karakter misalnya ditulis dengan diawali dan diakhiri dengan tanda petik tunggal, contohnya :

  ```
  #define HRF ‘A’
  ```

- Konstanta integer ditulis dengan tanda mengandung pemisah ribuan dan tak mengandung bagian pecahan, contohnya :
- ```
  #define MAX 10
  ```
- Konstanta real (*float* dan *double*) bisa mengandung pecahan (dengan tanda berupa titik) dan nilainya bisa ditulis dalam bentuk eksponensial (menggunakan tanda e), contohnya : 27.5f (untuk tipe *float*) atau 27.5 (untuk tipe *double*) dan 2.1e+5 (maksudnya 2,1 x 105).
- ```
  #define PHI 3.14f
  ```
- ```
  #define NILAI 8.75
  ```
- Konstanta string merupakan deretan karakter yang diawali dan diakhiri dengan tanda petik-ganda (“), contohnya :

  ```
  #define KALIMAT “Pemrograman Dasar C”
  ```
  ```

Operator

- Operator merupakan simbol atau karakter yang biasa dilibatkan dalam program untuk melakukan sesuatu operasi atau manipulasi, seperti menjumlahkan dua buah nilai, memberikan nilai ke suatu variabel, membandingkan kesamaan dua buah nilai.
- Berdasarkan jumlah operandnya :
  - Unary operator, contoh : `-c`

    operator yang hanya memiliki sebuah operand (yaitu C pada contoh ini).
  - Binary operator, contoh : `a + b`

    Sebagian operator C tergolong sebagai operator binary, yaitu operator yang dikenakan terhadap dua buah nilai *(operand)*.
  - Ternary operator, contoh : `? :`

```
hasil = (x > y) ? 0 : 1;
```

Operator Aritmatika

- Operator untuk operasi aritmatika yang tergolong sebagai operator binary adalah :
  - `*` perkalian
  - `/` pembagian (hasil bagi)
  - `%` sisa pembagian
  - `+` penjumlahan
  - `-` pengurangan
- Adapun operator yang tergolong sebagai operator unary.
  - `-` tanda minus
  - `+` tanda plus

Contoh operator aritmatika

```
/* Menghitung diskriminan pers kuadrat
  ax^2 + bx + c = 0 */
#include <stdio.h>
main()
{
    float a,b,c,d;

    printf("Masukkan nilai a, b dan c pisahkan
      dengan koma);
    scanf("%f %f %f", &a, &b, &c);
    d = b*b-4*a*c;
    printf("Diskriminan =%f\n", d);
}
```

Operator Modulus

- Operator yang telah dituliskan di atas, yang perlu diberi penjelasan lebih lanjut adalah operator modulus/sisa pembagian.
- Contoh :
  - Sisa pembagian bilangan 7 dengan 2 adalah 1 (7 % 2 → 1)
  - Sisa pembagian bilangan 6 dengan 2 adalah 0 (6 % 2 → 0)
  - Sisa pembagian bilangan 8 dengan 3 adalah 1 (8 % 3 → 2)
- Kegunaan operator ini diantaranya bisa dipakai untuk menentukan suatu bilangan bulat termasuk ganjil atau genap, berdasarkan logika : “Jika bilangan habis dibagi dua (sisanya nol), bilangan termasuk genap, sebaliknya termasuk ganjil”.

Operator Penurunan & Penaikan

- Masih berkaitan dengan operasi aritmatika, C menyediakan operator yang disebut sebagai operator penaikan dan operator penurunan, yaitu :

  ++ operator penaikan

  -- operator penurunan

- Operator penaikan digunakan untuk menaikkan nilai variabel sebesar satu. Penempatan operator terhadap variabel dapat dilakukan di muka atau di belakangnya, bergantung pada kondisi yang dibutuhkan oleh pemrogram. contohnya :

  x = x+1; → ++x atau x++

  y = y-1; → --y atau y--

- Berikut ini adalah contoh yang akan menunjukkan perbedaan pemakaian dan hasil dari ++x dengan x++ (atau pemakaian y-- dengan –y).

Contoh operator pemendekan

```
//penggunaan pre & post Increment operator
#include <stdio.h>
main()
{
   int count = 0, loop;

   loop = ++count;
   //count=count+1; loop=count;
   printf("loop = %d, count = %d\n", loop, count);
   loop = count++;
   //loop=count; count=count+1;
   printf("loop = %d, count = %d\n", loop, count);
}
```

Prioritas Operator Aritmatika

Tabel 2.3 Tabel prioritas operator aritmatika dan urutan pengerjaannya

PRIORITAS	OPERATOR	URUTAN PENGERJAAN
Tertinggi	( )	dari kiri ke kanan
	! ++ -- + -	dari kanan ke kiri *)
	* / %	dari kiri ke kanan
	+ -	dari kiri ke kanan *)
Terendah	= += -= *= /= %=	dari kanan ke kiri

*) Bentuk **unary** + dan **unary** − memiliki prioritas yang lebih tinggi daripada bentuk **binary** + dan **binary** −

Operator Penugasan *(Assignment)*

- Operator penugasan *(assignment operator)* digunakan untuk memindahkan nilai dari suatu ungkapan *(expression*) ke suatu identifier (variabel).
- Operator pengerjaan yang umum digunakan dalam bahasa pemrograman, termasuk bahasa C adalah operator sama dengan (=).
- Contohnya :

```
fahrenheit = celcius * 1.8 + 32;
```

- Maka '=' adalah operator penugasan yang akan memberikan nilai dari ungkapan : celcius * 1.8 + 32 kepada variabel fahrenheit**.**

  → ekspresi di sebelah kanan tanda '=' diproses sampai tuntas, kemudian hasilnya di-assign ke identifier/variabel di sebelah kirinya
- Bahasa C juga memungkinkan dibentuknya statemen penugasan menggunakan operator pengerjaan jamak dengan bentuk sebagai berikut :

  var1 = var2 = … = ekspresi ;

  Misalnya :

```
a = b = 15;
```

  maka nilai variabel 'a ' akan sama dengan nilai variabel 'b' akan sama dengan 15.

Operator Kombinasi (Pemendekan)

- C menyediakan operator yang dimaksudkan untuk memendekkan penulisan operasi penugasan semacam

Tabel 2.4 Seluruh kemungkinan operator kombinasi dan padanannya

```
x +=  2;   kependekan dari  x = x + 2;
x -= 2;    kependekan dari  x = x - 2;
x *=  2;   kependekan dari  x = x * 2;
x /=  2;   kependekan dari  x = x / 2;
x %=  2;   kependekan dari  x = x % 2;
x <<= 2;   kependekan dari  x = x << 2;
x >>= 2;   kependekan dari  x = x >> 2;
x &=  2;   kependekan dari  x = x & 2;
x |=  2;   kependekan dari  x = x | 2;
x ^=  2;   kependekan dari  x = x ^ 2;
```

Operasi I/O → stdio.h

- Ada beberapa fungsi standart yang digunakan untuk melakukan operasi I/O
- Di antaranya :
  - Output
    - `printf()` → menampilkan output di layar dengan format tertentu
    - `putchar()` → menampilkan output berupa satu karakter di layar
  - Input
    - `scanf()` → menerima input dari keyboard dengan format tertentu
    - `getchar()` → menerima input satu karakter dari keyboard

Fungsi Standart

- Fungsi adalah bagian dari program yang memiliki tugas khusus/tertentu
- Fungsi Standart adalah fungsi yang sudah *available* (tersedia, tinggal dipakai) dalam sebuah compiler yang diinstall
- User tinggal memanggil fungsi-fungsi standart ketika membutuhkannya
- User juga harus menyertakan file header yang mendefinisikan prototype dari fungsi-fungsi tsb pada *preprocessor include*

Fungsi Standart Untuk Operasi I/O

- Compiler C telah mendefinisikan beberapa fungsi standart berkenaan dengan operasi I/O
- Semua fungsi tsb prototype-nya didefinisikan pada file header : `stdio.h`
- Program yang memanggil fungsi-fungsi I/O tsb harus menyertakan file headernya dengan menuliskan :

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

pada posisi teratas (bagian *preprocessor*)

Fungsi Standart untuk Operasi Output : `printf()`

- Tugasnya menampilkan data ke layar dengan format tertentu
- Bentuk umumnya :

```
printf("string kontrol",argumen1,argumen2,…);
```

- String kontrol dapat berupa keterangan yang akan ditampilkan pada layar beserta penentu formatnya (seperti `%d, %f,%c`).
- Penentu format dipakai untuk memberi tahu kompiler mengenai jenis data yang akan ditampilkan.
- Argumen sesudah string kontrol (argumen1, argumen2,...)adalah data yang akan ditampilkan ke layar yang berupa variabel, konstanta atau bahkan sebuah ekspresi

Fungsi Standart untuk Operasi Output : `printf()`

- Teks apapun yang ada di antara 2 tanda petik ganda " ", akan dicetak TANPA MODIFIKASI, kecuali jika bertemu dengan karakter % atau \
- Karakter % menandakan format tampilan sedangkan karakter \ menandakan karakter khusus yang harus diterjemahkan terlebih dahulu sebelum ditampilkan

Penentu format tampilan

- Penentu format untuk data string atau karakter:
  - %c untuk menampilkan sebuah karakter
  - %s untuk menampilkan sebuah string

Penentu format tampilan

- Untuk menampilkan data bilangan, penentu format yang dipakai adalah sbb:

`%u`	untuk menampilkan data bilangan tak bertanda *(unsigned)* dalam bentuk desimal.
`%d`, `%i`	untuk menampilkan bilangan integer bertanda *(signed)* dalam bentuk desimal
`%o`	untuk menampilkan bilangan bulat tak bertanda dalam bentuk oktal.
`%x`, `%X`	untuk menampilkan bilangan bulat tak bertanda dalam bentuk heksadesimal (`%x` → notasi yang dipakai : a, b, c, d, e dan f sedangkan `%X` → notasi yang dipakai : A, B, C, D, E dan F)
`%f`	untuk menampilkan bilangan real dalam notasi : dddd.dddddd
`%e`, `%E`	untuk menampilkan bilangan real dalam notasi eksponensial
`%g`, `%G`	untuk menampilkan bilangan real dalam bentuk notasi seperti `%f`, `%E` atau `%F` bergantung pada kepresisian data (digit `0` yang tak berarti tak akan ditampilkan)
l	merupakan awalan yang digunakan untuk `%d`, `%u`, `%x`, `%X`, `%o` untuk menyatakan long int (misal `%ld`). Jika diterapkan bersama `%e`, `%E`, `%f`, `%F`, `%g` atau `%G` akan menyatakan *double*
L	Merupakan awalan yang digunakan untuk `%f`, `%e`, `%E`, `%g` dan `%G` untuk menyatakan *long double*
h	Merupakan awalan yang digunakan untuk `%d`, `%i`, `%o`, `%u`, `%x`, atau `%X`, untuk menyatakan *short int*.

Penentu format tampilan

- Penentuan panjang medan bagi tampilan data, dengan cara menyisipkan bilangan bulat sesudah tanda % dalam penentu format yang menyatakan panjang medan.
- Untuk data yang berupa bilangan bulat, misalnya:

```
printf("Abad %4d", 20);
```

`%4d` menyatakan medan untuk menampilkan bilangan **20** adalah sepanjang 4 karakter.

- Untuk data yang berupa bilangan real, spesifikasi medannya berupa → m.n

m = panjang medan m.n dan n = jml digit pecahan

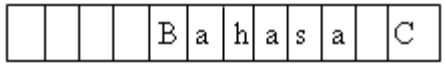
```
printf("Harga : Rp %8.2f\n", 500.0);
```

`%8.2f` menyatakan panjang medan dari bilangan real yang akan ditampilkan adalah 8 karakter dengan jumlah digit pecahan 2 buah.

Penentu format tampilan

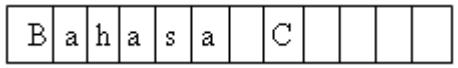
- Untuk data yang berupa string, contoh :

```
printf("%12s", "Bahasa C");
```

Hasilnya : 

- Penentu format yang mengandung panjang medan, *default*-nya menampilkan data berbentuk rata kanan terhadap panjang medan yang diberikan.
- Untuk menampilkan dalam bentuk rata kiri, maka sesudah tanda % pada penentu format perlu disisipkan tanda – (minus), contoh :

```
printf("%-12s", "Bahasa C");
```

Hasilnya : 

Karakter khusus pada tampilan layar

- Tanda \ pada string yang menjadi argumen `printf()` mempunyai makna yang khusus.
- Digunakan untuk menyatakan karakter khusus
- Di antaranya :
  - \n → menyatajab karakter baris-baru
  - \" → menyatakan karakter petik-ganda
  - \\ → menyatakan karakter backslash
  - \t → menyatakan karakter tab

Fungsi Standart untuk Operasi Output : `putchar()`

- Digunakan khusus untuk menampilkan sebuah karakter di layar.
- Penampilan karakter tidak diakhiri dengan perpindahan baris.
- Contoh :
  - `putchar('A');`

  menghasilkan keluaran yang sama dengan
  - `printf("%c",'A');`

Fungsi Standart untuk Operasi Input : `scanf()`

- Digunakan untuk memasukkan berbagai jenis data dari keyboard dengan format tertentu
- Bentuk `scanf()` menyerupai fungsi *`printf()`*.
- Fungsi ini melibatkan penentu format yang pada dasarnya sama digunakan pada *`printf()`*
- Bentuk umum
  - `scanf("string kontrol", daftar_argumen);`

Fungsi Standart untuk Operasi Input : `scanf()`

- Pada `scanf()`, **daftar_argumen** dapat berupa satu atau beberapa argumen dan haruslah berupa **ALAMAT/ADDRESS**.
- Untuk menyatakan alamat dari variabel, di depan variabel dapat ditambahkan tanda & (tanda & dinamakan sebagai operator alamat).
- Contoh :

```
scanf("%f", &radius);
```

berarti (bagi komputer) : "bacalah sebuah bilangan real (`%f`) dan tempatkan ke alamat dari **radius** (`&radius`)".

Fungsi Standart untuk Operasi Input : `getchar()`

- Digunakan khusus untuk menerima masukan berupa sebuah karakter dari keyboard
- Contoh :
  - `kar = getchar();`
- maka variabel `kar` akan berisi karakter yang diketikkan oleh user atau EOF *(end of file)* jika ditemui akhir dari file.

Latihan

1. Diketahui variabel-variabel sebagai berikut:

```
var_bulat = 32767;
var_pecahan1 = 339.2345678f;
var_pecahan2 = 3.4567e+40;
var_karakter = 'S';
```

Buat program untuk menampilkan semua variabel di atas.

2. Diketahui variabel-variabel sebagai berikut:

```
int  a = 12, b = 2, c = 3, d = 4;
```

Buat program untuk mencetak hasil :

```
a % b
a - c
a + b
a / d
a / d * d + a % d
a % d / d * a - c
```

Latihan

3. Buatlah program untuk mengkonversi suhu dari Celcius ke Fahrenheit dengan rumus

   F = C * 1.8 + 32

   Input : suhu dalam Celcius

   Output : suhu dalam Fahrenheit

4. Buat program untuk menginputkan satu karakter dari keyboard kemudian cetak karakter tersebut ke layar.
5. Buat program untuk menghitung luas lingkaran, dengan panjang jari-jari yang diinputkan dari keyboard. Definisikan sebuah konstanta PI dengan nilai 3.14f

Alhamdulillahi robbil ‘alamin