

Bab 2

Variable, Konstanta, dan Input/Output

Penamaan Variable

Variable merupakan wadah penyimpanan data-data yang digunakan oleh program. Ada beberapa aturan penamaan variable yang perlu diperhatikan :

- Nama sebuah variable harus terdiri dari huruf, angka, dan *underscore* (_).
- Nama variable harus diawali dengan huruf atau *underscore*.
- Nama variable *case sensitive*, artinya huruf kapital dan huruf kecil dianggap beda.
- Tidak boleh sama dengan keyword seperti : if, else, int, float, dll.

Biasanya penamaan variable pada program C menggunakan huruf kecil semua, sedangkan penamaan konstanta menggunakan huruf kapital semua.

Tips : Sebaiknya jangan menggunakan nama variable yang diawali dengan *underscore*, karena nama seperti ini biasanya dipakai pada *library*.

Tips : Gunakan *underscore* untuk penamaan variable yang panjang, supaya lebih mudah dibaca.

Variable Dasar

Ada 4 macam variable dasar yang dapat kita gunakan, yaitu :

char	tipe data numerik (8 bit) / karakter
int	tipe data numerik (16/32 bit)
float	tipe data bilangan pecahan
double	tipe data bilangan pecahan yang lebih presisi daripada float

Variable Turunan

Selain 4 macam tipe diatas, kita juga dapat menggunakan *modifier* seperti:

short	digunakan untuk menciptakan int 16 bit
long	digunakan untuk menciptakan int 32 bit
unsigned	digunakan untuk menyatakan variable yang isinya selalu positif

Konstanta

Konstanta adalah nilai yang tidak berubah. Konstanta angka dapat langsung dituliskan seperti : 1000, 344, 1, 0, dll. Konstanta karakter ditulis dengan batas kutip 1 ('), seperti '2', 'a', 'A', 'k', dll. Konstanta string ditulis dengan batas kutip 2 ("), seperti "Hello C World !", "Apa kabar ?", "Abc def ghij", dll.

Ada beberapa karakter khusus yang disebut *escape character*. Karakter-karakter ini dituliskan dengan tanda *back-slash* (\) dan diikuti kode tertentu. Karakter-karakter tersebut diantaranya :

\n	new line
\t	tab
\\	back-slash
\?	tanda tanya
\'	kutip satu
\"	kutip dua
\0	null

Deklarasi Variable

Variable dideklarasikan dengan format <tipe> <nama>. Untuk mendeklarasikan beberapa variable sekaligus, gunakan tanda koma (,) untuk memisahkan nama-namanya.

Contoh pendeklarasian variable :

```
int a;  
int x,y,z;
```

Anda bisa mengisi nilai awal pada variable yang baru dibuat dengan menggunakan tanda sama dengan (=). Contoh pengisian nilai variable :

```
int a = 10;  
int x = 3, y = 4, z = 5;
```

Deklarasi Variable Konstanta

Anda dapat membuat variable yang tidak dapat berubah isinya. Sebuah variable konstanta diberi nilai awal, lalu nilai tersebut tidak dapat dirubah lagi. Cara mendefinisikan konstanta adalah dengan menambahkan "const" diawal deklarasi.

Contoh pendeklarasian konstanta :

```
const int JARAK = 100;  
const int WAKTU = 30;
```

Fungsi Input/Output

Fungsi I/O pada bahasa C sebenarnya amat banyak. Dua fungsi yang populer digunakan adalah fungsi `scanf` dan fungsi `printf`. Pada bab 1 kita sudah mengenal sekilas tentang fungsi `printf`. Fungsi `printf` digunakan untuk output ke layar, sedangkan `scanf` sebaliknya, digunakan untuk input dari keyboard.

Fungsi `printf` yang sudah kita coba sebelumnya menerima sebuah argumen, yaitu kalimat yang ingin di-print. Bagaimana jika kita ingin mencetak isi dari variable ?

Cobalah program dibawah ini :

```
#include<stdio.h>  
  
int a = 10;  
char b = 'A';  
float c = 25.4567;  
  
int main()  
{  
    printf( "integer a =");  
    printf("%d", a);  
    printf("\n");  
  
    printf("karakter b = ");  
    printf("%c", b);  
    printf("\n");  
  
    printf("float c = ");  
    printf("%f", c);  
    printf("\n");  
}
```

```
    return 0;
}
```

Pada contoh diatas, kita menggunakan 2 buah argumen untuk mencetak isi variable. Argumen pertama diisi "%d", "%c", atau "%f". String ini disebut *formatting string*. Tipe variable `int` menggunakan *formatting* %d, `char` menggunakan %c, `float` dan `double` menggunakan %f.

Formatting string dapat juga disatukan dengan string biasa, sehingga program diatas dapat dituliskan sebagai :

```
#include<stdio.h>

int a = 10;
char b = 'A';
float c = 25.4567;

int main()
{
    printf( "integer a = %d\n", a);
    printf("karakter b = %c\n", b);
    printf("float c = %f\n", c);

    return 0;
}
```

atau bahkan :

```
#include<stdio.h>

int a = 10;
char b = 'A';
float c = 25.4567;

int main()
{
    printf("integer a = %d\nkarakter b = %c\nfloat c = %f\n", a,b,c);
    return 0;
}
```

Fungsi `scanf` merupakan kebalikan dari `printf`, namun cara kerjanya hampir sama. Fungsi ini menerima input dari keyboard. Sama dengan `printf`, `scanf` juga menggunakan *formatting string*. Perhatikan contoh dibawah ini :

```
#include<stdio.h>

int a;
char b;
float c;

int main()
{
    scanf("%d\n", &a);
    scanf("%c\n", &b);
    scanf("%f", &c);
}
```

```
printf("Hasil : %d, %c, %f\n", a,b,c);  
return 0;  
}
```

Program diatas menerima input dari keyboard sebanyak 3 buah, yaitu sebuah int a, sebuah char b, dan sebuah float c. Untuk memasukan input tersebut ke variable, gunakan tanda '&' diawal variable.

Saat kita meng-*input*, kita selalu menekan tombol *enter*, karena itu pada *formatting string* pertama dan kedua kita perlu menambahkan karakter '\n'. Hal ini perlu dilakukan supaya ketika kita menekan tombol *enter* (tanda selesai meng-*input*), karakter tersebut tidak dianggap sebagai *input* untuk `scanf` kedua, dst.

Pada `scanf` terakhir kita tidak perlu menambahkan karakter '\n', karena tidak perlu khawatir akan salah terbaca.