

Algoritma dan Pemrograman

bagian 1

2009

Modul ini menjelaskan tentang bahasa C dan apa saja yang dibutuhkan bila kita akan menulis suatu program dengan bahasa C. Editor yang dipakai adalah Turbo C++ 4.5. Struktur program yang akan dipakai adalah struktur program bahasa C.

**Pengenalan C
dan struktur
program**

LATAR BELAKANG

Bahasa C adalah sebuah bahasa pemrograman yang dipakai untuk membuat suatu program komputer. Bahasa C dirancang oleh DENNIS M. RITCHIE. Bahasa C adalah bahasa pemrograman prosedural. Disebut demikian karena bahasa C memiliki fungsi-fungsi yang akan dijalankan oleh program utama. Fungsi tersebut dapat dibagi menjadi 2, yaitu fungsi yang khusus dibuat dan digunakan untuk program tertentu yang dibuat dan fungsi yang terdapat di dalam header. Contoh header yang ada di dalam bahasa C adalah `stdio`, `iostream`, `stdlib`, `math`, `conio`, dll. Di dalam header tersebut telah tersedia berbagai macam fungsi, contohnya `printf`, `scanf`, `rand`, `sqrt`, `cin`, `cout`, dsb. Header tersebut harus diletakkan di dalam program dengan menggunakan *preprocessor directive* **#include**. Masing-masing fungsi tersebut memiliki aturan tersendiri dalam penggunaannya.

STRUKTUR PROGRAM

Yang namanya bahasa, pasti memiliki aturan-aturan dalam penggunaannya. Jadi, bila kita menyalahi aturan tersebut, maka komputer akan bingung dan pesan kita tidak tersampaikan dengan baik atau bahkan sama sekali tidak dimengerti oleh komputer. Bahasa C memiliki fungsi yang harus ada di dalamnya, yaitu fungsi `main`. Fungsi inilah yang akan menjalankan fungsi-fungsi yang lainnya (bila ada). Tanpa fungsi `main` tersebut, program tidak dapat dieksekusi tapi bisa dikompilasi. Atau bisa dikatakan bahwa tanpa fungsi `main`, program tidak akan mengalami error, tetapi program tersebut tidak akan menghasilkan output apa-apa.

```
<preprocessor directive>
void main(){
    <statement>;
    <statement>;
    ...
}
```

```
<preprocessor directive>
int main(){
    <statement>;
    <statement>;
    ...
    return 0;
}
```

Statement adalah perintah yang harus dikerjakan oleh komputer. Statement dapat berupa operasi aritmatika, permintaan input, atau hanya sebuah output kalimat tertentu. Beberapa contoh statement dalam bahasa C adalah sebagai berikut.

```
d = a + b;
scanf("%i", &a);
printf("Algoritma dan Pemrograman");
```

Statement dapat dibagi menjadi 4, yaitu:

1. **Statement kosong**
2. **Statement ungkapan**

3. Statement kendali

4. Statement jamak

Statement kosong adalah statement yang hanya terdiri dari tanda titik koma (;) dan tidak menghasilkan output apa pun.

```
for(int i=1;i<=10;i++); → tidak menghasilkan output apapun, hanya menghitung angka 1-10
```

Statement ungkapan adalah statement yang melakukan sesuatu dan diakhiri dengan tanda titik koma (;). Statement ini bisa berupa sebuah operasi atau sebuah perintah (input atau output).

Statement kendali adalah statement yang digunakan untuk mengendalikan proses suatu program. Statement ini bisa berupa percabangan (misal : if ... else ...) atau perulangan (misal : do ... while ...)

contoh 1:

```
if(i<0)
    printf("bilangan negatif");
else
    printf("bilangan bulat atau nol");
```

contoh 2:

```
do{
    printf("algoritma dan pemrograman");
    i++;
}while(i<=10);
```

Statement jamak adalah statement yang terdiri dari beberapa statement ungkapan. Statement jamak ditandai dengan tanda kurung kurawal { , }.

```
while(i<5)
{
    printf("praktikum");
    printf("algoritma");
    printf("dan");
    printf("pemrograman");
} } Statement jamak
```

Bila kita menambahkan fungsi ke dalam program yang kita buat, maka fungsi tersebut dapat kita letakkan *sebelum* maupun *sesudah* fungsi main, dengan catatan bila kita meletakkannya sebelum fungsi main, kita harus mendeklarasikannya terlebih dahulu. Untuk lebih jelasnya, kita akan mempelajarinya di pertemuan-pertemuan yang akan datang. Sebagai gambaran, lihat contoh penulisan fungsi di bawah ini.

```

#include<stdio.h>
int tambah (int c, int d);      //deklarasi fungsi
void main()
{
    int a=3,b=5;
    printf("hasil dari a+b adalah %d",tambah(a,b));
}

int tambah (int c, int d)
{
    int e;
    return c+d;
}

```

} fungsi

```

#include<stdio.h>
int tambah (int c, int d)
{
    int e;
    return c+d;
}
void main()
{
    int a=3,b=5;
    printf("hasil dari a+b adalah %d",tambah(a,b));
}

```

} fungsi

Di samping fungsi yang kita buat, ada juga fungsi yang sudah disediakan di dalam editor yang kita pakai, yaitu fungsi yang terdapat di dalam header. Berikut ini adalah beberapa fungsi yang terdapat di dalam header yang sering digunakan.

- a. `printf("...");`
Fungsi ini digunakan untuk menampilkan apa saja yang tertulis di dalam tanda petik("). Terdapat di dalam header `stdio`.
- b. `scanf("%i",&a);`
Fungsi ini digunakan untuk memberikan input data. Statement / fungsi tersebut memiliki arti user harus menginputkan sebuah nilai dengan tipe integer dan nilai tersebut akan ditampung di dalam variabel 'a'. Terdapat di dalam header `stdio`.
- c. `clrscr();`
Fungsi ini digunakan untuk membersihkan layar sehingga menjadi kosong. Terdapat di dalam header `conio`.
- d. `rand();`
Fungsi ini digunakan untuk memilih sebuah angka secara acak. Terdapat di dalam header `stdlib`.

- e. `gets(...)`;
Fungsi ini digunakan untuk memasukkan input bahkan berupa kalimat yang mengandung spasi. Terdapat di dalam header `stdio`.

Masih ada fungsi-fungsi bawaan yang lain yang akan kita temui.

IDENTIFIER

Apabila kita akan melakukan suatu operasi, pasti kita membutuhkan penampung untuk nilai yang akan kita operasikan. Penampung ini dinamakan **identifier**. Identifier dapat dibedakan menjadi konstanta dan variabel. Namun, selain itu identifier juga dapat berupa sebuah fungsi yang kita buat untuk program tertentu. **Konstanta** adalah suatu penampung nilai di mana nilainya tidak diubah selama program dijalankan, meskipun dilakukan operasi apapun untuk mengubahnya. Kebalikan dari konstanta, **variabel** adalah suatu penampung nilai yang nilainya dapat diubah sewaktu-waktu saat program dijalankan, misalnya melalui suatu operasi aritmatika. Dalam penggunaannya, konstanta dan variabel membutuhkan pendeklarasian. Deklarasi ini berfungsi untuk menentukan tipe data yang akan digunakan. Dalam membuat nama konstanta maupun variabel, terdapat beberapa aturan yang harus dipenuhi, yaitu:

- a. tidak boleh sama dengan nama keyword reserved, function, dan harus unik (berbeda antara yang satu dengan yang lain)
- b. maksimum 32 karakter. Bila lebih, maka karakter selebihnya tidak akan diperhatikan oleh komputer
- c. case sensitive: huruf besar dan huruf kecil dianggap berbeda
- d. karakter pertama harus huruf atau underscore (`_`), selebihnya boleh angka
- e. tidak boleh mengandung spasi.

Bentuk umum deklarasi konstanta : `#define <nama konstanta> <nilai>`;

Bentuk umum deklarasi variabel : `<tipe data> <nama variabel>`;

Keyword adalah identifier yang ditulis dalam huruf kecil yang telah didefinisikan oleh bahasa C. Menurut standar ANSI, ada 32 keyword dalam bahasa C, yaitu:

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

TIPE DATA

Tipe data yang dapat digunakan di dalam bahasa C adalah sebagai berikut.

Type	Length	Range
unsigned char	8 bits	0 to 255
char	8 bits	-128 to 127
short int	16 bits	-32.768 to 32.767

unsigned int	32 bits	0 to 4.294.967.295
int	32 bits	-2.147.483.648 to 2.147.483.648
unsigned long	32 bits	0 to 4.294.967.295
enum	16 bits	-2.147.483.648 to 2.147.483.648
long	32 bits	-2.147.483.648 to 2.147.483.648
float	32 bits	$3,4 \times 10^{-38}$ to $3,4 \times 10^{38}$
double	64 bits	$1,7 \times 10^{-308}$ to $1,7 \times 10^{308}$
long double	80 bits	$3,4 \times 10^{-4932}$ to $3,4 \times 10^{4932}$
near (pointer)	32 bits	not applicable
far (pointer)	32 bits	not applicable

Ada beberapa hal yang perlu diperhatikan saat kita menggunakan tipe data :

- integer → bila kita memasukkan bilangan pecahan ke dalam variabel dengan tipe data integer, maka bilangan tersebut akan dibulatkan ke bawah (bilangan di belakang koma dihilangkan)
- secara umum → bila kita menginputkan data yang melebihi panjang tipe data, maka data tersebut akan berulang. Misalkan kita menginputkan data -32768 ke dalam variabel dengan tipe data integer, maka yang akan tersimpan adalah -32767.

KARAKTER

Di dalam komputer, karakter (A, B, m, n , 1, 5, *, \$, dsb.) tidak ditulis sebagai karakter seperti yang kita ketikkan. Karakter di dalam komputer dibaca sebagai kode ASCII. Misalnya huruf A yang kita ketikkan melalui keyboard sebenarnya adalah kode ASCII dengan nilai 65 (01000001). Agar kita mengetahui kode ASCII dari karakter yang kita inputkan melalui keyboard, kita dapat melakukan **casting**, yaitu pemaksaan tipe data. Untuk mengetahui nilai ASCII dari A, kita dapat mengetikkan

```
char a='A';
printf("%c = %d",a,a); //dengan a bertipe data char dan berisi huruf A
```

Kita juga dapat melakukan hal yang sebaliknya, yaitu kita menginputkan nilai ASCII dan kita dapat mengetahui karakter apa yang muncul. Contohnya adalah sebagai berikut.

```
int a=65;
printf("%d = %c",a,a); //dengan a bertipe data int dan berisi 65
```

KARAKTER ESCAPE

adalah karakter yang diawali dengan tanda \ dan memiliki makna tertentu.

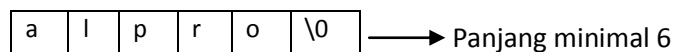
- \a Bunyi bel (speaker komputer)
- \b Mundur satu spasi (backspace)
- \f Ganti halaman (form feed)
- \n Ganti baris baru (new line)
- \r Ke kolom pertama baris yang sama (carriage return)
- \t Tabulasi horisontal
- \v Tabulasi vertikal
- \0 Nilai kosong (null)
- \' Karakter petik tunggal
- \"\" Karakter petik ganda
- \\ Garis miring terbalik (back slash)

STRING

String adalah kumpulan dari beberapa karakter (huruf, angka, simbol). Turbo C++ tidak memiliki tipe data string. String dapat dibuat dengan menggabungkan beberapa buah karakter menjadi satu. Oleh karena itu, string disebut sebagai *array of character* (kumpulan karakter). Meskipun tidak memiliki tipe data string, tetapi Turbo C++ dapat meminta input data string dan menampilkan data string dengan menggunakan %s.

```
char a[10]="alpro";
printf("%s",a);
```

Di dalam mendeklarasikan string (gabungan dari karakter), kita harus berhati-hati. Komputer akan berhenti membaca saat bertemu dengan NULL (\0). Jadi, saat mendeklarasikan string, kita harus menambah panjang array paling tidak sebanyak 1 untuk memberikan tempat kepada NULL. Sebagai ilustrasi, lihat gambar berikut.



Bila kita menginputkan string dengan kode format %s, maka yang terjadi adalah karakter setelah spasi tidak akan terbaca. Sebagai solusinya, kita dapat mengganti kode format %s menjadi %[^\n].