

# PERTEMUAN 4

## STRUKTUR DASAR BAHASA C

### 2.1 TUJUAN PRAKTIKUM

#### Tujuan Umum

Mahasiswa dapat memahami:

1. Struktur penulisan bahasa pemrograman
2. Sintaks assignment statement dan output statement,
3. Keperluan sebuah variable,
4. Tipe data standar bahasa pemrograman.
5. Mengenal berbagai operator dalam Bahasa C

#### Tujuan Khusus

Mahasiswa dapat :

1. Menuliskan sintaks instruksi : assignment statement, dan output Statement
2. Mendeklarasikan dan menggunakan variabel dalam berbagai tipe data dalam sebuah program
3. Memilih tipe data sesuai dengan kegunaan data tersebut.
4. Menulis program untuk menampilkan isi dari suatu variabel
5. Menulis program untuk menampilkan string yang mengandung karakter khusus
6. Membuat program sederhana yang melibatkan berbagai operator
7. Memberi komentar program

### 2.2 TEORI SINGKAT

#### 2.2.1 Tipe Data

Tipe data merupakan bagian program yang paling penting karena tipe data mempengaruhi setiap instruksi yang akan dilaksanakan oleh komputer. Misalnya saja 5 dibagi 2 bisa saja menghasilkan hasil yang berbeda tergantung tipe datanya. Jika 5 dan 2 bertipe integer maka akan menghasilkan nilai 2, namun jika keduanya bertipe float maka akan menghasilkan nilai 2.5000000. Pemilihan tipe data yang tepat akan membuat proses operasi data menjadi lebih efisien dan efektif.

Dalam bahasa C terdapat lima tipe data dasar, yaitu :

No	Tipe Data	Ukuran	Range (Jangkauan)	Format	Keterangan
1	char	1 byte	-128 s/d 127	%c	Karakter/string
2	int	2 byte	-32768 s/d 32767	%i , %d	Integer/bilangan bulat

No	Tipe Data	Ukuran	Range (Jangkauan)	Format	Keterangan
3	float	4 byte	-3.4E-38 s/d 3.4E+38	%f	Float/bilangan pecahan
4	double	8 byte	-1.7E-308 s/d 1.7+308	%lf	Pecahan presisi ganda
5	void	0 byte	-	-	Tidak bertipe

### 2.2.2 Variabel

Variabel adalah suatu pengenal (identifier) yang digunakan untuk mewakili suatu nilai tertentu di dalam proses program. Berbeda dengan konstanta yang nilainya selalu tetap, nilai dari suatu variable bisa diubah-ubah sesuai kebutuhan. Nama dari suatu variabel dapat ditentukan sendiri oleh pemrogram dengan aturan sebagai berikut :

1. Terdiri dari gabungan huruf dan angka dengan karakter pertama harus berupa huruf. Bahasa C bersifat case-sensitive artinya huruf besar dan kecil dianggap berbeda. Jadi antara **nim**, **NIM** dan **Nim** dianggap berbeda.
2. Tidak boleh mengandung spasi.
3. Tidak boleh mengandung simbol-simbol khusus, kecuali garis bawah (underscore). Yang termasuk symbol khusus yang tidak diperbolehkan antara lain : \$, ?, %, #, !, &, \*, (, ), -, +, = dsb
4. Panjangnya bebas, tetapi hanya 32 karakter pertama yang terpakai.

Contoh penamaan variabel yang benar :

NIM, a, x, nama\_mhs, f3098, f4, nilai, budi.

Contoh penamaan variable yang salah :

%nilai\_mahasiswa, 80mahasiswa, rata-rata, ada spasi, penting!

### 2.2.3 Karakter Khusus (Special Character)

Pada Bahasa C, pada umumnya karakter atau string dapat ditampilkan dengan menuliskan karakter / string tersebut secara langsung. Namun demikian, terdapat beberapa karakter khusus yang penulisannya sedikit berbeda. Berikut ini karakter khusus yang dikenal di bahasa C beserta penjelasannya.

Karakter Khusus	Penjelasan
\a	Untuk bunyi bell (alert)
\b	Mundur satu spasi (backspace)
\f	Ganti halaman (form feed)
\n	Ganti baris baru (new line)
\r	Menuju ke kolom pertama, baris yang sama (carriage return)
\v	Tabulasi vertikal

Karakter Khusus	Penjelasan
\t	Tabulasi horizontal
\0	Nilai kosong (null)
'	Kutip tunggal
"	Kutip ganda
\\"	Karakter garis miring (backslash)

#### 2.2.4 Deklarasi

Deklarasi diperlukan bila kita akan menggunakan pengenal (identifier) dalam program. Identifier dapat berupa variable, konstanta dan fungsi.

##### Deklarasi Variabel

Bentuk umum pendeklarasian suatu variable adalah :

```
Nama_tipe nama_variabel;
```

Contoh :

```
int x;
char y, huruf, nim[10];
float nilai;
double beta;
int array[5][4];
char *p;
```

##### Deklarasi Konstanta

Dalam bahasa C konstanta dideklarasikan menggunakan preprocessor #define. Contohnya :

```
#define PHI 3.14
#define nim "0111500382"
#define nama "Achmad Solichin"
```

##### Deklarasi Fungsi

Fungsi merupakan bagian yang terpisah dari program dan dapat diaktifkan atau dipanggil di manapun di dalam program. Fungsi dalam bahasa C ada yang sudah disediakan sebagai fungsi pustaka seperti printf(), scanf(), getch() dan untuk menggunakannya tidak perlu dideklarasikan. Fungsi yang perlu dideklarasikan terlebih dahulu adalah fungsi yang dibuat oleh programmer. Bentuk umum deklarasi sebuah fungsi adalah : Tipe\_fungsi nama\_fungsi(parameter\_fungsi); Contohnya : float luas\_lingkaran(int jari); void tampil(); int tambah(int x, int y);

## 2.2.5 Operator

### 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.

### Operator Aritmatika

Bahasa C menyediakan lima operator aritmatika, yaitu :

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

Catatan : operator % digunakan untuk mencari sisa pembagian antara dua bilangan. Misalnya :

$$\begin{aligned}9 \% 2 &= 1 \\9 \% 3 &= 0 \\9 \% 5 &= 4 \\9 \% 6 &= 3\end{aligned}$$

### Operator Hubungan (Perbandingan)

Operator Hubungan digunakan untuk membandingkan hubungan antara dua buah operand (sebuah nilai atau variable). Operator hubungan dalam bahasa C :

Operator	Arti	Contoh	
<	Kurang dari	$x < y$	Apakah x kurang dari y
$\leq$	Kurang dari sama dengan	$x \leq y$	Apakah x kurang dari sama dengan y
>	Lebih dari	$x > y$	Apakah x lebih dari y
$\geq$	Lebih dari sama dengan	$x \geq y$	Apakah x lebih dari sama dengan y
$=$	Sama dengan	$x == y$	Apakah x sama dengan y
$\neq$	Tidak sama dengan	$x != y$	Apakah x tidak sama dengan y

## Operator Logika

Jika operator hubungan membandingkan hubungan antara dua buah operand, maka operator logika digunakan untuk membandingkan logika hasil dari operator-operator hubungan. Operator logika ada tiga macam, yaitu :

- && : Logika AND (DAN)
- || : Logika OR (ATAU)
- ! : Logika NOT (INGKARAN)

## **2.2.6 Kata Tercadang (Reserved Word)**

Bahasa C standar ANSI memiliki 32 kata tercadang (reserved word) dan Turbo C menambahkannya dengan 7 kata tercadang. Semua reserved word tidak boleh digunakan dalam penamaan identifier (variable, nama fungsi dll). Kata tercadang yang tersedia dalam bahasa C adalah sbb (tanda \* menunjukkan kata tercadang pada Turbo C):

```
*asm, default, for, *pascal, switch, auto, do, goto, register,  
typedef, break, double, *huge, return, union, case, else, if,  
short, unsigned, *cdecl, enum, int, signed, void, char, extern,  
*interrupt, sizeof volatile const *far long static while  
continue float *near struct
```

## **2.2.7 Komentar Program**

Komentar program hanya diperlukan untuk memudahkan pembacaan dan pemahaman suatu program (untuk keperluan dokumentasi program). Dengan kata lain, komentar program hanya merupakan keterangan atau penjelasan program. Untuk memberikan komentar atau penjelasan dalam bahasa C digunakan pembatas /\* dan \*/ atau menggunakan tanda // untuk komentar yang hanya terdiri dari satu baris. Komentar program tidak akan ikut diproses dalam kompilasi program (akan diabaikan).

## PELAKSANAAN PRAKTIKUM

1. Tuliskan Program 2.1 berikut ini pada editor Dev-C++.

**Program 2.1 variabel1.cpp**

```
1 #include <stdio.h>
2 int main()
3 {
4     char nim[10];
5     char nama[30];
6     int nilai;
7
8     printf("NIM : %s", nim);
9     printf("NAMA : %s", nama);
10    printf("NILAI : %i", nilai);
11
12    return 0;
13 }
```

2. Jalankan program 2.1 di atas dan tuliskan apa yang tercetak di layar.

3. Pada Program 2.1, variabel yang dideklarasikan belum diisi dengan nilai tertentu (belum diinisialisasi). Lakukan perubahan Program 2.1 dengan menambahkan perintah untuk mengisi variabel nim, nama dan nilai seperti pada Program 2.2 berikut ini.

**Program 2.2 variabel2.cpp**

```
1 #include <stdio.h>
2 int main()
3 {
4     char nim[10];
5     char nama[30];
6     int nilai;
7
8     nim = "141150123";
9     nama = "Achmad Solichin";
10    nilai = 85;
11
12    printf("NIM : %s", nim);
13    printf("NAMA : %s", nama);
14    printf("NILAI : %i", nilai);
15
16    return 0;
17 }
```

4. Lakukan kompilasi Program 2.2 di atas dan perhatikan hasilnya. Apakah terjadi error? Apa error yang akan ditampilkan? Tuliskan error yang ditampilkan dan baris berapa terjadi error!

Error yang ditampilkan:

Error terjadi pada baris ke:

5. Kesalahan pada Program 2.2 terjadi karena cara pengisian variabel nim dan nama yang kurang tepat. Variabel nim dan nama dideklarasikan sebagai sebuah variabel bertipe char dan berupa array (ditunjukkan dengan adanya tanda kurung siku []). Pembahasan mengenai array akan dilakukan secara khusus pada Pertemuan ke-9.

6. Sekarang kita perbaiki kesalahan pada Program 2.2 dengan mengisi nim dan nama menggunakan fungsi `strcpy()`. Penggunaan fungsi `strcpy()` harus menyertakan header fungsi `<string.h>`. Ubahlah Program 2.2 menjadi Program 2.3 berikut ini.

**Program 2.3 variabel3.cpp**

```
1 #include <stdio.h>
2 #include <string.h>
3 int main()
4 {
5     char nim[10];
6     char nama[30];
7     int nilai;
8
9     strcpy(nim, "141150123");
10    strcpy(nama, "Achmad Solichin");
11    nilai = 85;
12
13    printf("NIM : %s", nim);
14    printf("NAMA : %s", nama);
15    printf("NILAI : %i", nilai);
16
17    return 0;
18 }
```

7. Jalankan Program 2.3 dan tuliskan apa yang tercetak di layar!

8. Tampilan Program 2.3 sedikit berantakan bukan? Mari kita buat lebih rapi dengan menambahkan karakter khusus `\n` (pindah baris) dan `\t` (tabulasi horizontal). Perhatikan Program 2.4 berikut ini dan jalankan!

**Program 2.4 variabel4.cpp**

```
1 #include <stdio.h>
2 #include <string.h>
3 int main()
4 {
5     char nim[10];
6     char nama[30];
7     int nilai;
8
9     strcpy(nim, "141150123");
10    strcpy(nama, "Achmad Solichin");
11    nilai = 85;
12
13    printf("NIM \t: %s", nim);
```

```

14     printf("\nNAMA \t: %s", nama);
15     printf("\nNILAI \t: %i", nilai);
16
17     return 0;
18 }
```

9. Jalankan Program 2.4 dan tuliskan apa yang tercetak di layar!

10. Selanjutnya kita akan ubah Program 2.4 menjadi Program 2.5 dengan menambahkan nilai uts, uas, tugas dan kehadiran serta menghitung nilai akhir yang diperoleh mahasiswa. Nilai akhir diperoleh dengan menggunakan rumus sebagai berikut:

$$\text{NILAI AKHIR} = 10\% \text{ kehadiran} + 20\% \text{ tugas} + 30\% \text{ uts} + 40\% \text{ uas}$$

Kita akan mendeklarasikan variabel kehadiran, tugas, uts dan uas bertipe integer dan variabel nilai\_akhir bertipe float (pecahan).

#### **Program 2.5 variabel5.cpp**

```

1 #include <stdio.h>
2 #include <string.h>
3 int main()
4 {
5     char nim[10];
6     char nama[30];
7     int kehadiran, tugas, uts, uas;
8     float nilai_akhir;
9
10    strcpy(nim, "141150123");
11    strcpy(nama, "Achmad Solichin");
12    kehadiran = 100;
13    tugas = 90;
14    uts = 83;
15    uas = 86;
16
17    nilai_akhir = (0.1 * kehadiran) + (0.2 * tugas) + (0.3
* uts) + (0.4 * uas);
18
19    printf("NIM \t: %s", nim);
20    printf("\nNAMA \t: %s", nama);
21    printf("\nKEHADIRAN \t: %i", kehadiran);
22    printf("\nTUGAS \t: %i", tugas);
23    printf("\nUTS \t: %i", uts);
```

```
24     printf("\nUAS \t: %i", uas);
25     printf("\nNILAI AKHIR \t: %.2f", nilai_akhir);
26
27     return 0;
28 }
```

11. Jalankan Program 2.5 dan tuliskan apa yang tercetak!

12. Untuk lebih mempermudah pembacaan program (dan untuk keperluan belajar), kita dapat menambahkan komentar-komentar program. Ubah Program 2.5 menjadi Program 2.6 berikut ini.

**Program 2.6 variabel6.cpp**

```
1 #include <stdio.h>
2 #include <string.h>
3 int main()
4 {
5     char nim[10];
6     char nama[30];
7     int kehadiran, tugas, uts, uas;
8     float nilai_akhir;
9
10    strcpy(nim, "141150123");
11    strcpy(nama, "Achmad Solichin");
12    kehadiran = 100;      //nilai kehadiran
13    tugas     = 90;       //nilai tugas
14    uts       = 83;       //nilai uts
15    uas      = 86;       //nilai uas
16
17    /* perhitungan nilai akhir
18       sesuai peraturan di UBL
19 */
20    nilai_akhir = (0.1 * kehadiran) + (0.2 * tugas) + (0.3
* uts) + (0.4 * uas);
21
22    //tampilkan data
23    printf("NIM \t: %s", nim);
24    printf("\nNAMA \t: %s", nama);
25    printf("\nKEHADIRAN \t: %i", kehadiran);
```

```
26     printf("\nTUGAS \t: %i", tugas);
27     printf("\nUTS \t: %i", uts);
28     printf("\nUAS \t: %i", uas);
29     printf("\nNILAI AKHIR \t: %.2f", nilai_akhir);
30
31     return 0;
32 }
```

13. Jalankan Program 2.6 dan lihat hasilnya. Bandingkan dengan hasil Program 2.5. Sama bukan? Hal tersebut menunjukkan bahwa komentar program tidak mempengaruhi hasil program. Komentar program diperlukan oleh programmer sendiri agar dalam mempermudah pembacaan program.

## 2.4 LATIHAN

Tuliskan dan jalankan beberapa program berikut ini dan tuliskan hasilnya di tempat yang sudah disediakan.

Program 2.7 unary.cpp

```
1 #include <stdio.h>
2 int main()
3 {
4     int A, B;
5     A = 5;
6     printf("A = %i", A);
7     printf("\nA = %i", A++);
8     printf("\nA = %i", A);
9
10    B = 10;
11    printf("\n\nB = %i", B);
12    printf("\nB = %i", ++B);
13    printf("\nB = %i", B);
14
15    return 0;
16 }
```

Hasil Program 2.7

**Program 2.8 lingkaran.cpp**

```
1 #include <stdio.h>
2 int main()
3 {
4     int jari;
5     float luas, keliling;
6     jari = '/';
7
8     luas = 3.14 * jari * jari;
9     keliling = 2 * 3.14 * jari;
10
11    printf("LUAS dan KELILING LINGKARAN");
12    printf("\nJari-jari = %i", jari);
13    printf("\nLUAS = %.3f", luas);
14    printf("\nKELILING = %.2f", keliling);
15
16    return 0;
17 }
```

**Hasil Program 2.8**