PRAKTIKUM 3
DASAR PEMROGRAMAN C

A. TUJUAN

1. Menjelaskan tentang beberapa tipe data dasar (jenis dan jangkauannya)
2. Menjelaskan tentang Variabel
3. Menjelaskan tentang konstanta
4. Menjelaskan tentang berbagai jenis operator dan pemakaiannya

B. DASAR TEORI

B.1. Tipe Data Dasar

Data merupakan 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,

Data berdasarkan jenisnya dapat dibagi menjadi lima kelompok, yang dinamakan sebagai tipe data dasar. Kelima tipe data dasar adalah:

- Bilangan bulat (integer)
- Bilangan real presisi-tunggal
- Bilangan real presisi-ganda
- Karakter
- Tak-bertipe (*void*), keterangan lebih lanjut tentang void dijelaskan dalam Bab Selanjutnya.

Kata-kunci yang berkaitan dengan tipe data dasar secara berurutan di antaranya adalah *int* (*short int, long int, signed int* dan *unsigned int*), *float, double*, dan *char*.
Tabel B-1 memberikan informasi mengenai ukuran memori yang diperlukan dan kawasan dari masing-masing tipe data dasar.

Tabel B-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 kawasan dari masint-masing tipe data adalah sebagai berikut :

Tabel B-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

Catatan :

- 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).

B.2 Variabel

B.2.1 Aturan Pendefinisan Variabel

Aturan penulisan pengenal untuk sebuah variabel, konstanta atau fungsi yang didefinisikan oleh pemrogram adalah sebagai berikut :

- Pengenal 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.

B.2.2 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 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 var_bulat1;
float var_pecahan1, var_pecahan2;
```

B.2.3 Memberikan Nilai ke Variabel

Untuk memberikan nilai ke variabel yang telah dideklarasikan, maka bentuk umum pernyataan yang digunakan adalah :

```
nama_variabel = nilai;
```

Contoh:

```
int var_bulat = 10;
double var_pecahan = 10.5;
```

B.2.4 Inisialisasi Variabel

Adakalanya dalam penulisan program, setelah dideklarasikan, variabel langsung diberi nilai awal. Sebagai contoh yaitu variabel **nilai** :

```
int nilai;
nilai = 10;
```

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

```
int nilai= 10;
```

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).

B.3 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.

- Konstanta karakter misalnya ditulis dengan diawali dan diakhiri dengan tanda petik tunggal, contohnya : 'A' dan '@'.
- Konstanta integer ditulis dengan tanda mengandung pemisah ribuan dan tak mengandung bagian pecahan, contohnya : –1 dan 32767.
- 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 10^5).
- Konstanta string merupakan deretan karakter yang diawali dan diakhiri dengan tanda petik-ganda ("), contohnya :"Pemrograman Dasar C".

B.4 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. Sebagian operator C tergolong sebagai operator binary, yaitu operator yang dikenakan terhadap dua buah nilai *(operand)*. Contoh :

```
a + b
```

Simbol + merupakan operator untuk melakukan operasi penjumlahan dari kedua *operand*-nya (yaitu a dan b). Karena operator penjumlahan melibatkan dua operator ini tergolong sebagai operator binary.

```
-c
```

Simbol − (minus) juga merupakan operator. Simbol ini termasuk sebagai operator unary, yaitu operator yang hanya memiliki sebuah operand (yaitu `c` pada contoh ini).

B.4.1. Operator Aritmatika

Operator untuk operasi aritmatika yang tergolong sebagai operator binary adalah :

*	perkalian
/	pembagian
%	sisa pembagian
+	penjumlahan
-	pengurangan

Adapun operator yang tergolong sebagai operator unary.

-	tanda minus
+	tanda plus

Contoh pemakaian operator aritmatika misalnya untuk memperoleh nilai diskriminan dari suatu persamaan kuadrat : **$D = b^2 - 4ac$**

```
/* File program : diskrim.c
 Menghitung diskriminan pers kuadrat  ax^2 + bx + c = 0 */

# include <stdio.h>

main()
{
    float a,b,c,d;
    a = 3.0f;
    b = 4.0f;
    c = 7.0f;

    d = b*b-4*a*c;
    printf("Diskriminan =%f\n",d);
}
```

Contoh eksekusi :

```
Diskriminan = -84.000000
```

Operator yang telah dituliskan di atas, yang perlu diberi penjelasan lebih lanjut adalah operator sisa pembagian. Beberapa contoh berikut kiranya akan memperjelas makna dari operator ini .

- 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".

B.4.2. Operator Penurunan dan 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, contohnya :

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

Bisa ditulis menjadi :

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

atau :

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

bergantung pada kondisi yang dibutuhkan oleh pemrogram. Di bawah ini adalah contoh yang akan menunjukkan perbedaan pemakaian dan hasil dari `++x` dengan `x++` (atau pemakaian `y--` dengan `--y`).

```
/* File program : pre_post.c
Contoh 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);
}
```

Contoh eksekusi :

```
loop = 1, count = 1
loop = 1, count = 2
```

B.4.3. Prioritas Operator Aritmatika

Tabel di bawah ini memberikan penjelasan mengenai prioritas dari masing- masing operator. Operator yang mempunyai prioritas tinggi akan diutamakan dalam hal pengerjaan dibandingkan dengan operator yang memiliki prioritas lebih rendah.

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 –**

B.4.4. Operator Penugasan

Operator penugasan *(assignment operator)* digunakan untuk memindahkan nilai dari suatu ungkapan *(expression*) ke suatu pengenal. 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.`

Bahasa C juga memungkinkan dibentuknya statemen penugasan menggunakan operator pengerjaan jamak dengan bentuk sebagai berikut :

pengenal1 = pengenal2 = … = ungkapan ;

Misalnya :

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

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

B.4.5 Operator Kombinasi (Pemendekan)

C menyediakan operator yang dimaksudkan untuk memendekkan penulisan operasi penugasan semacam

```
x = x + 2;
y = y * 4;
```

menjadi

```
x +=  2;
y *=  4;
```

Daftar berikut memberikan seluruh kemungkinan operator kombinasi dalam suatu pernyataan serta pernyataan padanannya.

Tabel B.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;`

C. TUGAS PENDAHULUAN

1. Jelaskan fungsi dibawah ini:
    - main()
    - printf()
    - scanf()
    - return 0;
2. Tentukan hasil beberapa ekspresi berikut dan jika ekspresi illegal berikan alasannya:
    - 2*6-7*23/5
    - 2*(6-7)*23/5
    - (5*4)/(3*5)
    - 45>(3+5)

D. PERCOBAAN

1. Buatlah program untuk menghitung luas segitiga, dimana terdapat 3 variabel yaitu luas, alas dan tinggi. Tipe data dari masing-masing variabel adalah float. Rumus luas segitiga yang digunakan adalah luas=0.5 * alas * tinggi. Adapun Tampilan hasilnya adalah :

```
Buatlah proram untuk menghitung luas segitiga
Masukkan alas: 10
Masukkan tinggi: 10
Jadi luas segitiga adalah 50
```

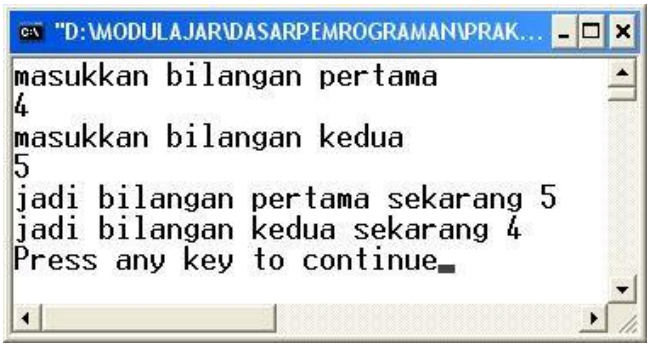
2. Cobalah program dibawah ini :

```
#include <stdio.h>
main()
 {
    char kar='a';
    kar=kar-32;
    printf("jadi hurufnya sekarang menjadi %c\n",kar);
 }
```

Dan tampilkan hasilnya!

3. Menukarkan 2 buah nilai A dan B!

Input: a,b;



4. Buatlah program yang membaca tiga buah bilangan yang masing-masing menyatakan panjang(cm), lebar(cm) dan tinggi(cm) sebuah kotak, menghitung volume kotak tersebut dan menampilkan hasilnya.

```
Masukkan panjang, lebar dan tinggi
Panjang(cm):5
Lebar(cm):3
Tinggi(cm):2
Jadi volumenya adalah 30 cm kubik
```

E. DELIVERABLES

- **Buatlah Penamaan Tugas C dengan TugasC_NIM_NAMA**
- **Buatlah Penamaan Tugas D dengan TugasD_NIM_NAMA**
- **Gabunglah tugas Anda pada Folder W02S03_TUGAS_NIM_NAMA.zip/rar**