



# Dasar Pemrograman C

---



# Objectives

---

- Menjelaskan tentang beberapa tipe data dasar
- Menjelaskan tentang Variabel
- Menjelaskan tentang konstanta
- Menjelaskan tentang berbagai jenis operator dan pemakaiannya
- Menjelaskan tentang instruksi I/O



# Tipe Data Dasar

---

- Data bisa dinyatakan dalam bentuk konstanta atau variabel.
  - Konstanta → nilainya tetap.
  - Variabel → nilainya dapat diubah-ubah selama eksekusi.
- Berdasarkan jenisnya, data dapat dibagi menjadi lima kelompok → dinamakan tipe data dasar, yaitu:
  - Bilangan bulat (integer)
  - Bilangan real presisi-tunggal (float)
  - Bilangan real presisi-ganda (double)
  - Karakter (char)
  - Tak-bertipe (*void*)



# Ukuran Memori untuk tipe data

Type_data	Jumlah bit	Range nilai	Keterangan
char	8	-128 s/d 127	Karakter
int (signed int)	16	-32768 s/d 32767	Bilangan bulat (integer)
short int	16	-32768 s/d 32767	Bilangan bulat.
Unsigned int	16	0 s/d 65535	Bilangan bulat tak bertanda
long int	32	-2147483648 s/d 2147483647	Bilangan bulat
float	32	1.7E-38 s/d 3.4E+38	Bilangan real (single)
double	64	2.2E-308 s/d 1.7E+308	Bilangan real (double)
void	0	-	Tak bertipe



# Variabel

---

- Aturan penulisan:
  - Nama harus diawali dengan huruf (A..Z, a..z), angka atau karakter garis bawah ( \_ ).
  - Selanjutnya dapat berupa huruf, digit (0..9) atau karakter garis bawah atau tanda dollar (\$).
  - Panjang nama variabel boleh lebih dari 31 karakter → hanya 31 karakter pertama yang akan dianggap.
  - nama variabel tidak boleh menggunakan nama yang tergolong sebagai kata-kata cadangan (*reserved words*) seperti *printf*, *int*, *if*, *while* dan sebagainya



# Deklarasi Variabel

---

- 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_data daftar_nama_variabel;`
- Contoh:  
`int var_bulat1;`  
`float var_pecahan1, var_pecahan2;`



## Beri Nilai Variabel

---

- Memberikan nilai ke variabel:

```
nama_variabel = nilai;
```

- Contoh:

```
var_bulat1 = 34;
```

```
var_pecahan1 = 34.52;
```



# Inisialisasi Variabel

---

- Inisialisasi nilai variabel

```
int nilai;
```

```
nilai = 10;
```

- Sama dengan:

```
int nilai = 10;
```





# Contoh Program

---

```
#include <stdio.h>
main()
{
    int jumlah;
    float harga_unit, harga_total;
    jumlah=10;
    harga_unit=17.5;
    harga_total=jumlah*harga_unit;
    printf("Harga total = %f\n",harga_total);
}
```

# Konstanta

- Konstanta menyatakan nilai tetap.
- Tidak perlu dideklarasikan.
- Juga mempunyai tipe data.
- Aturan penulisan:
  - Konstanta karakter → diawali dan diakhiri dengan tanda petik tunggal, Contoh : 'A' dan '@'.
  - Konstanta integer → ditulis dengan angka (tanpa tanda petik) tanpa mengandung pemisah ribuan dan tak mengandung bagian pecahan. Contoh : -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<sup>5</sup>).
  - Konstanta string merupakan deretan karakter yang diawali dan diakhiri dengan tanda petik-ganda ("). Contoh: "Program Dasar".



## Konstanta - *continued*

---

- Aturan penulisan konstanta:
  - Menggunakan keyword `#define`  
`#define <nama_konstanta> <nilai>`atau
  - Menggunakan keyword `const`  
`const <tipe_konstanta> <nama_konstanta>  
= <nilai>;`
- Contoh:  
`#define PI 3.14159`  
atau  
`const float PI = 3.14159;`

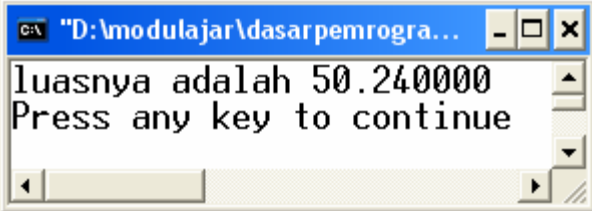
# Contoh Program

```
#include <stdio.h>
#define PI 3.14
```

```
main()
{
    float r=4;
    float luas;
    luas=PI*r*r;

    printf("luasnya adalah %f", luas);

}
```



A screenshot of a Windows command prompt window. The title bar shows the path "D:\modulajar\dasar pemrogra...". The window content displays the output of the program: "luasnya adalah 50.240000" followed by "Press any key to continue".



# Operator

---

- Simbol atau karakter → digunakan untuk melakukan sesuatu operasi atau manipulasi.
- Misal: menjumlahkan, mengurangi, membandingkan, memberikan nilai, dll.
- Jenis operator:
  - Operator Aritmatika
  - Operator Increment dan Decrement
  - Operator penugasan
  - Operator kombinasi



# Operator Aritmatika

---

- Terdiri dari dua jenis:
  - Operator binary

Operator	Fungsi
*	Perkalian
/	Pembagian
%	Sisa Pembagian
+	Penjumlahan
-	Pengurangan

- Operator unary
  - Tanda '-' (minus)
  - Tanda '+' (plus)



## Contoh program menggunakan operator aritmatika

---

```
# include <stdio.h>
main()
{
    int a,b,c;
    float d;
    a = 3 * 5;
    b = 10 % 3;
    c = 10 / 3;
    d = 10.0 / 3.0;
    printf("Nilai dari a = %d\n", a);
    printf("Nilai dari b = %d\n", b);
    printf("Nilai dari c = %d\n", c);
    printf("Nilai dari d = %f\n", d);
}
```



# Operator Increment dan Decrement

---

- Operator increment:  
'++'
- Operator  
decrement: '--'

operasi	arti
$x++/++x$	$x=x+1$
$y--/--y$	$y=y-1$



# Contoh program menggunakan operator increment

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

```
main()
```

```
{
```

```
    int count, loop;
```

loop = 1, count = 1

loop = 1, count = 2

```
    count = 0;
```

```
    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

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)

---

- Digunakan untuk memindahkan nilai dari suatu ungkapan (*expression*) ke suatu pengenalan.
- Operator pengerjaan yang umum digunakan dalam bahasa pemrograman, termasuk bahasa C adalah operator sama dengan (=).

**pengenal1 = pengenal2 = ... = ungkapan ;**

Contoh : `a=(b=1)+5;`



# Operator Kombinasi

---

- Digunakan untuk memendekkan penulisan operasi penugasan.

- Contoh:

```
x = x + 2 ;
```

```
y = y * 4 ;
```

- Dapat dipendekkan menjadi:

```
x += 2 ;
```

```
y *= 4 ;
```

# Operator Kombinasi

Operator Kombinasi	Arti padanannya
<code>x += 2;</code>	<code>x = x + 2;</code>
<code>x -= 2;</code>	<code>x = x - 2;</code>
<code>x *= 2;</code>	<code>x = x * 2;</code>
<code>x /= 2;</code>	<code>x = x / 2;</code>
<code>x %= 2;</code>	<code>x = x % 2;</code>
<code>x &lt;&lt;= 2;</code>	<code>x = x &lt;&lt; 2;</code>
<code>x &gt;&gt;= 2;</code>	<code>x = x &gt;&gt; 2;</code>
<code>x &amp;= 2;</code>	<code>x = x &amp; 2;</code>
<code>x  = 2;</code>	<code>x = x   2;</code>
<code>x ^= 2;</code>	<code>x = x ^ 2;</code>



# Fungsi printf()

---

- digunakan untuk menampilkan data ke layar.
- Bentuk umum pernyataan *printf()*:  
`printf("string kontrol", argumen1, argumen2, ...);`

Format untuk data string dan karakter :

%c     untuk menampilkan sebuah karakter

%s     untuk menampilkan sebuah string



## Fungsi printf()

---

- Fungsi puts() : menampilkan string

```
puts ("Selamat mencoba");
```

sama dengan

```
printf ("Selamat mencoba\n");
```

- Fungsi putchar() : menampilkan karakter

```
putchar ('F');
```

sama dengan

```
printf ("%c", 'F');
```



# Fungsi scanf()

---

- Digunakan untuk menerima input data dari keyboard.
- Bentuk *scanf()* → menyerupai fungsi *printf()*.
- Fungsi ini melibatkan penentu format yang pada dasarnya sama digunakan pada *printf()*.
- Bentuk umum fungsi *scanf()* adalah:  
`scanf("string kontrol",daftar_argumen);`



# Fungsi scanf()

- **daftar\_argumen** dapat berupa satu atau beberapa argumen dan haruslah berupa **alamat**.
- Misalnya hendak membaca bilangan real dan ditempatkan ke variabel `radius`, maka yang ditulis dalam `scanf()` adalah alamat dari **radius**.
- Untuk menyatakan alamat dari variabel, di depan variabel dapat ditambahkan tanda `&` (tanda `&` dinamakan sebagai operator alamat)
- Contoh :

```
scanf ("%f",&radius);  
scanf ("%d %d",&data1, &data2);
```



# Penentu format scanf()

Format	Arti
%c	membaca sebuah <b>karakter</b>
%s	membaca sebuah <b>string</b> (dibahas pada bab vii)
%i atau %d	membaca sebuah <b>integer desimal</b>
%e atau %f	membaca sebuah <b>bilangan real</b> (bisa dalam bentuk eksponensial)
%o	membaca sebuah <b>integer oktal</b>
%x	membaca sebuah <b>integer heksadesimal</b>
%u	membaca sebuah <b>integer tak bertanda</b>
l	awalan untuk membaca data long int (misal : %ld) atau untuk membaca data <b>double</b> (misal : %lf)
L	awalan untuk membaca data <b>long double</b> (misal : %Lf)
h	awalan untuk membaca data <b>short int</b>



# Fungsi scanf()

---

- Fungsi getch() : membaca karakter dan tidak ditampilkan.
- Fungsi getchar() : membaca karakter dan ditampilkan.

Contoh : `kar = getchar();`  
`scanf ("%c",&kar);`

## Contoh program menggunakan fungsi scanf

```
/* File program : bujursangkar.c
Menghitung luas dan keliling bujursangkar */

#include <stdio.h>
main()
{
    int luas, keliling, panjang_sisi;
    printf("Masukkan panjang sisi bujursangkar : ");
    scanf("%d", &panjang_sisi);
    luas = panjang_sisi * panjang_sisi;
    keliling = panjang_sisi * 4;
    printf("\nData bujursangkar\n");
    printf("Panjang sisi = %6d\n", panjang_sisi);
    printf("Luas          = %6d\n", luas);
    printf("Keliling       = %6d\n", keliling);
}
```



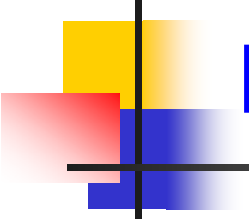
# Exercise

1. Mengapa nama-nama variabel di bawah ini tidak valid ?
  - a. value\$sum
  - b. exit flag
  - c. 3lotsofmoney
  - d. char

2. Berapakah hasil akhir dari program berikut :

```
#include <stdio.h>
main( )
{
int a = 22;

a = a + 5;
a = a-2;
printf("a = %d\n", a);
}
```



## Exercise - *continued*

---

3. Berapakah nilai  $x$  setelah pernyataan-pernyataan berikut dijalankan, apabila  $x$  bertipe *int* :
  - a.  $x = (2 + 3) - 10 * 2;$
  - b.  $x = (2 + 3) - (10 * 2);$
  - c.  $x = 10 \% 3 * 2 + 1;$
  
4. Nyatakan dalam bentuk pernyataan :
  - a.  $y = bx^2 + 0,5x - c$
  - b.  $Y = 0,3xy / 2a$



## Exercise - *continued*

---

5. Apa hasil eksekusi dari program berikut :

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