

# Topik Hari Ini

---

## Operasi Input dan Output:

- Jenis data *string*
- Fungsi *scanf* dan *printf*
- Fungsi *getchar* dan *putchar*
- Fungsi *gets* dan *puts*

## String (rentetan)

*String* is a sequence of zero or more characters enclosed in double quotes (" ").

*Example:*

```
"" /* A null string */  
"h"  
"Hello World\n"  
"HOW ARE YOU"  
"Good Morning!"
```

Use single quotes (') for character data.

Use double quotes (") for string data

*Example:*

'Hello World' *(This is wrong, because single quotes only for single character)*

*How about this?*

"H" *(is it true? If true, is it a character data or string data)*

Now, distinguish these. Are they the same?

1 Number 1

'1' Character 1

"1" String 1

How about these?

10

'10' Invalid, because a character  
can only hold single character

"10"

A string variable is declared using the following syntax:

```
char string_var[size];
```

Or

```
char string_var[size]="Initialized Value";
```

Or

```
char string_var[]="Initialized Value";
```

Examples:

```
char stra[50];
```

```
char strb[50]="Hello World"; // declare and initialize
```

```
char strc[]="Hello World"; // the size is optional
```

## Example program using string:

```
#include <stdio.h>

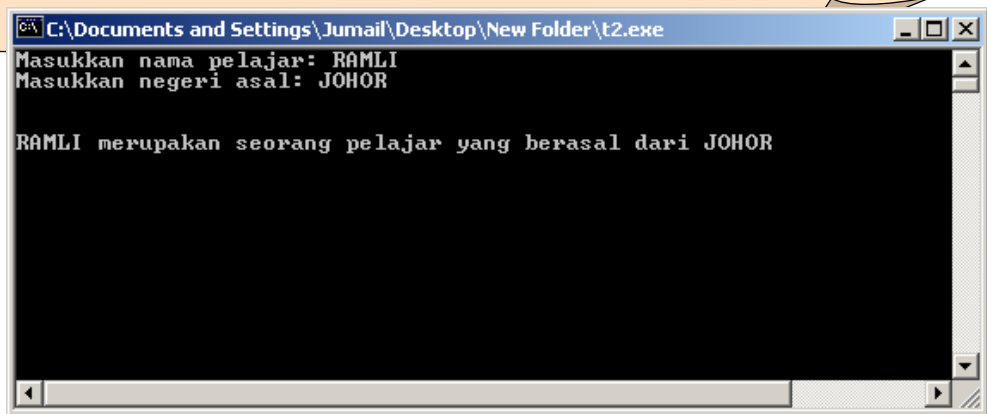
int main()
{
    char nama[50];
    char negeri[15];

    printf("Masukkan nama pelajar: ");
    scanf("%s",&nama);

    printf("Masukkan negeri asal: ");
    scanf("%s",&negeri);

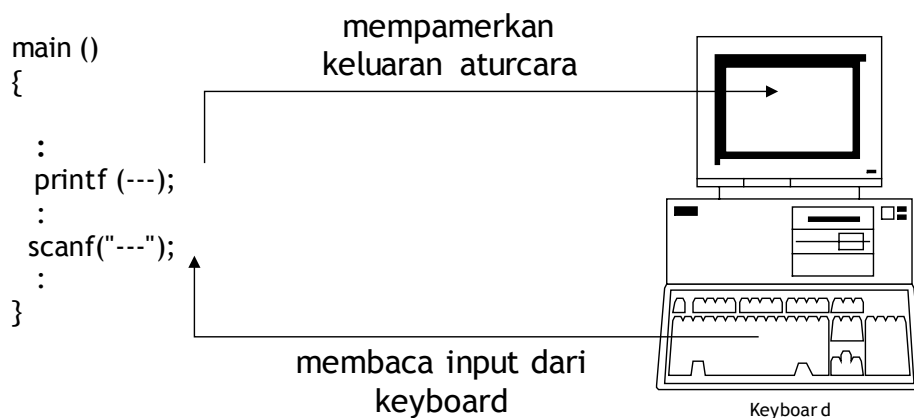
    printf("\n\n%s merupakan seorang pelajar yang berasal dari %s",nama, negeri);
    return 0;
}
```

### Output



## Operasi Input dan Output

- Kita telah menggunakan fungsi `printf (...)` untuk mencetak mesej ke skrin dalam format tertentu.
- Fungsi `scanf (...)` digunakan untuk membaca input dari keyboard dalam format tertentu.



Format specifier	Output
%d	Decimal
%c	Single character
%s	String
%f	Floating-point number (real number)
%e	Real number in exponent form
%E	Realnumber in exponent form and uppercase
%g	guna %f atau %e mana yg lebih pendek
%u	unsigned integer
%x	unsigned hexadecimal number
%X	unsigned hexadecimal and uppercase
%o	unsigned octal number
%%	print a percent mark (%)

**Contoh:**

```
char a='A';
printf("%c %d %x %0", a, a, a, a);
```

**Output:**

A 65 41 101

- Format pembolehkan yg dicetak boleh diubahsuai dgn cara memasukkan **pengubahsuai (modifier)** di antara % & **format specifier**.

Contohnya :

**%-6d, %5d, %6.2f**

Pengubahsuai	Makna
<b><i>digit</i></b>	Lebar medan minimum bagi mencetak pembolehkan. Medan yg lebih lebar akan digunakan jika nombor yg dicetak tidak muat dlm medan yg diberi.
<b><i>.digit</i></b>	Mewakili bilangan titik perpuluhan.
<b>-</b>	Pembolehkan akan dicetak bermula dari kiri ( <i>left justify</i> ).
<b><i>l</i></b>	Pembolehkan yg akan dicetak adalah jenis <i>long integer</i> .

Contoh :

**Arahan**

**Output**

<code>printf(" %d ", 987);</code>	<code> 987 </code>
<code>printf(" %2d ", 987);</code>	<code> 987 </code>
<code>printf(" %8d ", 987);</code>	<code>  987 </code>
<code>printf(" %-8d ", 987);</code>	<code> 987  </code>
<code>printf(" %0.2f ", 9876.54);</code>	<code> 9876.54 </code>
<code>printf(" %4.2f ", 9876.54);</code>	<code> 9876.54 </code>
<code>printf(" %3.1f ", 9876.54);</code>	<code> 9876.5 </code>
<code>printf(" %10.3f ", 9876.54);</code>	<code>  9876.540 </code>

- Selain fungsi `printf` & `scanf` C juga menyediakan fungsi-fungsi lain untuk operasi input & output. Fungsi2 ini boleh dicapai dari library (pustaka) `<stdio.h>` dan `<conio.h>`.

Contoh:

`conio.h` : `clrscr`    `getch`    `putch`  
`stdio.h` : `scanf`    `getchar`    `getc`    `putc`    `gets`    `puts`

## Fungsi `getchar()` & `putchar()`

- dicapai dari *library* `<stdio.h>`
- Fungsi `getchar` digunakan utk membaca satu aksara dari papan kekunci dan menunggu aksara `<ENTER>` dimasukkan. Aksara yang ditaipkan juga akan dipaparkan ke skrin.

Contoh:

```
ch = getchar ();
```

nilai yg dibaca dari papan kunci akan disimpan dalam *ch*

- Fungsi `putchar` digunakan utk mencetak satu aksara ke skrin.

**Contoh:**

```
putchar (ch);
```

*variable atau constant*



- Contoh Aturcara 1: membaca satu aksara dari *keyboard* & mencetak aksara tersebut & aksara selepasnya :

```
char huruf;  
  
huruf = getchar ();  
putchar (huruf); putchar (huruf+1);
```

**Output:** *katakan pengguna memasukkan E*

**EF**

### ○ **Contoh Aturcara 2:**

```
char hurufA = 0x41; // 0x means hexadecimal  
putchar (hurufA);  
putchar ('b');  
putchar ('\n');  
putchar ('C');
```

**Output:**

**Ab  
C**

## Fungsi getch() & putch()

- dicapai dari *library* <conio.h>

Oleh itu, jika hendak menggunakannya perlu

```
#include <conio.h>
```

- Fungsi `getch` digunakan utk membaca satu aksara dari papan kekunci tanpa menunggu <ENTER> dimasukkan. **Aksara yang dimasukkan tidak akan dipamer ke skrin.**

*Contoh :*

```
ch = getch ();
```

nilai yg dibaca dari *keyboard* akan disimpan dalam `ch`

- Fungsi `putch` digunakan utk mencetak satu aksara ke skrin.

*Contoh:*

```
putch (ch);
```

*character variable* atau *character constant* yang hendak dicetak

- Jika contoh dari `getchar` & `putchar` ditukarkan kpd `getch` & `putch` ia akan memberi keluaran yg sama, hanya cara masukan berbeza.

## Fungsi `getc()` & `putc()`

○ dicapai dari *library* `<stdio.h>`

○ Fungsi `getc` digunakan utk membaca satu aksara dari *standard input*. Contoh *standard input* adalah fail dan *keyboard*. Cara menulisnya adalah hampir sama dengan `getchar()` kecuali perlu ditambah dengan jenis *standard input* yang digunakan.

### **Syntax:**

```
getc(jenis_peranti)
```

**Contoh:** membaca aksara dari *keyboard*:

```
ch = getc (stdin);
```

○ Fungsi `putc` digunakan utk mencetak satu aksara ke skrin.

### **Syntax:**

```
putc ( ch, jenis_peranti);
```

Contoh: menggunakan fungsi `putc` utk mencetak aksara 'A' ke skrin.

```
putc('A', stdout);
```

○ Jika contoh dari `getchar` & `putchar` ditukarkan kpd `getc` & `putc` ia akan dilaksanakan dgn cara yg sama.

## Fungsi gets() & puts()

- dicapai dari *library* <stdio.h>
- Fungsi **gets** digunakan utk membaca string dari keyboard. Bacaan akan diakhiri jika *newlinecharacter* (\n) atau <ENTER> dimasukan. String akan disimpan dan *newlinecharacter* akan ditukar kepada *null character* (\0).
- Fungsi **puts** digunakan untuk mencetak *string* ke skrin dengan menambah *newline character* (\n) dipenghujung *string* tersebut.

### Masalah dengan aturcara sebelum ini:

```
#include <stdio.h>

int main()
{
    char nama[50];
    char negeri[15];

    printf("Masukkan nama pelajar: ");
    scanf("%s", &nama);

    printf("Masukkan negeri asal: ");
    scanf("%s", &negeri);

    printf("\n\n%s merupakan seorang pelajar yang berasal dari %s", nama, negeri);
    return 0;
}
```

Output

```
C:\Documents and Settings\Jumail\Desktop\New Folder\t2.exe
Masukkan nama pelajar: RAMLI AHMAD
Masukkan negeri asal: AHMAD
RAMLI merupakan seorang pelajar yang berasal dari AHMAD

Kenapa masalah ini berlaku?
```

## Penyelesaian: Gunakan gets menggantikan scanf

```
#include <stdio.h>

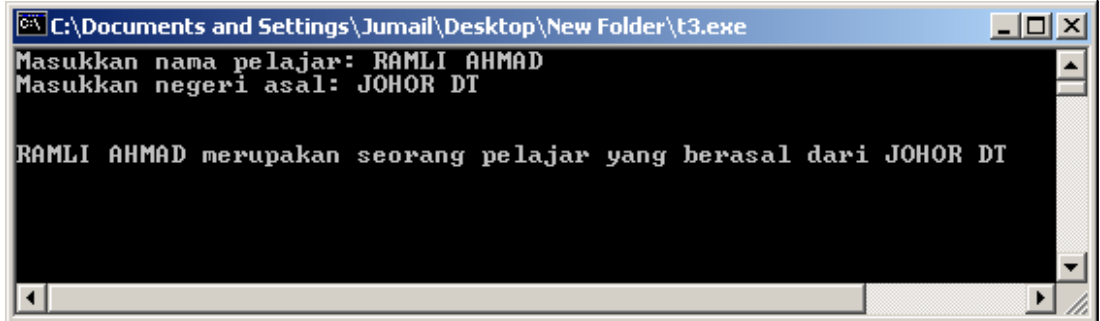
int main()
{
    char nama[50];
    char negeri[15];

    printf("Masukkan nama pelajar: ");
    gets(nama); // tidak perlu ada simbol &

    printf("Masukkan negeri asal: ");
    gets(negeri);

    printf("\n\n%s merupakan seorang pelajar yang berasal dari %s", nama, negeri);
    return 0;
}
```

Output



```
C:\Documents and Settings\Jumail\Desktop\New Folder\t3.exe
Masukkan nama pelajar: RAMLI AHMAD
Masukkan negeri asal: JOHOR DT

RAMLI AHMAD merupakan seorang pelajar yang berasal dari JOHOR DT
```

## Latihan Ulangkaji:

1. Apakah output bagi setiap pernyataan berikut :

- `printf("Panjang tali adalah : %2.2f m", 2.445);`
- `printf("%c %d %u", 66, 0x50, 'C');`
- `printf("%-8d\n%6.3f %-6.3f", 4356, 1.52, 1.52);`
- `putchar('\'); putchar('\'); putchar(66);`

## 2. Cari kesalahan (*errors*) dalam aturcara berikut:

```
include (stdio.h)
int main()
{
    int nom1, nom2;
    unsigned int jum1;
    char warna;

    jum1=-1;
    scanf("%d %d", nom1, num2);
    printf("%u", jum1);
    &warna=getchar();
    printf("%d %c %u", warna, warna, warna);

    return 0;
}
```