

# STRUKTUR DATA

## Pertemuan 1



Ratih Ngestrini, Nori Wilantika, Firdaus



# SKEMA PERKULIAHAN



Satuan Kredit: 2 SKS Teori, 1 SKS Praktikum



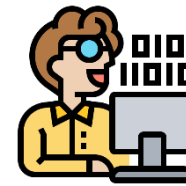
2 x 60'  
Belajar Mandiri



2 x 50' Teori  
(Offline/Online)



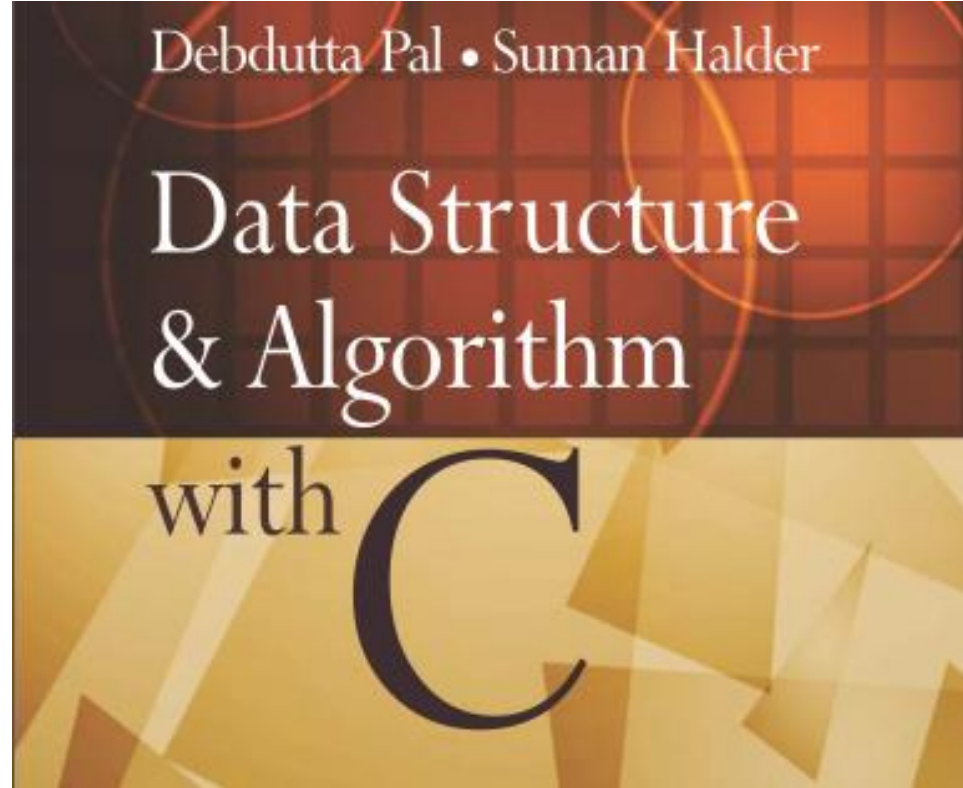
1 x 50'  
Praktikum  
(Modul)



1 x 120'  
Praktikum  
Mandiri



2 x 60'  
Tugas Terstruktur

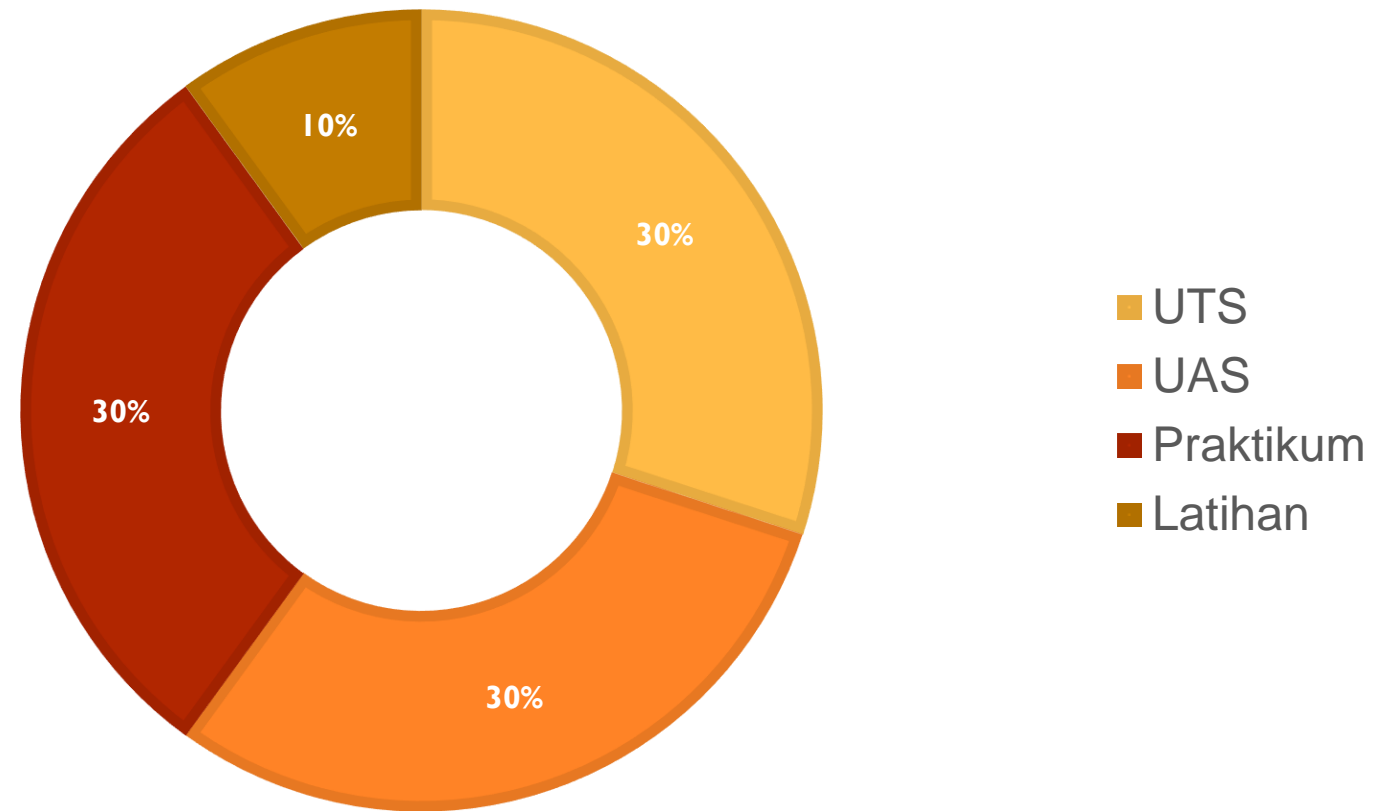


Pal, D. & Halder, S. (2018).  
*Data Structure & Algorithm with C.*  
Alpha Science International Ltd.



Sjukani, M. (2012).  
*Struktur Data (Algoritma & Struktur Data 2) dengan C, C++.* Mitra Wacana Media

# KOMPONEN PENILAIAN



# ATURAN-ATURAN UMUM

## KEHADIRAN

- Kehadiran seorang mahasiswa pada setiap sesi merupakan **tanggung jawab pribadi** mahasiswa tersebut. Politeknik Statistika STIS menetapkan aturan minimal 80% kehadiran untuk dapat diijinkan mengikuti UTS dan UAS.
- Pada pertemuan online, selama mengikuti zoom mahasiswa dalam posisi duduk, berpakaian rapi, dan mengaktifkan video. Nama yang ditampilkan di zoom: nomor absen\_nama, contoh: **1\_Nori Wilantika** (Tidak perlu menggunakan NIM)

## BIMBINGAN TAMBAHAN

- Gunakan fasilitas *Stream/Forum* pada *Google Classroom* atau hubungi asisten
- Mahasiswa diizinkan untuk menanggapi pertanyaan mahasiswa lain dan dihitung sebagai nilai partisipasi.
- *Private message* diperkenankan melalui email pada jam kerja (Senin-Jumat 07.30-16.00)



# PENGENALAN STRUKTUR DATA



# TIPE DATA

Apa itu Tipe Data?

Tipe data mengacu pada tipe data yang dimiliki variabel atau konstanta.

Secara umum, di bahasa pemrograman apapun, tipe data dibagi menjadi:

- Alphanumerical Data
- Numerical Data



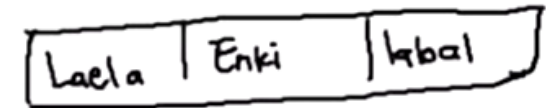
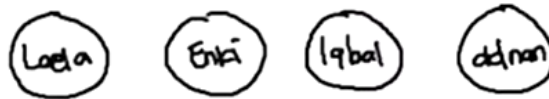
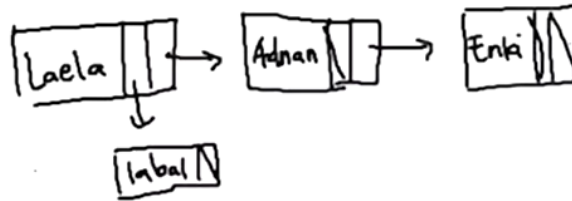
# Tipe Data VS Objek Data VS Struktur Data

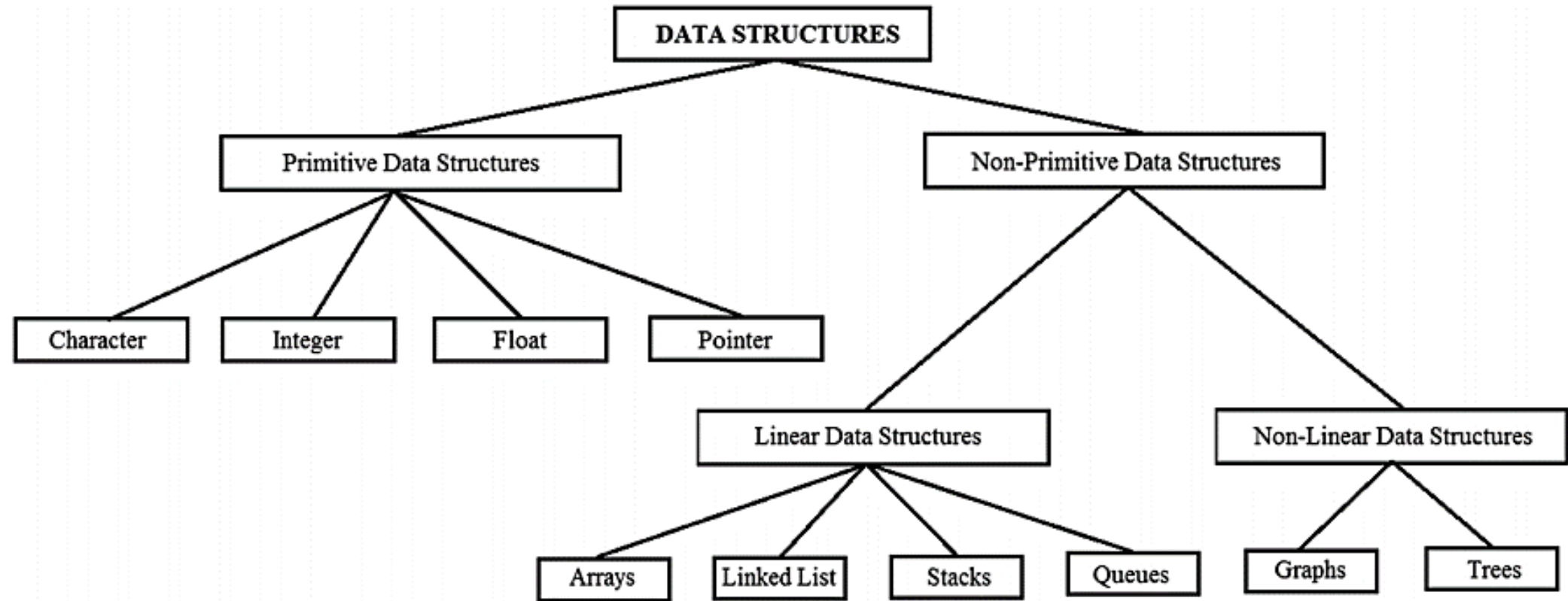
- **Tipe Data:** jenis data yang mampu ditangani oleh suatu bahasa pemrograman pada komputer. Misalnya di Pascal: integer, real dll.
- **Objek Data** adalah kumpulan elemen yang mungkin untuk suatu tipe data tertentu. Misal integer di pascal adalah bilangan antara -32768 s.d. 32767
- **Struktur Data** adalah cara penyimpanan, pengorganisasian , dan pengaturan data di dalam media penyimpanan komputer sehingga data tersebut dapat digunakan secara efisien.

# STRUKTUR DATA

Struktur data: cara menyimpan dan mengorganisasi sekumpulan data (secara logical) dengan aturan tertentu pada memori komputer.

Struktur data adalah model logika atau model matematis.



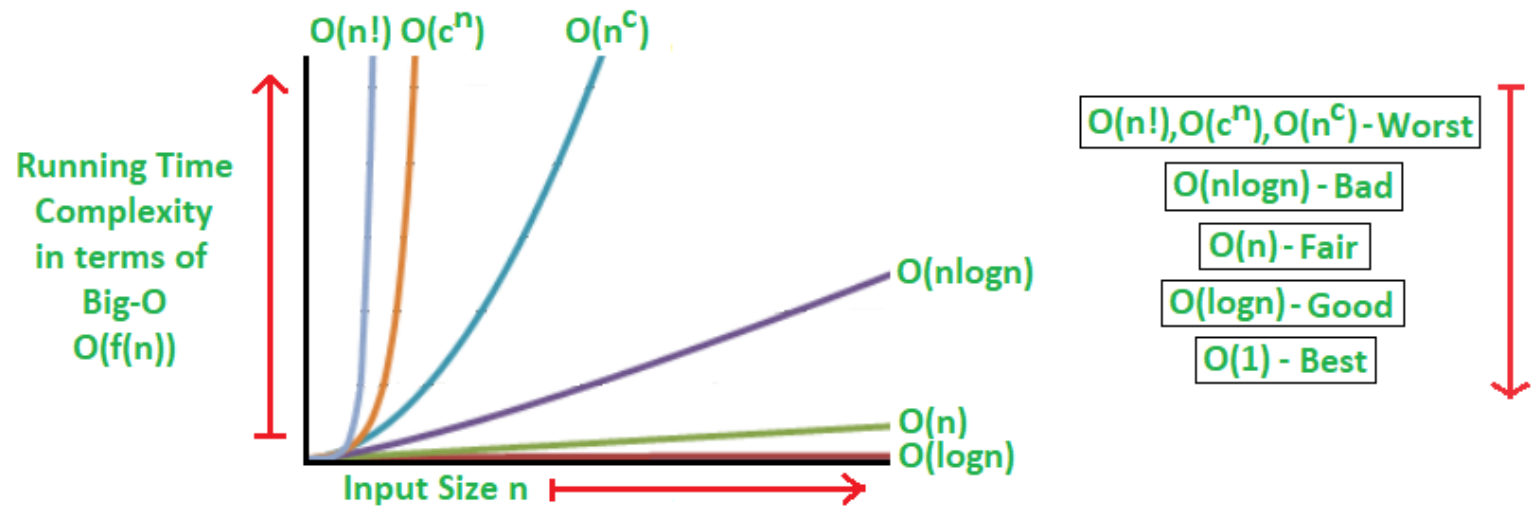


# TUJUAN STRUKTUR DATA

- Tujuan dari struktur data adalah untuk menyimpan, mengambil, dan memperbarui data secara efisien.
- Setiap programmer yang baik akan menggunakan cara yang paling efektif dan efisien dalam menyelesaikan suatu permasalahan, berarti harus bisa meminimalisir kompleksitas dari algoritma yang kita gunakan.
- Kompleksitas suatu algoritma dibagi menjadi 2, yaitu **Time Complexity** dan **Space Complexity**.
- **Time Complexity (O)** adalah seberapa lama waktu yang diperlukan untuk menjalankan suatu algoritma, dengan input tertentu (n). **Space Complexity** adalah seberapa besar memori yang kita gunakan untuk menjalankan suatu algoritma.

# Notasi “O Besar” (Big-O notation)

- $O(1)$  — Constant Time
- $O(\log n)$  — Logarithmic Time
- $O(n)$  — Linear Time
- $O(n^c)$  — Polynomial time
- $O(c^n)$  — Exponential Time



# MATA KULIAH STRUKTUR DATA

- Mata kuliah ini mengajarkan teknik dasar untuk mengabstraksikan data, membuat algoritma yang dapat mengakses data tersebut, dan memanipulasi struktur abstrak tersebut.
- Pada mata kuliah ini juga akan diperkenalkan analisa kompleksitas ruang dan waktu dalam mengimplementasikan sebuah algoritma.
- Topik-topik yang dibahas meliputi: array, pointer, tumpukan, antrian, antrian berprioritas, linked list, tree, graph, hash table, algoritma pengurutan dan pencarian.

Pertemuan 1	Pengenalan dan Manfaat struktur data
Pertemuan 2	Primitive and Abstract Data Types, Record dan Pointer
Pertemuan 3	Array,Alokasi Memori Dinamis
Pertemuan 4	Single Linked List
Pertemuan 5	Doubly Linked List
Pertemuan 6	Stack/Tumpukan
Pertemuan 7	Queue/Antrian

Pertemuan 8	Tree
Pertemuan 9	Tree
Pertemuan 10	Graph
Pertemuan 11	Searching
Pertemuan 12	Hash Table
Pertemuan 13	Sorting
Pertemuan 14	Sorting



# PENGENALAN BAHASA C





- **C** adalah bahasa pemrograman tingkat menengah. Dapat digunakan untuk pemrograman tingkat rendah (*assembly language*) seperti kernel, driver, dll dan juga mendukung fungsi – fungsi dalam bahasa pemrograman tingkat atas (Java, Python, Ruby, dll)
- **C** adalah bahasa pemrograman terstruktur (program dapat dipecah menjadi beberapa program yang lebih kecil – ***function***) . Contoh: program untuk mengelola nilai mahasiswa

```
main() Nilai Mahasiswa
      HitungNilai()
      TampilkanNilai()
      InputNilai()
      UpdateNilai()
      GetNilaiMahasiswaByNIM()
```

- **C** adalah *case-sensitive* (huruf kecil dan huruf besar berbeda/berpengaruh)
- **C** adalah *general purpose programming language* (bertujuan umum) untuk menyelesaikan banyak masalah, *tidak spesifik* masalah tertentu.



# Penulisan Bahasa C

```
#include <stdio.h>

int main() {
    /* my first program in C */
    printf("Hello, World! \n");

    return 0;
}
```

- Ekstensi file “.c”
- `include stdio.h` = header file (library) pada bahasa C yang digunakan untuk operasi input-output (stdio = Standar Input dan Output). Tanpa menggunakan library ini maka perintah-perintah input dan output tidak dapat dieksekusi. Fungsi – fungsi di `stdio.h` meliputi:

Library Functions:

<code>clearr()</code>	<code>fclose()</code>	<code>fccloseall()</code>	<code>fdopen()</code>	<code>fflush()</code>
<code>fgetc()</code>	<code>fgetchar()</code>	<code>fgetpos()</code>	<code>fgets()</code>	<code>flushall()</code>
<code>fopen()</code>	<code>fprint()</code>	<code>fputc()</code>	<code>fputchar()</code>	<code>fputs()</code>
<code>fread()</code>	<code>free()</code>	<code>freopen()</code>	<code>fscan()</code>	<code>fseek()</code>
<code>fsetpos()</code>	<code>ftell()</code>	<code>fwrite()</code>	<code>gets()</code>	<code>getw()</code>
<code>perror()</code>	<code>printf()</code>	<code>puts()</code>	<code>rename()</code>	<code>rewind()</code>
<code>scanf()</code>	<code>unlink()</code>			

# Library Bahasa C

Contoh beberapa header file untuk program C:

- **stddef.h** – mendefinisikan tipe/ekspresi standar
- **stdint.h** – mendefinisikan jenis integer dengan lebar tertentu
- **stdio.h** – mendefinisikan fungsi input-output standar
- **stdlib.h** – mendefinisikan fungsi konversi numerik dan alokasi memori
- **string.h** – mendefinisikan fungsi untuk penanganan string
- **math.h** – mendefinisikan fungsi matematika yang umum

# Penulisan Bahasa C

```
#include <stdio.h>

int main() {
    /* my first program in C */
    printf("Hello, World! \n");

    return 0;
}
```

- `int main()` Fungsi utama dimana program mulai dieksekusi
- `/* ...*/` atau `//` adalah *comments* dan akan diacuhkan oleh compiler
- `printf()` adalah fungsi yang akan menampilkan kata – kata “Hello, World!”
- `return 0` akan mengakhiri fungsi `main()`
- Setiap statement diakhiri dengan titik koma (;)

## Contoh:

```
// menuliskan kalimat
#include <stdio.h>

int main() {
    printf("Selamat, Anda telah berhasil\n");
    printf("membuat program C yang pertama!\n");
    return 0;
}
```

*Komentar, tidak diproses header*

*Pindah baris*

*Akhir perintah*

*Perintah untuk tampilkan ke layar*

# IDENTIFIER

- **Identifier** : untuk memberi nama/mengidentifikasi variabel, fungsi, dll
  - A - Z, a - z, *underscore*, dan digit 0 – 9
  - **Selain reserved words** berikut:

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

# Deklarasi Variabel

- **Deklarasi Variabel:** `data_type variable_name;`

Contoh:

```
int x, y, z;  
char flag;  
char ch;
```

- **Inisialisasi/Definisi Variabel:** `data_type variable_name = value;`

Contoh:

```
int x = 30;  
y = 100;  
z = y;  
char flag = 'x';  
ch = 'n';
```

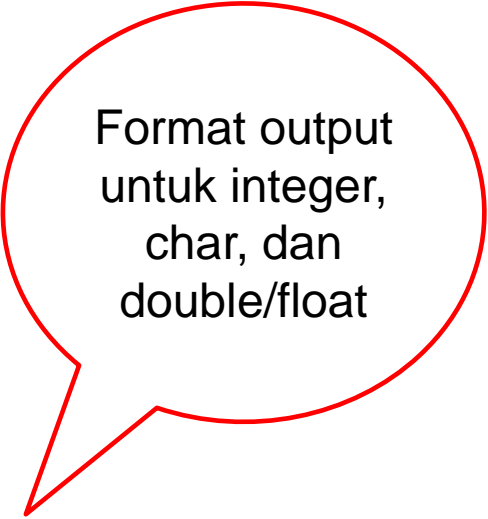


## ■ Deklarasi di bahasa C:

```
#include <stdio.h>
int main() {
    int a = -128;
    char ch2 = 'a';
    unsigned char uc = 'b';
    short s = 10;
    int i = 1000;
    unsigned int ui = 45555;
    long l = 1234567;
    unsigned long ul = 1234567898;
    float f = 3.5;
    double d = 23.9999;
    long double ld = 23.239;

    printf("Nilai dari variabel adalah : %d, %c, dan %f", a, ch2, d);

    return 0;
}
```



Format output  
untuk integer,  
char, dan  
double/float

# Variabel berdasarkan *Scope* dalam Program

## 1. Local Variable

- Ruang lingkup hanya di dalam fungsi dimana dia dideklarasikan
- Tidak dapat diakses di luar fungsi tersebut

## 2. Global Variable

- Ruang lingkup berada di seluruh program
- Didefinisikan di luar *main function* jadi bisa dipanggil di *main function* dan fungsi-fungsi lain
- Variabel-variabel ini dapat diakses dari mana saja dalam program ini

## 3. Environment Variable

- Variabel yang tersedia untuk semua aplikasi dan program
- Dapat diakses dimanapun di program C tanpa harus mendeklarasikan atau mendefinisikannya
- Akses dengan inbuilt function: `getenv()`, modifikasi: `setenv()`, assign: `putenv()`

# Local & Global Variable

```
#include <stdio.h>
int m = 22, n = 44;
int a = 50, b = 80;
```

Global Variables

```
int main()
{
    printf("Tampilkan semua variabel dari main function:");
    printf("\nvalue: m = %d,n = %d, a = %d, b = %d", m, n, a, b);

    test();
}
```

Fungsi Utama

```
void test()
{
    int x = 100, y = 200;
    printf("\n\nTampilkan semua variabel dari test function:");
    printf("\nvalue: m = %d,n = %d, a = %d, b = %d, x = %d, y = %d", m, n, a, b, x, y);
}
```

Local Variables

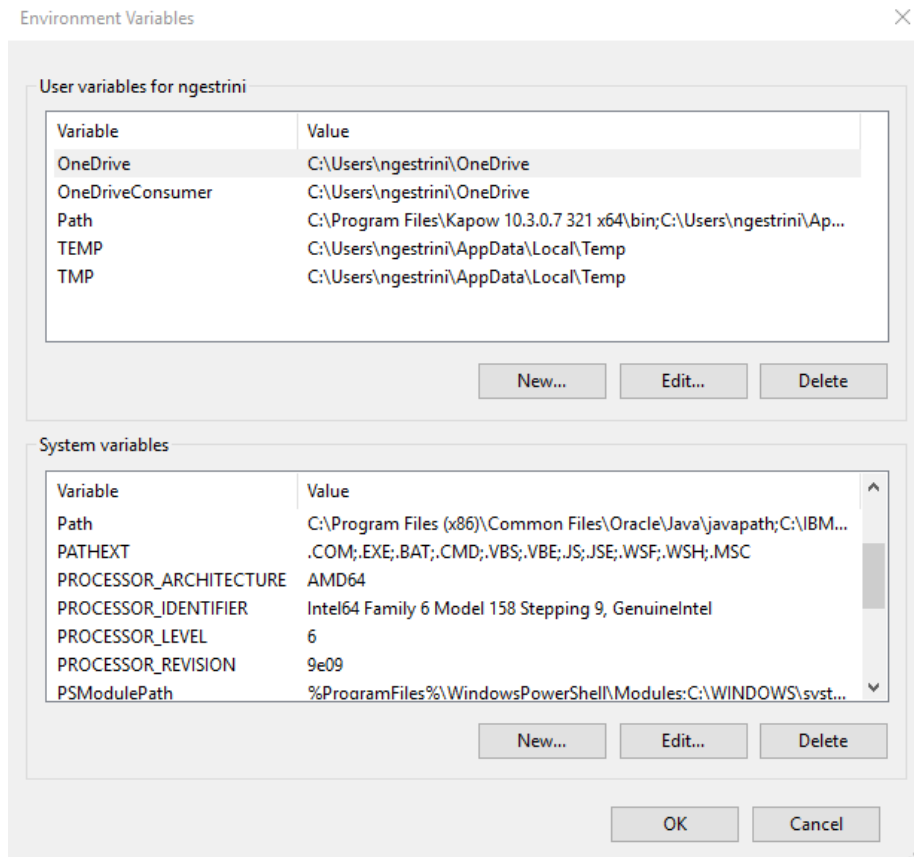
Fungsi test ()

## Hasil:

Tampilkan semua variabel dari main function:  
value: m = 22,n = 44, a = 50, b = 80

Tampilkan semua variabel dari test function:  
value: m = 22,n = 44, a = 50, b = 80, x = 100, y = 200

# Environment Variable



Variable	Value
TEMP	C:\WINDOWS\TEMP
TEST_EV	450
TMP	C:\WINDOWS\TEMP
USERNAME	SYSTEM
VS140COMNTOOLS	C:\Program Files (x86)\Microsoft Visu

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

getenv () = mengambil nilai env variabel

setenv () = mengisi nilai env variabel

```
int main()
{
    printf("Nilai awal TEST_EV = %d\n", getenv("TEST_EV"));
    setenv("TEST_EV", 300, 1);
    printf("Nilai akhir TEST_EV = %d\n", getenv("TEST_EV"));
    return 0;
}
```

Untuk Windows di control panel - system properties – environment variables

# Fungsi di Bahasa C

- Fungsi adalah suatu blok *statement/code* yang melakukan tugas tertentu

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

```
int jumlah(int a, int b);
```

*Function prototype*

```
int main()
```

```
{
```

```
    int n1, n2, sum;
```

```
    printf("Masukan angka 1: ");
```

```
    scanf("%d", &n1);
```

```
    printf("Masukan angka 2: ");
```

```
    scanf("%d", &n2);
```

```
    sum = jumlah(n1, n2);
```

*Function call*

```
    printf("sum = %d", sum);
```

```
    return 0;
```

```
}
```

```
int jumlah(int a, int b)
```

*Function definition*

```
{
```

```
    int hasil;
```

```
    hasil = a+b;
```

```
    return hasil;
```

*Return statement*

```
}
```

## Mengapa *function prototype* dibutuhkan?

- Fase 1 dari compiler (*lexical analysis*) akan membaca dari kiri ke kanan, atas ke bawah
- Menginformasikan ke compiler tentang nama fungsi, *return type*, dan parameter
- Jika fungsi didefinisikan sebelum `main()` maka *function prototype* tidak perlu

# Fungsi di Bahasa C

Fungsi didefinisikan sebelum `main()` :

```
#include<stdio.h>

void displayMessage() {
    printf("www.c4learn.com");
}

void main() {
    displayMessage();
}
```

**Void = Tidak *return* value apapun  
(seperti procedure dalam pascal)**

Fungsi didefinisikan setelah `main()` :

```
#include<stdio.h>

//Prototype Declaration
void displayMessage();

void main() {
    displayMessage();
}

void displayMessage() {
    printf("www.c4learn.com");
}
```

# Passing Arguments di Bahasa C

*pass by reference*



*fillCup(       )*

*pass by value*



*fillCup(       )*

[www.mathwarehouse.com](http://www.mathwarehouse.com)

- **Pass by reference** (Pointer Sebagai Parameter Fungsi) : alamat digunakan untuk mengakses parameter yang dipanggil fungsi, jika ada dilakukan perubahan dalam fungsi tsb maka akan mengubah nilai dalam alamat tsb
- **Pass by value** : mengakses nilai dari parameter yang dipanggil di fungsi



TERIMA KASIH