



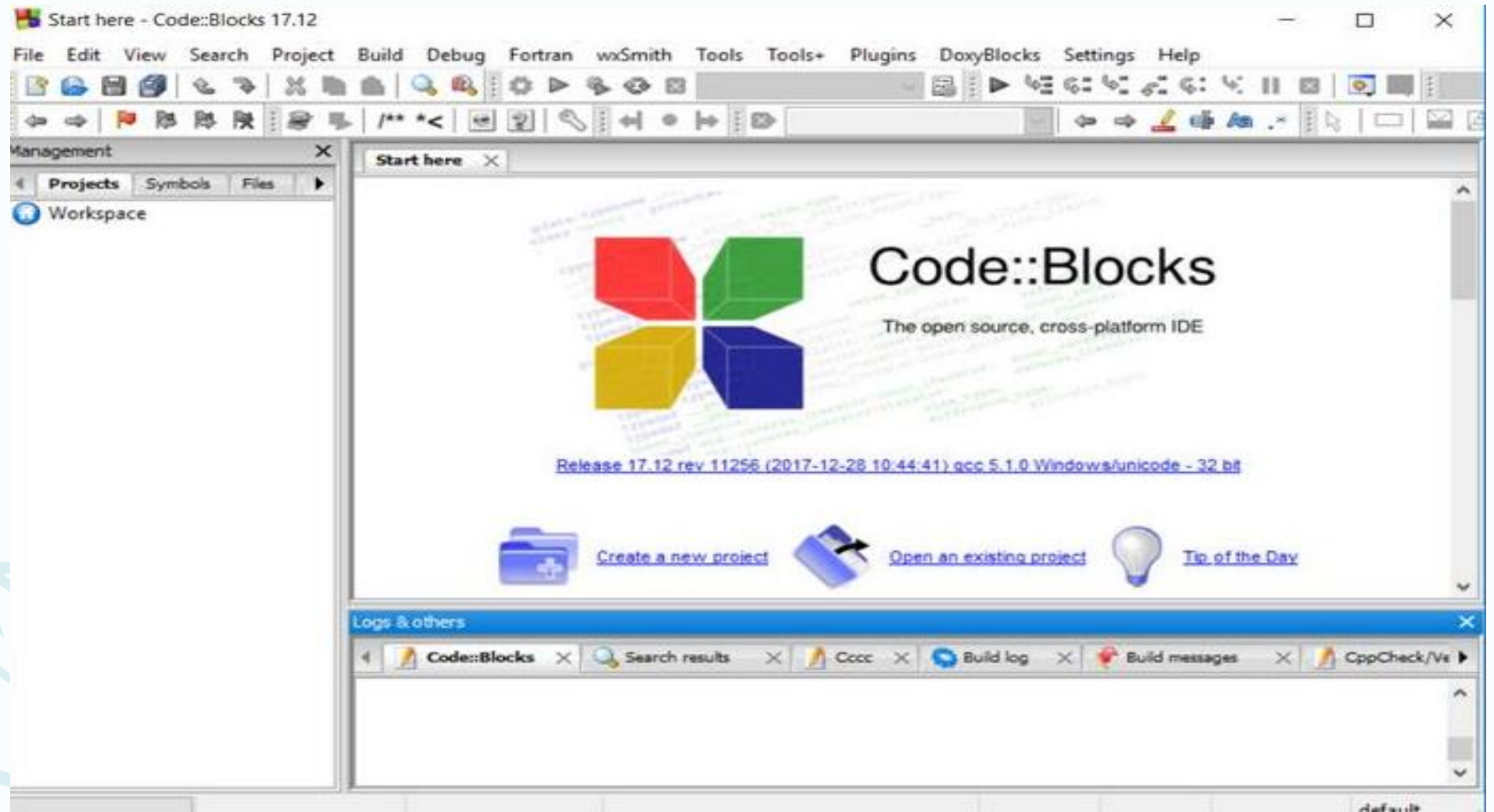
# Bahasa C

## 2 Bahasa Pemrograman C

---

- **Bahasa Pemrograman C** merupakan salah satu bahasa pemrograman komputer paling senior yang masih digunakan hingga saat ini.
- Dirilis pertama kali tahun 1972 oleh **Dennis Ritchie**, C menjadi “dasar” dari berbagai bahasa pemrograman yang lebih modern seperti **C++**, **C#**, **Java**, **PHP** hingga **JavaScript**.
- Bersama-sama dengan bahasa pemrograman **Pascal** dan **C++** ketiganya sering digunakan untuk belajar algoritma, yakni dasar dari pemrograman.
- Dengan mempelajari bahasa C, anda akan familiar dan lebih mudah saat berpindah ke bahasa pemrograman lain yang merupakan turunan dari bahasa C.

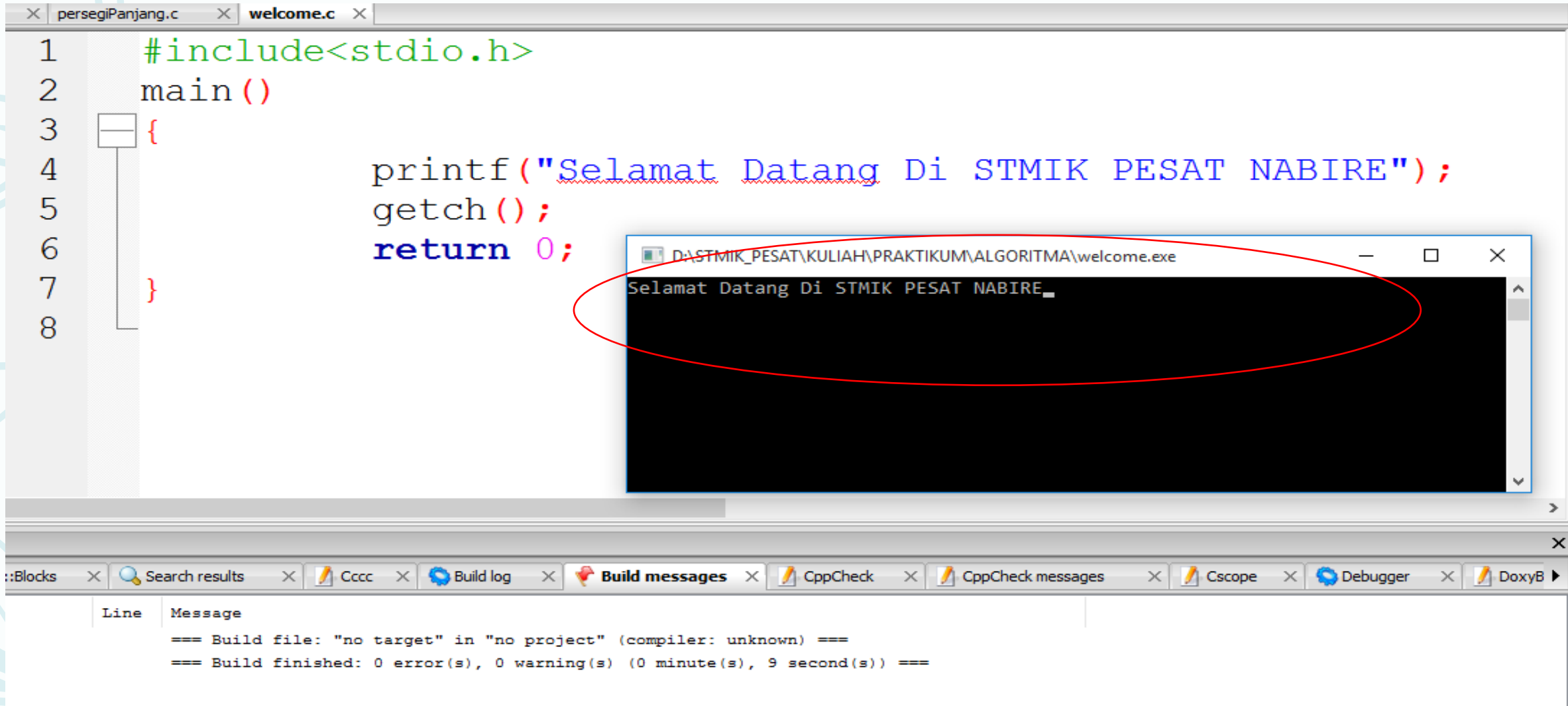
3



Tampilan dari CodeBlocks.

# Struktur Dasar Bahasa Pemrograman C

- 4 Kode program yang telah jalankan sebelumnya sangat sederhana, tapi sudah mewakili struktur dasar dari sebuah bahasa pemrograman C. Berikut kode program tersebut:



The image shows a screenshot of a C program editor and its execution. The editor window has two tabs: 'persegiPanjang.c' and 'welcome.c'. The 'welcome.c' tab is active, showing the following code:

```
1  #include<stdio.h>
2  main()
3  {
4      printf("Selamat Datang Di STMIK PESAT NABIRE");
5      getch();
6      return 0;
7  }
8
```

Below the code editor, there is a console window titled 'D:\STMIK\_PESAT\KULIAH\PRAKTIKUM\ALGORITMA\welcome.exe'. The console displays the output of the program: 'Selamat Datang Di STMIK PESAT NABIRE\_'. A red oval highlights the console window.

At the bottom of the screenshot, there is a status bar with various tabs: 'Blocks', 'Search results', 'Cccc', 'Build log', 'Build messages', 'CppCheck', 'CppCheck messages', 'Cscope', 'Debugger', and 'DoxyB'. The 'Build messages' tab is active, showing the following messages:

```
Line Message
=== Build file: "no target" in "no project" (compiler: unknown) ===
=== Build finished: 0 error(s), 0 warning(s) (0 minute(s), 9 second(s)) ===
```

# #include <stdio.h>

## 5

- Di baris paling awal, terdapat kode **#include**.
- Perintah **#include** digunakan untuk memasukkan sebuah file khusus yang memungkinkan kita mengakses berbagai fitur tambahan dalam bahasa C.
- Dalam contoh diatas, file **stdio.h** berisi kode program agar nantinya kita bisa mengakses perintah **printf**. File **stdio.h** sendiri merupakan singkatan dari *Standard Input/Output*.
- Dengan kata lain, agar di dalam kode program nanti kita bisa menggunakan perintah **printf**, dibagian paling atas kode program C harus terdapat baris **#include <stdio.h>**.
- File include ini juga sering disebut sebagai **header file**, dan karena itu pula menggunakan akhiran **.h**.
- Bahasa C menerapkan konsep *modular*, dimana fitur-fitur yang ada di pecah ke berbagai file. Jika ingin menggunakan perintah tertentu, panggil header file yang sesuai.
- Hasilnya, ukuran file program yang ditulis menggunakan bahasa C menjadi efisien. Kita hanya perlu menggunakan header file yang dibutuhkan saja. Namun kebalikannya, setiap ingin menggunakan perintah tertentu, harus men-include-kan file header yang dibutuhkan.

# int main(void) {}

- 6 – Satu-satunya perintah yang harus ada di setiap kode program bahasa C adalah **main()**.
- Struktur **main()** sendiri pada dasarnya merupakan sebuah fungsi (*function*). Isi dari function ini diawali dan diakhiri dengan tanda kurung kurawal " { " dan " } ". Di dalam tanda kurung inilah "isi" dari kode program penyusun fungsi **main()** ditulis.
  - Kode **int** sebelum **main()** menandakan **nilai kembalian** atau hasil akhir dari function **main()**. Kode **int** merupakan singkatan dari **integer**, yakni tipe data angka bulat.
  - Dengan demikian, kode program **main()** yang saya tulis diatas harus menghasilkan sebuah angka bulat (menggunakan perintah **return** yang akan kita bahas sesaat lagi).
  - Sedangkan tambahan **void** ke dalam **main(void)** menandakan bawah fungsi **main()** tidak membutuhkan nilai input (bahasa inggris void = kosong).
  - Jika anda agak bingung dengan penjelasan ini, bisa dianggap bahwa **int main(void) { }** adalah perintah yang mengawali setiap kode program bahasa C.

# printf(“Selamat Datang ....”);

- Perintah **printf** digunakan untuk menampilkan sesuatu ke layar. Perintah ini merupakan bagian dari **stdio.h**, sehingga jika kita ingin menggunakannya, harus terdapat baris perintah **#include <stdio.h>** di bagian paling awal kode program bahasa C.
- Teks yang ingin ditampilkan ditulis dalam tanda kurung dan di dalam tanda kutip dua, seperti: **printf(“Selamat ....”);** Hasil dari perintah ini, akan tampil teks **Selamat Datang ...** di layar. Tapi apa fungsi tambahan karakter **\n**?
- Jika ditulis di dalam teks, karakter “ \ ” dikenal sebagai **escape character**. Fungsinya untuk menampilkan karakter yang tidak bisa ditulis. Sebagai contoh, **\n** merupakan perintah untuk menulis *newline character*, yakni karakter penanda baris baru.
- Artinya, perintah **printf(“Selamat Datang \n”)** akan menampilkan teks **“Selamat Datang”**, kemudian pindah ke baris baru. Bahasa C mendukung berbagai *escape character* yang nantinya juga akan kita pelajari.
- Setelah tanda kurung penutup perintah **printf**, harus ditutup dengan tanda titik koma (*semi-colon*), yakni tanda “ ; ”. Setiap perintah bahasa C, harus diakhiri dengan tanda ini, kecuali beberapa perintah khusus. Lupa menambahkan tanda titik koma di akhir sebuah perintah merupakan error yang sangat sering terjadi.



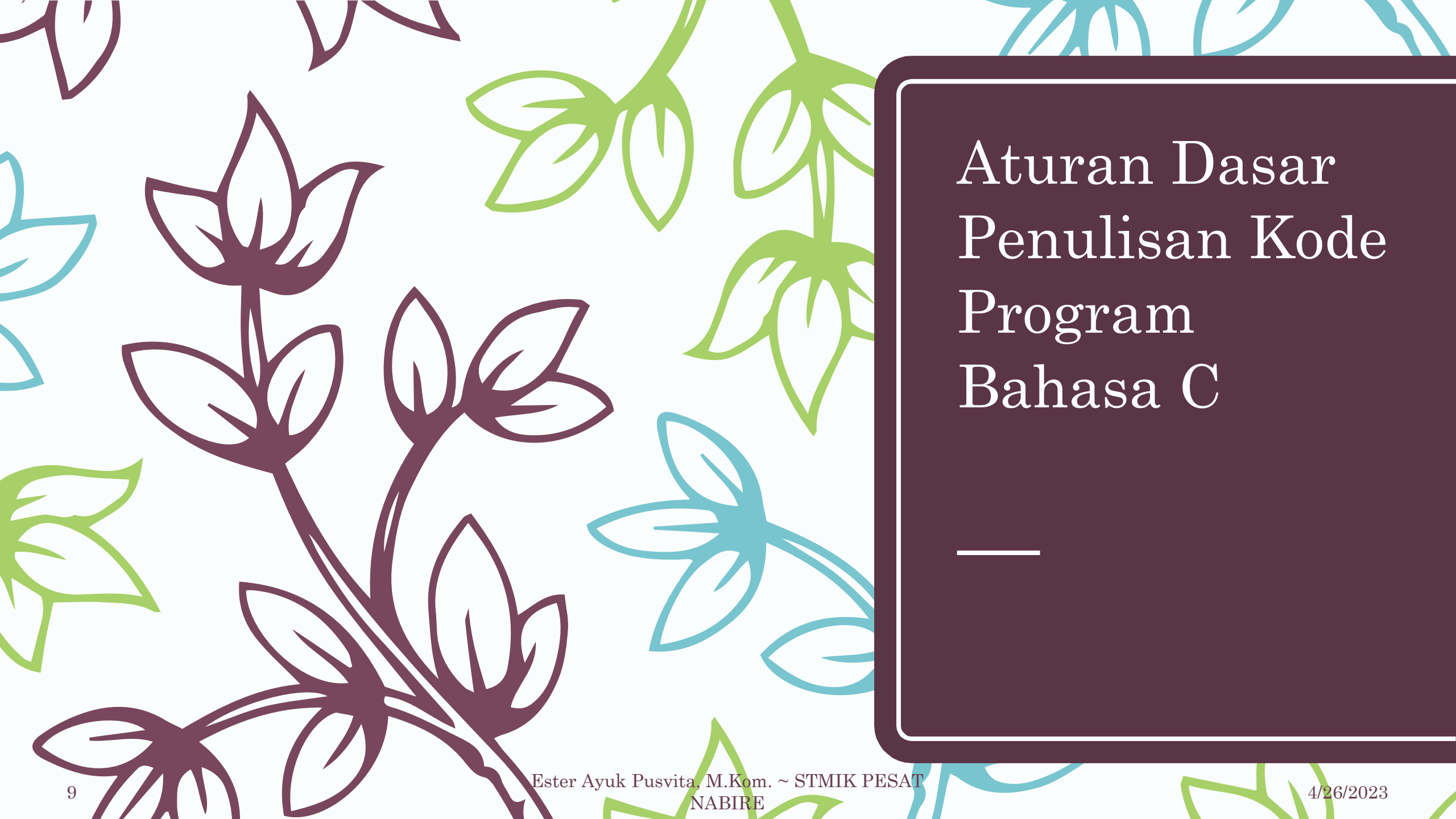
## 8 return 0;

- Perintah **return 0**; berhubungan dengan kode **int main(void)** sebelumnya. Disinilah kita menutup *function main()* yang sekaligus mengakhiri kode program bahasa C.

---

- **Return 0** artinya kembalikan nilai **0** (nol) ke sistem operasi yang menjalankan kode program ini. Nilai 0 menandakan kode program berjalan normal dan tidak ada masalah (EXIT\_SUCCESS).
- Kita juga bisa menulis return 1, return 99, return -1, dll. Nilai-nilai ini nantinya bisa digunakan oleh sistem operasi atau program lain. Nilai return selain 0 dianggap terjadi error atau sesuatu yang salah (EXIT\_FAILURE).
- *Apakah perintah Return 0 ini harus ditulis? **Harus ditulis!*** jika kita berpatokan ke struktur bahasa C yang ideal. Namun beberapa compiler (termasuk **Code:Blocks** yang saya gunakan), akan “memaafkan” jika perintah ini tidak ditulis dan menambahkan perintah return 0 secara otomatis (tidak disarankan).





# Aturan Dasar Penulisan Kode Program Bahasa C

---

# 10 Perbedaan Huruf Besar / Kecil dalam Bahasa C

- Pengertian sederhana dari **case sensitivity** adalah perbedaan antara huruf besar dan huruf kecil. Istilahnya, bahasa C termasuk bahasa yang **case sensitif**. Dalam bahasa C, huruf besar dan kecil dianggap berbeda. Perintah **printf** tidak bisa ditulis menjadi **Printf**.

```
#include <stdio.h>
int main(void)
{
    Printf("Hello, World!\n");    /* error !! */
    return 0;
}
```

Untuk penulisan variabel juga akan berbeda antara huruf besar dan kecil. Variabel **jumlah**, **JUMLAH**, dan **Jumlah** adalah 3 variabel yang berlainan. Aturan ini berbeda jika dibandingkan bahasa pemrograman PASCAL yang bersifat *case insensitif* (tidak membedakan huruf besar dan kecil).

# 11 Cara Penulisan Komentar di dalam Bahasa C

---

- **Komentar** atau *comment* adalah ‘kode program’ yang ditambahkan untuk memberi keterangan/penjelasan mengenai cara kerja program. Komentar tidak akan diproses oleh compiler C dan berfungsi untuk memberi keterangan tambahan (terutama jika kode program yang ditulis cukup rumit)
- Untuk membuat komentar di dalam kode program bahasa C, menggunakan tanda `/*` dan `*/`. Seluruh karakter yang ada diantara kedua tanda ini akan dianggap sebagai komentar dan diabaikan pada saat proses *compiler*.
- Berikut contoh penulisan komentar dalam bahasa C:

12

```
gIPanjang.c x welcome.c x contoh.c x
#include <stdio.h>
int main(void)
{
    /* Tampilkan pesan Hello World */
    printf("Hello, World!\n");
    return 0;
}
```

Komentar

Hasilnya adalah

```
D:\STMIK_PESAT\KULIAH\PRAKTIKUM\ALGORITMA\contoh.exe
Hello, World!

Process returned 0 (0x0)   execution time : 0.167 s
Press any key to continue.
```

- Komentar juga sering digunakan untuk membuat semacam “copyright” kode program yang ditulis di awal, seperti contoh kode program berikut:

13

```
1  /* Pembuat  : STMIK PESAT NABIRE
2     * Tujuan  : Menampilkan pesan "Hello, World!"
3     * Bahasa  : C
4     * Tanggal Pembuatan: 7 Mei 2021
5     */
6
7  #include <stdio.h>
8  int main(void)
9  {
10     printf("Hello, World!\n");
11     return 0;
12 }
```

Komentar “//” hanya berlaku untuk satu baris saja. Jika ingin membuat komentar lebih dari 1 baris, tanda “//” harus ditulis beberapa kali. Contohnya sebagai berikut:

Bahasa pemrograman C++ memperkenalkan alternatif pembuatan komentar, yakni dengan tanda “//”. Umumnya, compiler bahasa C juga mendukung bahasa C++, sehingga kita juga bisa menggunakan tanda “//” untuk membuat komentar dalam bahasa C.

```
persegipanjang.c  welcome.c  *contoh.c
//Pembuat  : STMIK PESAT NABIRE
//Tujuan  : Menampilkan pesan "Hello, World!"
//Bahasa  : C
//Tanggal Pembuatan: 7 Mei 2021

#include <stdio.h>
int main(void)
{
    printf("Hello, World!\n");
    return 0;
}
```

# Pengertian Statement dalam Bahasa Pemrograman C

14

- **Statement** dalam bahasa C adalah sebuah baris perintah. Setiap baris perintah (*statement*) dalam bahasa C harus diakhiri dengan tanda titik koma ( ; ).
- Berikut beberapa contoh statement dalam bahasa pemrograman C:

```
#include <stdio.h>

int main() {
    int Luas, P, L;
    printf("\nPROGRAM MENGHITUNG LUAS PERSEGI PANJANG\n");
    printf("\n=====");
    printf("\nMasukkan Panjang :");
    scanf("%d", &P);
    printf("\nMasukkan Lebar :");
    scanf("%d", &L);
    Luas=P*L;
    printf("\nLuas Persegi panjang adalah %d ", Luas);
}
```

```
Select D:\STMIK_PESAT\KULIAH\PRAKTIKUM\ALGORITMA\persegi...
PROGRAM MENGHITUNG LUAS PERSEGI PANJANG
=====
Masukkan Panjang :10
Masukkan Lebar :4
Luas Persegi panjang adalah 40
Process returned 32 (0x20)   execution time : 6.983 s
Press any key to continue.
```

```
contoh.c X persegiPanjang.c X
#include <stdio.h>
int main() {
    int Luas, P, L;
    printf("\nPROGRAM MENGHITUNG LUAS PERSEGI PANJANG\n");
    printf("\n=====");
    printf("\nMasukkan Panjang :");
    scanf("%d", &P);
    printf("\nMasukkan Lebar :");
    scanf("%d", &L);
    Luas=P*L;
    printf("\nLuas Persegi panjang adalah %d ", Luas);
}
```

Dengan menggunakan perintah **scanf**, kita bisa membuat program yang lebih interaktif, yakni meminta data dari user / pengguna. Data ini nantinya bisa disimpan ke dalam variabel dan diolah lebih lanjut untuk kemudian ditampilkan kembali.

Sama seperti **printf**, perintah **scanf** juga merupakan function yang butuh beberapa argumen. Berikut format dasar penggunaan fungsi **scanf**:

Perintah **scanf**, atau lebih tepatnya *function scanf()* adalah perintah bahasa C untuk menerima masukan ke dalam program, yakni sebagai sarana **input** dari pengguna.

**scanf**(kode\_format, &nama\_variabel\_penampung)



# 16

## **scanf**(kode\_format, &nama\_variabel\_penampung)

---

Bagian **kode\_format** adalah format untuk tipe data inputan. Kode format ini sama seperti yang dipakai untuk fungsi **printf**, misalnya kode “%d” untuk tipe data **integer**, atau “%c” untuk tipe data **char**.

Bagian **nama\_variabel\_penampung** adalah nama variabel yang digunakan untuk menampung nilai inputan. Variabel ini harus sudah di deklarasikan sebelumnya.

Perhatikan penambahan tanda ‘&’ diawal variabel penampung. Tanda ini merujuk ke *pointer* untuk alamat memory dari variabel tersebut. Untuk tipe data dasar seperti **int**, **float** dan **char**, tanda ‘&’ harus disertakan. Untuk beberapa tipe data seperti **string**, tidak perlu ditambahkan tanda ‘&’.

Sama seperti **printf**, fungsi **scanf** juga bukan bagian dari inti bahasa C, tapi berasal dari library **stdio.h**. Karena itu kode **#include <stdio.h>** harus ditulis agar bisa menggunakan perintah **scanf**.

## contoh kode program bahasa C untuk penggunaan perintah scanf:

contoh.c x persegiPanjang.c x

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

```
int main(void)
```

```
{
```

```
    int harga;
```

```
    printf("Masukkan harga barang: ");
```

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

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

```
    printf("Harga barang adalah: %d", harga);
```

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

```
    return 0;
```

```
}
```

D:\STMIK\_PESAT\KULIAH\PRAKTIKUM\ALGORITMA\contoh...

Masukkan harga barang: 3000

Harga barang adalah: 3000

Process returned 0 (0x0) execution time : 6.939 s

Press any key to continue.

```
contoh.c x persegiPanjang.c x
#include <stdio.h>

int main(void)
{
    int harga;

    printf("Masukkan harga barang: ");
    scanf("%d", &harga);

    printf("\n");
    printf("Harga barang adalah: %d", harga);
    printf("\n");

    return 0;
}
```

Terakhir, nilai dari variabel harga ini saya tampilkan kembali dengan perintah **printf("Harga barang adalah: %d", harga)**. Berikut contoh hasilnya:

Di awal kode program, saya mendefinisikan 1 buah variabel **harga** yang diset sebagai **int**. Artinya, variabel harga hanya bisa diisi dengan angka bulat.

Selanjutnya terdapat baris **printf("Masukkan harga barang: ")**. Ini digunakan untuk menampilkan teks sebagai keterangan agar pengguna menginput sesuatu.

Proses pembacaan data di tangani oleh perintah **scanf("%d",&harga)**. Disini, cursor akan berhenti dan menunggu kita menginput suatu nilai. Nilai ini akan disimpan ke dalam variabel **harga**. Tanda **"%d"** adalah sebagai kode format kalau nilai inputan harus berupa **integer**.

```
D:\STMIK_PESAT\KULIAH\PRAKTIKUM\ALGORITMA\contoh...
Masukkan harga barang: 3000

Harga barang adalah: 3000

Process returned 0 (0x0)   execution time : 6.939 s
Press any key to continue.
```

```

int main(void)
{
    int harga;
    float nilai_ip;
    char huruf;

    printf("Harga barang: ");
    scanf("%d", &harga);

    printf("Nilai IP: ");
    scanf("%f", &nilai_ip);

    printf("Huruf pertama nama anda: ");
    scanf("%c", &huruf);

    printf("\n");
    printf("harga = %d , nilai_ip = %f dan huruf = %c",
           harga, nilai_ip, huruf);
    printf("\n");

    return 0;
}

```

Yang juga harus menjadi catatan, ketika karakter yang diinput bukan angka, bahasa C akan mengkonversi karakter tersebut. Misalnya diinput angka 2500.25 (pecahan), yang akan di ambil hanya angka 2500 saja. Karena variabel harga hanya bisa menampung angka bulat.

Kode program ini mirip seperti sebelumnya, hanya saja kali ini saya membuat 3 variabel bertipe **int**, **float** dan **char**. Setelah itu terdapat 3 perintah **scanf** untuk menerima input untuk ketiga variabel ini.

```
Harga barang: 3000  
Nilai IP: 70  
Huruf pertama nama anda:  
harga = 3000 , nilai_ip = 70.000000 dan huruf =  
  
Process returned 0 (0x0)   execution time : 4.204 s  
Press any key to continue.
```

- Akan tetapi, jika anda menjalankan kode program diatas, terdapat 1 masalah. Ketika kita menekan tombol enter setelah menginput angka untuk variabel **nilai\_ip**, program langsung berakhir tanpa sempat berhenti untuk menerima input untuk variabel **huruf**.
- Hal ini terjadi karena karakter “Enter” akan dibaca sebagai inputan untuk variabel **huruf** yang di set sebagai **char**.
- Solusinya, tambahkan 1 spasi di dalam baris **scanf(“%c”,&huruf)** menjadi:

```
int main(void)
{
    int harga;
    float nilai_ip;
    char huruf;

    printf("Harga barang: ");
    scanf("%d", &harga);

    printf("Nilai IP: ");
    scanf("%f", &nilai_ip);

    printf("Huruf pertama nama anda: ");
    scanf(" %c", &huruf);

    printf("\n");
    printf("harga = %d , nilai_ip = %f dan huruf = %c",
           harga, nilai_ip, huruf);
    printf("\n");

    return 0;
}
```

Beri spasi

D:\STMIK\_PESAT\KULIAH\PRAKTIKUM\ALGORITMA\contoh2.exe

Harga barang: 5000

Nilai IP: 34.22

Huruf pertama nama anda: E

harga = 5000 , nilai\_ip = 34.220001 dan huruf = E

Process returned 0 (0x0) execution time : 10.959 s

Press any key to continue.

# 22 Pengertian Keyword dalam Bahasa Pemrograman C

---

- **Keyword** adalah kata kunci yang menjadi dasar perintah bahasa C. Keyword ini tidak bisa digunakan sebagai **identifier** (*variabel, konstanta* maupun nama dari sebuah *fungsi*).
- Sebagai contoh, perhatikan statement berikut:

```
1 | int jumlah;  
2 | return 0;
```

Disini, kata **int** dan **return** merupakan *keyword* dan memiliki makna tertentu di dalam bahasa C.

Terdapat beberapa kata kunci (*keyword*) dalam bahasa pemrograman C, yakni:



# 23

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

# 24 Pengertian Identifier dalam Bahasa Pemrograman C

---

- **Identifier** adalah nama untuk menandakan “sesuatu” sepanjang kode program. Contoh dari **identifier** adalah *variabel*, *konstanta* dan *fungsi*.

- Perhatikan statement berikut:

```
int jumlah;  
double total_penjualan;
```

- Disini, **jumlah** dan **total\_penjualan** adalah *variabel* dan termasuk ke dalam **identifiers**.

- 
- Secara umum, kita bebas ingin menulis nama **identifier**, namun terdapat beberapa aturan:
    1. **Identifier** harus selain dari *keyword* (yang terdapat di tabel sebelum ini). Sebagai contoh, kita tidak bisa memakai kata **int** sebagai nama variabel, karena **int** merupakan *keyword* untuk menandakan tipe data *integer*.
    2. **Identifier** bisa terdiri dari huruf, angka dan karakter *underscore* / garis bawah ( \_ ).
    3. Karakter pertama dari **identifier** hanya bisa berupa huruf dan underscore, tidak bisa berupa angka. Meskipun boleh, tapi tidak disarankan menggunakan karakter underscore sebagai awal dari identifier.
    4. Beberapa compiler ada yang membatasi panjang **identifier** maksimal 31 karakter.

# 26 Tugassssssssssssssssssssss

---

- Buatlah program untuk mencetak perhitungan sederhana !
- Buatlah satu program (meng – inputkan) data, program bebas dengan catatan tidak boleh sama antara satu dengan yang lain !