



MODUL #05

PEMROGRAMAN TERSTRUKTUR

IDENTIFIER (PENGENAL) PADA C++

NUR ALAMSYAH

IDENTIFIER

identfier adalah suatu pengenalan atau pengidentifikasi yang kita deklarasikan agar compiler dapat mengenalinya

Mengenal Jenis-jenis Identifier di dalam C++

Identifier dapat berupa nama variabel, konstanta, fungsi, kelas, template, maupun namespace.

Pada bagian ini kita akan membahas tentang identifier yang berperan sebagai **variabel** dan **konstanta** saja

Dalam menentukan atau membuat *identifier* dalam program, harus memperhatikan hal-hal berikut:

- Karena bahasa C++ bersifat *case sensitive*, maka C++ akan membedakan variabel yang ditulis dengan huruf KAPITAL dan huruf kecil. harga
Harga
HARGA

- Identifier tidak boleh berupa angka atau diawali dengan karakter yang berupa angka

Contoh:

```
long 1000; // SALAH karena identifier berupa angka
long 2X;  // SALAH karena identifier diawali oleh
           //oleh karakter berupa angka
long X2;  // BENAR karena identifier tidak diawali angka
```

- Identifier tidak boleh mengandung spasi

Contoh :

```
int Bilangan Bulat;      // SALAH, karena mengandung spasi
int Bilangan_Bulat      // BENAR
int BilanganBulat        // BENAR
int _BilanganBulat       // BENAR
```

- Identifier tidak boleh menggunakan karakter-karakter simbol (#, @, ?, !, \$, dll)

```
long !satu;              // SALAH
long dua@;               // SALAH
long ti#ga;              // SALAH
```

- Identifier tidak boleh mengandung spasi

Contoh :

```
int Bilangan Bulat;      // SALAH, karena mengandung spasi
int Bilangan_Bulat      // BENAR
int BilanganBulat        // BENAR
int _BilanganBulat       // BENAR
```

- Identifier tidak boleh menggunakan karakter-karakter simbol (#, @, ?, !, \$, dll)

```
long !satu;              // SALAH
long dua@;               // SALAH
long ti#ga;              // SALAH
```

- Identifier tidak boleh menggunakan kata kunci (keyword) yang terdapat pada C++

```
long break;              // SALAH
                          // karena menggunakan kata kunci break
long return;             // SALAH
                          // karena menggunakan kata kunci return
```

- Nama identifier sebaiknya disesuaikan dengan kebutuhannya, artinya jangan sampai orang lain bingung hanya karena salah dalam penamaan identifier.

Berdasarkan jenisnya identifier sendiri dibagi menjadi dua bagian yaitu

1. **konstanta**
2. **variabel**

1. Konstanta

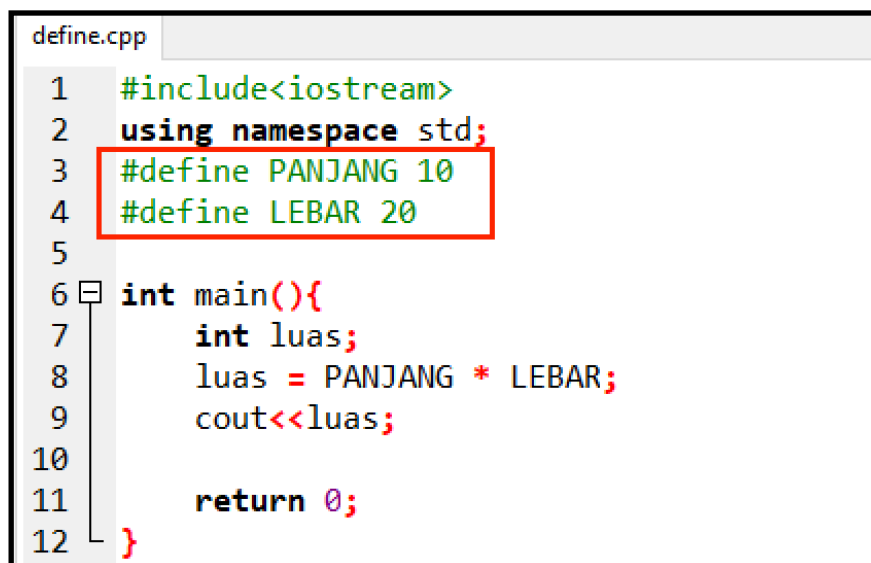
Konstanta adalah jenis identifier yang bersifat konstan atau tetap, artinya nilai dari konstanta didalam program tidak dapat diubah. Konstanta berguna untuk menentukan nilai yang merupakan tetapan, misalnya nilai pi (π), kecepatan cahaya dan lainnya.

Dalam bahasa C++, terdapat dua buah cara untuk membuat konstanta, yaitu:

1. Dengan menggunakan *preprocessor directive* `#define`
2. Menggunakan kata kunci `const`.

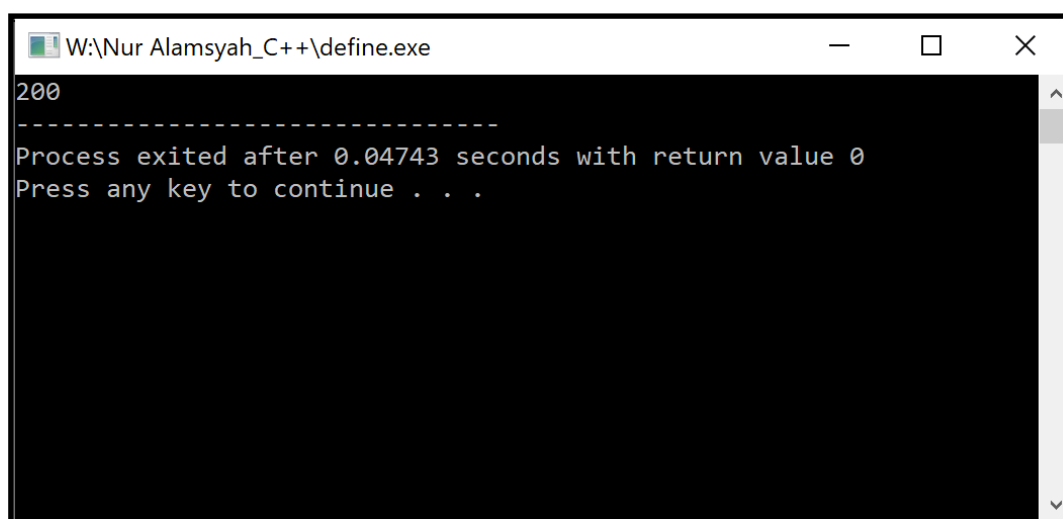
Menggunakan Preprocessor directive **#define**

Praktikkan: Buat Folder baru beri nama Pertemuan 5 Identifier, Buat file baru bernama **"define.cpp" dan simpan difolder**



```
define.cpp
1  #include<iostream>
2  using namespace std;
3  #define PANJANG 10
4  #define LEBAR 20
5
6  int main(){
7      int luas;
8      luas = PANJANG * LEBAR;
9      cout<<luas;
10
11     return 0;
12 }
```

Hasil Tampilan:



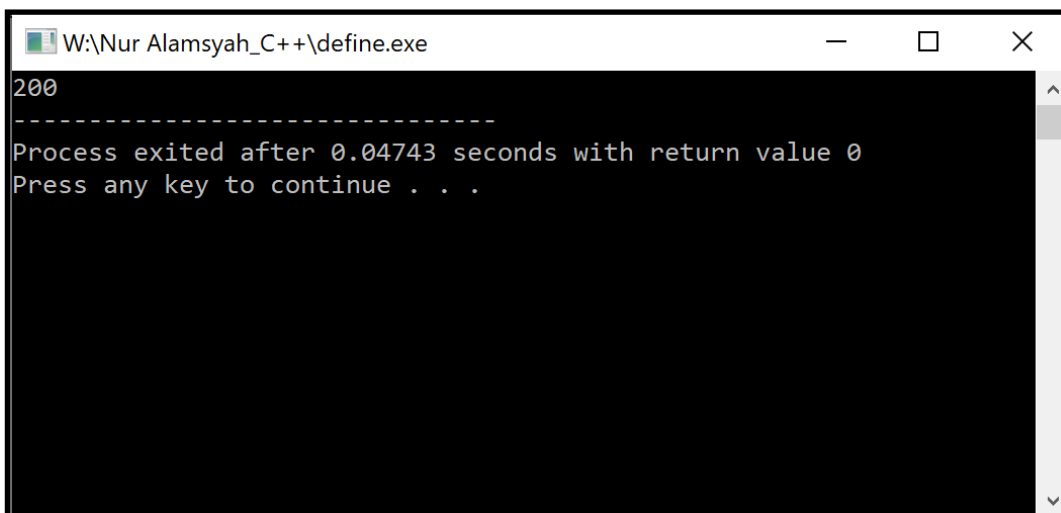
```
W:\Nur Alamsyah_C++\define.exe
200
-----
Process exited after 0.04743 seconds with return value 0
Press any key to continue . . .
```

Menggunakan *const*

Praktikkan: Buat file baru bernama "*const.cpp*" dan simpan difolder

```
const.cpp
1  #include<iostream>
2  using namespace std;
3
4  int main(){
5      const int PANJANG = 10;
6      const int LEBAR = 20;
7
8      int luas = PANJANG * LEBAR;
9      cout<<luas;
10
11     return 0;
12 }
```

Hasil Tampilan:



```
W:\Nur Alamsyah_C++\define.exe
200
-----
Process exited after 0.04743 seconds with return value 0
Press any key to continue . . .
```

2. Variabel

Berbeda dengan konstanta yang mempunyai nilai tetap, **variabel** adalah sebuah identifier (pengenal) yang mempunyai nilai dinamis. Arti kata 'dinamis' disini bermaksud bahwa nilai variabel tersebut dapat kita ubah sesuai kebutuhan dalam program.

Berikut bentuk umum pendeklarasian variabel dalam C++.

```
tipe_data nama_variabel
```

Contoh:

```
int x;
```

Atau langsung mendeklarasikan 3 buah variabel

```
int x, y, z ;
```

Inisialisasi Variabel

Dalam konteks ini, **inisialisasi** dapat didefinisikan sebagai **proses pengisian nilai awal** (nilai default) kedalam suatu variabel. Dalam C++, pengisian nilai dilakukan dengan menggunakan operator sama dengan (=).

Bentuk umum yang digunakan untuk melakukan **inisialisasi variabel** adalah sebagai berikut:

```
tipe_data nama_variabel = Nilai awal;
```

Contoh:

```
int x = 7;
```

Atau langsung mendeklarasikan 3 buah variabel

```
int x=7, y=5, z=8;
```

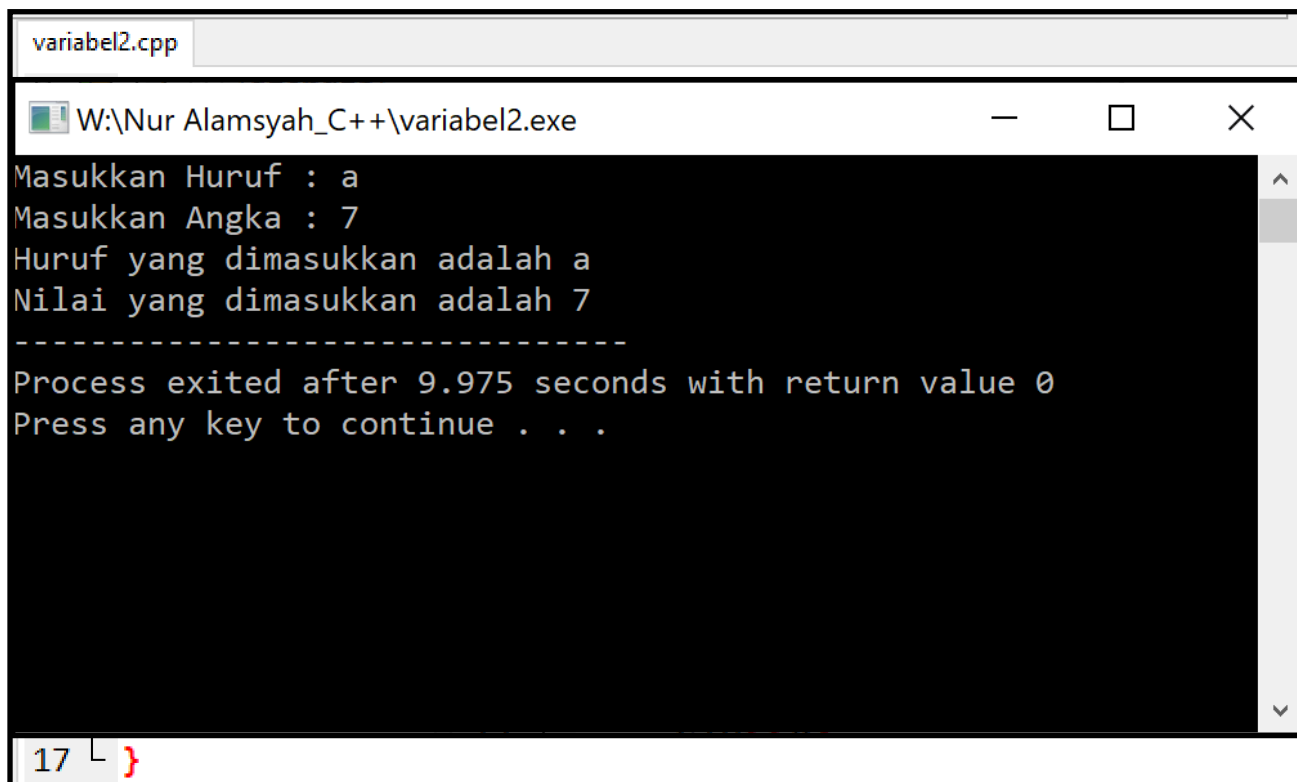
Praktikkan: Buat file baru bernama “variabel.cpp” dan simpan difolder

```
variabel.cpp
1  #include<iostream>
2  using namespace std;
3
4  int main(){
5      char karaktersaya = 'A';
6      int panjang = 20;
7      int lebar = 10;
8      float phi = 3.14;
9      bool a;
10
11     int luas = panjang * lebar;
12     cout<<luas;
13
14     return 0;
15 }
```

Hasil Tampilan:

```
W:\Nur Alamsyah_C++\define.exe
200
-----
Process exited after 0.04743 seconds with return value 0
Press any key to continue . . .
```

Praktikkan: Buat file baru bernama “variabel2.cpp” dan simpan difolder



```
variabel2.cpp
W:\Nur Alamsyah_C++\variabel2.exe
Masukkan Huruf : a
Masukkan Angka : 7
Huruf yang dimasukkan adalah a
Nilai yang dimasukkan adalah 7
-----
Process exited after 9.975 seconds with return value 0
Press any key to continue . . .
17 }
```

Hasil Tampilan:

Variabel Global vs Variabel Lokal

Berdasarkan ruang lingkupnya, variabel dibedakan menjadi dua: **global dan lokal**. Penentuan variabel untuk dijadikan sebagai variabel global atau lokal tergantung dari kasus program yang dihadapi.

Variabel Global

Kita telah mengetahui bahwa dalam bahasa C++ selalu terdapat fungsi utama `main()`. Apabila kita mendeklarasikan sebuah **variabel diluar fungsi `main()` (atau fungsi lain)**, maka dengan sendirinya compiler akan menganggap variabel tersebut sebagai **variabel global**.

Variabel Lokal

Berbeda dengan variabel global, **variabel lokal** adalah variabel yang hanya dikenal oleh suatu fungsi saja. Proses deklarasi **variabel lokal dilakukan didalam lingkup fungsi** yang dimaksud.

Contoh variabel Global

Praktikkan: Buat file baru bernama “variabel_global.cpp” dan simpan difolder

```
variabel_global.cpp
1  #include<iostream>
2  using namespace std;
3
4  int x;
5
6  //kita membuat 2 buah fungsi utama yaitu fungsi test() dan fungsi main()
7  //membuat fungsi bernama test()
8  void test(){
9      x = 20;
10     cout<<"Nilai x didalam fungsi test() adalah : "<<x<<endl;
11 }
12
13 //membuat fungsi bernama main()
14 int main(){
15     x = 10;
16     cout<<"Nilai x didalam fungsi main() adalah : "<<x<<endl;
17
18     //memanggil fungsi test()
19     test();
20
21     return 0;
22 }
```

Hasil

Tampilan:

```
W:\Nur Alamsyah_C++\variabel_global.exe
Nilai x didalam fungsi main() adalah : 10
Nilai x didalam fungsi test() adalah : 20

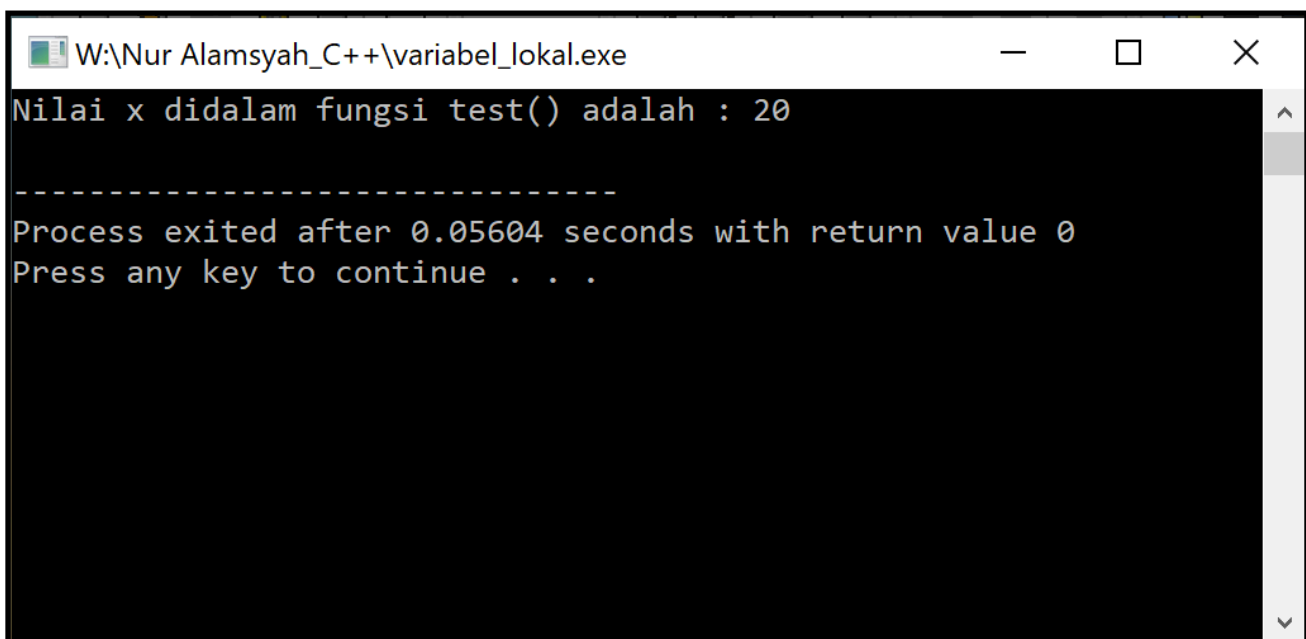
-----
Process exited after 0.04644 seconds with return value 0
Press any key to continue . . .
```

Contoh variabel lokal

Praktikkan: Buat file baru bernama **"variabel_lokal.cpp"** dan simpan difolder

```
variabel_lokal.cpp
1  #include<iostream>
2  using namespace std;
3
4  //kita membuat 2 buah fungsi utama yaitu fungsi test() dan fungsi main()
5  //membuat fungsi bernama test()
6  void test(){
7      int x; // variabel x ini hanya dikenal oleh fungsi test()
8
9      x=20;
10     cout<<"Nilai x didalam fungsi test() adalah : "<<x<<endl;
11 }
12
13 //membuat fungsi bernama main()
14 int main(){
15     // x tidak dapat digunakan di fungsi main()
16     //cout<<"Nilai x didalam fungsi main() adalah : "<<x<<endl;
17
18     //memanggil fungsi test()
19     test();
20
21     return 0;
22 }
23
```

Hasil Tampilan:



```
W:\Nur Alamsyah_C++\variabel_lokal.exe
Nilai x didalam fungsi test() adalah : 20
-----
Process exited after 0.05604 seconds with return value 0
Press any key to continue . . .
```