

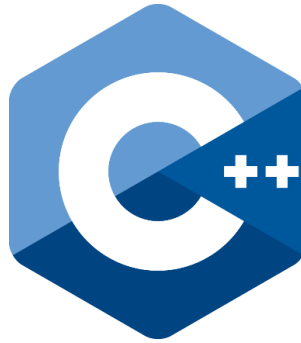


Modul 2



Pemrograman Terstruktur

Menggunakan



Dasar Input dan Output pada C++
Komentar, Identifier dan Tipe data

Oleh
NUR ALAMSYAH

**PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS ISLAM KALIMANTAN
MUHAMMAD ARSYAD AL BANJARI
BANJARMASIN
2019**



BAB IV

DASAR INPUT & OUTPUT (I/O) PADA C++

4.1 Output pada C++ (sintaks: cout)

Dalam pemrograman C++ kita akan sering menggunakan sintaks `cout` untuk menampilkan data ke alat output khususnya data berupa teks.

Untuk menggunakan keyword `cout` kita membutuhkan *Insertion Operator*:

yaitu 2 buah tanda lebih kecil `<<` di antara keyword dengan ekspresi.

Operator Insertion `<<` (2 buah tanda kurang dari) untuk `cout`

Berikut contoh programnya:

```
1 #include <iostream>
2 using namespace std;
3
4 int main()
5 {
6     cout << "Welcome to C++";
7
8     return 0;
9 }
```

4.2 Input pada C++ (sintaks: cin)

Jika fungsi `cout` kita gunakan untuk menampilkan output maka bisa ditebak bahwa fungsi `cin` akan kita gunakan untuk melakukan perintah input dasar text.

Untuk menggunakan keyword `cin` kita membutuhkan *Extraction Operator* dengan 2 buah tanda lebih dari `>>` yang diletakan di antara keyword `cin` dan memori.

Operator Extraction `>>` (2 buah tanda lebih dari) untuk `cin`.

Untuk itu, **kita harus mendeklarasikan variabel terlebih dahulu.**

Karena nantinya, data yang diberikan oleh pengguna dalam operasi pemasukan (input) akan disimpan di dalam memori variable yang telah kita deklarasikan.

Silahkan perhatikan baris kode berikut:

```
1 #include <iostream>
2 using namespace std;
3     tipe data     Variabel
4 string nama;
5
6 int main()
7 {
8     cout << "Masukan Nama: ";
9     cin >> nama;
10
11     cout << "Nama saya adalah " << nama;
12 }
```

Dari contoh program diatas,kita telah mendeklarasikan variabel `nama` bertipe `string`. Sedangkan `cin` untuk meyimpan nilai masukan dari pengguna ke variabel `nama`.

Lalu nilai/value dari variabel `nama` ditampilkan dengan menggunakan fungsi `cout`.

BAB V

KOMENTAR DAN IDENTIFIER

5.1 Macam-Macam Komentar

Komentar adalah catatan atau dokumenasi yang ditulis oleh programmer untuk sebagai pengingat atau penjelasan ketika membaca sebuah baris kode.

Dalam bahasa C++ ada dua tanda yang dapat di gunakan untuk sebuah komentar:

1. Yang pertama ada tanda yang di gunakan untuk komentar satu baris saja, untuk tanda komentar satu baris, menggunakan tanda dua garis miring `//`.
2. dan yang ke dua tanda yang di gunakan untuk komentar lebih dari satu baris. Sedangkan untuk komentar lebih dari satu baris, menggunakan tanda `/*...*/`.

Perhatikan contoh berikut:

Source Code
<pre>//ini kode satu baris /* ini komentar beberapa baris */</pre>

5.2 Pengertian, Jenis dan Contoh Identifier dalam C++

Identifier adalah **suatu pengenalan** atau pengidentifikasi yang kita deklarasikan agar compiler dapat mengenalinya.

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.

Identifier yang berperan sebagai variabel dan konstanta berfungsi untuk menampung sebuah nilai yang digunakan dalam program. dilakukan untuk mempermudah proses penanganan data atau nilai, misalnya untuk memasukkan dan menampilkan nilai, sebagai gambaran, dibawah ini adalah contoh program yang menggunakan dua buah identifier didalamnya.

```

identifier.cpp
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      char teks[20];
7      int x;
8      cout << "Masukan sebuah kata      : "; cin>>teks;
9      cout << "Masukan sebuah angka    : "; cin>>x;
10     cout<<teks<<endl;
11     cout<<x;
12
13     return 0;
14 }

```

Pada program diatas kita mempunyai **dua buah identifier**, yaitu **teks** dan **x**. Pada saat program dijalankan, identifier tersebut akan digunakan untuk menyimpan nilai yang dimasukkan dari keyboard.

Dalam C++ sendiri dalam **pembuatan Identifier** kita dapat menuliskannya dengan karakter sebagai berikut:

- Huruf "a" sampai "z"
- Huruf "A" sampai "Z"
- Bilangan antara "0" sampai "9"
- Underscore " _ "

6 Ketentuan membuat Identifier

Dalam menentukan atau membuat identifier(pengenal) dalam program, kita harus memperhatikan hal-hal berikut:

1. Case sensitive

Karena bahasa C++ bersifat case sensitive, maka C++ juga akan membedakan identifier yang ditulis dengan huruf kapital dan huruf kecil.

Misalnya identifier **X** tentunya akan berbeda dengan identifier **x**.

- Harga
 - HARGA
 - harga
- } **Tiga variabel berbeda**

2. Tidak boleh diawali dengan Angka

Identifier tidak boleh diawali dengan karakter yang berupa angka, berikut contohnya:

```
long 1000; // SALAH karena identifier berupa angka
long 2x;   // SALAH karena identifier diawali angka
long x2;   // BENAR karena identifier tidak diawali angka
```

3. Tidak menggunakan spasi

Identifier tidak boleh mengandung spasi, biasanya spasi diganti dengan underscore "_", berikut contohnya:

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

4. Tidak menggunakan karakter simbol

Identifier tidak boleh menggunakan karakter-karakter simbol (#,\$,%,^,!,@,?, dll), berikut contohnya:

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

5. Tidak menggunakan kata kunci (keyword)

Identifier tidak boleh menggunakan kata kunci (keyword) yang terdapat pada C++, berikut contohnya:

```
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** dibagi menjadi dua bagian yaitu **konstanta** dan **variabel**.

5.2.1 Konstanta

Konstanta adalah jenis identifier yang bersifat konstan atau tetap, artinya nilai dari konstanta didalam program tidak dapat diubah.

Nilai akan ditetapkan saat sebelum kompilasi program, data tersebut tidak akan bisa diubah sepanjang kode sumber program dan di saat program tersebut sedang berjalan (runtime).

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

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

Contoh source code menggunakan **#define**

```
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 }
```

Perlu diperhatikan bahwa penggunaan **#define** tidak diakhiri dengan tanda titik koma (;).

Contoh source code menggunakan **const**

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

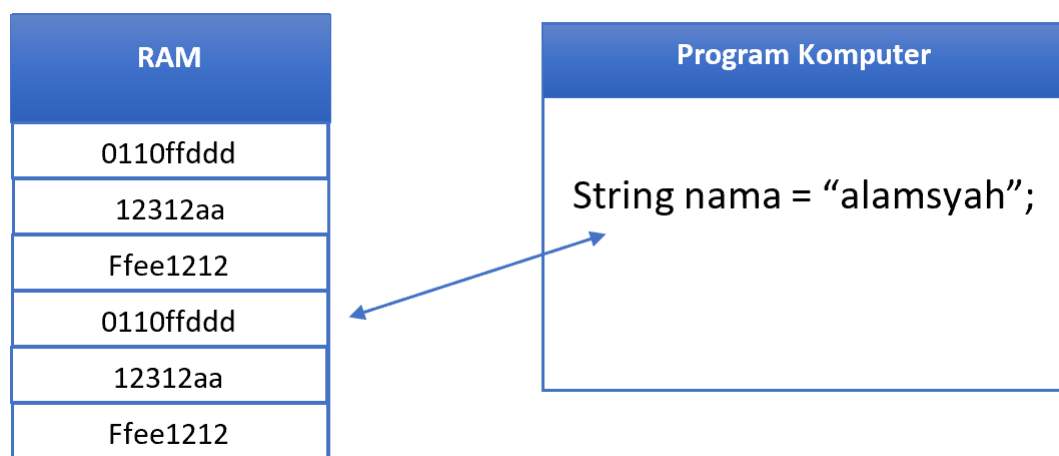
5.2.2 Variabel

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

Semua program komputer yang sedang berjalan akan menyimpan data sementara di dalam **RAM** (Random Access Memori).

Bagaimana cara program menyimpan nilai ke RAM? Jawabannya dengan menggunakan variabel.

Semakin banyak variabel yang kamu buat semakin besar pula memori yang akan digunakan di dalam RAM.



Jadi dapat disimpulkan, variabel adalah sebuah nama lokasi penyimpanan di dalam memori.

Berikut bentuk umum pendeklarasian variabel dalam C++.

tipe_data nama_variabel;

Contoh:

```
int A;
```

Atau

```
int A;  
int B;  
int C;
```

bisa juga seperti ini `int A, B, C;` asal memiliki tipe data sama

Inisialisasi Variabel

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

```
int A = 20;
```

Atau bisa juga seperti ini

```
int A=20, B=30, C=40;
```

Contoh 1 source code inisialisasi variabel

```
variabel.cpp
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      int x;
6      cout<<" nilai x sebelum diberi nilai : "<< x <<endl;
7      x=20;
8      cout<<" nilai x setelah diberi nilai : "<< x <<endl;
9      x=80;
10     cout<<" nilai x setelah diberi nilai baru lagi : "<< x <<endl;
11
12     return 0;
13 }
```

Contoh 2 source code inisialisasi variabel

```
inisialisasi_var.cpp
4
5  int main ()
6  {
7      // Deklarasi variabel bertipe int
8      // Dengan nama ABC
9      // Tanpa Inisialisasi
10     int ABC;
11
12     // Contoh Deklarasi dan Inisialisasi
13     // int ABC = 10;
14
15     // Menampilkan nilai variabel ABC
16     // Sebelum dilakukan Pengisian nilai (Assignment)
17     cout<<"Nilai ABC Sebelum Assignment : "<<ABC<<endl;
18
19     // Mengisi nilai kedalam variabel ABC
20     ABC = 10;
21
22     // Menampilkan nilai variabel ABC
23     // Setelah dilakukan Pengisian nilai (Assignment)
24     cout<<"Nilai ABC Setelah Assignment : "<<ABC<<endl;
25
26     return 0;
27 }
```

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.

```
var_global.cpp
1  #include <iostream>
2  using namespace std;
3
4  int A;
5
6  void test(){
7      A = 20;
8      cout<<"Nilai A didalam fungsi test(): "<<A<<endl;
9  }
10 int main(){
11     A = 10;
12     cout<<"Nilai A didalam fungsi main(): "<<A<<endl;
13     test();
14
15     return 0;
16 }
```

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.

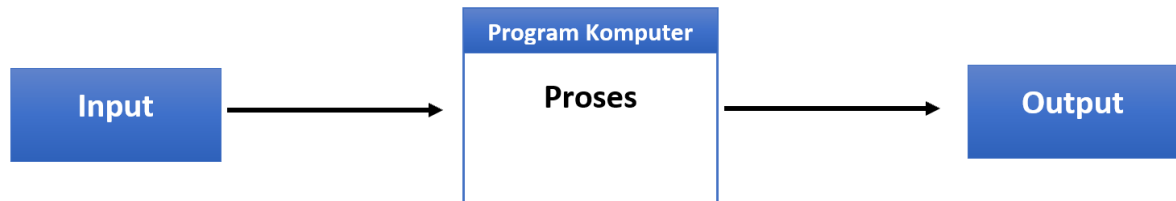
local.cpp

```
1  #include <iostream>
2  using namespace std;
3
4
5
6  void test(){
7      int A;
8      A = 20;
9      cout<<"Nilai A didalam fungsi test(): "<<A<<endl;
10 }
11 int main(){
12     //A = 10;
13     //cout<<"Nilai A didalam fungsi main(): "<<A<<endl;
14     test();
15
16     return 0;
17 }
```

BAB VI

TIPE DATA PADA C++

Inti dari sebuah program komputer adalah menerima input, melakukan pemrosesan, dan menghasilkan output.



Nilai input bisa kita dapatkan dari keyboard, file, kamera, mikrofon, dan sebagainya.

Sementara output dapat kita tampilkan ke monitor, cetak ke dokumen, atau ke dalam sebuah file.

Pada tahap pemrosesan, program membutuhkan bantuan variabel untuk menyimpan nilai sementara.

Sama seperti waktu kita berpikir, kita membutuhkan beberapa ingatan untuk memproses informasi.

Ketika anda mendeklarasikan sebuah variabel atau konstanta, anda harus bisa menentukan tipe data apa yang cocok untuk program kita.

Ketepatan pemilihan tipe data pada variabel atau konstanta akan sangat menentukan pemakaian sumberdaya komputer (terutama memori komputer).

Salah satu tugas penting seorang programmer adalah memilih tipe data yang sesuai untuk menghasilkan program yang efisien dan berkinerja tinggi.

Ada banyak tipe data yang tersedia, tergantung jenis bahasa pemrograman yang anda pakai. Namun secara umum dapat dikelompokkan menjadi dua:

1. **Tipe data primitive** - adalah tipe data dasar yang tersedia secara langsung pada suatu bahasa pemrograman. Sebagai **contoh tipe data integer dan char**.
2. **Tipe data composite** - adalah tipe data bentukan yang terdiri dari dua atau lebih tipe data primitive. **Contohnya tipe data string** (bentukan dari tipe data char), **array** (larik), **struct** (struktur) dan **enum** (enumerasi).

Penjelasan:

Tipe data Primitive

Pada dasarnya tipe data primitive dibagi menjadi 3 jenis:

- **Tipe Data Angka** - untuk angka dan berhubungan dengan aritmetika.
- **Tipe Data Karakter** - untuk karakter dan angka bukan untuk operasi aritmetika.
- **Tipe Data Logika** - untuk logika benar (*true*) atau salah (*false*).

Pada dasarnya C++ memiliki beberapa tipe data built-in yang langsung anda gunakan. Dan berikut 7+ tipe data primitif dalam bahasa pemrograman C++ :

Tipe Data	Keyword
Boolean	bool
Character	char
Integer	int
Floating point	float
Double	double
Valueless	void
Wide character	wchar_t

1. **Bool (Boolean)**
adalah tipe data yang digunakan untuk menentukan false dan true, tipe data ini sama seperti bilangan biner hanya ada dua angka saja yaitu 0 dan 1.
2. **Char (karakter)**
adalah tipe data untuk karakter yang sering digunakan untuk tipe data yang menggunakan huruf dan angka sebagai datanya.
3. **Int (Integer)**
adalah tipe data untuk numerik yang sering digunakan untuk data berupa angka.
4. **Float (Floating Point)**
adalah tipe data untuk numerik yang digunakan untuk data berupa angka pecahan.
5. **Double (Double Floating Point)**
adalah tipe data untuk numerik yang digunakan untuk data berupa angka pecahan yang nilai decimalnya dua kali lebih banyak.

6. Void (Kosong)

adalah tipe data yang tidak bertipe karena mempunyai ukuran 0 byte biasanya digunakan untuk tipe data kosong seperti membuat fungsi.

7. Wchar_t

adalah tipe data yang digunakan sama seperti char yang menyimpan karakter besar, biasanya digunakan untuk menyimpan karakter yang tidak ada di huruf abjad misalnya huruf japan, china, korea, dll.

Beberapa macam tipe data dapat dimodifikasi sesuai kegunaannya dengan menambahkan fungsi ini didepan tipe data :

1. **Short** : memodifikasi jangkauan nilai ke dalam bit yang lebih kecil (16bit).
2. **Long** : memodifikasi jangkauan nilai ke dalam bit yang lebih besar (32bit).
3. **Signed** : memodifikasi jangkauan nilainya negatif dan positif.
4. **Unsigned** : memodifikasi jangkauan nilainya positif di mulai dari nol.

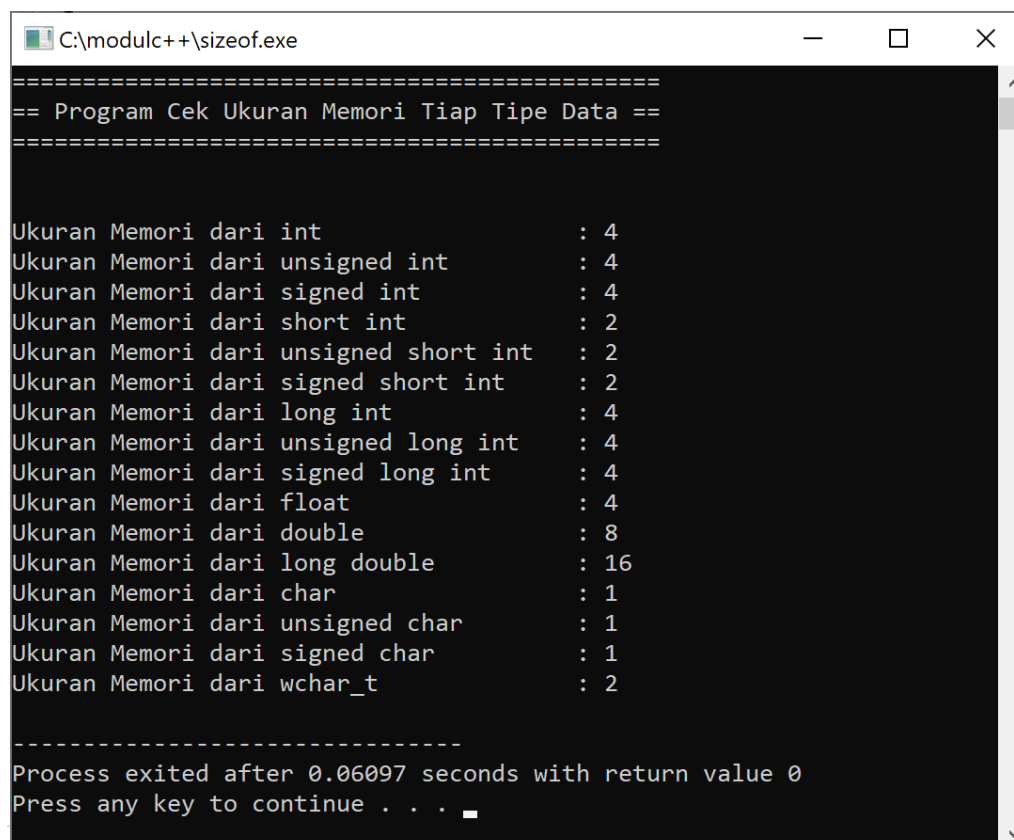
Tipe Data	Ukuran Memori	Jangkauan
char	1 byte	-127 to 127 or 0 to 255
unsigned char	1 byte	0 to 255
signed char	1 byte	-127 to 127
int	4 bytes	-2147483648 to 2147483647
unsigned int	4 bytes	0 to 4294967295
signed int	4 bytes	-2147483648 to 2147483647
short int	2 bytes	-32768 to 32767
unsigned short int	Range	0 to 65,535
signed short int	Range	-32768 to 32767
long int	4 bytes	-2,147,483,648 to 2,147,483,647
signed long int	4 bytes	-2,147,483,648 to 2,147,483,647
unsigned long int	4 bytes	0 to 4,294,967,295
float	4 bytes	+/- 3.4e +/- 38 (~7 digits)
double	8 bytes	+/- 1.7e +/- 308 (~15 digits)
long double	8 bytes	+/- 1.7e +/- 308 (~15 digits)
wchar_t	2 atau 4 bytes	1 wide character

Ukuran Memori dan Jangkauannya

Untuk membuktikannya ukuran memori dan jangkauannya anda bisa memeriksanya dengan fungsi kode `sizeof()`

```
sizeof.cpp
2  using namespace std;
3
4  int main()
5  {
6      cout<<"=====\n";
7      cout<<"== Program Cek Ukuran Memori Tiap Tipe Data ==\n";
8      cout<<"=====\n\n";
9
10     cout<<"Ukuran Memori dari int \t\t\t: "<<sizeof(int)<<endl;
11     cout<<"Ukuran Memori dari unsigned int \t: "<<sizeof(unsigned int)<<endl;
12     cout<<"Ukuran Memori dari signed int \t\t: "<<sizeof(signed int)<<endl;
13     cout<<"Ukuran Memori dari short int \t\t: "<<sizeof(short int)<<endl;
14     cout<<"Ukuran Memori dari unsigned short int \t: "<<sizeof(unsigned short int)<<endl;
15     cout<<"Ukuran Memori dari signed short int \t: "<<sizeof(signed short int)<<endl;
16     cout<<"Ukuran Memori dari long int \t\t: "<<sizeof(long int)<<endl;
17     cout<<"Ukuran Memori dari unsigned long int \t: "<<sizeof(unsigned long int)<<endl;
18     cout<<"Ukuran Memori dari signed long int \t: "<<sizeof(signed long int)<<endl;
19     cout<<"Ukuran Memori dari float \t\t: "<<sizeof(float)<<endl;
20     cout<<"Ukuran Memori dari double \t\t: "<<sizeof(double)<<endl;
21     cout<<"Ukuran Memori dari long double \t\t: "<<sizeof(long double)<<endl;
22     cout<<"Ukuran Memori dari char \t\t: "<<sizeof(char)<<endl;
23     cout<<"Ukuran Memori dari unsigned char \t: "<<sizeof(unsigned char)<<endl;
24     cout<<"Ukuran Memori dari signed char \t\t: "<<sizeof(signed char)<<endl;
25     cout<<"Ukuran Memori dari wchar_t \t\t: "<<sizeof(wchar_t)<<endl;
26
27     return 0;
28 }
```

Hasil tampilan console nya:



```
C:\modulc++\sizeof.exe
====
== Program Cek Ukuran Memori Tiap Tipe Data ==
=====

Ukuran Memori dari int                : 4
Ukuran Memori dari unsigned int        : 4
Ukuran Memori dari signed int          : 4
Ukuran Memori dari short int           : 2
Ukuran Memori dari unsigned short int  : 2
Ukuran Memori dari signed short int    : 2
Ukuran Memori dari long int            : 4
Ukuran Memori dari unsigned long int   : 4
Ukuran Memori dari signed long int     : 4
Ukuran Memori dari float               : 4
Ukuran Memori dari double              : 8
Ukuran Memori dari long double         : 16
Ukuran Memori dari char                : 1
Ukuran Memori dari unsigned char       : 1
Ukuran Memori dari signed char         : 1
Ukuran Memori dari wchar_t             : 2

-----
Process exited after 0.06097 seconds with return value 0
Press any key to continue . . .
```

#1 Tipe Data Boolean (bool)

Boolean adalah salah satu tipe data yang hanya memiliki dua pilihan yaitu **True** (1) atau **False** (0). Tipe data ini biasanya digunakan untuk memberikan kondisi pada program.

...atau bisa juga memastikan kebenaran dari sebuah operasi.

Besarnya memori yang dibutuhkan tipe data **bool** yaitu 1 byte atau 8 bit.

Berikut ini contoh program C++ menggunakan tipe data bool:

```
bool.cpp
1  #include <iostream>
2  using namespace std;
3
4  int main()
5  {
6      int angka;
7      bool hasil;
8      cout << "Masukan angka = "; cin >> angka;
9      hasil = angka > 10;
10     cout << hasil;
11 }
```

Pada contoh program diatas, kita menggunakan 2 buah variabel yaitu variabel **angka** dengan tipe data integer, dan variabel **hasil** dengan tipe data boolean.

Nah, disini saya akan mengambil nilai/value untuk variabel **hasil** dengan membandingkan nilai pada variabel **angka** terhadap bilangan 10.

Apabila nilai pada variabel **angka** lebih dari 10 maka **hasil** bernilai 1 (true) dan jika **angka** lebih kecil dari 10 maka **hasilnya** bernilai 0 (false).

#2 Tipe Data Character (char)

Character adalah salah satu tipe data yang memungkinkan kita untuk memesan memori berformat text (huruf, angka, dan simbol) dengan karakter tunggal.

Besarnya memori yang dibutuhkan tipe data **char** yaitu 1 byte atau 8 bit.

Perlu diingat bahwa tipe data **char** hanya dapat menyimpan data berbentuk karakter dan hanya satu karakter, oleh karena itu apabila anda memasukan lebih dari 1 karakter maka nilai yang akan tersimpan hanya karakter pertama.

Berikut ini contoh program C++ menggunakan tipe data char:


```

char.cpp
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      char nilai;
6
7      cout << "Masukan nilai (A/B/C/D): "; cin>>nilai;
8      cout << "Nilai anda:" << nilai;
9
10     return 0;
11
12 }

```

#3 Tipe Data Integer (int)

Integer adalah salah satu tipe data numerik yang memungkinkan kita untuk menyimpan data dalam bentuk bilangan bulat.

Besarnya memori yang dibutuhkan tipe data **int** yaitu 4 byte atau 32 bit. Berikut ini contoh program C++ menggunakan tipe data int:

```

int.cpp
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      int x,y,z;
6      x=3; y=4;
7
8      z=x*y;
9      cout << "Hasil perkalian: " << z;
10
11     return 0;
12 }

```

Dengan menggunakan tipe data integer hal ini memungkinkan kita untuk melakukan sejumlah operasi aritmetika seperti perkalian dan lain sebagainya.

Pada contoh diatas, saya menggunakan 3 buah variabel beripe integer sebagai berikut: **x** bernilai 3, **y** bernilai 4, dan **z** sebagai hasil hasil perkalian x dan y.

#4 Tipe Data Floating Point (float)

Floating Point adalah tipe data numerik yang memungkinkan untuk menyimpan nilai dalam memori bersifat bilangan pecahan atau real, maupun eksponensial.

Besarnya memori yang dibutuhkan tipe data **float** yaitu 4 byte atau 32 bit. Berikut ini contoh program C++ menggunakan tipe data float:

```

float.cpp
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      float jari, hasil ;
6      const float p=3.14;
7
8      cout << "Masukan Jumlah jari-jari = "; cin >> jari;
9      hasil = (jari * p) * 2;
10
11     cout << "Keliling dari Lingkaran adalah " << hasil;
12
13     return 0;
14
15 }

```

#5 Tipe Data Double Floating Point (double)

Double Floating Point sama seperti float yaitu salah satu tipe data yang bersifat menyatakan bilangan pecahan atau real, maupun eksponensial.

Bedanya adalah penyimpanan angka maksimal lebih besar daripada float dan otomatis double juga akan membutuhkan memori yang lebih besar.

Besarnya memori yang dibutuhkan tipe data **double** yaitu 8 byte atau 64 bit. Berikut ini contoh program C++ menggunakan tipe data double:

```

double.cpp
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      double jari, hasil ;
6      const double p=3.1428;
7
8      cout << "Masukan Jumlah jari-jari = "; cin >> jari;
9      hasil = jari*(jari * p);
10     cout << "Luas lingkaran: " << hasil;
11
12     return 0;
13 }

```

#6 Tipe Data String (string)

String merupakan tipe data text (huruf, angka, dan simbol) yang memungkinkan kita menyimpan nilai dengan bentuk text, kumpulan dari character.

Besarnya memori yang dibutuhkan tipe data **string** yaitu 4 byte atau 32 bit. Berikut ini contoh program C++ menggunakan tipe data string:

```
string.cpp
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      string nohp;
6
7      cout << "Masukan nomor HP: "; cin >> nohp;
8      cout << "Nomor HP anda: " << nohp;
9  }
```

Sama seperti halnya tipe data char, dalam tipe data string kita bisa menggunakan karakter dan angka dengan ketentuan tidak dapat dilakukan operasi aritmetika.

Namun perbedaannya, jika dalam tipe data char kita hanya mampu menyimpan nilai satu karakter untuk tiap variabel, hal ini tidak berlaku pada tipe data string.

Hal ini di karenakan **String** merupakan tipe data composite.

#7 Tipe Data Valueless (void)

Valueless adalah salah satu tipe data yang berarti “tidak ada” atau “tidak mempunyai tipe data”. Namun disini kita belum akan membahasnya lebih detail.

Void termasuk katagori tipe data namun kita tidak bisa menggunakannya pada variabel biasa, void biasanya digunakan pada **function** yang tidak mempunyai return value.

Besarnya memori yang dibutuhkan tipe data **void** yaitu 1 byte atau 8 bit.

Membuat Porgram C++ dengan Variabel dan Tipe Data

Silahkan buat sebuah file baru bernama **biodata.cpp**, kemudian isi dengan kode berikut:

biodata.cpp

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      // deklarasi tipe data variabel
6      string nama;
7      int umur;
8      char jenis_kelamin;
9
10     // --- proses input ---
11     cout << "Siapakah namamu?" << endl;
12     cout << "jawab: ";
13     // menyimpan data ke variabel
14     getline(cin,nama);
15
16     cout << "Berapa umurmu?" << endl;
17     cout << "jawab: ";
18     // menyimpan data ke variabel
19     cin >> umur;
20
21     cout << "Jenis kelamin [L/P]: ";
22     // menyimpan data ke variabel
23     cin >> jenis_kelamin;
24
25     // --- proses output ---
26     cout << "Salam kenal, " << nama << " Sekarang engkau berusia ";
27     cout << umur << " dan kau berjenis kelamin "<< jenis_kelamin;
28
29     return 0;
30 }
```