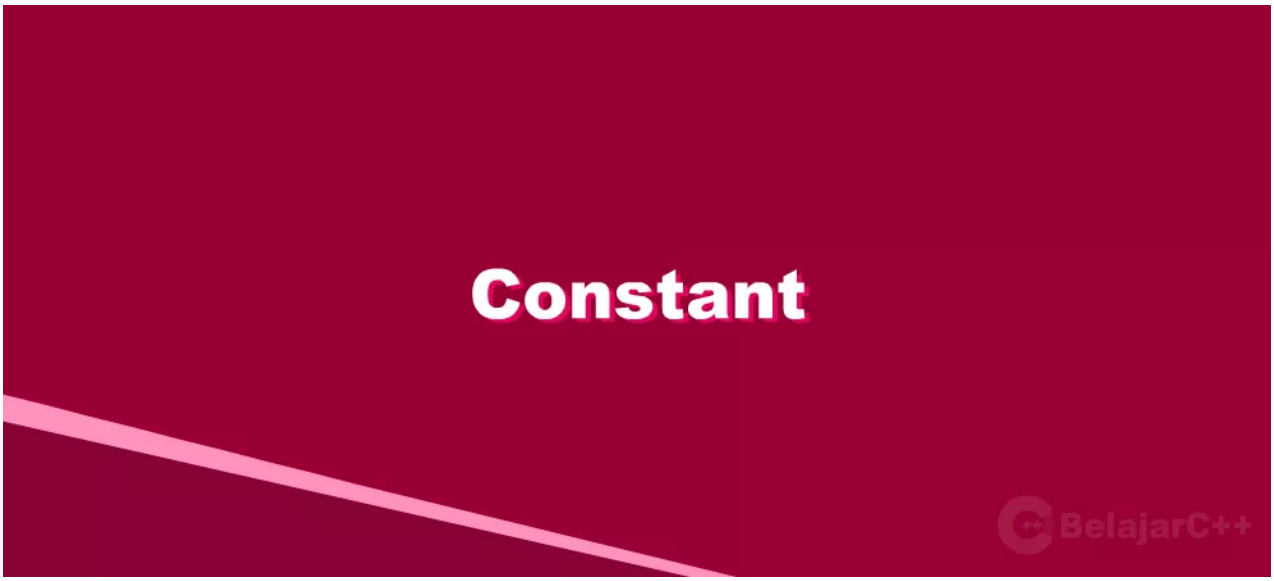
Konstanta

Posted on Januari 22, 2018 | Last Modified Januari 22, 2018



Pengertian Konstanta

Konstanta adalah tetapan dari suatu nilai yang bersifat tetap. Nilai atau data akan ditetapkan saat sebelum kompilasi program , data tersebut tidak akan bisa diubah sepanjang kode sumber program dan di saat program tersebut sedang berjalan (runtime). Tidak bisa diubah oleh pengguna maupun programmer itu sendiri.

Pengertian Literal

Literal adalah jenis yang paling jelas dari konstanta. Mereka digunakan untuk mengekspresikan nilai-nilai tertentu dalam kode sumber dari sebuah program.

```
x = 2 ;
```

nilai 2 adalah merupakan sebuah literal constant. Dan dibawah ini merupakan macam-macam Literal Constant.

Integer Literal

Integer Literal adalah bilangan konstanta yang mengidentifikasi nilai-nilai integer. Dalam penulisan Integer Literal tidak membutuhkan dua buah tanda petik dua (“) seperti apa yang dilakukan pada string. Untuk melakukan Integer Literal dilakukan dengan hanya menulis angka dari nilai yang dimaksud, contoh : 123, angka tersebut akan dibaca sebagai decimal base, yang ditulis tanpa menggunakan special karakter (seperti penggunaan tanda petik dua di awal dan di akhir nilai).

Selain angka Desimal, C ++ juga memungkinkan untuk penggunaan nomor Oktal “Basis 8” yang penulisanya di awali dengan “0”, Hexadecimal “Basis 16” yang penulisanya diawali dengan “0x”, dan Biner(C++14) “Basis 2” yang penulisanya diawali dengan “0b”.

Prefix	Sistem Bilangan
	Desimal
0	Oktal
0x	Hexsadesimal
0b	Biner

Contoh Penulisan

```
75            //Decimal
0113          //Octal
0x4d          //Hexadecimal
0b110101      //Biner
```

Contoh Program

```
int main ()
{
    int x = 75;   //Decimal
    x = 0113;     //Octal
    x = 0x4d;     //Hexadecimal
    x = 0b110101; //Biner

    return 0;
}
```

Integer Literal juga dapat memiliki akhiran (Suffix) yang menyatakan type modifikasi . tapi untuk pemberian konstanta biasanya hal ini dapat dimengerti otomatis oleh kompiler tanpa menggunakan akhiran tersebut. di bawah ini adalah beberapa akhiran yang dapat digunakan dalam Integer Literal:

Suffix	Keterangan
u atau U	Unsigned Int
l atau L	Long Int
ul atau (Ul, uL, UL, lu, Lu, lU, LU)	Unsigned Long Int
ll atau (Ll, lL, LL)	Long Long Int
Ull atau (uLL, Ull, ULL, llu, llU, LLu, or LLU)	Unsigned Long Long Int

Contoh Penulsan

```
65            //int
65u          //unsigned
65l           //long
65ul         //unsigned long
65ll          //long long
65ull        //unsigned long long
65U         //unsigned
65LU       //unsigned long
65LLU    //unsigned long long
```

Contoh Program

```
#include <iostream>
int main ()
{

    int x = 65; //Int
    x = 65u;    //Unsigned
    x = 65l;    //Long
    x = 65ul;   //Unsigned Long
    x = 65ll;   //Long Long
    x = 65ull;  //Unsigned Long
    x = 65llu;  //Unsigned Long Long
    return 0;
}
```

Floating-point literal

Floating-point literal adalah suatu literal yang memiliki bagian integer, decimal, pecahan, dan eksponen. Anda dapat mewakili floating point literal dalam bentuk decimal atau bentuk eksponensial.

Sementara mewakili menggunakan bentuk desimal, Anda harus menyertakan titik desimal, eksponen, atau keduanya, dan saat mewakili menggunakan bentuk eksponensial, Anda harus menyertakan bagian integer, bagian pecahan, atau keduanya. Eksponen ditandai dan diperkenalkan oleh e(Huruf kecil) atau E(Huruf Besar).

Contoh Penulisan

```
314159E-5L    // Legal
3.14159       // 3.14159
6.02e23       // 6.02 x 10^23
1.6e-19       // 1.6 x 10^-19
3.0           // 3.0
```

Dasar data type dari floating-point literal adalah double. Floating-point literal dengan tipe data float atau long double bisa diwakili menggunakan akhiran sebagai berikut

Suffix	Data Type
f atau F	Float
l atau L	Long Double

Contoh Penulisan

```
3.14159L   // long double
6.02e23f   // float
```

Literal Boolean

Ada dua literal Boolean dan mereka adalah bagian dari kata kunci C ++ standar:
True : Sebuah nilai yang berarti Benar (1)
False : Sebuah nilai yang berarti Salah (0).
Kita dapat mewakili kata kunci True dengan angka 1, dan kata kunci False dengan angka 0.

```
bool myVarTrue = true;   // 1
bool myVarFalse = false; //0
```

Character dan String literals

Character Literal diapit dengan tanda kutip tunggal dan String Literal diapit dengan tanda kutip ganda, hall itu digunakan untuk membedakan mereka dari identitas suatu variabel / object atau standar kata kunci pada c++. Dalam Character atau String Literals dapat memuat karakter polos (misalnya, a-z,0-9 dan symbol lainya), Escape Sequances (misalnya, \t, \n dan lain-lain) dan karakter universal (misalnya, \u02C0).

Contoh Penulisan

```
‘a’                                 //Character Literal
‘B’                                 //Character Literal
“BelajarC++”                        //String Literal
“BelajarCpp\nTempat Belajar C++”    //String Literal
```

Pada Character dan String Literal terdapat code untuk mewakili karakter khusus yang sulirt atau tidak bisa dinyatakan langsung dalam kode sumber program. Kode untuk mewakili karakter khusus yang dimaksud selalu di awali dengan tanda backslash (\), Seperti Newline (\n), tab (\t) dan lain-lain. Semua itu disebut sebagai Escape Sequances atau Escape Code.

Contoh Penulisan

```
‘\n’
‘\t’
“Kanan \t Kiri”
“satu\ndua\ntiga”
```

Di dalam komputer setiap karakter diwakili dengan kode numerik, dan kode untuk pewakilan karakter ter kebanyakan mengikuti ASCII Encoding. Dari hal itu kita dapat mewakili karakter dengan kode numeric yang penulisanya diawali dengan tanda backslash (\), kode tersebut dapat dinyatakan sebagai oktal (basis 8) atau hexadecimal (basis 16). Untuk menggunakan kode numerik oktal kita hanya menulinya digit kode tersebut langsung setelah tanda backslash (\), dan untuk hexsadesimal kita harus menambahkan tanda x ditengah (diantara) backslash (\) dank ode numerik hexsadesimal (misal : \x4f).

Dalam penulisan String Literal kita juga dapat melakukan penulisan dengan cara terpisah. Hal ini sama dan akan akan menghasilkan satu string.

Contoh Penulisan

```
“Satu” “\t” “Dua”
Penulisan tersebut sama seperti penulisan
“Satu\tDua”
```

Dan di dalam string literal kita juga dapat melakukan penulisan di baris selanjutnya. Hal itu dapat kita lakukan menggunakan tanda backslash (\) pada akhir dari baris tersebut, tanda tersebut akan menggabungkan string selanjutnya yang ada pada baris-baris dibawahnya.

Contoh Penulisan

```
“Hallo, \
Selamat datang di\
BelajarC++”
```

Contoh penulisan di atas akan menghasilkan.

```
“Hallo, Selamat Datang di BelajarC++”
```

String merupakan kumpulan dari character (char). Dalam char kita juga dapat menentukan jenis karakter yang akan kita pakai. Dengan menambahkan awalan (Prefix) sebagai berikut

Prefix	Character Type
L	wchar_t
U	char32_t
u	char16_t

Dan untuk string memiliki dua awalan tambahan.

Prefix	Keterangan
R	Raw String (String Mentah)
u8	String literal encoded dalam eksekusi menggunakan UTF-8

Const Keyword

Dalam bahasa pemrograman C++ terdapat dua cara untuk membuat nilai yang memiliki nilai tetap, yaitu menggunakan const keyword dan define preprocessor directive. Untuk const keyword berfungsi untuk merubah sifat deklarasi seperti variabel atau object menjadi memiliki sifat tetap, data atau nilai pada deklarasi itu tidak akan bisa untuk diubah. Hal ini biasanya digunakan untuk memnyatakan sebuah nilai yang memang sudah standard seperti nilai PI, kecepatan cahaya dan lain-lain, dengan keyword const kita dapat membuat data pada deklarasi itu tetap terjaga.

Untuk mendirikan sebuah deklarasi konstanta dirikan sebuah variabel seperti biasa dan kita tambahkan keyword “const” yang biasanya diletakan pada awal dari deklarasi.

Bentuk Penulisan

```
const dataType identitas = inisialisasi
```

Contoh Penulisan

```
const float PI = 3.14;
```

Untuk menyatakan variabel konstanta kita diwajibkan untuk melakukan inisialisasi pada variabel konstanta tersebut.

Contoh Program

```
int main ()
{
    const float PI = 3.14;
    PI = 20;

    return 0;
}
```

Contoh Program di atas akan memperlihatkan fungsi dari keyword const. pada baris ke 3 kita dirikan variabel float “PI” dengan sifat konstanta dan pada baris selanjutnya kita coba untuk mengubah nilai dari variabel PI tersebut. saat kompilasi, compiler akan menghasilkan pesan error bahwa variabel PI hanya dapat dibaca dan tidak bisa diubah.

Untuk identifier dari sebuah deklarasi konstanta biasanya para programmer menggunakan huruf besar pada identitas deklarasi untuk mempermudah mereka dalam mengenali sifat deklarasi yang akan mereka panggil.

#define Preprocessor Directive

Selain cara yang ada di atas (const keyword). kita juga bisa menggunakan “#define” untuk membuat konstanta. #define Digunakan untuk mendefinisikan suatu nilai tertentu kepada suatu nama konstanta.

Bentuk Penulisan

```
#define NAMA_KONSTANTA nilai
```

Keterangan :
NAMA_KONSTANTA : digunakan untuk identitas dari konstanta tersebut, dalam pemberian identitas sebaiknya ditulis dengan menggunakan huruf besar, guna untuk membedakannya dengan nama_variabel biasa (nilai yang dapat diubah).

nilai : adalah tempat dimana kita dapat mendefinisikan konstanta tersebut dengan nilai, kita tidak hanya dapat mengisinya dengan sebuah nilai (Angka/Karakter/String) tapi kita juga dapat mengisinya dengan sebuah pernyataan.

nilai pada #define dapat berupa:

```
#define PI 3.14 //Numerik
#define HURUF ‘B’ //Karakter
#define JABATAN “INSTRUCTOR” //String
#define CETAK std::cout<<“Belajar C++”; //Pernyataan
#define LUAS_KUBUS (n*n) //Ekspresi
```

Setelah #define didirikan di dalam program, untuk menggunakanya kita cukup tuliskan NAMA_KONSTANTAnya saja. Ketika kompilasi semua pemanggilan nama KONSTANTA #define akan diganti menjadi nilai dari KONSTANTA #define tersebut.

Contoh Program

```
#include <iostream>
#define PI 3.14159 //Konstanta
#define NEWLINE cout<<endl; //Konstanta

using namespace std;

int main ()
{
    const char NAMA[]="Belajarpp.com"; //Konstanta
    double hasil;


    hasil = (PI * 14)*2;
    cout<<NAMA;
    NEWLINE
    cout<<hasil;

   return 0;
}
```

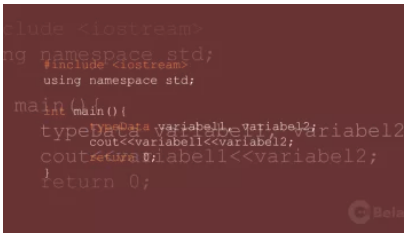

Terkait

Pernyataan if
Sebuah program akan menjalankan sebuah tugas tertentu jika memenuhi sebuah kondisi. Contoh sederhana: jika sebuah nilai merupakan
Januari 29, 2018
dalam "C"

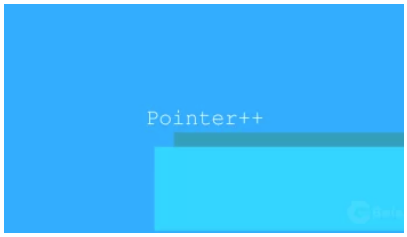
Assignment Operator
Januari 22, 2018
dalam "C++"

Pernyataan Switch
Januari 23, 2018
dalam "C++"

Array
Januari 23, 2018
dalam "C++"



Variabel
Januari 22, 2018
dalam "C++"



Aritmetika pada Pointer
Januari 23, 2018
dalam "C++"