**FUNGSI KEYWORD-KEYWORD dan CONTOH PROGRAMNYA DALAM BAHASA C++**

**Struktur Program C/C++**

Untuk dapat memahami bagaimana suatu program ditulis, maka struktur dari program harus dimengerti terlebih dahulu, atau sebagai pedoman penulis program (programmer) bagaimana seharusnya program tersebut ditulis.

Struktur dari program C dapat diihat sebagai kumpulan dari sebuah atau lebih fungsi-fungsi. Fungsi pertama yang harus ada di di program C yang sudah ditentukan namanya, yaitu fungsi main(). Artinya program C minimal memiliki satu fungsi (fungsi main()). Berikut ini adalah struktur dari program C.

|  |  |
| --- | --- |
| main()  {  statemen\_1;  statemen\_2;  ……..  statemen\_n;  } | Fungsi Utama |
| fungsi\_lain()  {  statemen\_statemen;  } | Fungsi-fungsi lain  yang ditulis oleh pemrogram komputer |

Gambar 1.1 Struktur program C

Keterangan :

1.       Dimulai dari tanda { hingga tanda } disebut tubuh fungsi / blok.

2.       Tanda ( ) digunakan untuk mengapit argumen fungsi, yaitu nilai yang dilewatkan ke fungsi. Pada fungsi main( ) tidak ada argumen yang diberikan, maka tidak ada entri di dalam ( ).

3.       Kata void menyatakan bahwa fungsi ini tidak memiliki nilai balik.

4.       Tanda { menyatakan awal eksekusi program dan tanda } menyatakan akhir eksekusi program.

5.       Didalam tanda { } bisa tergantung sejumlah unit yang disebut pernyataan (statemen). Umumnya pernyataan berupa instruksi untuk :

• Memerintah komputer melakukan proses menampilkan string ke layar.

• menghitung operasi matematika.

• membaca data dari keyboard.

• dll.

Berikut ini adalah suatu program C yang sederhana :

Contoh 1 :

Sisi atas merupakan *source code*, yang dapat diberi nama *project1.*cpp dan sisi bawah adalah hasilnya setelah di-kompile dan di-eksekusi.

Program diatas merupakan salah satu program paling sederhana dalam C++, tetapi dalam program tersebut mengandung komponen dasar yang selalu ada pada setiap pemrograman C++. Jika dilihat satu persatu :

**a.      // my first program in C++**

Baris ini adalah komentar. semua baris yang diawali dengan dua garis miring (**//**) akan dianggap sebagai komentar dan tidak akan berpengaruh terhadap program. Dapat digunakan oleh programmer untuk menyertakan penjelasan singkat atau observasi yang terkait dengan program tersebut.

**b.      #include <iostream.h>**

Kalimat yang diawali dengan tanda (#) adalah are *preprocessor directive*. Bukan merupakan baris kode yang dieksekusi, tetapi indikasi untuk kompiler. Dalam kasus ini kalimat **#include <iostream.h>** memberitahukan preprocessor kompiler untuk menyertakan header filestandard **iostream**. File spesifik ini juga termasuk library deklarasi standard I/O pada C++ dan file ini disertakan karena fungsi-fungsinya akan digunakan nanti dalam program.

**c.       int main ()**

Baris ini mencocokan pada awal dari deklarasi fungsi **main**. fungsi **main** merupakan titik awal dimana seluruh program C++ akan mulai dieksekusi. Diletakan diawal, ditengah atau diakhir program, isi dari fungsi main akan selalu dieksekusi pertama kali. Pada dasarnya, seluruh program C++ memiliki fungsi **main**.

**d.      main** diikuti oleh sepasang tanda kurung **()** karena merupakan fungsi. pada C++, semua fungsi diikuti oleh sepasang tanda kurung **()** dimana, dapat berisi argumen didalamnya. Isi dari fungsi **main** selanjutnya akan mengikuti**,**berupa deklarasi formal dan dituliskan diantara kurung kurawal (**{}**), seperti dalam contoh.

**e.      cout << "Hello World";**

Intruksi ini merupakan hal yang paling penting dalam program contoh. **cout** merupakan standard output stream dalam C++ (biasanya monitor). **cout** dideklarasikan dalam header file iostream.h, sehingga agar dapat digunakan maka file ini harus disertakan.

**f.** Perhatikan setiap kalimat diakhiri dengan tanda semicolon (**;**). Karakter ini menandakan akhir dari instruksi dan harus disertakan pada setiap akhir instruksi pada program C++ manapun.

**g.      return 0;**

Intruksi **return** menyebabkan fungsi **main()** berakhir dan mengembalikan kode yang mengikuti instruksi tersebut, dalam kasus ini **0**. Ini merupakan cara yang paling sering digunakan untuk mengakhiri program.

Tidak semua baris pada program ini melakukan aksi. Ada baris yang hanya berisi komentar (diawali //), baris yang berisi instruksi untuk preprocessor kompiler (Yang diawali **#**),kemudian baris yang merupakan inisialisasi sebuah fungsi (dalam kasus ini, fungsi **main**) dan baris yang berisi instruksi (seperti, **cout <<**), baris yang terakhir ini disertakan dalam blok yang dibatasi oleh kurung kurawal (**{}**) dari fungsi **main**.

**A.     FILE JUDUL / HEADER FILE**

File judul (header file) merupakan file yang berisi dengan prototype (judul, nama dan sintak) dari sekumpulan fungsi-fungsi pustaka tertentu. Jadi file ini hanya berisi dengan prototype dari fungsi-fungsi pustaka, sedangkan fungsi-fungsi pustakanya sendiri disimpan di file pustaka (library file dengan extention file .LIB). Misalnya prototype dari fungsi-fungsi pustaka printf() dan scanf() terdapat di file judul stdio.h, sehingga jika fungsi-fungsi ini akan digunakan di program, maka nama file judulnya harus dilibatkan dengan menggunakan **preprocessor directive #include**. File judul stdio.h berisi prototype fungsi-fungsi pustaka untuk operasi input dan output standar. Ada dua cara untuk melibatkan file judul di suatu program C, yaitu sebagai berikut :

#include <stdio.h> atau #include “stdio.h”

**B.     NAMA PENGENAL**

Nama pengenal (Identifier name) adalah nama-nama yang ditentukan sendiri oleh pembuat program. yang nantinya nama tersebut digunakan dalam pemrograman. Fungsinya untuk menyatakan : Variabel, Tipe Data, Konstanta, Fungsi, Label dan Obyek.

Aturan penamaan pengenal :

1. Karakter pertama harus huruf atau garis bawah.

2. Karakter berikutnya boleh huruf, bilangan, atau garis bawah.

3. Panjang maksimal 32 karakter.

4. Nama pengenal tidak boleh sama dengan kata kunci (reserved word)

5. Pengenal bersifat case sensitif.

Huruf kecil dan huruf besar (kapital) pada suatu pengenal tidak dianggap sama :

Contoh : NAMA, Nama, nama = menyatakan pengenal yang berbeda.

Contoh penamaan :

|  |  |
| --- | --- |
| Benar | Salah dan alasanya |
| Nama | 2semester ( tidak boleh diawali angka ) |
| NAMA | nama-barang ( tanda – tidak diperbolehkan ) |
| Nama\_barang | #barang (simbol # tidak diperbolehkan ) |
| Kuartal\_2 | Nama barang ( tidak boleh mengandung spasi ) |

**C.      KOMENTAR PROGRAM**

Komentar merupakan bagian terpenting dari program. Kehadirannya sangat membantu pemrograman ataupun orang lain dalam memahami program, karena berupa penjelasan-penjelasan mengenai program atau bagian-bagian program. Hal ini penjelasannya bisa berupa :

                        • tujuan / fungsi program

                        • saat program dibuat atau direvisi

                        • keterangan-keterangan lain tentang kegunaan sejumlah pernyataan dalam program.

Pada C suatu komentar di awali dengan tanda ( /\* ) dan diakhiri dengan tanda ( \*/ ). Semua string atau tulisan yang terletak sesudah tanda /\* hingga akhir baris dengan sendirinya akan diperlukan sebagai keterangan. Bagi kompiler hal ini tidak berguna dan akan di abaikan pada saat kompilasi.

**a.      Preprocessor directive**

Preprocessor directive disebut juga pengarah compiler karena fungsinya untuk mengatur proses kompilasi.

**#include** merupakan satu jenis pengarah *preprocessor* yang digunakan untuk membaca *file* yang dinamakan *file* judul (*header file*). **iostream, conio.h, stdio.h, windows.h, math.h,** dan **string.h** merupakan *file header* yang merupakan *s*tanda*rd library* dari C++. Setiap *file header* berhubungan dengan perintah masukan (*input*), perintah keluaran (*output*), dan fungsi-fungsi yang digunakan dalam suatu program.

**b.      Keyword**

Keyword adalah kata kunci yang digunakan dalam pemrograman untuk fungsi-fungsi tertentu. Dalam pemrograman c++ keyword dikenali oleh compiler dengan cara mendeklarasikan file headernya, tetapi ada juga keyword yang tidak perlu untuk mendeklarasikan file header. Keyword merupakan pernyataan singkat dan mempunyai fungsi tertentu sehingga sangat fleksibel dan mudah dalam penggunaan dalam pemrograman. Berikut merupakan keyword dasar dalam c++.

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

**1.      Auto**

Mendefinisikan sebuah variabel lokal sebagai yang memiliki lifetime lokal. Keyword auto menggunakan syntaks sebagai berikut:;

[auto] definisi-data. Contoh:

#include <stdio.h>

#include <conio.h>

int main(void)

{

for(; ;)

{

if (getche() == 'a')

{

**auto int t;**

for (t=0; t<'a'; t++)

printf("%d ", t);

break;

}

}

return 0;

}

**2.      Break**

Membuat kontrol keluar dari pernyataan majemuk. Pernyataan break membuat alur yang sedang dijalankan keluar dari pernyataan iterasi seperti while, do atau switch. Sintaksnya hanya:;

break; contoh:

while (x < 100)

{

x = get\_new\_x();

if (kbhit()) break; /\* tombol keyboard ditekan \*/

process(x);

}

**3.      case**

Merupakan bagian dari pernyataan percabangan switch dimana keyword case adalah sebagai pernyataan pilihan yang selanjutnya akan dijalankan jika ekspresi switch memenuhi suatu konstanta case. Syntaksnya adalah sebagai berikut:;

switch (ekspresi) {

case ekspresi-konstanta1:

alur pernyataan1;

break;

case ekspresi-konstanta2:

alur pernyataan2;

break;

case ekspresi-konstanta3:

alur pernyataan3;

break;

.

..

}

Alur diatas dapat dijelaskan secara sederhana layaknya pilihan ganda, jika ekspresi yang mengikuti pernyataan switch sesuai dengan ekspresi-konstanta1 maka pernyataan1 akan dijalankan, jika sesuai dengan ekspresi-konstanta2 maka pernyataan2 akan dijalankan, jika sesuai dengan ekspresi-konstanta3 maka pernyataan3 akan dijalankan dan seterusnya. Jika tidak ada ekspresi-konstanta yang mengikuti pernyataan case dalam lingkup switch diatas yang memenuhi ekspresi maka alur tidak akan melakukan eksekusi apapun.;

**4.      char**

Merupakan type data mendasar dalam C/C++ yang menampung data berjenis karakter atau teks. Dapat juga dijadikan sebagai penanda type data nilai kembalian suatu fungsi. Contoh sintaksnya adalah sebagai berikut:;

char huruf;

huruf='h';

Syntaks diatas, baris pertama adalah mendeklarasikan variabel huruf bertipe data char yang mempunyai panjang satu karakter. Sedangkan baris kedua adalah pernyataan penugasan yaitu mengisi nilai variabel huruf menjadi sama dengan h. Adapun pengunaan keyword char untuk mendeklarasikan tipe data dari nilai kembalian suatu fungsi:;

char fungsiku(int x, int y, char z);

Arti dari pernyatan diatas adalah kita mendeklarasikan fungsi dengan 3 buah parameter argumen (x bertipe data integer, y bertipe data integer, dan z bertipe data character) yang mempunyai nilai kembalian dengan tipe data character.;

**5.      const**

Mendeklarasikan/membuat nilai dari variabel atau pointer menjadi tidak bisa diubaha (konstanta). Berbeda dengan deklarasi variabel biasa, deklarasi konstanta harus langsung disertai dengan assigment nilai dari variabel konstanta tersebut. Contoh:;

const x=2;

Syntaks diatas adalah kita mendeklarasikan variabel konstanta x dengan nilai sama dengan 2.;

**6.      continue**

Adalah pernyataan yang membuat alur melewati lingkup iterasi seperti while, do, for saat itu dan melanjutkan ke nilai iterasi selanjutnya. Contoh penggunaannya adalah sebagai berikut:;

for (i = 0; i < 50; i++)

{

if (array[i] == 0)

continue;

array[i] = array[i]/2;

}

Fungsi alur diatas adalah merubah semua nilai dari elemen array menjadi setengahnya tetapi tidak jika nilai elemen array itu sama dengan nol.

**7.      dafault**

Merupakan bagian dari penggunaan keyword switch, penjelasannya adalah pada lingkup pernyataan switch, alur akan mencari nilai dari berbagai kontanta case yang disajikan yang sesuai dengan ekspresi yang mengikuti keyword switch, jika tiak ada dari berbagai pernyataan case yang disajikan maka pernyataan dibawah keyword default -lah yang akan dijalankan. Contoh penggunaannya adalah sebagai berikut:;

switch (operand)

{

case MULTIPLY:

x \*= y; break;

case DIVIDE:

x /= y; break;

case ADD:

x += y; break;

case SUBTRACT:

x -= y; break;

case INCREMENT2:

x++;

case INCREMENT1:

x++; break;

case EXPONENT:

case ROOT:

case MOD:

printf ("Not implemented!\n");

break;

default:

printf("Bug!\n");

exit(1);

}

**8.      do**

Merupakan bagian dari pernyataan iterasi while...do yang fungsinya adalah mengulang berbagai pernyataan dengan kriteria pengulangan. Jika kriterianya terpenuhi maka pengulangan akan dilanjutkan, tetapi jika tidak maka pengulangan akan dihentikan. Contoh penggunaannya:;

x = 1;

do

{

x++;

} while (x <= 100);

Penjelasan alur diatas adalah variabel x akan terus bertambah nilainya satu selama nilai x lebih kecil atau sama dengan 100.;

**9.      double**

Adalah salah satu tipe data floating number dalam bahasa C/C++. Bedanya dengan tipe data float, double mempunyai titik presisi ganda. Contoh penggunaannya adalah sebagai berikut:;

double modal;

Code diatas artinya kita mendeklarasikan variabel modal dengan tipe data double.;

**10. else**

Adalah bagian dari pernyataan kondisional if. Alur dibawah keyword else akan dijalankan jika ekspresi yang menyertai keyword if bernilai false. Contohnya adalah sebagai berikut:;

if (ekspresi) pernyataan1;

else pernyataan2;

Jika ekspresi bernilai true maka pernyataan1 akan dijalankan, tetapi jika tidak maka pernyataan2 akan dijalankan. Bentuk lain dari pernyataan kondisional if..else adalah sebagai berikut:;

if (ekspresi)

{

pernyataan1;

pernyataan2;

…

…

}

else

{

pernyataan3;

pernyataan4;

…

…

}

**11. enum**

Membuat tipe data enumerasi berjenis data integer yang mempunyai banyak nilai konstan yang telah didefinisikan.;

enum [tag] {nama [=nilai], ...};

Berikut adalah contoh penggunaannya:;

enum nomor {Satu =1 , Dua = 2, Tiga = 3, Empat = 4};

**12. extern**

Menunjukkan bahwa sebuah identifier didefinisikan di tempat lain. Dalam pemrograman modular, extern sangat bermanfaat jika kita ingin mengidentifikasi suatu variabel global yang telah dideklarasikan di tempat/file source code lainnya. Maka dalam pemrograman multifile, kita bisa mendeklarasikan semua variabel global pada suatu file lalu mengidentifikasikannya di file lain. File satu:;

int x;

int main(void)

{

x=1;

cout << x << endl;

return 0;

}

File dua:;

extern int x;

int func32(void)

{

int y;

y=x;

cout << y << endl;

return 0;

}

Source code diatas menunjukkn bahwa File dua mengidentifikasi variabel x dari File satu yang telah dideklarasikan.;

**13. float**

Sama seperti double, namum hanya mempunyai titik presisi tunggal.;

float laba;

**14. for**

Adalah salah satu pernyataan iterasi selain while..do.;

for (i=0; i<100; i++) sum += x[i];

**15. goto**

Merupakan pernyataan untuk memerintah alur yang sedang berjalan agar berpindah ke baris lain sesuai label. Contohnya adalah:;

Mulai:

pernyataan;

..

.

goto Mulai;

Alur akan mengulangi eksekusi ke pernyataan setelah mencapai baris goto Mulai;.;

**16. if**

Merupakan pernyataan kondidional, keyword if digunakan untuk eksekusi kondisional. Secara mendasar sintaksnya adalah sebagai berikut:;

if (ekspresi)

pernyataan;

Secara lanjut dapat ditulis sebagai berikut:;

if (ekspresi)

{

pernyatanan1;

..

.

}

else

{

pernyatan2;

..

.

}

**17. int**

Adalah tipe data standar berjenis bilangan bulat. Jangkauan tipe data int adalah −32,767 sampai 32,767. Deklarasinya adalah sebagai berikut:;

int x;

**18. register**

Memberitahu compiler untuk menyimpan variabel yang dinyatakan dalam register CPU. Penggunaannya adalah sebagai berikut:;

register definisi-data;

Contoh penggunaannya adalah sebagai berikut:;

register int i;

**19. return**

Adalah pernyataan yang berfungsi untuk memberikan nilai kembalian suatu fungsi lalu alur akan keluar dari fungsi tersebut. Sintaksnya adalah sebagai berikut: return nilai; Contoh penggunaannya adalah sebagai berikut:;

bool invers(bool exp)

{

return !exp;

}

**20. signed**

Merupakan type data modifier yang membuat suatu tipe data mempunyai jangkauan dari nilai minus batas atasnnya hingga nilai plus dari baas atasnnya. Penggunaannya adalah sebagai berikut:;

signed int var1;

Sebenarnya juga dapat ditulis seperti dibawah ini tanpa mengubah fungsinya:;

int var1;

sizeof Adalah fungsi yang mengembalikan nilai ukuran dalam byte dari ekspresi atau type data yang diberikan. Contoh penggunaannya adalah sebagai berikut:;

x=sizeof(int);

**21. static**

Mempertahankan nilai variabel untuk bertahan hidup setelah ruang lingkup berakhir. Kata kunci static dapat diterapkan untuk kedua definisi data dan fungsi:;

static data-definition;

static function-definition;

;

**22. struct**

Mengelompokkan variabel dalam record tunggal. Sintaksnya adalah sebagai berikut:;

struct [nama-struktur]

{

[type nama-variabel] ;

...

} [variabel-struktur] ;

Contoh:;

struct person

{

string nama, nomor\_hp, alamat;

int umur, tinggi\_badan;

};

Contoh diatas kita mendeklarasikan struktur dengan nama person yang mempunyai anggota data bertipe string 3 buah dan bertipe integer 2 buah. Deklarasi anggota data dengan tipe data sama dipisahkan dengan tanda koma dan titik koma jika tipe datanya berbeda. [variabel-struktur] dapat diisi ataupun tidak (opsional). Penggunaan tipe data struktur diatas dapat seperti ini:;

struct person orang[100];

Atau jika kita ingin mendeklarasikan variabel array orang langsung setelah mendeklarasikan strukturnya maka dapat ditulis seperti ini:;

struct person

{

string nama, nomor\_hp, alamat;

int umur, tinggi\_badan;

}orang[100];

**23. switch**

Merupakan pernyataan percabanagan, menyebabkan kontrol melakukan percabangan terhadap daftar pernyataan yang disediakan dalam blok pernyataan. Sintaknya adalah:;

switch (ekspresi) pernyataan

Secara lanjut dapat ditulis:;

switch (ekpresi)

{

case ekspresi-konstanta1:

..

case ekspresi-konstanta2:

..

}

**24. typedef**

Mendefinisikan/membuat tipe data baru. Sinaksnya adalah:

typedef definisi-type identifier;

Contoh penggunaannya adalah sebagai berikut:;

typedef unsigned char byte;

typedef char str40[41];

typedef struct {float re, im;} complex;

typedef char \*byteptr;

typedef int (\*fncptr)(int);

Setelah definisi diatas, maka kita dapat mendeklarasikan:;

byte m, n;

str40 myStr;

complex z1, z2;

byteptr p;

fncptr myFunc;

Sama seperti kita mendeklarasikan:;

unsigned char m, n;

char myStr[41];

struct {float re, im;} z1, z2;

char \*p;

int (\*myFunc)(int);

**25. union**

Mengelompokkan variabel yang berbagi ruang penyimpanan yang sama. Sebuah serikat mirip dengan sebuah struct, kecuali union memungkinkan kita untuk menentukan variabel yang berbagi ruang penyimpanan. Sintaks untuk mendefinisikan serikat adalah:;

union [union-type-name]

{

type variable-names;

...

} [union-variables] ;

Contoh,;

union short\_or\_long

{

short i;

long l;

} a\_number;

**26. unsigned**

Merupakan type modifier yang dapat diterapkan ke tipe data standar. Seperti namanya unsigned (tanpa tanda) maka jika kita mendeklarasi variabel menggunakan type modifier unsigned maka jangkauan variabel tersebut hanya akan bermula dari 0 hingga batas atas tipe data dasar tersebut dikali 2. Contohnya jika kita mendeklarasikan variabel dengan tipe int yang mempunyai jangkauan – 32,767 sampai 32,767 maka jika kita awali dengan type modifier unsigned variabel itu akan mempunyai jangkauan 0 sampai 6,5534. Contoh deklarasi variabel dengan modifier unsigned adalah seperti dibawah ini:;

unsigned int x;

**27. void**

Adalah type data kosong. Jika kita gunakan dalam deklarasi fungsi maka artinya fungsi tersebut tidak mempunyai nilai kembalian:;

void hello (char \*nama)

{

printf("Hallo, %s bagaimana kabar anda?", nama);

}

Jika dimasukkan ke deklarasi parameter fungsi maka artinya fungsi tersebut tidak memerlukan parameter apapun.;

void hello (void)

{

printf("Hallo bagaimana kabar anda?”);

}

**28. volatile**

Menunjukkan bahwa variabel dapat diubah oleh rutinitas latar belakang. Syntaksnya adalah sebagai berikut:;

volatile data-definition;

**29. while**

Adalah pernyataan iterasi/pengulangan dimana suatu alur akan diulang ketika kondisi yang menyertai while bernilai true. Sintaksnya adalah:;

while (ekspresi) pernyataan;

Ketika ekspresi bernilai true maka pernyataan akan terus dijalankan hingga ekspresi bernilai false. Contoh:;

while (\*p == ' ') p++;

**D.     STATEMENT**

Suatu statemen (statement) adalah pernyataan yang menyebabkan suatu tindakan akan dilakukan oleh komputer. Tindakan tersebut dapat berupa tindakan untuk menghitung, menampilkan hasil, menerima input data, mengendalikan proses program dan lain-lain. Suatu statemen dibahasa C di tulis dengan diakhiri oleh tanda titik koma (;).

Contoh :

X = X + 1;

printf(“ Nilai X = %f \n”,X);

**E. TIPE DATA**

Bahasa C memiliki lima tipe data utama, yaitu karakter (char), integer (int), Floating point (float0, double floating(double) dan valualess/kosong (void). Untuk int, float, double dan char dapat dikombinasikan dengan modifier signed, unsigned, long, short, maka hasilnya sbb : Penanda signed tidak perlu dituliskan.

|  |  |  |
| --- | --- | --- |
| Tipe | Lebar | Jangkauan nilai |
| Int |  |  |
| Signed int | 16 bit | -32768 s/d 32767 |
| Signed short int | 16 bit | -32768 s/d 32767 |
| Signed long int | 32 bit | -2147483647 s/d 2147483647 |
| Unsigned int | 16 bit | 0 s/d 65535 |
| Unsigned short int | 16 bit | 0 s/d 65535 |
| Unsigned long int | 32 bit | 0 s/d 4294967295 |
| Float |  |  |
| Float | 32 bit | 3.4E-38 s/d 3.4E+38 |
| Double | 64 bit | 1.7E-308 s/d 1.7E+308 |
| Long double | 80 bit | 3.4E-4932 s/d 3.4E+4932 |
| Char |  |  |
| Signed char | 8 bit | -128 s/d 127 |
| Unsigned char | 8 bit | 0 s/d 255 |

**F.      KONSTANTA**

Adalah suatu nilai yang tidak pernah berubah selama proses program. Dibagi menjadi :

a.       Numeric integer

Dapat berupa integer bertanda dan integer tidak bertanda. Untuk unsigned integer diberi huruf u atau U pada akhir nilai, untuk signed long diberi tanda l atau L dan unsigned long diberi tanda ul atau UL.

Contoh :

43 (signed int), -123 (signed int), -75000L (signed long int), 261062UL (unsigned long int).

b.      Numerik pecahan

Merupakan nilai numeric yang dapat mempunyai pecahan dibelakang titik decimal. Bilangan yang diberikan dapat diikuti huruf F untuk float atau huruf L untuk long double.

Contoh :

1.5 (float) , 253.67(float), 12.34E-99 (double), 1000.1F (float), 1.2E+4930 (long double).

c.       String dan character.

Merupakan nilai yang ditulis diantara tanda petik ganda untuk string dan tunggal untuk karakter.

Contoh :

“A” string A dan ‘A’ karakter A.

d.      Karakter Escape

Karakter ini digunakan untuk menampilkan hasil tertentu dan diawali dengan tanda ‘\’. Berikut table karakter escape :

|  |  |
| --- | --- |
| Karakter | Fungsi |
| \a | Bunyi bell atau alert |
| \b | Mundur 1 spasi (backspace) |
| \f | Form feed (ganti halaman) |
| \n | New line (ganti baris) |
| \r | Carriage return (ke kolom pertama,baris yg sama |
| \t | Tabulasi horizontal |
| \v | Tabulasi vertical |
| \0 | Nilai kosong (null) |
| \’ | Tanda petik |
| \” | Petik ganada |
| \\ | Backslash |

e. Konstanta Bernama

            Konstanta bernama adalah suatu pengenal yang berhubungan dengan nilai tetap (variabel konstan). C++ memungkinkan pendefinisian suatu konstanta bernama. Hal ini dilakukan dengan menggunakan kata kunci const.

Bentuk umum :

**const** *tipe\_data* nama\_konstanta = *nilai*;

            Contoh :

**const** int bulan = 12;

**const** float phi = 3.14285;

**G.     VARIABEL**

Variabel adalah suatu pengenal yang digunakan untuk menyimpan suatu nilai dan nilai dari variabel tersebut dapat berubah-ubah selama proses program.

**a.      Mendeklarasikan Variabel**

            Bentuk umum deklarasi

***typedata*** namavariabel;

Contoh :

**int** jumlah;

**float** rata\_rata;

**char** nama;

**b.      Pendeklarasian dan penugasan variabel**

Nilai awal suatu variabel dapat langsung diberikan pada saat mendeklarasikannya.

                        Bentuk umum

***typedata*** namavariabel = nilai;

Contoh :

**int** jumlah = 5;

**float** rata\_rata = 2.5;

**char** karakter = ‘A’

**char** string[10] ;

**H.    BAGIAN PENGARAH COMPILER**

Contoh program C++ :

Merupakan statement praprosesor, disebut juga pengarah compiler karena berfungsi mengatur proses kompilasi. **IOSTREAM.H** merupakan file program yang mengandung deklarasi kelaskelas yang diperlukan oleh objek **cout**. File-file dengan ekstensi .H yang berisi deklarasi fungsi-fungsi standar C ini, disebut secara umum sebagai file header.

Beberapa pengarah compiler adalah :

1. # define

2. # include

3. # if, # else, # elif, # endif

4. # ifdef, # ifndef

  **Pengarah Kompiler # Define**

Untuk mendefinisikan suatu pengenal / konstanta yang nantinya akan digantikan oleh praprosesor saat program dikompilasi.

Contoh Program :

**# define SIZE** 30

Int array [**SIZE**] ;

For (register int i = 0 ; i < SIZE ; i++)

{

Cout << array [ i ] ;

}

**I.        PERINTAH KELUARAN**

#include <stdio.h> //untuk mengaktifkan perintah printf dan scanf

#include <conio.h> //untuk mengaktifkan perintah getch()

Sebelumnya, Untuk menampilkan hasil output dibutuhkan kode format, berikut adalah daftar kode format:

|  |  |
| --- | --- |
| **Kode format** | **Kegunaan** |
| %c | Menampilkan sebuah karakter |
| %s | Menampilkan nilai string |
| %d | Menampilkan nilai decimal integer |
| %i | Menampilkan nilai decimal integer |
| %u | Menampilkan nilai decimal integer tidak bertanda (unsigned integer) |
| %ld | Menampilkan nilai decimal long integer |
| %lu | Menampilkan nilai decimal long integer tak bertanda |
| %li | Menampilkan nilai decimal long integer |
| %hu | Menampilkan nilai decimal short integer tak bertanda |
| %hi | Menampilkan nilai decimal short integer |
| %x | Menampilkan nilai heksa decimal integer |
| %o | Menampilkan nilai okta integer |
| %f | Menampilkan nilai pecahan / float |
| %e | Menampilkan nilai float scientific |
| %g | Sebagai pengganti %f atau %e tergantung yang terpendek |
| %lf | Menampilkan nilai pecahan double |
| %le | Menampilkan nilai pecahan double |
| %lg | Menampilkan nilai pecahan double |
| %p | Menampilkan suatu alamat memory untuk pointer |

**Penjelasan** Perintah standar output yang disediakan oleh Borland C++, diantaranya adalah :

                     **printf()**

                     **puts()**

                     **putchar()**

                     **cout()**

**a.      printf()**

**Penjelasan:** Fungsi ***printf()*** merupakan fungsi keluaran yang paling umum digunakan untuk menampilkan informasi kelayar.

**Bentuk Penulisan**

String-Kontrol dapat berupa keterangan yang akan ditampilkan pada layar beserta penentu format. Penentu format dipakai untuk memberi tahu kompiler mengenai jenis data yang dipakai dan akan ditampilkan.

Argumen ini dapat berupa variabel, konstanta dan ungkapan.

Tabel 2.4. Penentu Format Printf()

|  |  |
| --- | --- |
| **TIPE DATA** | **Penentu Format Untuk *printf()*** |
| Integer | %d |
| Floating Point |  |
| Bentuk Desimal | %f |
| Bentuk Berpangkat | %e |
| Bentuk Desimal dan Pangkat | %g |
| Double Precision | %lf |
| Character | %c |
| String | %s |
| Unsigned Integer | %u |
| Long Integer | %ld |
| Long Unsigned Integer | %lu |
| Unsigned Hexadecimal Integer | %x |
| Unsigned Octal Integer | %o |

printf(“%c merupakan abjad yang ke - %d”,’b’,2);

Contoh-1

**a. Penggunaan Penentu Lebar Field**

**Penjelasan:** Bila ingin mencetak atau menampilkan data yang bertipe data FLOAT atau pecahan, tampilan yang tampak biasanya kurang bagus. Hal tersebut dapat diatur lebar field-nya dan jumlah desimal yang ingin dicetak. Berikut bentuk penulisannya :

Output yang akan dihasilkan, jika tidak menggunakan panentu lebar field adalah

Bilangan A = 7.500000

Bilangan B = 243.210007

**b.      puts()**

**Penjelasan**: Perintah *puts()* sebenarnya sama dengan *printf()*, yaitu digunakan untuk *mencetak string* ke layar. *puts()* berasal dari kata *PUT STRING*.

Perbedaan antara *printf()* dengan *puts()* adalah :

Tabel 2.6. Perbedaan fungsi puts() dengan printf()

|  |  |
| --- | --- |
| ***printf()*** | ***puts()*** |
| Harus menentukan tipe data untuk data string, yaitu %s | Tidak Perlu penentu tipe data string, karena fungsi ini khusus untuk tipe data string. |
| Untuk mencetak pindah baris, memerlukan notasi ‘ \n ‘ | Untuk mencetak pindah baris tidak perlu notasi ‘ \n ‘ , karena sudah dibeikan secara otomatis. |

Contoh-3

**c.       putchar()**

**Penjelasan:** Perintah *putchar()* digunakan untuk menampilkan sebuah karakter kelayar. Penampilan karakter tidak diakhiri dengan pindah baris.

Contoh-5

**d.      cout()**

**Penjelasan:** Fungsi ***cout()*** merupakan sebuah objeck didalam Borland C++ digunakan untuk menampilkan suatu data kelayar. Untuk menggunakan fungsi cout() ini, harus menyertakan file header **iostream.h** .

Contoh-6

**Fungsi Manipulator**

**Penjelasan** Manipulator pada umumnya digunakan untuk mengatur tampilan layar, untuk mengguakan manipulator ini file header yang harus disertakan file header **iomanip.h** . Ada beberapa fungsi manipulator yang disediakan oleh Borland C++, antara lain.

|  |  |
| --- | --- |
|                      endl                       end                       flush()                       dec()                       hex()                       oct() |                      setbase()                       setw()                       setfill()                       setprecision()                       setosflags() |

Berikut akan dibahas beberapa fungsi manipulator, diantaranya :

**a. endl**

**Penjelasan: *endl*** merupakan suatu fungsi manipulator yang digunakan untuk menyisipkan karakter NewLine atau mengatur pindah baris. Fungsi ini sangat berguna untuk piranti keluaran berupa file di disk. File header yang harus disertakan adalah file header **iostream.h** .

**b. ends**

**Penjelasan: *ends*** merupakan suatu fungsi manipulator yang digunakan untuk menambah karakter null ( nilai ASCII NOL ) kederetan suatu karakter. Fungsi ini akan berguna untuk mengirim sejumlah karakter kefile didisk atau modem dan mangakhirinya dengan karakter NULL.. File header yang harus disertakan adalah file header **iostream.h** .

**c. dec, oct dan hex**

**Penjelasan: *dec, oct dan hex*** merupakan suatu fungsi manipulator yang digunakan untuk menampilkan data dalam bentuk desimal, oktal dan hexadesimal. File header yang harus disertakan adalah file header **iomanip.h** .

**d. setprecision ()**

**Penjelasan:** Fungsi ***setprecision()*** merupakan suatu fungsi manipulator yang digunakan untuk mengatur jumlah digit desimal yang ingin ditampilkan. Fungsi ini biasa pada fungsi **cout()**, file header yang harus disertakan adalah file header **iomanip.h** .

**J.       Perintah Masukan**

**Penjelasan:** Perintah standar input yang disediakan oleh Borland C++, diantaranya adalah :

                     **scanf()**

                     **gets()**

                     **cout()**

                     **getch**

                     **getche()**

**a.      scanf()**

**Penjelasan:** Fungsi ***scanf()*** digunakan untuk memasukkan **berbagai jenis data**. Bentuk Umum dari fungsi ini adalah :

|  |
| --- |
|  |
|  |  |

**Keterangan** simbol **&** merupakan pointer yang digunakan untuk menunjuk kealamat variabel memori yang dituju.

Tabel 2.7. Penentu Format scanf()

|  |  |
| --- | --- |
| **TIPE DATA** | **Penentu Format Untuk *scanf()*** |
| Integer | %d |
| Floating Point |  |
| Bentuk Desimal | %e atau %f |
| Bentuk Berpangkat | %e atau %f |
| Double Precision | %lf |
| Character | %c |
| String | %s |
| Unsigned Integer | %u |
| Long Integer | %ld |
| Long Unsigned Integer | %lu |
| Unsigned Hexadecimal Integer | %x |
| Unsigned Octal Integer | %o |

**b.      gets()**

**Penjelasan:** Fungsi ***gets()*** digunakan untuk memasukkan **data string**. Bentuk Umum dari fungsi ini adalah :

|  |
| --- |
|  |
|  |  |

Perbedaan antara ***scanf()*** dengan ***gets()*** adalah :

Tabel 2.8. Perbedaan scanf() dengan gets()

|  |  |
| --- | --- |
| ***scanf()*** | ***gets()*** |
| Tidak dapat menerima string yang mengandung spasi atau tab dan dianggap sebagai data terpisah | Dapat menerima string yang mengandung spasi atau tab dan masing dianggap sebagai satu kesatuan data. |

**c.       cin ()**

**Penjelasan:** Fungsi ***cin()*** merupakan sebuah objeck didalam C++ digunakan untuk memasukkan suatu data. Untuk menggunakan fungsi cin() ini, harus menyertakan file header **iostream.h** .

Contoh-13

**d.      getch ()**

**Penjelasan:** Fungsi ***getch()*** (*get character and echo*) dipakai untuk membaca sebuah karakter dengan sifat karakter yang dimasukkan tidak perlu diakhiri dengan menekan tombol ENTER, dan karakter yang dimasukan tidak akan ditampilkan di layar. *File header* yang harus disertakan adalah **conio.h**.

Contoh-14

**e.      getche()**

**Penjelasan:** Fungsi ***getche()*** dipakai untuk membaca sebuah karakter dengan sifat karakter yang dimasukkan tidak perlu diakhiri dengan menekan tombol ENTER, dan karakter yang dimasukan ditampilkan di layar. File header yang harus disertakan adalah **conio.h**.

Contoh-15Selain itu kedua fungsi ini dapat digunakan untuk menahan agar tidak langsung balik kembali kedalam listing program dan hasil dari program yang di eksekusi dapat dilihat tanpa menekan tombol ALT – F5. Karena fungsi *getch()* merupakan fungsi masukkan, jadi sebelum program keluar harus menginputkan satu buah karakter.

# [Fungsi-Fungsi Dasar C/C++](http://puthryy.blogspot.co.id/)

## Minggu, 02 Maret 2014

### FUNGSI KEYWORD-KEYWORD dan CONTOH PROGRAMNYA DALAM BAHASA C++

Diposkan oleh [Putri nur Ihsani](https://plus.google.com/107248947265955632211) di [15.45](http://puthryy.blogspot.co.id/2014/03/fungsi-keyword-keyword-dan-contoh_2.html)

Label: [PUTRI NUR IHSANI](http://puthryy.blogspot.co.id/search/label/PUTRI%20NUR%20IHSANI)

<http://puthryy.blogspot.co.id/2014/03/fungsi-keyword-keyword-dan-contoh_2.html>