**BAHASA PEMROGRAMAN C++**

Albert Soerjonoto / X-I / 2

**1. 1. Sejarah Bahasa C**

Sejarah perkembangan dan latar belakang munculnya bahasa C adalah seperti gambar di bawah ini:

Boleh dikatakan bahwa akar dari bahasa C adalah bahasa BCPL(***Basic Combined Programming Language)*** yang dikembangkan oleh *Martin Richards* pada tahun 1967. Kemudian berdasar pada bahasa BCPL ini *Ken Thompson* yang bekerja di Bell Telephone Laboratories (Bell Labs) mengembangkan bahasa B pada tahun 1970. Saat itu bahasa B telah berhasil diimplementasikan di komputer DEC PDP-7 dengan operating system (OS) UNIX. Pada tahun 1972, peneliti lain di Bell Labs bernama *Dennis Ritchie* menyempurnakannya menjadi bahasa C.

Pada tahun 1978, *Dennis Ritchie* bersama dengan *Brian Kernigha*n mempublikasikan buku yang kemudian menjadi legenda dalam sejarah perkembangan bahasa C, yang berjudul *The C Programming Language*. Buku ini diterbitkan oleh Prentice Hall, dan pada saat ini telah diterjemahkan dalam berbagai bahasa di dunia. Boleh dikatakan bahwa buku ini adalah buku yang paling banyak direfer orang dan dijadikan buku panduan tentang pemrograman bahasa C sampai saat ini. Teknik dan gaya penulisan bahasa C yang merefer kepada buku ini kemudian terkenal dengan sebutan *K&R C* atau *Classic C* atau *Common C*.

Seiring dengan berkembang pesatnya bahasa C, banyak vendor mengembangkan kompiler C menurut versi masing-masing. Hal ini menggerakkan ANSI (*American National Standards Institute*) pada tahun 1983 untuk membuat suatu komite yang kemudian diberi nama *X3J11*, yang betujuan untuk membuat definisi standar bahasa C yang lebih modern dan komprehensif, dengan memperbaiki *syntax* dan *grammar* bahasa C. Usaha ini berhasil diselesaikan 5 tahun kemudian, yaitu ditandai dengan lahirnya standard ANSI untuk bahasa C yang kemudian terkenal dengan sebutan *ANSI C* pada tahun 1988.

**1.2. Mengapa Memakai Bahasa C**

Sampai saat ini, bahasa C telah berhasil digunakan untuk mengembangkan berbagai jenis permasalahan pemrograman, dari level operating system (unix, linux, ms dos, dsb), aplikasi perkantoran (text editor, word processor, spreadsheet, dsb), bahkan sampai pengembangan sistem pakar (*expert system*). Kompiler C juga telah tersedia di semua jenis platform komputer, mulai dari Macintosh, UNIX, PC, Micro PC, sampai super komputer.

C bisa disebut bahasa pemrograman tingkat menengah (*middle level programming language*). Arti tingkat (level) disini adalah kemampuan mengakses fungsi-fungsi dan perintah-perintah dasar bahasa mesin/hardware (*machine basic instruction set*). Semakin tinggi tingkat bahasa pemrograman (misalnya: java), semakin mudahlah bahasa pemrograman dipahami manusia, namun membawa pengaruh semakin berkurang kemampuan untuk mengakses langsung instruksi dasar bahasa mesin. Demikian juga sebaliknya dengan bahasa pemrograman tingkat rendah (misalnya: assembler), yang semakin sulit dipahami manusia dan hanya berisi perintah untuk mengakses bahasa mesin.

Dalam perspektif mudahnya dipahami manusia, C bisa digolongkan dalam bahasa tingkat tinggi, namun C juga menyediakan kemampuan yang ada pada bahasa tingkat rendah, misalnya operasi bit, operasi byte, pengaksesan memori, dsb.

Beberapa alasan mengapa memakai bahasa C adalah terangkum dibawah.

**🖰*C* adalah *bahasa pemrograman yang paling populer saat ini***

Dengan banyaknya programmer bahasa C, membawa pengaruh semakin mudahnya kita menemukan pemecahan masalah yang kita dapatkan ketika menulis program dalam bahasa C. Pengaruh positif lain adalah semakin banyaknya kompiler yang dikembangkan untuk berbagai platform (berpengaruh ke portabilitas).

**🖰 C ad*alah bahasa pemrograman yang memiliki portabilitas tinggi***

Program C yang kita tulis untuk satu jenis platform, bisa kita kompile dan jalankan di platform lain dengan tanpa ataupun hanya sedikit perubahan. Ini bisa diwujudkan dengan adanya standarisasi ANSI untuk C.

**🖰*C adalah bahasa pemrograman dengan kata kunci (keyword) sedikit***

Kata kunci disini adalah merupakan fungsi ataupun kata dasar yang disediakan oleh kompiler suatu bahasa pemrograman. Hal ini membawa pengaruh semakin mudahnya kita menulis program dengan C. Pengaruh lain dari sedikitnya kata kunci ini adalah proses eksekusi program C yang sangat cepat. C hanya menyediakan 32 kata kunci seperti terangkum dibawah:

**🖰*C adalah bahasa pemrograman yang fleksibel***

Dengan menguasai bahasa C, kita bisa menulis dan mengembangkan berbagai jenis program mulai dari operating system, word processor, graphic processor, spreadsheets, ataupun kompiler untuk suatu bahasa pemrograman.

**🖰*C adalah bahasa pemrograman yang bersifat moduler***

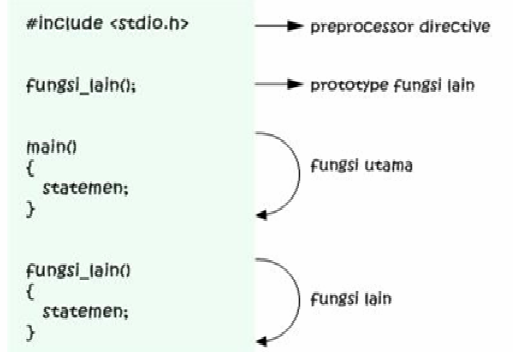
Program C ditulis dalam *routine* yang biasa dipanggil dengan fungsi. Fungsi-fungsi yang telah kita buat, bisa kita gunakan kembali (*reuse*) dalam program ataupun aplikasi lain.

Berbicara tentang C++ biasanya tidak lepas dari C, sebagai bahasa pendahulunya.Pencipta C adalah yang sudah di jelaskan di atas,dan sekitar satu dekade setelahnya diciptakanlah C++, oleh Bjarne Stroustrup dariLaboratorium Bell, AT&T, pada tahun 1983. C++ cukup kompatibel dengan bahasapendahulunya C. Pada mulanya C++ disebut “ a better C “. Nama C++ sendiri diberikanoleh Rick Mascitti pada tahun 1983, yang berasal dari operator increment pada bahasa C.Keistimewaan yang sangat berari dari C++ ini adalah karena bahasa ini mendukungpemrograman yang berorientasi objek ( OOP / Object Oriented Programming)

**1.3. Struktur Program Bahasa C++**

Program bahasa C++ adalah suatu program terdiri dari satu atau lebih fungsi-fungsi. Fungsi utama dan harus ada pada program C++ yang kita buat adalah fungsi **main()**. Fungsi main() ini adalah ***fungsi pertama yang akan diproses pada saat program di kompile dan dijalankan, sehingga bisa disebut sebagai fungsi yang mengontrol fungsi-fungsi lain***. Karena struktur program C++ terdiri dari fungsi-fungsi lain sebagai program bagian (*subroutine*), maka bahasa C++ biasa disebut sebagai *bahasa pemrograman terstruktur*. Cara penulisan fungsi pada program bahasa C++ adalah dengan memberi nama fungsi dan kemudian dibuka dengan kurang kurawal buka (**{**) dan ditutup dengan kurung kurawal tutup (**}**). Fungsi-fungsi lain selain fungsi utama bisa dituliskan setelah atau sebelum fungsi utama dengan deskripsi prototype fungsi pada bagian awal program. Bisa juga dituliskan pada file lain yang apabila kita ingin memakai atau memanggil fungsi dalam file lain tersebut, kita harus menuliskan header filenya, dengan *preprocessor directiv*e **#include**. File ini disebut file pustaka (*library file*). Untuk lebih jelas tentang struktur program bahasa C++ ini, silakan melihat pada Gambar 3 dibawah.

**Gambar 3: Struktur Dasar Program C++**

****

**Contoh Program :**

#include<stdio.h>

#include<iostream.h>

#include<conio.h>

garis();

main()

{

char a[10];

cout<<"Masukkan Nama :";

scanf("%s",&a);

printf("Hello %s \nMau Kemana? Aku ikut donk... ",a);

getch();

}

garis()

{

cout<<”===============”<<endl;

}

**Beberapa File Header**

|  |  |
| --- | --- |
| alloc.h | Untuk manajemen memori |
| bios.h | Untuk memanggil routines ROM BIOs |
| complex.h | Untuk fungsi matematika komplek (dengan C++) |
| conio.h | Untuk memanggil console DOS I/O (input output) |
| float.h | Berisi parameter untuk routing float point |
| graphics.h | Untuk prototype fungsi grafik |
| iostream.h | Untuk routing I/O basicdi C++ |
| math.h | Untuk prototype fungsi matematika |
| stdio.h | Mendefinisikan tipe dan macro untuk paket standar I/O yang dikembangkan oleh Kernighan & Ritchie |
| stdlib.h | Untuk beberapa fungsi rutin umum : konversi dll. |

**Escape Squence**

|  |  |
| --- | --- |
| \a | Alert (peringatan/ bunyi bel) |
| \n | Baris baru (new line) |
| \b | Backspace |
| \r | Catridge return (enter |
| \f | Formfeed |
| \t | Tab |
| \v | Vertical tab |
| \’ | Tanda petik tunggal |
| \” | Tenda petik |

**Komentar**

/\* ...... \*/ menandakan untuk komentar yang lebih dari satu baris

// menandakan untuk komentar satu baris

// ----------------------------------

// ini adalah komentarku yang pertama

// ----------------------------------

atau

/\* ----------------------------------

ini adalah komentarku yang pertama

---------------------------------- \*/

**1.4. Bahasa C++ mempunyai elemen dasar:**

1. **Pengenal(identifier)**

Nama yang digunakan sebagai tempat untuk variabel, fungsi dan konstanta yang didefinisikan oleh pemrogram.

**Ketentuan-ketentuan khusus:**

* 1. Jgn diawali dengan angka
  2. Tidak boleh menggunakan operator2 aritmatika/karakter2 khusus.
  3. Tidak boleh ada spasi
  4. Tidak menggunakan kata-kata yanng dimiliki oleh bahasa C++
  5. Huruf besar dan huruf kecil dibedakan
  6. Panjang maksimal suatu variabel 32 karakter

**Contoh :**

Halo123

Rata\_rata\_Nilai\_Ujian\_Untuk\_Siswa\_Kelas\_X

1. **Kata kunci(keyword)**

Kata-kata yang digunakan oleh kompiler dan tidak digunakan untuk pemberian variabel, nama pengenal atau nama fungsi.

Contoh Kata Kunci:

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. **Tipe data dasar**

Tipe data merupakan bagian program yang paling penting karena tipe data mempengaruhi setiap instruksi yang akan dilaksanakan oleh computer. Misalnya saja 5 dibagi 2 bisa saja menghasilkan hasil yang berbeda tergantung tipe datanya. Jika 5 dan 2 bertipe integer maka akan menghasilkan nilai 2, namun jika keduanya bertipe float maka akan menghasilkan nilai 2.5000000. Pemilihan tipe data yang tepat akan membuat proses operasi data menjadi lebih efisien dan efektif.

Data berdasarkan jenisnya dapat dibagi menjadi tiga tipe data dasar, yaitu:

* 1. **Tipe integer**

Digunakan untuk menyimpan data bernilai numerik atau bilangan bulat, diideklarasikan dengan **int, long int**

Batasan int : -32768 byte sampai 32767 byte

Batasan long int : -2147483648 sampai 2147483649

* 1. **Tipe float**

Digunakan untuk menyimpan data bernilai numerik pecahan atau bilangan real, diideklarasikan dengan **float**

Batasan float : 3.4e-38 sampai 3.4e+38

* 1. **Tipe char**

Digunakan untuk menyimpan data berupa karakter, yakni gabungan huruf, simbol dan angka, dideklarasikan dengan **char**

Batasan char : -128 sampai 127

* 1. **Void**

Jenistipe data tidakbertipe.Digunakanuntukfungsi main.

Dalam bahasa C terdapat lima tipe data dasar, yaitu :

**No** **Tipe Data**  **Ukuran** **Range**  **Format**  **Keterangan**

1 char 1 byte -128 s/d 127 %c Karakter/string

2 int 2 byte - 32768 s/d 32767 %i , %d Integer/bilangan bulat

3 float `4 byte - 3.4E-38 s/d 3.4E+38 %f Float/bilangan pecahan

4 double 8 byte - 1.7E-308 s/d1.7+308 %lf Pecahan presisi ganda

5 void 0 byte - - Tidak bertipe

6 String - - %s String

C++ memiliki 7 tipe data dasar :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tipe**  **Data** | **Ukuran**  **Memori** | **Jangkauan Nilai** | **Format** | **Jumlah**  **Digit** |
| Char | 1 Byte | -128 s.d 127 |  |  |
| Int | 2 Byte | -32768 s.d 32767 |  |  |
| Short | 2 Byte | -32768 s.d 32767 |  |  |
| Long | 4 Byte | -2,147,435,648 s.d 2,147,435,647 |  |  |
| Float | 4 Byte | 3.4 x 10-38 s.d 3.4 x 10+38 |  | 5 – 7 |
| Double | 8 Byte | 1.7 x 10-308 s.d 1.7 x 10+308 |  | 15 – 16 |
| Long  Double | 10 Byte | 3.4 x 10-4932 s.d 1.1 x 10+4932 |  | 19 |

1. **Variabel**

Adalah suatu pengenal untuk menyimpan nilai tertentu yang dapat diubah-ubah.

Syarat pemberian namavariabel sama dengan dengan pemberian nama pengenal.

1. **Konstanta**

Adalah suatu pengenal untuk menyimpan nilai tertentu yang tetap/tdk dpt diubah.

*a. Menggunakan keyword const*

Contoh :const float PI = 3.14152965;

Berbedadengan variable, konstantabernamatidakdapatdiubahjikatelahdiinisialisasi

b. *Menggunakan* #define

Contoh : #define PI 3.14152965

Keuntunganmenggunakan #define apabiladibandingkandengan**const**adalahkecepatankompilasi, karenasebelumkompilasidilaksanakan, kompilerpertamakalimencari symbol #define (olehsebabitumengapa # dikatakanpreprocessor directive) danmenggantisemua Phi dengannilai 3.14152965.

**ContohPenggunaanKonstanta :**



1. **Pernyataan**

Adalah suatu instruksi / perintah yang akan dikerjakan selama eksekusi program berlangsung.

1. **Operator**

Operator adalah simbol yg biasa dilibatkan dalam program untuk operasi atau manipulasi.

***Sifat Operator***

|  |  |  |
| --- | --- | --- |
| Sifat | Keterangan | Contoh |
| Unary | Melibatkan satu operand | -1 |
| Binary | Melibatkan dua operand | 1+2 |
| Ternary | Melibatkan tiga atau lebih operand | (a>b)?a:b |

Contoh (ternary):

**3 + 2 - 1**

operator operand

**1. a. Operator Aritmatika** dikenal sebagai operator yang digunakan untuk operasi perhitungan matematika atau disebut juga operator binary.

|  |  |  |  |
| --- | --- | --- | --- |
| Operator | Prioritas | Keterangan | Contoh |
| \* | + - - Tertinggi | Perkalian | 2 \* 3 |
| / | - | Pembagian | 7 / 2 |
| % | \* / % | Sisa pembagian | 7 % 2 |
| + | + - Terendah | Penjumlahan | 5 + 4 |
| - |  | Pengurangan | 5 – 4 |

**1. b. Operator Penurunan Dan Penaikan** adalah operator yang digunakan untuk menurunkan dan menaikkan nilai variabel sebesar satu, penempatannya dapat dilakukan di muka atau di belakannya.

|  |  |
| --- | --- |
| Operator | Keterangan |
| + + | Operator Penaikkan |
| - - | Operator Penurunan |

Contoh :

X = x + 1; dapat ditulis menjadi + + x (increment pre fixed)

atau x + + (increment post fixed)

Y = y – 1; dapat ditulis menjadi y - - (decrement post fixed)

atau - - y (decrement pre fixed)

**2. Operator Penugasan/Penunjukkan** adalah operator pemberian nilai berupa simbol sama dengan (=)

|  |  |
| --- | --- |
| Operator | Keterangan |
| = | Memberikannilai |
| \*= | Mengalikanvariabel di sisikiridengannilai di sisikanan |
| /= | Membagikanvariabel di sisikiridengannilai di sisikanan |
| %= | Memperolehsisapembagianantaravariabeldisisikiridengannilai di sisikanan |
| += | Menambahkanvariabel di sisikiridengannilai di sisikanan |
| -= | Mengurangivariabel di sisikiridengannilai di sisikanan |

Contoh :

* A = 25; //artinya A diberi nilai 25 desimal.
* C = sin(B); //artinya C diberi nilai fungsi sin
* C \*= 3; //artinya C diberi nilai C sebelumnya dikali 3 (ekivalen dengan C=C\*3)
* C += 14; //artinya C diberi nilai C sebelumnya ditambah 14 (ekivalen dengan C=C+14)

**3. Operator Relasi** adalah operator untuk membandingkan dua buah nilai.

|  |  |
| --- | --- |
| Operator | Keterangan |
| = = | Sama Dengan ( bukan penugasan) |
| != | Tidak Sama Dengan |
| > | Lebih Dari |
| < | Kurang Dari |
| >= | Lebih Dari atau Sama Dengan |
| <= | Kurang Dari atau Sama Dengan |

**4. Operator Logika** adalah digunakan untuk menghubungkan dua buah ungkapan kondisi menjadi sebuah ungkapan kondisi. Operator ini berupa:

* + && (operator logika AND)
  + || (operator logika OR)
  + ! (operator logika NOT)

**1.5. Perintah Output dan Input**

*Ada beberapa perintah Output(mencetak ke layar) yaitu* :

**printf();**

Fungsi keluaran yang paling umum digunakan untuk menampilkan informasi ke layar.

Bentuk Umum :

**printf(“penentu format”, argumen\_1,argumen\_2);**

**printf(“penentu format”, variabel\_1,variabel\_2);**

***Penentu format printf(), adalah:***

|  |  |
| --- | --- |
| Tipe data | Penentu format |
| Integer | **%d atau %i** |
| Float desimal | **%f** |
| Float pangkat | %e |
| Float desimal dan pangkat | %g |
| Character | **%c** |
| String | **%s** |
| Long integer | **%ld** |

**puts();**

adalah perintah sama dengan printf() yaitu untuk mencetak string ke layar, asal kata put string, tidak perlu %s dan \n.

***Perbedaanfungsiputs() denganprintf()***

|  |  |
| --- | --- |
| ***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 :** #include <stdio.h>

#include <conio.h>

main()

{

char a[4] = "BSI";

clrscr();

puts("Saya Kuliah di. ");

puts(a);

}

**putchar();**

adalah perintah untuk menampilkan sebuah karakter ke layar, penampilannya tidak pindah baris.

**Contoh :** #include <stdio.h>

#include <conio.h>

main()

{

putchar('B');

putchar('S');

putchar('I');

}

**cout<<;**

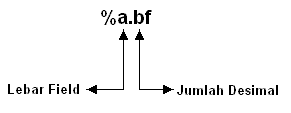
adalah sebuah perintah untuk menampilkan data ke layar, harus menyertakan file header iostream.h.

Bentuk Umum:

**cout<<"…pernyataan…"<<nama\_variabel<<perintah;**

*Di bawah ini perintah input(memasukkan data melalui keyboard) yaitu :*

**Catatan :** 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 :



**Contoh :** #include <stdio.h>

#include <conio.h>

main()

{

float a = 7.50, b = 243.21;

printf("Bilangan A = %f \n", a);

printf("Bilangan B = %f", b);

}

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

**Bilangan A = 7.500000**

**Bilangan B = 243.210007**

**Contoh:** #include <stdio.h>

#include <conio.h>

main()

{ float a = 7.50, b = 243.21;

printf("Bilangan A = %4.1f \n", a);

printf("Bilangan B = %4.1f", b);}

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

**Bilangan A = 7.5**

**Bilangan B = 243.2**

*Di bawah ini perintah input(memasukkan data melalui keyboard) yaitu :*

* **scanf()**
* **gets()**
* **cin>>**
* **getch()**
* **getche()**

**scanf()**

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

**scanf(**"**penentu format", &nama-variabel);**

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

***Penentu Format scanf()***

|  |  |
| --- | --- |
| **TIPE DATA** | **Penentu Format Untuk *scanf()*** |
| Integer | **%d atau %i** |
| 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 |

**Contoh :** # include <stdio.h>

# include <conio.h>

main()

{

int a, b, c = 0 ;

printf("Masukan Nilai A = "); scanf("%d",&a);

printf("Masukan Nilai B = "); scanf("%d",&b);

c = a + b;

printf("Hasil Penjumlahan = %d",c);

}

**gets()**

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

**gets(**nama-variabel-array**);**

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

|  |  |
| --- | --- |
| ***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. |

**Contoh :** # include <stdio.h>

# include <conio.h>

main()

{

char nm1[20];

char nm2[20];

puts("Masukan nama ke - 1 = ");

gets(nm1);

printf("Masukan nama ke - 2 = ");

scanf("%s",&nm2);

printf("\n\n");

puts("Senang Berkenalan Dengan Anda ..");

puts(nm1);

printf("Senang Berkenalan Dangan Anda ..%s", nm1);

puts("Senang Berkenalan Dangan Anda ..");

puts(nm1);

printf("\n\n");

puts("Senang Berkenalan Dangan Anda ..");

puts(nm2);

printf("Senang Berkenalan Dangan Anda ..%s", nm2);

}

**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:** # include <stdio.h>

# include <conio.h>

# include <iostream.h>

main()

{

float a, b, c;

cout<<"Masukan Nilai A : ";

cin>>a;

cout<<"Masukan Nilai B : ";

cin>>b;

c = a + b;

cout<<"Masukan Nilai C : "<<c<<endl;

}

**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:** # include <stdio.h>

# include <conio.h>

main()

{

char kar;

printf("Masukan Sebuah Karakter Bebas = ");

kar = getch();

printf("\nTadi Anda Memasukan karakter %c", kar);

getch();

}

**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 :** # include <stdio.h>

# include <conio.h>

main()

{

char kar;

printf("Masukan Sebuah Karakter Bebas = ");

kar = getche();

printf("\nTadi Anda Memasukan karakter %c", kar);

getch ();

}

**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 C++, antara lain.

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

Berikut akan dibahas beberapa fungsi manipulator, diantaranya :

**endl**

***Penjelasan* *endl*** merupakan suatu fungsi manipulator yang digunakanuntukmenyisipkankarakterNewLineataumengaturpindahbaris. Fungsi ini sangat berguna untuk piranti keluaran berupa file di disk. File header yang harus disertakan adalah file header **iostream.h** .

**Contoh:** # include <stdio.h>

# include <conio.h>

# include <iostream.h>

main()

{

float a, b, c;

a=7.5; b=8.4; c=0

cout<<"Masukan Nilai A : "<<a<<**endl**;

cout<<"Masukan Nilai B : "<<b<<**endl**;

c = a + b;

cout<<"Masukan Nilai C : "<<c<<**endl**;

getch();

}

**dec, octdan hex**

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

**Contoh :** # include <stdio.h>

# include <conio.h>

# include <iostream.h>

# include <iomanip.h>

main()

{

int nilai = 10;

cout<<"Nilai = "<<nilai<<endl;

cout<<"Nilai ke Octal = "<<oct<<nilai<<endl;

cout<<"Nilai ke Hexadesimal = "<<hex<<nilai<<endl;

cout<<"Nilai ke Desimal = "<<dec<<nilai<<endl;

getch();

}

**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** .

**Contoh :** # include <stdio.h>

# include <conio.h>

# include <iostream.h>

# include <iomanip.h>

main()

{

float a, b, c;

a = 25.77;

b = 23.45;

c = a \* b;

cout<<setiosflags(ios::fixed);

cout<<setprecision(1)<<c<<endl;

cout<<setprecision(2)<<c<<endl;

cout<<setprecision(3)<<c<<endl;

cout<<setprecision(4)<<c<<endl;

cout<<setprecision(5)<<c<<endl;

getch();

}