**Header dan Fungsi main()**

Beberapa bagian yang harus di perhatikan adalah ***header*** dan ***fungsi***. **Header**adalah diana kita memasukkan  library yang ada dalam C++  kedalam kode program kita.**Fungsi** adalah “**tempat**” dimana kita bisa “bereksperimen”. Dimana didalamnya terdapat semua ***statement***yang kita buat. Program yang paling sederhana adalah:

#include

#include

void main () {

                        cout << “selamat pagi\n”;

getch();

                  }

Pada baris pertama ada perintah #include . Tanda ‘#’ **adalah** *preprocessor directive***.**

**Preprocessor directive** adalah perintah – perintah yang diberikan kepada compiler untuk melakukan definisi, memasukkan file library, dll.

Contoh – contoh dari preprocessor ini adalah:

         #define maximum\_number 100000

         #include “fileku.h”

         #ifdef maximum\_number

#undef maximum\_number

#define maximum\_number\_2 1000

#endif

Itu artinya kita memasukkan **(*include*)** library**iostream**ke dalam kode program kita.**iostream** itu adalah header yang dibutuhkan untuk “kegiatan” input dan output dari program.

Perintah cout sendiri ada karena kita memasukkan iostream ke dalam kode program kita dan juga using namespace std; dimana ada *member* coutdi dalamnya.

Berikut beberapa header yang terdapat dalam C++:

         ctime =  header waktu

         cstdio = header untuk perintah C seperti printf dan scanf

         cmath = header untuk operasi matematika seperti sin, cos, sqrt, dan tan

         cstring = header untuk memanipulasi string seperti strcpy, strcat.

         =berisi makro dan informasi untuk diagnosis yang membantu          pemeriksaan kesalahan proses dalam program.

         =mengandung prototipe fungsi untuk menguji properti dari karakter, dan dapat digunakan untuk mengubah karakter huruf kecil ke besar.

         =berisi batasan di dalam sistem bilangan bertipe float.  berisi batasan sistem integral.

         =berisi prototipe fungsi dan informasi lainnya yang dapat memodifikasi sistem data pada saat program diproses.

         =berisi prototipe fungsi untuk pustaka matematika.

         =berisi prototipe fungsi yang menuntun urutan pemanggilan fungsi.

         =berisi prototipe fungsi dan makro untuk menangani kondisi yang bermacam-macam saat program diproses.

         =mendefinisikan makro untuk menangani daftar argumen fungsi yang nilai dan tipenya tidak diketahui.

         =berisi definisi umum dari tipe yang digunakan C untuk membentuk perhitungan tertentu.

         =berisi prototipe fungsi untuk fungsi pustaka baku input/output.

         =berisi prototipe fungsi untuk konversi nilai ke text atau sebaliknya, alokasi memori, bilangan acak, dan utilitas lainnya.

         =berisi prototipe fungsi untuk pemrosesan string.

         =berisi prototipe fungsi dan tipe untuk memanipulasi data waktu (jam dan tanggal)

main () **adalah** fungsi utama (*main*) yang harus ada dalam setiap program C++ kita. Di sini yang digunakan adalah dengan main tanpa nilai kembalian dan tanpa paramater.

Jika diperhatikan ada perintah \n pada akhir “kalimat” selamat pagi.”\n” **adalah**perintah dalam banyak bahasa pemrograman sebagai baris baru (*new line*).

Perintah dengan menggunkan *backslash* (‘\’) juga ada beberapa jenis, diantaranya:

         ‘\a’ = untuk menghasilkan bunyi alert

         ‘\t’= tabulasi

         ‘\0’ = *null terminated*dalam array

Tanda ‘{‘ juga harus ada dalam setiap fungsi C++, itu menandakan awal dari fungsi / program dan ‘}’ menandakan akhir dari fungsi / program.

**PENULISAN DALAM BAHASA C++**

Penulisan dalam bahasa C++ adalah *case* *sensitive*. Yang berarti, C++ membedakan huruf kecil dan kapital. Yang berarti (lagi): variabel\_pertama tidak sama denganvariabel\_Pertama

Komentar dalam C++ ditulis dengan ‘//’ untuk komentar dengan panjang satu baris atau ‘/\*’ dan ‘\*/’ untuk komentar lebih dari satu baris, dimana tanda /\* merupakan “pembuka” komentar dan \*/ adalah “penutup” komentar. Komentar adalah cataatan ang ditulis oleh programmer untuk sebagai pengingat atau penjelasan ketika membaca sebuah kode. Kata – kata dalam sebuah komentar tidak akan dieksekusi oleh compiler.

Beberapa kesalahan dalam penulisan dalam C++  adalah:

         int int; /\*salah karena nama sebuah variabel tidak boleh sama dengan seuah tipe data (int)\*/

         int Jam\_9; //salah karena nama variabel dimulai dengan huruf besar

         int 90jam; /\*nah, ini juga salah karena nama sebuah variabel tidak boleh dimulai dengan bilangan\*/

Penulisan yang benar dalam C++ adalah:

         int myint; //betul

         char kata\_2; //betul juga...

         double bilangan\_KeDuA; /\*walaupun nama identifiernya  aneh, tapi ini juga betul :) \*/

Setiap statement dalam C++ harus diberi tanda ‘;’ pada akhir dari statement. Contoh penulisan statement yang benar dalam C++ adalah:

int a, b, c;

a = 1;

b = 2;

c = a + b;

Sedangkan penulisan yang salah adalah:

int a, b, c

karena itu dianggap error oleh  compiler, karena merupakan statement yang tidak lengkap.

**TIPE DATA DAN VARIABEL**

Tipe – tipe data yang dikenal dalam C++ adalah sebagai berikut:

**int**

int adalah tipe data bilangan bulat (*integer*). Contoh penggunaan tipe int adalah:

int a = 1;

int num1 = 2, num2 = 4;

Perhatikan bahwa dalam nilai – nilai yang masuk di *assign* (masukkan) ke dalam variabel – variabel bertipe int adalah angka – angka / bilangan bulat. Seperti namanya,int **hanya** bisa bernilai bulat dan bukan pecahan (desimal). Bagaimana kalau yang tipenya int lalu yang dimasukkan adalah hasil pembagian yang tidak bulat hasilnya? Maka:

int hasil\_bagi\_bulat = 5 / 2;

Hasilnya:

2

Ini karena 5 dibagi 2 = 2,5 kemudian karena hasil\_bagi\_bulat adalah integer (bilangan bulat) 2,5 kemudian dibulatkan menjadi 2.

Untuk bilangan desimal sendiri ada tipe tertentu untuk bilangan ini.

**VOID**

Variabel yang berupa void merupakan variable yang tanpa isi tipe data dan pengembalian nilai

Menyatakan tipe kosong untuk :  
1. mendeklarasikan fungsi yang tidak mengembalikan nilai apapun.  
2. mendeklarasikan fungsi yang tidak menerima parameter apapun.  
3. bila diawali dengan operator \*, menyatakan penunjuk terhadap  
sembarang tipe data.

Contoh deklarasi void :  
Void cctrputs (char\*, int ) ; atau  
Main (void) ; atau  
Void\* action ;  
Int ivalue = 100 ;  
Action = &ivalue ;

**FLOAT**

Variable float digunakan untuk menunjukkan bilangan yang berupa pecahan,namun jangkauan datanya masih rendah.

Menyatakan bilangan pecahan/real, maupun eksponensial. Dalam keadaan  
default, bilang floting point dianggap bertipe double.  
Rentang nilai float mulai 3,4 E -38 sampai 3,4 E +38

**double**

adalah tipe yang khusus untuk bilangan yang tidak bulat alias desimal / pecahan. Contoh penggunaannya adalah:

double mydouble = 5.00;

double pecahan\_pertama = 2.5, pecahan\_kedua = 9.9;

Rentang nilai double mulai 1,7 E -308 sampai 1,7 E +308

**char**

adalah tipe data karakter yang bisa menyimpan mulai dari: a – z, A – Z, ‘!’, ‘@’, dsb. Penggunannya ± sama dengan tipe -  tipe yang sebelumnya:

char huruf = ‘c’;

char inisial1 = ‘A’, inisial2 = ‘W’;

char simbol = ‘#’;

**bool**

adalah tipe yang hanya “menyimpan” nilai “benar” dan “salah” yaitu, *true* dan *false*(boolean). Contoh:

bool benar\_atau\_salah = true;

**string**

adalah tipe yang khusus untuk menampung semua masukan dari user / gabungan karakter yang sangat panjang. Misalnya:

Char text [ ]

**STRUCT, UNION**  
Digunakan untuk mendeklarasikan sekelompok data yang memiliki tipe yang  
berlainan. Struct : elemennya ada dilokasi memori yang berbeda, dan union :  
elemennya ada dilokasi memori yang sama.

                              Struct tipestruktur  
            {  
              Tipeanggota1 namaAnggota1 ;  
              Tipeanggota2 namaAnggota2 ;  
              ………….  
            }  
              namaStruktur ;

**VARIABEL**

Variabel adalah suatu pengenal (identifier) yang digunakan untuk mewakili

suatu nilai tertentu di dalam proses program. Berbeda dengan konstanta

yang nilainya selalu tetap, nilai dari suatu variable bisa diubah-ubah sesuai

kebutuhan.

Nama dari suatu variable dapat ditentukan sendiri oleh

pemrogram dengan aturan sebagai berikut :

Terdiri dari gabungan huruf dan angka dengan karakter pertama

harus berupa huruf. Bahasa C bersifat case-sensitive artinya huruf

besar dan kecil dianggap berbeda. Jadi antara **nim, NIM**dan **Nim**

dianggap berbeda.

Tidak boleh mengandung spasi.

Tidak boleh mengandung symbol-simbol khusus, kecuali garis bawah

(underscore). Yang termasuk symbol khusus yang tidak

diperbolehkan antara lain : $, ?, %, #, !, &, \*, (, ), -, +, = dsb

Panjangnya bebas, tetapi hanya 32 karakter pertama yang terpakai.

Contoh penamaan variabel yang benar :

NIM, a, x, nama\_mhs, f3098, f4, nilai, budi, dsb.

Contoh penamaan variable yang salah :

%nilai\_mahasiswa, 80mahasiswa, rata-rata, ada spasi, penting!, dsb.

Contoh penulisan variable:

#include

#include

void main()

{

int a, b;

int hasil;

a=8;

b=1;

hasil=a-b;

cout<

              getch()

         }

**OPERATOR ARITMATIKA**

Operator aritmatika yang sering digunakan dalam C++ adalah:

         ‘+’ = penjumlahan

         ‘-‘ = pengurangan

         ‘\*’ = perkalian

         ‘/’ = pembagian

         ‘%’ = sisa bagi (modulo)

Contoh dari pemakaian masing – masing operator tersebut adalah sebagai berikut:

#include

#include

void main ()

{

      int satu, dua, tiga, empat, lima;

      satu = 1 + 0;

      dua = 4 - 2;

      tiga = 1 \* 3;

      empat = 20 / 5;

      //mencari sisa bagi dari 25 / 10

      lima = 25 % 10;

      cout << satu << “\n”;

      cout << dua << “\n”;

      cout << tiga << “\n”;

      cout << empat << “\n”;

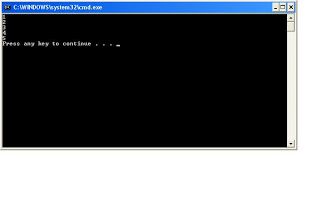
      cout << lima << “\n”;

      getch();

}

Hasilnya adalah:

Adapun operator lain dalam borlan C++ yaitu:

[](http://4.bp.blogspot.com/_SsaDsK9u96w/S1VI3C9-NfI/AAAAAAAAADA/Szh9hhBPlok/s1600-h/untitled.JPG)

1.Operator Majemuk

‘+=’ artinya x = x+y

‘\*=’ artinya x = x\*y

‘/=’  artinya x = x/y

‘%=’artinya x = x % y

2. operator penaikan dan penurunan

Operator ini biasanya dilambangkan dengan tanda ++ jika mengalami penaikan dan – jika mengalami penurunan. Penaikan dan penurunan yang terjadi hanya 1 variabel.

Biasanya saat menulis program kita temukan seperti ini :

            a++;

     a+1;

     a=a+1;

3.operator relasional

Operator ini biasanya bertujuan untuk mencari nilai kebenaran apakah “true” atau “false” dalam bahasa C++ biasanya dikatakan bool.

Adapun beberapa operator relasinal yaitu :

* > artinya lebih besar
* >= artinya lebih besar atau sama dengan
* < artinya lebih kecil
* <= artinya lebih kecil atau sama dengan
* = = artinya sama dengan
* != artinya tidak sama dengan

Contoh :

5>3                        bernilai **true**

9<=1          bernilai **false**

4. operator logical

* !           not
* \* /        perkalian pembagian
* &&      AND
* ||           OR

**STRUKTUR KONTROL**

**STRUKTUR KONTROL**

**Ada beberapa struktur control ada beberapa yaitu:**

**1.Seleksi kondisional (if ………. else……..)**

Fungsi kondisi (if) : Digunakan untuk membandingkan suatu pernyataan.

Contohnya :

If(kondisi)

Pada kondisi diisi dengan pernyataan yang diinginkan

            Contoh: (nilai<60)

Pada statement diisi dengan perintah yang ingin di jalankan jika kondisi terpenuhi

            Contoh : printf(“nilai:c”);

Jadi disimpulakan contoh:

If(nilai<60)

     Print(“nilai anda :C”);

Namun jika menggunakan if…else akan memuat dua perintah yaitu perintah dimana jika penyataan terpenuhi dan tidak terpenuhi.

Penulisan :

if(kondisi)

else

//apabila statement A tidak terpenuhi

Contoh

Bila jumlah mahasiswa yang membolos lebih dari 10 maka tampilkan hasil “tidak boleh mengikuti ujian”,jika tidak maka “boleh mengikuti ujian”.

if(bolos>10)

  printf(“tidak boleh ujian”);

else

  printf(“boleh ujian);

jika ingin menggunakan alternatif kondisi lebih dari satu dapat digunakan else if.

Contoh:

if(ukuran>=80)

  printf(“XL”);

else if(ukuran<=20)

  printf(“M”);

else

  printf(“S”);

**2.PERULANGAN (LOOPING)**

perulangan digunakan untuk melakukan suatu proses secara berulang-ulang, jika suatu kondisi dipenuhi atau sebaliknya.

Proses looping yang menggunakan penghitungan dapat dibuat menggunakan pernyataan FOR.

Bentuk umum :

for (ungkapan1; keadaan; ungkapan 3)

{

pernyataan;

}

Bentuk diatas berarti:

a)      Ungkapan 1 adalah inisialisasi terhadap variabel pengendali looping

b)      Keadaan adalah kondisi untuk keluar dari looping

c)      Ungkapan 3 adalah pengatur kenaikan atau penurunan nilai variabel pengendali looping

Contoh :

#include

#include

void main ()

{

   for (int n=10; n>0; n--)

{

   cout<<”, “;

}

   cout<<”STOP!”;

   getch ();

}

Hasilnya adalah:

10, 9, 8, 7, 6, 5, 4, 3, 2, 1, STOP!

**3.Perintah do....while**

Perintah ini menyatakan pengulangan proses selama kondisi tertentu.

**Bentuk umum**:

Do

{

Pernyataan;

Pernyataan;

}

While (keadaan)

Setiap loop dikerjakan maka kondisi akan dicek. Jika masih benar, proses loop dilakukan lagi dan jika salah maka proses loop berhenti dan berlanjut pada perintah selanjutnya.

contoh:

#include

#include

void main ()

{

   int a;

   a =1;

   do

{

   cout<<”ayu”;

   a++

}  while (a<=5);

   getch();

}

**4.Perintah WHILE**

Perintah WHILE prinsipnya sama dengan perintah DO...WHILE hanya pengujian kondisinya terletak pada awal loop.

Bentuk umum:

While (keadaan)

Pernyataan;

Atau

While (keadaan)

{

Pernyataan;

Pernyataan;

}

* Pernyataan dapat berupa pernyataan tunggal atau beberapa pernyataan yang dibatasi dengan tanda {}.
* Pernyataan itu akan dijalankan jika kondisinya masih benar.
* Keadaan disini boleh merupakan sembarang pernyataan yang mengandung nilai suatu variabel, pernyataan binari, pernyataan aritmatik, atau nilai return hasil dari memanggil suatu fungsi.

Contoh:

#include

#include

void main ()

{

int n;

cout<<”masukkan angka untuk mulai : “;

cin>>n;

while (n>0)

{

cout<< n << “, “;

--n;

}

cout <<”STOP!”;

getch ();

}

Hasilnya adalah:

Masukkan angka untuk mulai : 4

4, 3, 2, 1, STOP!

**5.Perintah CONTINUE**

Perintah ini digunakan untuk mengarahkan jalannya program ke iterasi (proses) berikutnya pada loop yang sama

Bentuk umum:

While (kondisi)

{

. . . .

. . . .

. . . .

CONTINUE;

. . . .

. . . .

}

. . . .

Contoh program perintah CONTINUE:

#include

#include

void main ()

{

for (int n=10; n>0; n--)

{

if (n==5) continue;

cout<<<”, “;

}

cout<<”STOP!”;

getch ();

}

**Hasilnya adalah:**

10, 9, 8, 7, 6, 5, 4, 3, 2, 1, STOP!

**6.Struktur Selektif (switch)**

Logika menggunakkan switvh sama dengan logika menggunakkan if .

Bentuk Umum            :

switch (pilihan)

{

  case nilai 1:

  blok pernyataan1

  break;

  case nilai 2 :

  blok pernyataan nilai 2

  break;

  -

  -

  default :

  blok pernyataan default

}

Contoh :

            switch (a)

{

          case 1:

          cout<<”a adalah 1”;

          break;

          case 2:

          cout<<”a adalah 2”;

          break;

          default:

          cout<<”error”;

}

**FUNGSI**

**Fungsi**

Fungsi merupakan blok dari kode yang dirancang untuk melakukan tugas khusus.

Adapun tujuan dari pembuatan fungsi ini adalah :

         Program menjadi terstruktur.

         Program yang besar dapat dipecah menjadi beberapa program yang lebih kecil, yang setiap satu program kecil tersebut mempunyai tugas tertentu.

         Dapat mengurangi duplikasi kode.

         Fungsi dapat dipanggil dari program atau fungsi yang lain.

**Dasar Fungsi**

Pada umumnya fungsi memerlukan masukan yang dinamakan argumen atau parameter. Hasil akhir fungsi  akan berupa sebuah nilai (nilai balik fungsi).

bentuk umum:

Penentu-tipe nama\_fungsi (daftar parameter)

Deklarasi parameter

{

          Tubuh fungsi

}

Penentu tipe berfungsi untuk menentukan tipe keluaran fungsi yang dapat berupa salah satu tipe data C++ yang berlaku, misalnya char atau int. Default tipe fungsi yang tidak disebutkan dianggap sebagai int.

Contoh:

            Inisialisasi ()

     {

          Return (0)

     }

**Pada fungsi diatas dijelaskan sebagai berikut**:

         Penentu tipe fungsi tidak disebutkan, berarti keluaran bertipe int.

         Inisialisasi adalah nama fungsi.

         Tanda () menyatakan bahwa fungsi tidak memiliki parameter.

         Tanda { dan } merupakan awal dan akhir fungsi.

         Return (0) merupakan pernyataan dalam tubuh fungsi.

Suatu fungsi cukup didefinisikan sekali tetapi dapat digunakan beberapa kali. Jika  tubuh fungsi banyak mengandung pernyataan maka pemakaian fungsi dapat menghindari duplikasi kode dan menghemat penulisan program maupun kode dalam memori. Jadi fungsi merupakan semacam subprogram.

Adapun bentuk umumnya adalah sebagai berikut:

Inisialisasi ()

{

Return (0);

}

main()

{

Int x,y;

.... . .

x=inisialisasi();

............. . .

y=inisialisasi()

}

**contoh implementasi fungsi**

/\*program:fungsi1.cpp \*/

#include

/\*deklarasi fungsi\*/

Int findMax(int n1, int n2);

Void PrintMax (int m);

Main ()

{

          Int i = 5;

          Int j = 7;

          Int k;

          k = FindMax (i,j);

          PrintMax(k);

Return 0;

     }

/\* definisi fungsi \*/

Int FindMax (int n1, int n2)

{

          If (n1 >n2)

          {

              Return n1;

          }

          Else

          {

              Return n2

          }

}

Void PrintMax (int m)

{

     Printf (“Bilangan yang terbesar adalah :%d\n”,m);

}

Bila program diatas dijalankan, maka hasilnya adalah sebagai berikut:

Bilangan yang terbesar adalah : 7

**Penjelasan:**

* Program diatas adalah contoh fungsi untuk menampilkan bilangan terbesar diantara 2 buah bilangan.
* Dimana sebelum bilangan pertama dengan bilangan kedua dibandingkan, terlebih dahulu didefinisikan fungsi yaitu:

int findmax (int n1, n2)

     {

          if (n1>n2)

          {

              return n1;

          }

          else

          {

              return n2;

          }

Jika bilangan pertama waktu dibandingkan dengan bilangan kedua menghasilkan hasil terbesar atau angkanya lebih besar maka bilangan pertama yang terbesar. Sedangkan sebaliknya.

* Untuk bilangan pertama diwakili i yang nilainya 5 dan nantinya akan mengisi n1.
* bilangan kedua yang diwakili j yang nilainya 7 dan nantinya akan mengisi n2.

**Parameter Formal dan Parameter Aktual**

* **Parameter formal** adalah variabel yang ada pada daftar parameter dalam fungsi.
* **Parameter aktual** adalah parameter yang dapat berupa variabel atau konstanta maupun ungkapan yang dipakai dalam pemanggilan fungsi.

cara melewatkan parameter adalah :

         Pemanggilan dengan nilai

         Pemanggilan dengan referensi

Contoh :

a...x

            b...y

Dengan cara ini nilai parameter aktual tidak dapat berubah sekalipun nilai parameter formal berubah-ubah sebab x merupakan salinan dari A dan y salinan dari B.

Pada saat pemanggilan suatu fungsi, misalnya:

a bernilai 20 → x juga bernilai 20

            b bernilai 30 → y juga bernilai 30

Selanjutnya nilai x dan y diproses tetapi nilai a dan b akan tetap.

Pemanggilan dengan referensi merupakan usaha melewatkan alamat dari suatu variabel ke dalam fungsi. Dengan pengubahan dilakukan di dalam fungsi untuk mengubah variabel di luar fungsi.

Perhatikan:

Tukar (int \*px, int \*py)

{

          Int z;

          Z=\*px;

          \*px=\*py;

          \*py=z;

          . . . . . . . . .

}

Parameter aktualnya:

            Tukar (&a, &b);

Dalam deklarasi parameternya:

            Int \*px, int \*py

Dalam hal ini, px menunjukkan variabel a dan py menunjukkan variabel b.

(\*adalah tanda pointer)

Dapat digambarkan sebagai berikut:

px                                                                                             py

   alamat a                                                                                            alamat b

               a                                                                                           b

Apabila penggalan program tersebut, kita implementasikan pada program, hasilnya adalah sebagai berikut:

**contoh parameter formal dan aktual**

/\* program: fungsi2.cpp\*/

#include

Void tukar (int x, int y);

Main()

     Int a, b;

     a=99;

     b=11;

     printf (“nilai sebelum pemanggilan fungsi:\n”);

     printf (“nilai a=%d nilai b=%d\n\n”,a,b);

     tukar (a, b);

     printf (“nilai sesudah pemanggilan fungsi: \n”);

     printf (“nilai a=%d nilai b=%d\n\n”,a,b);

}

Void tukar (int px, int py)

{

     Int z;

     z=px;

     px=py;

     py=z;

     printf (“nilai diakhir fungsi : \n”);

     printf (“nilai px=%d niali py=%d\n\n”,px,py);

}

Hasilnya :

Nilai sebelum pemanggilan fungsi:

            nilai a=99 nilai b=11

            nilai diakhir fungsi:

            nilai px=11 nilai py=99

Nilai sesudah pemanggilan fungsi:

            nilai a=99 nilai b=11

**penjelasan:**

Setelah px menunjukkan a dan py menunjukkan b, proses penukaran isi a dan b dilakukan dengan cara sebagai berikut:

            z=px;

            px=py;

py=z;

Dengan melalui tiga pernyataan diatas, nilai a dan nilai b dapat diubah dalam fungsi.

**Rekursi**

Fungsi rekursif adalah suatu fungsi yang memanggil dirinya sendiri, artinya fungsi tersebut dipanggil di dalam tubuh fungsi itu sendiri.

Fungsi faktorial, yang menghitung nilai faktorial dari suatu bilangan bulat positif, merupakan pokok bahasan yang memudahkan pemahaman fungsi rekursif. Berikut adalah fungsi faktorial yang diselesaikan dengan cara biasa:

int faktorial (int n)

     {

     int counter, hasil = 1;

     for (counter = n ; counter >= 1 ; counter--)

          hasil \*= counter;

     return hasil;

}

Fungsi tersebut menunjukkan bahwa nilai faktorial dihitung menggunakan looping sehingga melakukan proses sebagai berikut:

Hasil = 1;

            Hasil = hasil \* n; artinya hasil = n;

            Hasil =hasil \* (n-1) artinya hasil = n x (n-1);

Demikian seterusnya sampai n bernilai 1, atau jika dituliskan sekaligus menjadi:

Faktorial = n! = n x (n-1) x (n-2) .... x 1;

Fungsi ini dapat dituliskan dalam bentuk:

            Faktorial (n) = n! = n x (n-1) !;

Yang menunjukan sifat rekursif dari suatu fungsi, yaitu (n-1)!. Oleh karena itu, fungsi faktorial yang telah ditulis dalam program C++ sebelumnya, dapat ditulis kembali dalam bentuk rekursif sebagai berikut:

int faktorial (int n)

     {

     if (n == 0)

     return 1;

     else

     return (n \* faktorial (n-1));

     }

**Program contoh rekursi untuk faktorial**

/\*program: fungsi3.cpp\*/

#include

Int faktorial (int n)

{

If (n==1)

           Return (1);

Else

           Return (n\*faktorial (n-1));

}

Main ()

{

Int x;

Printf (“mencari nilai faktorial\n”);

Printf (“masukkan nilai x:”) ; scanf (“%d”,&x) ;

Printf (“nilai faktorial dari %d=%d\n”,x,faktorial (x) );

}

Hasilnya:

Mencari nilai faktorial

Masukkan nilai x: 4

Nilai faktorial dari 4=24

**Penjelasan:**

Fungsi tersebut menunjukkan bahwa nilai faktorial dihitung menggunakan looping sehingga melakukan proses sebagai berikuit:

Faktorial (1) = 1;

Faktorial (2) = 1 \* 2=2

Faktorial (3) = 3\*faktorial (2)

                = 3x2=6

Faktorial (4) =  4\*faktorial (3)

                   =4x6=2

**ARAY**

**Pengertian**

**Array** adalah tipe data terstruktur yang berguna untuk menyimpan sejumlah data yang bertipe sama. Bagian penyusun aray disebut elemen array yang masing-masing elemen dapat di akses tersendiri melalui index array yang dapat berupa integer atau string.

|  |  |
| --- | --- |
| D[0] | -45 |
| D[1] | 6 |
| D[2] | 0 |
| D[3] | 72 |
| D[4] | 1543 |
| D[5] | 43 |
| D[6] |  |

contoh variabel bernama D yang mempunyai 7 lokasi memori yang semuanya bertipe int.

Masing-masing nilai dalam setiap lokasi mempunyai identitas berupa nama c dan nomor indeks yang dituliskan di dalam tanda kurung siku ‘[ ]’. Sebagai contoh, 4 adalah nilai dari D[6].

**DEKLARASI ARRAY**

**Variabel array dideklarasikan dengan mencantumkan tipe dan nama variabel yang diikuti dengan banyaknya lokasi memori yang ingin dibuat.**

 Dengan demikian, deklarasi untuk variabel array D di atas adalah:

**Int d[7];**

C++ secara otomatis akan menyediakan lokasi memori sesuai dengan yang dideklarasikan,nomor indeks selalu dimulai dari 0. Nilai suatu variabel array dapat juga diinisialisasi secara langsung pada saat deklarasi, misalnya:

**Int d[7] = {-45, 0, 6, 72, 1543, 43, 4**};

setiap lokasi memori dari variabel array c langsung diisi dengan nilai-nilai yang dituliskan di dalam tanda kurung kurawal.

Banyaknya lokasi memori dapat secara otomatis disediakan sesuai dengan banyaknya nilai yang akan dimasukkan, seperti contoh berikut yang tentunya membuat variabel array dengan 10 lokasi memori:

**Int x[ ] = {10, 15, 12, 5, 13, 9, 6, 17, 25, 31};**

**Contoh program array**

/\*Program:array1. Cpp\*/

     #include

     #define SIZE 12

     main() {

int a [SIZE] = {1, 3, 5, 4, 7, 2, 99, 16, 45, 67, 89, 45};

          int indeks, total = 0;

          for (indeks=0 ; indeks<=SIZE-1 ; indeks++)

              total += a [indeks];

printf (“\nTotal setiap elemen array adalah %d”, total);

          return 0;

     }

Bila program diatas dijalankan, akan muncul hasil:

**Total setiap elemen array adalah 383**

Adapun keterangan dari program diatas adalah sebagai berikut:

Hasil penjumlahan setiap elemen array diperoleh dari jumlah data atau elemen array sebanyak 12 buah yang sudah didefinisikan pada awal program yaitu #define SIZE 12. Kemudian setiap elemen array dari a[0] yang berisi data 1, a[1] yang berisi data 3 dijumlah sampai dengan a[11] yang berisi data 45. Proses penjumlahan dilakukan pada loop dimulai dari 0 sampai data yang terakhir atau elemen terakhir.

**ARRAY DIMENSI SATU**

Struktur array mempunyai satu dimensi, sehingga variabelnya disebut dengan variabel array berdimensi satu

**Bentuknya**:

            Tipe nama\_var [ukuran];

**Dengan:**

            Tipe     : menyatakan jenis elemen array (int, char,unsigned, dll)

            Ukuran            : menyatakan jumlah maksimal elemen array.

**Contoh:**

            Float nilai\_angka[5];

Pada turbo C++, array disimpan dalam memori secara berurutan. Elemen pertama berindeks nol digambarkan sebagai berikut:

|  |
| --- |
| Nilai\_angka[0] |
| Nilai\_angka[1] |
| Nilai\_angka[2] |
| Nilai\_angka[3] |
| Nilai\_angka[4] |

Masing-masing bertipe float dan berjumlah 5 elemen.

Selain itu, deklarasi array juga dapat berupa:

            Static int angka [9] = {1, 2, 3, 4, 5, 6, 7, 8, 9}

Sesuai dengan deklarasi array diatas, maka isi variabel array telah ditentukan yaitu:

            angka[0] bernilai 1

            angka[1] bernilai 2

            angka[2] bernilai 3

            angka[3] bernilai 4

            angka[4] bernilai5

            angka[5] bernilai 6

            angka[6] bernilai 7

            angka[7] bernilai 8

            angka[8] bernilai9

            }

**ARRAY DIMENSI DUA**

Array berdimensi lebih dari satu, yang sering disebut dengan array berdimensi dua.

Sebagai contoh, sebuah matrik C berukuran 2 x 3 dapat dideklarasikan dalam C seperti berikut:

int c[2][3] = {{2,4,1},{5,3,7}};

akan menempati lokasi memori dengan susunan sebagai berikut:

|  |  |  |  |
| --- | --- | --- | --- |
|  | 0 | 1 | 2 |
| 0 | 2 | 4 | 1 |
| 1 | 5 | 3 | 7 |

 Dan definisi variabel untuk setiap elemen tersebut adalah:

|  |  |  |  |
| --- | --- | --- | --- |
|  | 0 | 1 | 2 |
| 0 | b[0][0] | b[0][1] | b[0][2] |
| 1 | b[1][0] | b[1][1] | b[1][2] |

Sebagai implementasi dari keterangan diatas, perhatikan program berikut ini:

**Contoh program array dimensi dua**

/\*Program: array3 .cpp\*/

#include

void printArray (int [] [3]),

main() {

     int matrik1 [2] [3] = { {1, 2, 3}, {4, 5, 6} }

           matrik2 [2] [3] = { 1, 2, 3, 4, 5},

           matrik3 [2] [3] = { {1, 2}, {4} };

     printarray (matrik1);

printarray (matrik2);

printarray (matrik3);

return 0;

}

void printArray (int a[] [3]) {

     int i, j;

     for (i=0; i<=1; i++) {

          for (j=0; j<=2; j++)

              printf(“%d”, a[i][j] );;

          printf (“\n”);

     }

}

Bila program diatas dijalankan, akan muncul hasil:

123

456

123

450

120

400

**Penjelasan:**

Dari program diatas untuk matriks 1, penulisannya adalah 123 456, sedangkan pada matriks 2 penulisannya adalah 123 450. 0 disini mempunyai arti tempat yang disediakan untuk data kolom ke 3 dari baris ke dua tidak diisi. Sedangkan matriks 3 penulisannya adalah 120 400. Dari matriks 3 disini kita bisa melihat bahwa pada baris pertama kolom ketiga data tidak diisi dan dianggap 0 dan pada baris kedua kilom kedua dan ketiga juga tidak diisi juga diisi 0.

Dalam program tersebut, juga digunakan fungsi untuk menampung hasil penjumlahan matriks.

**ARRAY DIMENSI BANYAK**

Array ini seperti array dimensi dua tetapi dapat memiliki ukuran yang lebih besar. Sebenarnya array dimensi banyak ini tidak terlalu sering digunakan, tetapi sewaktu-waktu kalau dimensi yang dibutuhkan banyak, maka array ini sangat memegang peranan yang penting.

**STRING**

**STRING**

**Definisi String**

String merupakan bentuk data yang biasa dipakai dalam pemrograman guna menampung dan memanipulasi data teks. Di dalam  C++, string bukan tipe data tersendiri tetapi hanya merupakan jenis khusus dari array.

String dipergunakan dalam konstanta dan variabel. Suatu konstanta string ditulis dengan diawali dan diakhiri oleh tanda petik. Misalnya:

“turbo C++”

Konstanta string akan disimpan berurutan dalam memori dan diakhiri tanda \0 atau NULL da setiap karakter akan menempati memori sebesar 1 byte.

 Contoh:

..... Turbo C++ \0. . .

Variabel string adalah variabel yang dipakai untuk menyimpan data string,

contoh:

Char nama [20] ;

Instruksi diatas berarti deklarasikan variabel nama dengan panjang maksimal 20 karakter dan bertipe array char.

**Fungsi pada STRING**

Pada C++ disediakan sejumlah fungsi makro yang didefinisikan dengan praprosesor #define.

Makro-makro yang didefinisikan dalam type.h adalah:

1.  Isalnum ()

Fungsi makro ini akan menghasilkan nilai benar (bukan nol) kalau argumennya adalah huruf besar, huruf kecil dan karakter digit ‘0’ sampai ‘9’.

Bentuknya adalah sebagai berikut:

Tipe isalnum (tipe argumen) ;

Contohnya adalah sebagai berikut:

                Int isalnum (int d);

2.  Isalpha ()

Fungsi makro ini akan menghasilkan nilai benar (bukan nol) kalau argumennya adalah huruf bukan angka.

Bentuknya adlah sebagai berikut:

                Tipe isalpha (tipe argumen)

3.  Isdigit ()

Fungsi makro ini akan menghasilkan nilai benar (bukan nol) kalau argumennya adalah karakter digit ‘0’ sampai ‘9’

Bentuknya adalah sebagai berikut:

                Tipe isdigit (tipe argumen)

4.  Islower ()

Fungsi makro ini akan menghasilkan nilai benar (bukan nol) kalau argumennya adalah huruf kecil.

Bentuknya adalah sebagai berikut:

                Tipe islower (tipe argumen)

5.  Isupper ()

Fungsi makro ini akan menghasilkan nilai benar (bukan nol) kalau argumennya adalah hurf besar.

Bentuknya adalah sebagai berikut:

                Tipe isupper (tipe argumen)

6.  Tolower ()

Fungsi makro ini akan menghasilkan huruf kecil jika diisi huruf besar, jika diisi bukan huruf besar maka akan menghasilkan sama dengan argumen itu sendiri.

Bentuknya adalah sebagai berikut:

                Tipe tolower (tipe argumen)

7.  Toupper ()

Fungsi makro ini kebalikan dari tolower yaitu akan menghasilkan huruf besar jika diisi huruf kecil  tetapi jika diisi selain huruf besar akan menghasilkan sama dengan argumen yang diisikan tersebut.

Bentuknya adalah sebagai berikut:

                Tipe toupper (tipe argumen)

8.  Strcpy ()

Fungsi ini dipakai untuk menyalin string asal ke variabel string tujuan, variabel tujuan harus memiliki ukuran yang dapat menampung seluruh karakter string asal.

Bentuknya adalah sebagai berikut:

                Strcpy (tujuan, asal)

9.  Strlen ()

Fungsi ini dipakai untuk menghitung jumlah karakter argumennya.

Bentuknya adalah sebagai berikut:

    Char S[ ] = “HALLO”

Cout<<”panjangnya : \n”, strlen (S);

Maka hasilnya adalah 5

10.  Strcat ()

Fungsi ini dipakai untuk menambah string asal ke bagian akhir string tujuan.

Bentuknya:

Strcat (tujuan, asal)

11.  Strcmp ()

Fungsi ini dipakai untuk membandingkan string str 1 dengan string str 2. Hasil dari fungsi ini bertipe int berupa nilai negatif (str1 kurang dari str2), nol(str1 = str2), positif (str 1 lebih dari str 2).

**Contoh program:**

#include

#include

#include

void main()

{

char kalimat [50];

int jum\_kecil=0;

int jum\_kapital=0;

int jum\_pungtuasi=0;//tanda pungtuasi (seperti !/?)

int jum\_digit=0; //karakter digit ( 0 sampai 9)

clrscr();

cout<<"masukkan string apa saja "<

cin.getline (kalimat,50);

for (int i=0; kalimat[i]; i++)

{

if (islower (kalimat [i]))

jum\_kecil++;

if (isupper (kalimat[i]))

jum\_kapital++;

if (ispunct (kalimat[i]))

jum\_pungtuasi++;

if (isdigit (kalimat[i]))

jum\_digit++;

}

cout<

cout<<"JUMLAH \n";

cout<<"Huruf kecil   : "<<

cout<<"Huruf kapital : "<<

cout<<"pungtuasi     : "<<

cout<<"Digit         : "<<

getch();

}

**Contoh toupper dan tolower:**

#include

#include

#include //toupper-tolower

void main()

{

char kata[20];

clrscr();

cout<<"masukkan sebuah kata : ";

cin.getline (kata,20);

for (int i = 0; kata [i]; i++)

kata [i]=toupper (kata [i]);

//kata [i]=tolower (kata [i]);

cout<<

getch();

}