**MENGENAL LOGIKA DAN C++**

**Riya Widayanti**

**Pengenalan C++**

Bahasa C++ merupakan bahasa generasi ketiga, setelah memahami algoritma selanjutnya permasalahan tersebut dibuat ke dalam bahasan pemrograman.

Sebelum memasuki bagasa pemrograman ini, terlebih dahulu dilakukan pengenalan Bahasa pemrograman C++.



Program C++ tidak pernah lepas dari fungsi, sebab salah satu dasar penyusun blok C++, minimal harus ada fungsi main(). Dimulai dengan tanda buka kurung “{“ dan diakhiri dengan ttutup kurawal “}”.

Sedangkan statemen yang berada diantara kurung tersebut disebut dengan isi program/pernyataan. Di mana setiap pernyataan harus di akhiri dengan tanda titik koma”;”.

**Mengenal cout**

Pengenal cout merupakan sebuah obyek di dalam C++, yang digunakan untuk mengarahkan data ke dalam standar output(cetak pada layar)

cout<<”Selamat Datang di dunia C++\n”;

Tanda << (dua buah tanda kurang dari berurutan) merupakan sebuah operator yang disebut operator “penyisipan/peletakan”. Opertor ini akan mengarahkan operand (data) yang terletak di sebelah kanannya ke obyek yang terletak disebelah kiri.

**\n adalah karakter pindah baris(newline).**

**Penjelasan tentang #include <namafile>**

#include ini bukan merupakan pernyataan melainkan suatu pengarah praprosesor yang tersedia pada c++. Bentuk umumnya adalah:

#include <namafile>

Baris tersebut menginstruksikan kepada kompiler untuk menyisipkan file lain dalam hal ini file yang berakhiran .h (file header) yaitu file yang berisi berbagai deklarasi contohnya:

* #include <iostream.h> : diperlukan pada program yang melibatkan obyek cout
* #include <conio.h> : diperlukan bila melibatkan clrscr(), yaitu perintah untuk membersihkan layar.
* #include <iomanip.h> : diperlukan bila melibatkan setw() yang bermanfaat untuk mengatur lebar dari suatu tampilan data.
* #include <math.h> : diperlukan pada program yang menggunakan operasi sqrt() yang bermanfaat untuk operasi matematika kuadrat.

**Komentar**

Merupakan penjelasan dalam suatu program, dimana tidak akan dieksekusi oleh program. Bentuk komentar dapat dilihat dalam contoh berikut ini:

**Kasus 1.2**

.Anda diminta untuk mencetak ke layar tulisan berikut ini:

Selamat datang di dunia C++

Selamat belajar Algoritma dan Pemrograman

//Belajar Program C++

// \*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*

#include <iostream.h>

#include <conio.h>

void main()

{

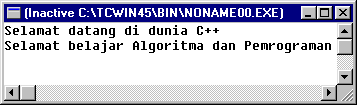
clrscr();

cout<<"Selamat datang di dunia C++\n";

cout<<"Selamat belajar Algoritma dan Pemrograman";

}

Hasil program di atas adalah:



Selain menggunakan //, komentara pada c++ juga dapat ditulis bentuk”

/\* Komentar \*/

/\* Komentar Anda

Panjang sekali \*/

Untuk lebih jelasnya dapat dicoba pada saat praktikum agar dapat diketahui perbedaannnya.

**2. Struktur Dasar Algoritma**

Algoritma berisi langkah-langkah penyelesaian suatu masalah. Langkah-langkah tersebut dapat berupa runtunan aksi (*sequence*), pemilihan aksi (*selection*), pengulangan aksi (*iteration*) atau kombinasi dari ketiganya. Jadi struktur dasar pembangunan algoritma ada tiga, yaitu:

1. Struktur Runtunan/Sequence

Digunakan untuk program yang pernyataannya *sequential* atau urutan.

1. Struktur Pemilihan/Selection

Digunakan untuk program yang menggunakan pemilihan atau penyeleksian kondisi.

1. Struktur Perulangan/Repeatition/Looping

Digunakan untuk program yang pernyataannya akan dieksekusi berulang-ulang.

**2.2. Tahapan dalam Pemrograman**

Langkah-langkah yang dilakukan dalam menyelesaikan masalah dalam

pemrograman dengan komputer adalah:

1. **Definisikan Masalah**

Berikut adalah hal-hal yang harus diketahui dalam analisis masalah supaya kita mengetahui bagaimana permasalahan tersebut:

a. Kondisi awal, yaitu *input* yang tersedia.

b. Kondisi akhir, yaitu *output* yang diinginkan.

c. Data lain yang tersedia.

d. Operator yang tersedia.

e. Syarat atau kendala yang harus dipenuhi.

Contoh kasus:

Menghitung biaya percakapan telepon di wartel. Proses yang perlu diperhatikan adalah:

a. *Input* yang tersedia adalah jam mulai bicara dan jam selesai bicara.

b. *Output* yang diinginkan adalah biaya percakapan.

c. Data lain yang tersedia adalah besarnya pulsa yang digunakan dan biaya per pulsa.

d. Operator yang tersedia adalah pengurangan (-), penambahan (+), dan perkalian (\*).

e. Syarat kendala yang harus dipenuhi adalah aturan jarak dan aturan waktu.

1. **Buat Algoritma dan Struktur Cara Penyelesaian**

Jika masalahnya kompleks, maka dibagi ke dalam modul-modul. Tahap penyusunan algoritma seringkali dimulai dari langkah yang global terlebih dahulu. Langkah global ini diperhalus sampai menjadi langkah yang lebih rinci atau detail. Cara pendekatan ini sangat bermanfaat dalam pembuatan algoritma untuk masalah yang kompleks. Penghalusan langkah dengan cara memecah langkah menjadi beberapa langkah. Setiap langkah diuraikan lagi menjadi beberapa langkah yang lebih ederhana. Penghalusan langkah ini akan terus berlanjut sampai setiap langkah sudah cukup rinci dan tepat untuk dilaksanakan oleh pemroses.

1. **Menulis Program**

Algoritma yang telah dibuat, diterjemahkan dalam bahasa komputer menjadi sebuah program. Perlu diperhatikan bahwa pemilihan algoritma yang salah akan menyebabkan program memiliki untuk kerja yang kurang baik. Program yang baik memiliki standar penilaian:

a. Standar teknik pemecahan masalah

- Teknik *Top-Down*

Teknik pemecahan masalah yang paling umum digunakan. Prinsipnya adalah suatu masalah yang kompleks dibagi-bagi ke dalam beberapa kelompok masalah yang lebih kecil. Dari masalah yang kecil tersebut dilakukan analisis. Jika dimungkinkan maka masalah tersebut akan dipilah lagi menjadi subbagian-subbagian dan setelah itu mulai disusun langkah-langkah penyelesaian yang lebih detail.

- Teknik *Bottom-Up*

Prinsip teknik *bottom up* adalah pemecahan masalah yang kompleks dilakukan dengan menggabungkan prosedur-prosedur yang ada menjadi satu kesatuan program sebagai penyelesaian masalah tersebut.

b. Standar penyusunan program

- Kebenaran logika dan penulisan.

- Waktu minimum untuk penulisan program.

- Kecepatan maksimum eksekusi program.

- Ekspresi penggunaan memori.

- Kemudahan merawat dan mengembangkan program.

- *User Friendly*.

- *Portability.*

- Pemrograman modular.

1. **Mencari Kesalahan**

a. Kesalahan sintaks (penulisan program).

b. Kesalahan pelaksanaan: semantik, logika, dan ketelitian.

1. **Uji dan Verifikasi Program**

Pertama kali harus diuji apakah program dapat dijalankan. Apabila program tidak dapat dijalankan maka perlu diperbaiki penulisan sintaksisnya tetapi bila program dapat dijalankan, maka harus diuji dengan menggunakan data-data yang biasa yaitu data yang diharapkan oleh sistem. Contoh data ekstrem, misalnya, program menghendaki masukan jumlah data tetapi *user* mengisikan bilangan negatif. Program sebaiknya diuji menggunakan data yang relatif banyak.

1. **Dokumentasi Program**

Dokumentasi program ada dua macam yaitu dokumentasi internal dan dokumentasi eksternal. Dokumentasi internal adalah dokumentasi yang dibuat di dalam program yaitu setiap kita menuliskan baris program sebaiknya diberi komentar atau keterangan supaya mempermudah kita untuk mengingat logika yang terdapat di dalam instruksi tersebut, hal ini sangat bermanfaat ketika suatu saat program tersebut akan dikembangkan. Dokumentasi eksternal adalah dokumentasi yang dilakukan dari luar program yaitu membuat *user guide* atau buku petunjuk aturan atau cara menjalankan program tersebut.

1. **Pemeliharaan Program**
   1. Memperbaiki kekurangan yang ditemukan kemudian.
   2. Memodifikasi, karena perubahan spesifikasi.

**Elemen Dasar C++**

1. **Himpunan karakter**

Himpunan karakter pada c++, terdiri dari huruf (A – Z), digit( 0 – 9), maupun simbol-simbol( -, +, \*, dan lain sebagainya)

1. **Pengenal(Identifier)**
   * Variabel
   * Konstanta bernama
   * Tipe data
   * fungsi
     + Penamaan Pengenal berupa satu atau beberapa karakter huruf, digit dan garis bawah “\_” serta berawalan huruf atau garis bawah.
     + Panjang maksimal nama pengenal pada C++ tergantung pada kompilernya. Misal Borland C++ : 32 karakter dan turbo C++ : 31 karakter.
     + Disarankan agar pemberian nama pengenal menggunakan kata yang berarti dan mudah dibaca.
     + Pada c++, huruf kecil dan huruf kapital pada suatu pengenal tidak dianggap sama, hal ini disebut dengan istilah **case sensitif.**
     + Beberapa hal yang *tidak diperkenankan* dalam penamaan pengenal:
   * Diawali dengan angaka
   * Menggunakan tanda -, #
   * Mengandung Spasi
   * Mengandung kata kunci (do, auto, break, char, double, defalt, dll)
2. **Tipe Data Dasar**

|  |  |
| --- | --- |
| Tipe Data | Ukuran Memori |
| char | 1 byte |
| Int | 2 byte |
| long | 3 byte |
| float | 4 byte |
| double | 8 byte |
| long double | 10 byte |

Mengetahui Ukuran Memori dari tipe data, lihat kasus berikut ini:

**Kasus 2.1**

// Mengetahui Ukuran Memori

/\* dari segala tipe data

pada C++ \*/

#include <iostream.h>

#include <conio.h>

void main()

{

clrscr();

cout<<"Ukuran memori char :"<<sizeof(char)<<"\n";

cout<<"Ukuran memori int :"<<sizeof(int)<<"\n";

cout<<"Ukuran memori long :"<<sizeof(long)<<"\n";

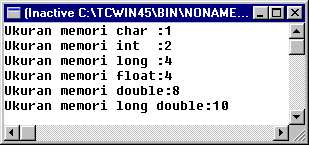
cout<<"Ukuran memori float:"<<sizeof(float)<<"\n";

cout<<"Ukuran memori double:"<<sizeof(double)<<"\n";

cout<<"Ukuran memori long double:"<<sizeof(long double)<<"\n";

}

Hasil Program di atas adalah



1. **Konstanta dan Variabel**

**Variabel** digunakan untuk menyimpan suatu nilai dimana dapat diubah selama eksekusi program berlangsung.

**Konstanta** adalah nilai yang tetap.

Variabel yang digunakan dalam program haruslah dideklarasikan terlebih dahulu, dalam hal ini mengenalkan ke program dan menentukan jenis data yang disimpan di dalamnya.

Apabila suatu pendeklarasian menyebabkan pengalokasian memori, maka pendeklarasian tersebut dinamakan pendefinisian.

Yang jelas pendefinisian berarti pula pendeklarasian, tetapi tidak berlaku sebaliknya.

***Deklarasi Variable***

Bentuk umumnya :

**Type data nama\_variabel ;**

**Contoh:**

int jumlah;

char jenis\_kelamin;

float harga;

float bayar;

*Bisa ditulis*

float harga, bayar;

***Pemberian nilai ke variabel***

jumlah = 20;

jumlah = 15;

harga = 4500;

bayar = jumlah \* harga;

Dari contoh di atas dapat disusun suatu algoritma:

Contoh: Buatlah algoritma untuk menentukan bayar dimana merupakan kalkulasi dari jumlah barang yang dibeli dikalo dengan harga barang.

1. Definisikan variabel jumlah bertipe int
2. Definisikan variabel harga dan bayar bertipe float
3. Tentukan nilai jumlah
4. Tentukan nilai harga
5. Bayar = jumlah \* harga
6. Tampilkan/cetak bayar

Untuk algoritma di atas dapat dipecahkan dengan program sebagai berikut:

#include <iostream.h>

#include <conio.h>

void main()

{

int jumlah;

float harga, bayar;

clrscr();

jumlah = 10;

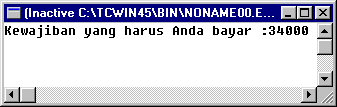
harga = 3400;

bayar = jumlah \* harga;

cout<<"Kewajiban yang harus Anda bayar :"<<bayar;

}

Hasil program di atas



Untuk jumlah yang diberi nilai dua kali, maka data jumlah yang terakhirlah yang akan diproses, Contoh

#include <iostream.h>

#include <conio.h>

void main()

{

int jumlah;

float harga, bayar;

clrscr();

jumlah = 10;

jumlah = 20;

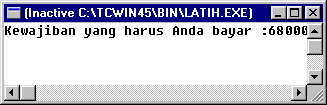
harga = 3400;

bayar = jumlah \* harga;

cout<<"Kewajiban yang harus Anda bayar :"<<bayar;

}

Hasil program di atas



Variabel & konstanta *bertipe char*, dapat menampung 1 karakter saja.

#include <iostream.h>

#include <conio.h>

void main()

{

char latih;

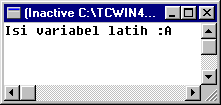
clrscr();

latih = 'A';

cout<<"Isi variabel latih :"<<latih;

}

Hasil dari program di atas:



1. **Inisialisasi Variabel**

Memberikan nilai pada variabel setelah didefinisikan.

int jumlah;

jumlah = 20;

Bisa dituliskan

int jumlah = 20;

**Contoh dengan kasus di atas:**

#include <iostream.h>

#include <conio.h>

void main()

{

int jumlah = 20;

float harga, bayar;

clrscr();

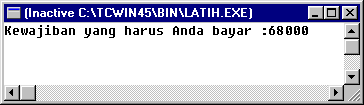
harga = 3400;

bayar = jumlah \* harga;

cout<<"Kewajiban yang harus Anda bayar :"<<bayar;

}

Hasil dari program di atas:



**Operator Matematika**

**Operator Binary**

* Bila variabel digunakan untuk menyimpan data bilangan bulat saja, tipe data yang digunakan adalah **int** atau **long (bergantung data yang ditampung)**. Operasi yang berlaku adalah

|  |  |
| --- | --- |
| Operator | Keterangan |
| \* | Perkalian |
| / | Pembagian |
| % | Modulus atau sisa pembagian |
| + | Pertambahan |
| - | Pengurangan |

* Bila variabel menampung data bilangan pecahan maka digunakan tipe **float** atau **double**. Operasi yang berlaku adalah:

|  |  |
| --- | --- |
| Operator | Keterangan |
| \* | Perkalian |
| / | Pembagian |
| + | Pertambahan |
| - | Pengurangan |

Lihat contoh kasus berikut untuk meihat perbedaannya:

**Kasus 3.1**

#include <iostream.h>

#include <conio.h>

void main()

{

int bulat=10;

float pecahan = 10, hasil1, hasil2, hasil3;

clrscr();

hasil1 = bulat /3;

hasil2= bulat % 3;

hasil3 = pecahan/3;

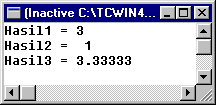
cout<<"Hasil1 = "<<hasil1<<endl;

cout<<"Hasil2 = "<<hasil2<<endl;

cout<<"Hasil3 = "<<hasil3<<endl;

}

Hasil dari program di atas:



**Operator Unary**

|  |  |
| --- | --- |
| Operator | Keterangan |
| + | Tanda plus |
| - | Tanda - |
| ++ | Increment/penaikan |
| -- | Decrement/penuruna |

Operator penaikan digunakna untuk **menaikkan** nilai variable sebesar satu, sedangkan penurunan dipai untuk **menurunkan** nilai bvariabel sebesari satu.

Contoh:

|  |  |  |
| --- | --- | --- |
| a = a + 1 | ++a | a++ |
| b = b + 1 | ++b | b++ |
| s =10 + r  r = r + 1 | - | s = 10 + r++ |
| r = r + 1  s =10 + r | s = 10 + ++ r | - |

**Kasus 3.2**

#include <iostream.h>

#include <conio.h>

void main()

{

int r =10, s;

clrscr();

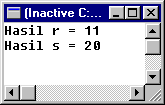
s = 10 + r++;

cout<<"Hasil r = "<<r<<endl;

cout<<"Hasil s = "<<s<<endl;

}

Hasil dari program di atas:



**Kasus 3.3**

#include <iostream.h>

#include <conio.h>

void main()

{

int r =10, s;

clrscr();

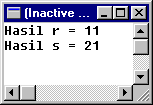
s = 10 + ++r;

cout<<"Hasil r = "<<r<<endl;

cout<<"Hasil s = "<<s<<endl;

}

Hasil dari program di atas:



**Operator Hubungan**

Digunakan untuk membandingkan dua elemen nilai dan akan dihasilkan nilai perbandingannya yaitu betul (bernilai satu) atau salah (bernilai nol). Operator sering digunakan pada penyeleksian suatu konddisi di struktur dasar **if.**

|  |  |
| --- | --- |
| Operator | Keterangan |
| > | Lebih besar dari |
| >= | Lebih besar sama dengan |
| < | Lebih kecil dari |
| <= | Lebih kecil sama dengan |
| = = | Sama dengan |
| != | Tidak sama dengan |

**Contoh:**

A = 5

B = 4

C = 10

Apakah A >B jawabnya adalah Benar

Apakah B > C jawabnya adalah Salah

**Operator Logika**

|  |  |
| --- | --- |
| Operator | Keterangan |
| && | AND |
| || | OR |
| ! | NOT |

Perbandingan operator logik

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | A && B | A || B |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 |

**cin**

Objek ini bermanfaat untuk membaca data dari standar input (dari keyboard). Bentuk umumnya:

cin>>var;

Tanda >> disebut operator ‘pengambilan dari’

**Kasus 3.4**

Buatlah algoritma dan program untuk menampilka keterangan Lulus atau tidaknya seorang siswa dengan ketentuan nilai akhir dan tugas yang ditempuh. Bila nilai akhir>70 **dan** tugas >=75, maka siswa tersebut akan lulus, jika tidak maka siswa tersebut tidak lulus.

Algoritma dari permasalahan di atas adalah:

1. Masukkan nilai akhir
2. Masukan nilai tugas
3. Jika nilai akhir >70 **dan** tugas >=75, jika tidak langkah 5
4. Cetak Lulus, ke langkah 6
5. Cetak Tidak lulus
6. Selesai

Bila dipecahkan dalam program adalah sebagai berikut:

#include <iostream.h>

#include <conio.h>

void main()

{

int nil\_akhir, tugas;

cout<<"Masukan Nilai Akhir:";

cin>>nil\_akhir;

cout<<"Masukan NIlai Tugas:";

cin>>tugas;

if (nil\_akhir>70 && tugas >=75)

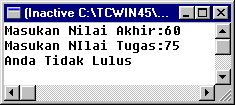
cout<<"Anda Lulus";

else

cout<<"Anda Tidak Lulus";

}

Hasil dari program di atas bila dimasukkan nilai akhir 60 dan tugas 75 adalah:



Tapi bila ketentuan soal di atas kita rubah, bila nilai akhir >70 **atau** nilai tugas >=75 maka Lulus, selainnya tidak lulus.

Buatlah program untuk mennyelesaikan masalah tersebut.

**Kasus 3.5**

Buatlah algortima dan flowchart untuk mencari luas segi empat, di man dengan rumus panjang kali lebar. Panjang dan lebar diinput secara bebas.

Jawab:

1. Masukan nilai panjang
2. Masukan nilai lebar
3. Hitung luas = panjang \* lebar
4. Tampilkan luas

Program untuk masalah diatas adalah

#include <iostream.h>

#include <conio.h>

#include <iomanip.h>

void main()

{

int panjang, lebar, luas;

clrscr();

cout<<"Masukan panjang persegi: ";

cin>>panjang;

cout<<"Masukan lebar persegi: ";

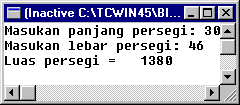
cin>>lebar;

luas = panjang \* lebar;

cout<<"Luas persegi = "<<setw(6)<<luas<<endl;

}

Hasil dari program di atas:



Latihan:

1. buatlah algoritma dan program untuk menghitung volume dan luas permukaan bila apabila masukkannya berupa jari-jari yang bertipe float. Keluaran yang diinginkan adalah Luas dan volume boal dengna tipe float.

Petunjuk:

V = 4/3 \* pi \* jari-jari \* jari-jari\*jari-jari

L = 4 \* pi \* jari-jari \* jari-jari

1. Mencari bilangan yang terbesar dari 2 bilangan yang diinputMenentukan bilangan genap atau ganjil dari bilangan yang diinputkan
2. Tuliskan program untuk menampilkan jumlah hari,jam, menit dan detik, dengan memasukkannya adalah detik. Sebagai contoh masukan adalah 1000 detik. Keluarannya nanti akan dihasilkan detik, jam, menit, hari.

Petunjuk;

1 hari = 24 jam

1 jam = 60 menit

1 menit = 60 detik

**Operator Matematika**

**Operator Binary**

* Bila variabel digunakan untuk menyimpan data bilangan bulat saja, tipe data yang digunakan adalah **int** atau **long (bergantung data yang ditampung)**. Operasi yang berlaku adalah

|  |  |
| --- | --- |
| Operator | Keterangan |
| \* | Perkalian |
| / | Pembagian |
| % | Modulus atau sisa pembagian |
| + | Pertambahan |
| - | Pengurangan |

* Bila variabel menampung data bilangan pecahan maka digunakan tipe **float** atau **double**. Operasi yang berlaku adalah:

|  |  |
| --- | --- |
| Operator | Keterangan |
| \* | Perkalian |
| / | Pembagian |
| + | Pertambahan |
| - | Pengurangan |

Lihat contoh kasus berikut untuk meihat perbedaannya:

**Kasus 3.1**

#include <iostream.h>

#include <conio.h>

void main()

{

int bulat=10;

float pecahan = 10, hasil1, hasil2, hasil3;

clrscr();

hasil1 = bulat /3;

hasil2= bulat % 3;

hasil3 = pecahan/3;

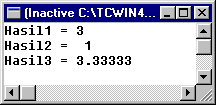
cout<<"Hasil1 = "<<hasil1<<endl;

cout<<"Hasil2 = "<<hasil2<<endl;

cout<<"Hasil3 = "<<hasil3<<endl;

}

Hasil dari program di atas:



**Operator Unary**

|  |  |
| --- | --- |
| Operator | Keterangan |
| + | Tanda plus |
| - | Tanda - |
| ++ | Increment/penaikan |
| -- | Decrement/penuruna |

Operator penaikan digunakna untuk **menaikkan** nilai variable sebesar satu, sedangkan penurunan dipai untuk **menurunkan** nilai bvariabel sebesari satu.

Contoh:

|  |  |  |
| --- | --- | --- |
| a = a + 1 | ++a | a++ |
| b = b + 1 | ++b | b++ |
| s =10 + r  r = r + 1 | - | s = 10 + r++ |
| r = r + 1  s =10 + r | s = 10 + ++ r | - |

**Kasus 3.2**

#include <iostream.h>

#include <conio.h>

void main()

{

int r =10, s;

clrscr();

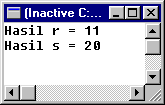
s = 10 + r++;

cout<<"Hasil r = "<<r<<endl;

cout<<"Hasil s = "<<s<<endl;

}

Hasil dari program di atas:



**Kasus 3.3**

#include <iostream.h>

#include <conio.h>

void main()

{

int r =10, s;

clrscr();

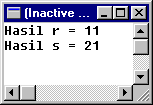
s = 10 + ++r;

cout<<"Hasil r = "<<r<<endl;

cout<<"Hasil s = "<<s<<endl;

}

Hasil dari program di atas:



**Operator Hubungan**

Digunakan untuk membandingkan dua elemen nilai dan akan dihasilkan nilai perbandingannya yaitu betul (bernilai satu) atau salah (bernilai nol). Operator sering digunakan pada penyeleksian suatu konddisi di struktur dasar **if.**

|  |  |
| --- | --- |
| Operator | Keterangan |
| > | Lebih besar dari |
| >= | Lebih besar sama dengan |
| < | Lebih kecil dari |
| <= | Lebih kecil sama dengan |
| = = | Sama dengan |
| != | Tidak sama dengan |

**Contoh:**

A = 5

B = 4

C = 10

Apakah A >B jawabnya adalah Benar

Apakah B > C jawabnya adalah Salah

**Operator Logika**

|  |  |
| --- | --- |
| Operator | Keterangan |
| && | AND |
| || | OR |
| ! | NOT |

Perbandingan operator logik

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | A && B | A || B |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 |

**cin**

Objek ini bermanfaat untuk membaca data dari standar input (dari keyboard). Bentuk umumnya:

cin>>var;

Tanda >> disebut operator ‘pengambilan dari’

**Kasus 3.4**

Buatlah algoritma dan program untuk menampilka keterangan Lulus atau tidaknya seorang siswa dengan ketentuan nilai akhir dan tugas yang ditempuh. Bila nilai akhir>70 **dan** tugas >=75, maka siswa tersebut akan lulus, jika tidak maka siswa tersebut tidak lulus.

Algoritma dari permasalahan di atas adalah:

1. Masukkan nilai akhir
2. Masukan nilai tugas
3. Jika nilai akhir >70 **dan** tugas >=75, jika tidak langkah 5
4. Cetak Lulus, ke langkah 6
5. Cetak Tidak lulus
6. Selesai

Bila dipecahkan dalam program adalah sebagai berikut:

#include <iostream.h>

#include <conio.h>

void main()

{

int nil\_akhir, tugas;

cout<<"Masukan Nilai Akhir:";

cin>>nil\_akhir;

cout<<"Masukan NIlai Tugas:";

cin>>tugas;

if (nil\_akhir>70 && tugas >=75)

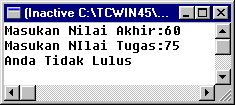
cout<<"Anda Lulus";

else

cout<<"Anda Tidak Lulus";

}

Hasil dari program di atas bila dimasukkan nilai akhir 60 dan tugas 75 adalah:



Tapi bila ketentuan soal di atas kita rubah, bila nilai akhir >70 **atau** nilai tugas >=75 maka Lulus, selainnya tidak lulus.

Buatlah program untuk mennyelesaikan masalah tersebut.

**Kasus 3.5**

Buatlah algortima dan flowchart untuk mencari luas segi empat, di man dengan rumus panjang kali lebar. Panjang dan lebar diinput secara bebas.

Jawab:

1. Masukan nilai panjang
2. Masukan nilai lebar
3. Hitung luas = panjang \* lebar
4. Tampilkan luas

Program untuk masalah diatas adalah

#include <iostream.h>

#include <conio.h>

#include <iomanip.h>

void main()

{

int panjang, lebar, luas;

clrscr();

cout<<"Masukan panjang persegi: ";

cin>>panjang;

cout<<"Masukan lebar persegi: ";

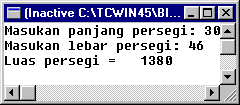
cin>>lebar;

luas = panjang \* lebar;

cout<<"Luas persegi = "<<setw(6)<<luas<<endl;

}

Hasil dari program di atas:



Latihan:

1. buatlah algoritma dan program untuk menghitung volume dan luas permukaan bila apabila masukkannya berupa jari-jari yang bertipe float. Keluaran yang diinginkan adalah Luas dan volume boal dengna tipe float.

Petunjuk:

V = 4/3 \* pi \* jari-jari \* jari-jari\*jari-jari

L = 4 \* pi \* jari-jari \* jari-jari

1. Mencari bilangan yang terbesar dari 2 bilangan yang diinput
2. Menentukan bilangan genap atau ganjil dari bilangan yang diinputkan
3. Tuliskan program untuk menampilkan jumlah hari,jam, menit dan detik, dengan memasukkannya adalah detik. Sebagai contoh masukan adalah 1000 detik. Keluarannya nanti akan dihasilkan detik, jam, menit, hari.

Petunjuk;

1 hari = 24 jam

1 jam = 60 menit

1 menit = 60 detik

**Pemilihan(Selection)**

Kita dapat menggunakan struktur kontrol pilihan dalam *pseudocode* untuk menggambarkan sebuah pilihan diantara dua atau lebih tindakan, tergantung kepada sebuah kondisi adalah **benar** atau **salah**. Kondisi ini berdasarkan pada sebuah perbandingan 2 hal, dan biasanya diekspresikan dengan salah satu **operator hubungan/relational** berikut ini :

**< kurang dari > lebih dari**

**= sama dengan <= kurang atau sama dengan**

**>= lebih besar atau sama <> tidak sama**

Struktur pemilihan ini merupakan suatu implementasi algoritma dalam pemecahan suatu masalah dengan berbagai alternatif jawaban yang tersedia dan Anda harus mengambil keputusan untuk memilih salah satu jawaban yang tepat.

C++ menyediakan dua perintah pengambilan keputusan yaitu perintah IF dan perintah SWITCH yang disertai dengan CASE

1. **Perintah IF – Tanpa Else**

Bentuk Umum Algoritma

**If *kondisi* then**

Pernyataan (jika kondisi bernilai benar)

**End if**

Dalam Bahasa Pemrograman C++

**if (kondisi)**

**pernyataan;**

Hal paling penting dalam penggunaan **IF** adalah pemahaman soal yang selanjutnya diterjemahkan dalam *kondisi*, yaitu membuat formula perbandingan dengan menggunakan operator hubungan.

Contoh:

IF jenis\_kelamin = ‘w’ THEN

jumlah\_wanita = jumlah\_wanita + 1

END IF

Atau dalam C++

if (jenis\_kelamin ==’w’)

{ jumlah\_wanita=jumlah \_wanita +1;}

Pasangan kurung kurawal (paranthesis) untuk pernyataan yang jumlahnya satu adalah optinal namun apabila *pernyataan majemuk(lebih dari satu wajib*, digunakan tanda { untuk mengawalinya dan tanda } untuk mengakhiri pernyataaan pada perintah IF.

if (kondisi)

{

pernyataan1;

pernyataan2;

pernyataan3;

…..

pernyataan ke n;

}

1. **Perintah IF.. Else**

Bentuk umum IF...Else

**IF** kondisi **THEN**

Pernyataan1;

**ELSE**

Pernyataan2;

**END IF**

Dalam Bahasa C++

if (kondisi)

Pernyataan1;

else

Pernyataan2;

Sedangkan untuk majemuk (lebih dari satu pernyataan)

if (kondisi)

{

Pernyataan\_1;

Pernyataan\_2

...

Pernyataan\_n;

}

else

{

Pernyataan\_5;

Pernyataan\_6;

.....

Pernyataan\_n;

}

Bentuk ini mempunyai arti **jika kondisi benar** maka pernyataan 1 akan dikerjakan dan **jika kondisi salah** (tidak terpenuhi) maka pernyataan 2 yang akan dikerjakan.

**Kasus 4.1:**

Algoritma mencetak **bilangan terbesar** dari dua bilangan

1. Masukkan bilangan1
2. Masukkan bilangan2
3. IF bilangan1 > bilangan2 THEN

Cetak “Bilangan terbesar adalah bilangan1”

Cetak bilangan1

ELSE

Cetak “Bilangan terbesar adalah bilangan2”

Cetak bilangan2

END IF

1. Selesai

Untuk kasus di atas dipecahkan dalam program sebagai berikut:

#include <iostream.h>

#include <conio.h>

void main()

{

int bil1, bil2;

cout<<"Masukan bilangan 1:";

cin>>bil1;

cout<<"Masukan bilangan 2:";

cin>>bil2;

if (bil1>bil2)

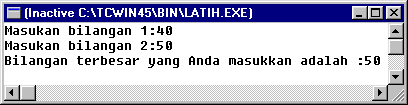
cout<<"Bilangan terbesar yang Anda masukkan adalah :"<<bil1;

else

cout<<"Bilangan terbesar yang Anda masukkan adalah :"<<bil2;

}

Hasil program tersebut bila dimasukkan nilai ke variabel bil1 = 40 dan ke variabel bil2 = 50 maka:



**Kasus 4.2 IF ..Else dengan pernyataan yang dikalkulasi**

Warung “Murah Meriah” memberikan diskon menarik untuk pelanggannya jika pembelian mencapai Rp. 2.000.000 maka mendapat diskon 5%, dan Bonus = Kaos.

1. Masukkan pembelian
2. Jika pembelian lebih besar sama dengan Rp. 2.000.000, maka Dis = 5%\*2.000.000 dan Bonus = Kaos.
3. Tampilkan Pembelian
4. Tampilkan diskon
5. Tampilkan total
6. Tampilkan Bonus

Program untuk kasus di atas adalah:

#include <iostream.h>

#include <conio.h>

void main()

{

long pembelian,disk, total;

disk = 0;

total = 0;

cout<<"Masukan pembelian:";

cin>>pembelian;

if (pembelian>=2000000)

disk=0.05\*pembelian;

total = pembelian-disk;

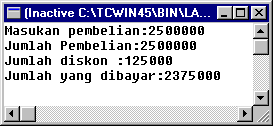
cout<<"Jumlah Pembelian:"<<pembelian<<endl;

cout<<"Jumlah diskon :"<<disk<<endl;

cout<<"Jumlah yang dibayar:"<<total<<endl;

}

Hasil program bila dimasukkan pembelian 2500000 maka



1. **NESTED IF – IF Bersarang LINIER (IF dengan kondisi lebih dari satu)**

Kombinasi pernyataan IF adalah pernyataan IF yang berisi banyak kondisi dan sejenis. Untuk lebih jelasnya perhatikan contoh kasus 4.3.

**Kasus 4.3 dengan menggunakan lebih dari satu kondisi**

Algoritma untuk mencetak warna, bila diketahui warna kuning untuk kode 1, Biru untuk kode 2 sedangkan ungu untuk kode 3

1. Masukkan Kode warna
2. IF Kode\_warna = 1 THEN

Cetak “ Kuning”

ELSE IF Kode\_warna = 2 THEN

Cetak “Biru”

ELSE

CETAK “Ungu”

1. Selesai

Program untuk masalah di atas adalah:

#include <iostream.h>

#include <conio.h>

void main()

{

int kode;

cout<<"Masukan Kode Warna:";

cin>>kode;

if (kode ==1)

cout<<"Warna yang Anda sukai adalah Kuning";

else if (kode ==2)

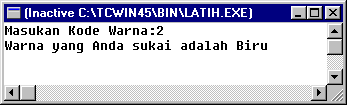
cout<<"Warna yang Anda sukai adalah Biru";

else

cout<<" Warna yang Anda sukai adalah Ungu";

}

Hasil program bila dimasukkan kode 2 maka



Lakukan perubahan program bila kode warna yang dimasukkan selain angaka 1, 2, dan 3, maka akan tercetak pernyataan “Kode yang Anda masukkan salah”

1. **NESTED IF – IF Bersarang NON-LINIER**

Kombinasi pernyataan IF adalah pernyataan IF yang berisi banyak kondisi dan beda untuk jenis kondisinya. Setiap pernyataan bisa menggunakan operator logika AND (DAN) atau OR (ATAU). Jika kondisi dikombinasikan dengan menggunakan penghubung AND, kedua kondisi harus bernilai benar untuk membuat kondisi kombinasi bernilai benar. Jika kondisi dikombinasikan dengna menggunakan penghubung OR, salah satu dari konsidi bernilai benar maka pernyataan bisa dikerjakan.

Perintah ini sering disebut dengan nested-if (if bersarang). Perintah ini bentuk umumnya adalah:

IF kondisi THEN

IF kondisi2 THEN

Pernyataan1;

Else

Pernyataan2;

**Kasus 4.4**

Suatu rental VCD memberlakukan ketentuan sebagai berikut:

Bila menyewa lebih dari atau sama dengan 10 dan menjadi anggota tetap maka mendapat diskon 0.10 tapi bila bukan member dia mendapatkan diskon 0.05.

Bila menyewa lebih dari atau sama dengan 5 dan menjadi anggota tetap maka mendapat diskon 0.05 dan bila bukan member tidak mendapatkan diskon, selainnya itu maka tidak mendapatkan diskon.

**Petunjuk:**

Bila menjadi Anggota kode yang dimasukkan 1

Bila bukan Anggota kode yang dimasukkan 2

Algoritma untuk masalah tersebut adalah:

Masukan kode\_anggota

Masukan jumlah\_CD

**IF jumlah\_cd >= 10 THEN**

IF kode\_anggota = 1 THEN

Diskon = 0.1

ELSE

Diskon = 0.05

END IF

**ELSE IF jumlah\_cd>=5 THEN**

IF kode\_anggota = 1 THEN

Diskon = 0.05

ELSE

Diskon = 0

END IF

**ELSE**

Diskon =0

Kasus tersebut dapat dipecahkan dengan menggunakan operator logika “AND”, algoritmanya menjadi

**IF jumlah\_cd >= 10 AND IF kode\_anggota = 1 THEN**

Diskon = 0.1

**ELSE IF jumlah\_cd >= 10 AND IF kode\_anggota = 2 THEN**

Diskon = 0.05

**ELSE IF jumlah\_cd>=5 AND IF kode\_anggota = 1 THEN**

Diskon = 0.05

**ELSE**

Diskon =0

Untuk kasus di atas bila dipecahkan dalam c++ adalah:

#include <iostream.h>

#include <conio.h>

void main()

{

int jml\_sewa, kd\_anggota;

float harga,disk,bayar, tot;

harga = 3000;

disk = 0;

cout<<"Masukan Jumlah Kaset yang Disewa:";

cin>>jml\_sewa;

cout<<"Masukan Kode Anggota:";

cin>>kd\_anggota;

bayar = jml\_sewa \* harga;

if (jml\_sewa >= 10)

if (kd\_anggota == 1)

{

cout<<"Anda diskon 10 %";

disk = bayar \* 0.1;

}

else

{

cout<<"Anda diskon 5 %";

disk = bayar \* 0.05;

}

else if (jml\_sewa >= 5)

if (kd\_anggota == 1)

{

cout<<"Anda diskon 5 %";

disk = bayar \* 0.05;

}

else

cout<<"Anda Tidak Diskon";

else

cout<<"Anda Tidak Diskon";

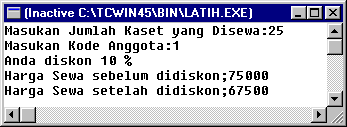
tot = bayar - disk;

cout<<"\nHarga Sewa sebelum didiskon;"<<bayar;

cout<<"\nHarga Sewa setelah didiskon;"<<tot;

}

Hasil program bila dimasukkan jumlah kaset 25 dan kode anggota = 1 adalah:



Perhatikan program berikut ini untuk menyelesaikan kasus 5.4 tunjukkan perbedaannya dan jelaskan

#include <iostream.h>

#include <conio.h>

void main()

{

int jml\_sewa, kd\_anggota;

float harga,disk,bayar, tot;

harga = 3000;

disk = 0;

cout<<"Masukan Jumlah Kaset yang Disewa:";

cin>>jml\_sewa;

cout<<"Masukan Kode Anggota:";

cin>>kd\_anggota;

bayar = jml\_sewa \* harga;

if (jml\_sewa >= 10 && kd\_anggota == 1)

{

cout<<"Anda diskon 10 %";

disk = bayar \* 0.1;

}

else if (jml\_sewa >= 10 && kd\_anggota == 2)

{

cout<<"Anda diskon 5 %";

disk = bayar \* 0.05;

}

else if (jml\_sewa >= 5 && kd\_anggota == 1)

{

cout<<"Anda diskon 5 %";

disk = bayar \* 0.05;

}

else if (jml\_sewa >= 5 && kd\_anggota == 2)

cout<<"Anda Tidak Diskon";

else

cout<<"Anda Tidak Diskon";

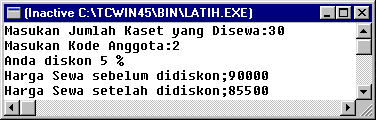
tot = bayar - disk;

cout<<"\nHarga Sewa sebelum didiskon;"<<bayar;

cout<<"\nHarga Sewa setelah didiskon;"<<tot;

}

Hasil dari program tersebut bila dimasukkan jmlah sewa 30 dan kode anggota = 2 adalah:



**Perintah SWITCH**

Perintah SWITCH merupakan pernyataaan yang dirancang untuk menangani pengambilan keputusan yang melibatkan sejumlah alternatif yang diantaranya untuk menggantikan pernyataan IF bersarang LINIER dengan menggunakan operator hubungan “=” atau “samadengan” saja.

Bentuk Umum

Switch (ekspresi)

{

Case konstanta1:

Pernyataan1;

Break;

Case konstanta2:

Pernyataan2;

Break;

Case konstanta3:

Pernyataan3;

Break;

…….

…….

Case konstanta ke n:

Pernyataan ke n;

Break;

Default:

Pernyataan x

}

Bentuk ekspresi dapat bertipe integer atau karakter begitu juga untuk konstanta.

Pengujian pada switch akan dimulai dari konstanta1. Apabila nilainya cocok dengan ekspresi maka pernyataan1 akan dijalankan. Kalau tidak cocok maka akan beralih ke pernyataaan2 dan seterusnya sampai hasil yang diinginkan.

Perintah break menyebabkan eksekusi menyebabkan ekskursi diarahkan ke akhir switch kalau nilai konstanta1 tidak sama dengan nilai ekspresi dan kemudian baru diteruskan pengujian dengna konstanta2 dan seterusnya. Jika sampai pada pengujian akhir tidak ada yang cocok maka default akan dijalankan.

Jika pernyataan break tidak diikutsertakan pada setiap case maka walupun konstanta cocok dengan ekspresi, maka setelah pernyataan pada case tersebut dikerjakan akan dilanjutkan ke case berikutnya sampai ditemui pernyataan break pada akhir switch.

**Kasus 5.4**

Misalkan ada permasalahan dimana Anda diminta untuk membuat algoritma dan pemrograman untuk menentukan menu makanan berdasakan kose yang dimasukkan. Adapun kode yang dimasukkan mulai dari A, B, C, D, E, F dan G. Input dari permasalahan di atas adalah kode dan keluaranya adalah menu makanana. Untuk lebih jelasnya perhatikan keterangna berikut:

Jika kode = A, maka menu = Yogya

Jika kode = B, maka menu = Padang

Jika kode = C, maka menu = Cina

Jika kode = D, maka menu = Jawa Timur

Jika kode = E, maka menu = Eropa

Jika kode = F, maka menu = Laut

Jika kode = G, maka menu = Jepang

Algoritma untuk kasus di atas adalah:

1. Masukkan kode
2. Jika kode = A, maka menu = Yogya
3. Jika kode = B, maka menu = Padang
4. Jika kode = C, maka menu = Cina
5. Jika kode = D, maka menu = Jawa Timur
6. Jika kode = E, maka menu = Eropa
7. Jika kode = F, maka menu = Laut
8. Jika kode = G, maka menu = Jepang
9. Tampilkan kode dan hari

Program untuk kasus di atas dipecahakan dengan **if** adalah:

#include <iostream.h>

#include <conio.h>

#include <ctype.h>

void main()

{

char kdmenu;

cout<<"Masukkan Kode Menu [A - G]:";

cin>>kdmenu;

kdmenu = toupper(kdmenu);

if (kdmenu =='A')

cout<<"Khas Yogya";

else if (kdmenu =='B')

cout<<"Khas Padang";

else if (kdmenu =='C')

cout<<"Khas Cina";

else if (kdmenu =='D')

cout<<"Khas Jawa Timur";

else if (kdmenu =='E')

cout<<"Khas Eropa";

else if (kdmenu =='F')

cout<<"Khas Laut";

else if (kdmenu =='G')

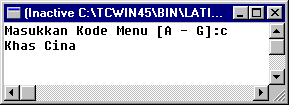
cout<<"Khas Jepang";

else

cout<<"Kode yang Anda Masukkan Salah";

}

Hasil program bila dimasukkan kdmenu c:



Program untuk kasus di atas dipecahakan dengan **case** adalah:

#include <iostream.h>

#include <conio.h>

#include <ctype.h>

void main()

{

char kdmenu;

cout<<"Masukkan Kode Menu [A - G]:";

cin>>kdmenu;

kdmenu = toupper(kdmenu);

switch (kdmenu)

{

case 'A':

cout<<"Khas Yogya";

break;

case 'B':

cout<<"Khas Padang";

break;

case 'C':

cout<<"Khas Cina";

break;

case 'D':

cout<<"Khas Jawa Timur";

break;

case 'E':

cout<<"Khas Eropa";

break;

case 'F':

cout<<"Khas Laut";

break;

case 'G':

cout<<"Khas Jepang";

break;

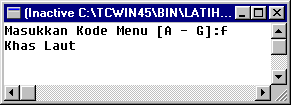
default:

cout<<"Kode yang Anda Masukkan Salah";

}

}

Hasil program bila dimasukkan kode baik huruf kecil ataupun huruf besar adalah:



**Latihan**

1. Buatlah algoritma dan program dalam bentuk menu, yang mampu menyelesaikan:
   1. Luas dan Keliling bujur sangkar
   2. Luas dan Keliling persegi panjang
   3. Luas dan keliling lingkaran

Ketentuan:

Pilihan dalam menu yaitu:

B apabila pilihan jatuh pada menu luas dan keliling bujur sangkar

P apabila pilihan jatuh pada menu luas dan keliling persegi panjang

L apabila pilihan jatuh pada menu Luas dan keliling lingkaran

1. Buatlah program untuk menentukan total pembayaran mahasiswa dalam 1 tahun dan jenis kuliah yang diambil, jika diberlakukan kondisi sebagai berikut:

|  |  |
| --- | --- |
| Kode | Keterangan |
| 1 | Kursus = “Komputer”  Jumlah = 50000\*sks  Semester = Jumlah\*2 |
| 2 | Kursus = “Bahasa”  Jumlah = 30000\*sks  Semester = Jumlah\*2 |
| 3 | Kursus = “Pajak”  Jumlah = 20000\*sks  Semester = Jumlah\*2 |

Input:

Kode dan sks yang diambil

Output

Nama kursus, jumlah dan semester

Ketentuan:

Apabila kode dimasukkan:

1 maka kursus = Komputer

2 maka kursus = Bahasa

selainnya kursus Pajak.

1. Buatlah algoritma dan program yangn dapat mengkonversikan sebulah nilai huruf A – E menjadi nilai 4-0

Petunjuk:

A 4

B 3

C 2

D 1

E 0

1. Buatlah algoritma dan program untuk menentukan gaji seorang karyawan dengan ketetnuan sebagai berikut:
   1. Gaji pokok, jenis kelamin dan jumkah anak diinputkan
   2. Tunjangan anak diberikan bila karyawan telah menikah dan mempunyai anak
   3. Bagi karyawan laki-laki yang telah menikah diberi tunjangna keluarga
   4. Apabila anak kurang dari 3 atau gaji lebih kecil dari Rp. 200.000 maka akan mendapatkan tambahan gaji 50.000,-
2. Input nomor pegawai, gaji perjam dan jumlah jam kerja selama seminggu. Program kemudian menghitung gaji mingguan pegawai dan mencetak bersama dengan data masukan. Menurut aturan perusahaan, tidak ada pegawai yang digaji melebihi dari 60 jam perminggu, dan gaji perjam maksimum adalah $25.00. jika bekerja lebih dari 35 jam perminggu, maka pembayaran untuk lembur perjam dikalkulasikan dengan satu setengah kali. Apabila gaji perjam melebihi batas, maka data masukan dan pesan yang sesuai akan dicetak dan gaji mingguan pegawai tidak diperhitungkan.)

**2. Struktur Pengulangan**

Struktur pengulangan terdiri atas dua bagian:

1. kondisi pengulangan, yaitu ekspresi bolean yang harus dipenuhi untuk melaksakan pengulangan.
2. Isi atau badan pengulangan, yaitu satu atau lebih pernyataan (aksi ) yang akan diulang.

Di samping itu, struktur pengulangan biasanya disertai dengan bagian:

1. inisialisasi, yaitu pernyataan yang dilakukan sebelum pengulangna dilakukan pertama kali
2. terminasi, yaitu aksi yang dilakukan setelah pengulangan selesai dilaksanakan

inisialisasi dan teminasi tisak selalu harus ada, namun berbagia kasus inisalisasi umumnya diperlukan.

Perintah atau notasi dalam struktur pengulangan adalah:

1. Perintah for
2. Perintah do .. while
3. Perintah while
4. Perintah Continue dan break
5. Perintah Go To

**2.1. Perintah for**

Perintah for digunakan untuk menghasilkan pengulangan sejumlah kali tanpa penggunaan kondisi apapun. Perintah ini dapat digunakan bila anda sudah tau berapa kali Andaakan mengulang satu atau beberapa pernyataan. Bentuk umum pernyataan for adalah

Dalam algoritma

For peubah nilai 🡨 nilai awal to nilai akhir di

Aksi

End

**Dalam bahasa pemrograman C++**

For (peubah=nilai awal; peubah=nilai akhir, keadaan)

{

Aksi;

}

keterangan:

1. peubah 🡨 nilai awal mempunyai arti sebagai inisialisasi
2. peubah 🡨 nilai akhir mempunyai arti sebagai terminasi/konsidi untuk keluar dari looping
3. keadaan mempunyai dua pengertian menaik (ascending) atau menurun (descending)
4. Jumlah aksi diulang sebanyak nilai akhir- nilai awal + 1

**Kasus 6.1**

Buatlah algoritma dan program untuk mencetak nama Anda sebanyak 10 kali

**Jawab:**

Algoritm untuk memecahkan masalah tersebut di atas adalah:

1. k bertipe integer
2. for k =1 to 10 do
3. cetak “Nama Anda”
4. end

**Program**

#include<iostream.h>

void main()

{

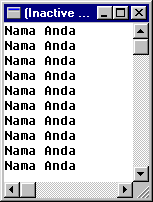
int k;

for(k=1;k<=10;k++)

cout <<"Nama Anda\n";

}

Hasil program:



**Latihan 1:**

1. Buatlah program untuk mencetak deret 1 – 10 ?

1 3 5 7 9 11 13 15 17 19

1. Buatlah program untuk mencetak deret genap 10 suku?

2 4 6 8 10 12 14 16 18 20

1. Buatlah program untuk mencetak deret kuadrat 5 suku?

1 4 9 16 25

**2.2. Perintah Do..While**

Perintah ini menyatakan pengulangan proses selama kondisi tertentu. Bentuk umumnya adalah sebagai berikut:

Do

Pernyataan;

While (keadaan);

Atau

Do

{

Pernyataan;

Pernyataan;

}

while (keadaan)

Setiap pengulangan dikerjakan maka kondisi akan dicek. Jika masih benar, proses loop dilakukan lagi dan jika salah maka proses loop berhenti dan berlanjut pada perintah selanjutnya. Lebih jelaskan perhatikan contoh berikut ini:

**Kasus 6.2**

Buatlah algortima dan pemrogramn untuk mencetak tanggal lahir Anda sebanyak 10 kali.

Jawab:

Algoritma dari permasalahan di atas adalah:

1. Tentukan nilai awal perhitungan = 0
2. Lakukan pengulangan
3. Cetak tanggal lahir
4. Nilai perhitungan bertambah Satu
5. Ulangi langkah sampai nilai perhitungan kurang dari 10

Program menggunakan Do..While

#include<iostream.h>

void main()

{

int A=0;

do

{

cout<<"Belajar Pemrograman C++ \n";

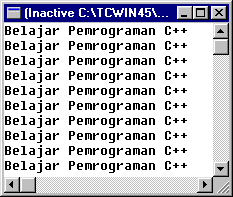
A++;

}

while (A<10);

}

**Hasil Program:**



**Latihan 2**

Kerjakan latihan 1 di atas dengan menggunakan perintah Do..while, tapi deret tersebut menurun.

**2.3. Perintah While**

Perintah While ini prinsipnya sama dengan perintah Do..While hanya pengujian kondisinya terletak pada awal loop.

Bentuk umumnya adalah sebagai berikut:

While (keadaan)

{

Pernyataan;

Pernyataan;

…

}

Pernyataan dapat berupa pernyataan tunggal atau beberapa pernyataan yang dibatasi dengan tanda {}. Pernyataan itu akan dijalankan jika kondisinya masih benar.

**Kasus 3:**

Buatlah algoritma dan pemrograman untuk mencetak deret berikut ini :

10 9 8 7 6 5 4 3 2 1

Algoritma untuk kasus di atas:

1. Tentukan nilai awal
2. Lakukan proses pengulangan selama i > 0
3. Cetak bilangan
4. Ulangi langkah 2 sampai batas akhir

Program untuk masalah tersebut adalah:

#include<iostream.h>

void main()

{

int A=10;

while (A>0)

{

cout<<A<<" ";

A--;

}

}

**2.4. Perintah continue dan break**

Pada beberapa kasus, Anda mungkin ingin menghentikan suatu pengulangan di tengah jalan untuk menghentikan pengulangan, gunakanlah pernyataan break. Anda bisa menggunakan pernyataan break pada pernyataan for, do..while dan while do. Perhatikan program berikut ini:

#include<iostream.h>

void main()

{

int x=0;

while (x>=0)

{

x++;

cout<<x<<"\n";

if (x>100)

break;

}

}

Bila dilihat sekilas, looping di atas seperti tidak akan pernah berakhir karena x selalu ditambah 1 padahal kondisi yang harus dipenuhi adalah (x>=0). Namun bila dijalankan, setelah x lebih besar dari pada 100, pengulangan akan berhenti. Hal ini dikarenakan adanya pernyataan break.

Pernyataan continue adalah bentuk pernyataaan khusus lain yang digunakan pada pengulangan. Pernyataan ini digunakan untuk mengembalikan aliran program ke pengujian kondisi pengulangan. Dengan kata lain, pernyataan di bawah continue akan diabaikan.

Berikut ini contoh program yang menggunakan perintah Continue:

#include<iostream.h>

void main()

{

int i=0;

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

{

if(i%3!=0)

continue;

cout<<i<<" ";

}

}

Program tersebut di atas akan menghasilkan semua bilangan kelipatan iga yang kurang dari 50. Mengapa bisa demikian? Karena pada saat nilai pencacah pengulangan atidak habis dibagi 3, program akan memeanggil pernyataan continue, akibatnya cout<<i<<" "; tidak dilaksanakan.

**Kasus 4**

Buatlah algoritma dan program untuk menampilkan bilangan dari 0 sampai 6, ketika proses pencetakan bilangan pada 4, bilangan 4 ini akan dilewati dan dilanjutkan dengan bilangan 5 dan pada bilangan 6 program akan berhenti sehingga hasil terakhir adalah 0, 1, 2, 3, 5, 6

Algoritma dari kasus di atas adalah sebagai berikut:

1. Tentukan nilai awal, batas akhir dan pertambahan nilai
2. Lakukan pengulangan sesuai dengan langkai
3. Jika dalam pengulangan bilangan bernilai dengan 4, maka pengulangan berhenti dan dilanjutkan ke bilangan selanjutnya.
4. Cetak bilangan
5. jika dalam pengulangan bernila 6, maka pengulangan dihentikan

Program untuk kasus di atas adalah:

#include<iostream.h>

void main()

{

int i=0;

for (i=0;i<10;i++)

{

if(i==4)

continue;

cout<<"Bilangan: "<<i<<"\n";

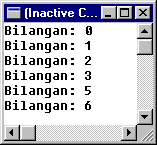
if (i==6)

break;

}

}

Hasil dari program itu adalah:



Dari program di atas, dapat dilihat pengulangan sari suatu bilangan sebannyak 10 kali. Etapi pada pengulangan i=4, ada perintah continue. Dengan perintah ini, maka program langsung meloncat ke loop berikutnya dan ketika sampai pengulangan i=6 ada perintah break. Otomatis program akan berhenti dan tidak sampai ke i=10.

**2.5. Perintah GOTO**

Perintah GOTO digunakan untuk mengalihkan proses menuju ke suatu tabel tertentu. Lebih jelaskan perhatikan contoh berikut ini:

**Kasus 5**

Buatlah algoritma dan program untuk menampilkan tulisan “C++ sangat mudah sekali” sebanyak 10 kali.

Algoritma kasus tersebut adalah:

1. Tentukan nilai awal pencetakan = 1
2. Tentukan nama label dimana kondisi untuk pencetakan tulisan dikenakan padanya
3. Cetak tulisan
4. Lakukan pengujian, jika nilai awal pencetakan kurang dari 10 ulangi langkah 3
5. Jika nilai pencetakan lebih dari 10, proses pencetakan dihentikan dan program akan selesai.

Program untuk kasus di atas adalah

#include<iostream.h>

void main()

{

int n=1;

cetak:

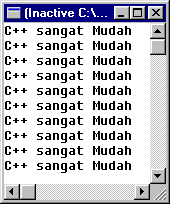
cout<<"C++ sangat Mudah\n";

if (n++<10)

goto cetak;

}

Hasil dari program tersebut di atas adalah:



**LATIHAN 3**

1. Rancanglah algoritma dan program untuk menampilkan semua bilangan ganjil kurang dari 100?
2. Tuliskan program untuk menampilkan pola-pola bintang seperti berikut ini:

\*

\* \*

\* \* \*

\* \* \* \*

n = 4

1. Buatlah algoritma dan program untuk menentukan vokal dan konsonan dari kalimat yang diinputkan?
2. Buatlah program untuk mencari bilangan terbesar dari 5 buah bilangan yang diinputkan?
3. Buatlah algoritma dan program untuk menentukan sisa hasil bagi pembagian antara bilangan yang dimasukkan dengan bilangan pembagi. Apabila sisa baginya=0 maka dicetak tidak ada dan kalu ada sisa baginya, maka sisa bagi tersebut ditampilkan.?
4. Tuliskan program untuk menampilkan pola-pola bintang seperti berikut ini:

1 = 1

1+2 = 3

1+2+3 = 6

1+2+3+4 = 10