**NAMA : ANITA MONITA**

**NIM : 190250501011**

**PRODI : SISTEM INFORMASI (A)**

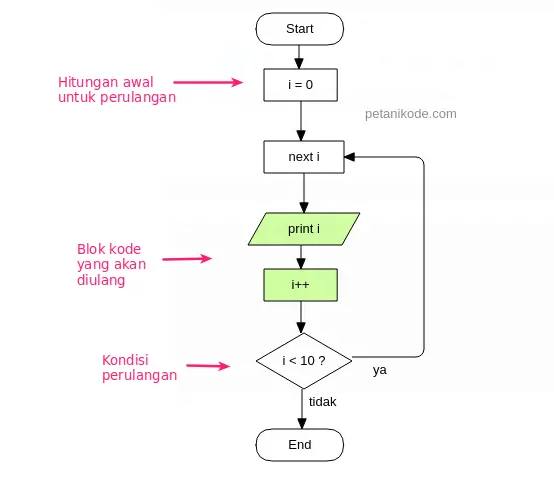
1. **Perulangan Do-While**

Perulangan menggunakan do-while akan melakukan pengecekan kondisi di akhir blok statement. Dalam hal ini pengulangan juga hanya akan dilakukan jika kondisi yang didefinisikan didalamnya terpenuhi (bernilai benar).Namun biar kondisi tidak terpenuhi,maka berproses pengulangan ini minimal akan dilakukan satu kali. Perulangan do/while sama seperti perulangan while.

Perbedaanya:

Perulangan do/while akan melakukan perulangan sebanyak 1 kali terlebih dahulu, lalu mengecek kondisi yang ada di dalam kurung while.

Flow chart perulangan do/while:



Bentuk kodenya seperti ini:

do {

// blok kode yang akan diulang

} while (<kondisi>);

Jadi perbedaanya:

Perulangan do/while akan mengecek kondisi di belakang (sesudah mengulang), sedangkan while akan mencek kondisi di depan atau awal (sbelum mengulang).

contohnya:

#include <iostream>

using namespace std;

int main(){

char ulangi = 'y';

int counter = 0;

do {

printf("Apakah kamu mau mengulang?\n");

printf("Jawab (y/t): ");

cin >> ulangi;

// increment counter

counter++;

} while(ulangi == 'y');

printf("\n\n----------\n");

printf("Perulangan Selesai!\n");

printf("Kamu mengulang sebanyak %i kali.\n", counter);

return 0;

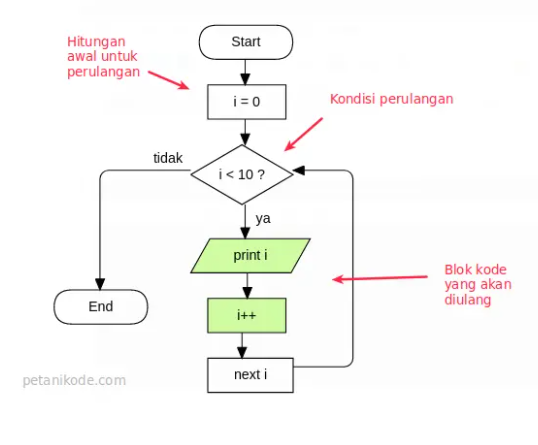
}

Contoh tersebut sama seperti contoh pada perulangan while.

1. **Perulangan While**

Perulangan menggunakan while akan melakukan pengecekan kondisi diawal blok statement. Dalam hal ini pengulangan hanya akan dilakukan jika kondisi yang didefinisikan didalamnya terpenuhi (bernilai benar). Jika kondisi yang didefinisikan bernilai salah,maka statement dalam blok tidak akan dieksekusi atau dijalankan. Perulangan while merupakan perulangan yang termasuk dalam perulangan uncounted loop. Perulangan while juga dapat menjadi perulangan yang counted loop dengan memberikan counter di dalamnya.

Bentuk flow chart-nya sama seperti flow chart for.



contohnya:

#include <iostream>

using namespace std;

int main(){

char ulangi = 'y';

int counter = 0;

// perulangan while

while(ulangi == 'y'){

printf("Apakah kamu mau mengulang?\n");

printf("Jawab (y/t): ");

cin >> ulangi;

// increment counter

counter++;

}

printf("\n\n----------\n");

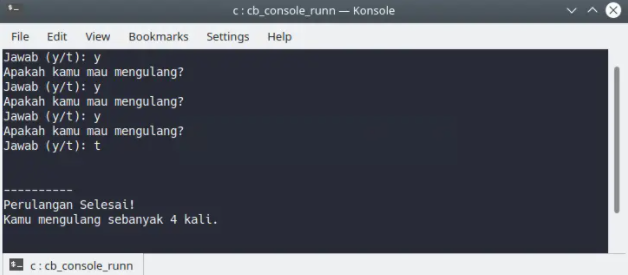
printf("Perulangan Selesai!\n");

printf("Kamu mengulang sebanyak %i kali.\n", counter);

return 0;

}

Hasilnya:



perhatikan blok kode while:

// perulangan while

while(ulangi == 'y'){

printf("Apakah kamu mau mengulang?\n");

printf("Jawab (y/t): ");

cin >> ulangi;

// increment counter

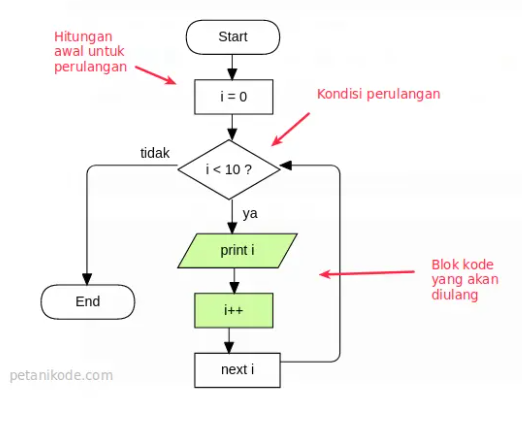
counter++;

}

Perulangan akan terjadi selama variabel ulangi brenilai y. Lalu kita menggunakan fungsi scanf() untuk mengambil input. Selama kita menjawab y pada input, maka perulangan akan terus dilakukan.Tapi kalau kita jawab yang lain, maka perulangan akan dihentikan. Karena kondisi perulangannya tidak terpenuhi.

1. **Perulangan For**

Perulangan for merupakan perulangan yang termasuk dalam couted loop, karena sudah jelas berapa kali ia akan mengulang.



Bentuknya kodenya seperti ini:

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

printf("Perulangan ke-%i\n", i);

}

Yang perlu diperhatikan adalah kondisi yang ada di dalam kurung setelah kata for.

Kondisi ini akan menentukan:

Hitungan akan dimulai dari 0 (i = 0); Hitungannya sampai berapa? Sampai i < 10; Lalu di setiap perulangan i akan bertambah +1 (i++). Variabel i pada perulangan for berfungsi untuk menyimpan nilai hitungan. Jadi setiap perulangan dilakukan nilai i akan selalu bertambah satu. Karena kita menentukannya di bagian i++.

Contoh program loop dengan for:

#include <iostream>

using namespace std;

int main(){

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

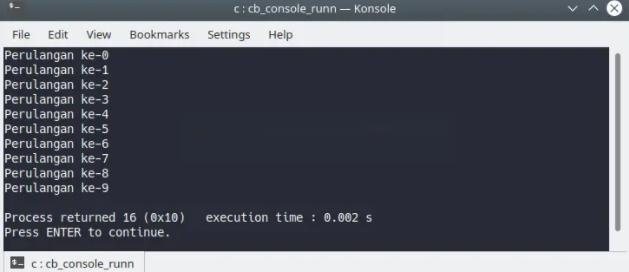
printf("Perulangan ke-%i\n", i);

}

return 0;

}

hasil outputnya:



Apakah nama variabelnya harus selalu i?

Tidak.

Kita juga bisa menggunakan nama lain.

Misalnya:

#include <iostream>

using namespace std;

int main(){

for(int counter = 0; counter < 10; counter += 2){

printf("Perulangan ke-%i\n", counter);

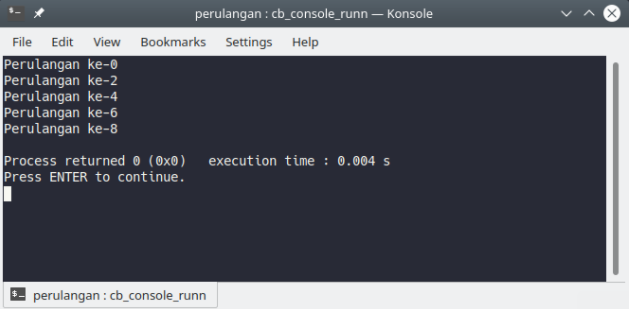
}

return 0;

}

Pada contoh tersebut, kita melakukan perulangan dimulai dari nol 0. Lalu di setiap perulangan nilai variabel couter akan ditambah 2 (counter+=2).

Hasilnya:



Bagaimana kalau counter perulangannya dimulai dari akanga yang lebih besar sampai yang ke paling kecil?

Ini biasanya kita buat saat ingin menghitung mundur…

Caranya gampang.

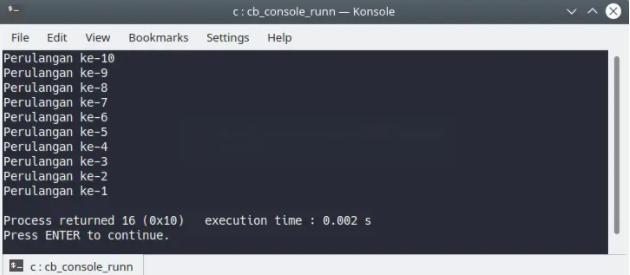
Kita tinggal isi nilai counter dengan nilai terbesarnya. Misalnya kita akan mulai hitungan dari 10 sampai ke 0. Maka nilai counter, kita isi awalnya dengan 10. Lalu di kondisi perbandingannya, kita berikan counter > 0. Artinya perulangan akan dilakukan selama nilai counter lebih besar dari 0. Lalu kita kurangi (-1) nilai counter di setiap perulangan (counter--).

for(int counter = 10; counter > 0; counter--){

printf("Perulangan ke-%i\n", counter);

}

Hasilnya:



Mengapa tidak sampai nol (0)?

Karena kondisi yang kita berikan counter > 0. Apabila counter bernilai 0, maka kondisi ini akan menjadi false.

Kecuali kita menggunakan operator lebih besar sama dengan (>=), maka jika counter bernilai 0, kondisi akan menjadi true.