Perulangan

Tim Olimpiade Komputer Indonesia

Pendahuluan

Melalui dokumen ini, kalian akan:

- Memahami konsep perulangan.
- Mempelajari struktur **for** dan **while** pada C++.

Motivasi

- Hari ini, Pak Dengklek ingin menyambut N ekor bebeknya yang baru lahir dari telur.
- Diberikan N, cetak tulisan "halo dunia!" sebanyak N kali!
- Contoh untuk N = 3:

```
halo dunia!
halo dunia!
halo dunia!
```

Motivasi (lanj.)

- Solusi "if (N = 1) <cetak satu kali>, else if (N = 2) <cetak dua kali>, ..." tidak mungkin digunakan, karena N bisa jadi sangat besar.
- Kita membutuhkan suatu struktur yang memungkinkan untuk mengulangi serangkaian pekerjaan!
- C++ menyediakan struktur perulangan berupa for dan while.

Perulangan: for

- Biasanya digunakan ketika kita tahu berapa kali perulangan perlu dilakukan.
- Pada C++, strukturnya:

```
for (<kondisi_awal>; <kondisi_ulang>; <perubahan>) {
  <perintah 1>;
  <perintah 2>;
  ...
}
```

- <kondisi_awal> dapat diisi dengan inisialisasi variabel untuk perulangan.
- <kondisi_ulang> biasanya berupa ekspresi yang menghasilkan `boolean`, untuk menandakan apakah perulangan sudah patut diberhentikan.
- <perubahan> merupakan bagian yang dieksekusi pada akhir setiap siklus perulangan.
- Penjelasan berikutnya dengan contoh akan meningkatkan pemahaman kalian.

Contoh Program: for.cpp

- Ketikkan program berikut dan coba jalankan:

```
#include <cstdio>

int main() {
  int N;
  scanf("%d", &N);

  for (int i = 0; i < N; i++) {
    printf("tulisan ini dicetak saat i = %d\n", i);
  }
  printf("akhir dari program\n");
}
```

- Masukkan berbagai nilai N, misalnya 1, 2, 10, dan 0.

Penjelasan Program: for.cpp

- Misalnya kita memasukkan `N` = 5.
- Pertama kali dijalankan, `i` dibuat dan diisi nilai 0.
- Kedua, C++ memeriksa apakah kondisi ulang tercapai. Berhubung `i` kurang dari `N`, maka bagian dalam for dilaksanakan dan tulisan dicetak saat `i` = 0.
- Setelah itu, akhir dari struktur for ditemukan. C++ akan mengeksekusi bagian perubahan, yakni menambah `i` dengan 1, lalu kembali ke awal dari for.
- Jika `i` masih kurang dari `N`, maka perintah di dalamnya akan kembali dilaksanakan.
- Dengan demikian, tercetaklah tulisan saat `i` = 1, 2, dan seterusnya hingga `N`-1.

Penjelasan Program: for.cpp (lanj.)

- Jika `i` sudah lebih dari `N`, perulangan akan berhenti dan C++ akan menjalankan perintah sesudah for tersebut.
- Pada contoh ini, mencetak tulisan "akhir dari program".

Masa Hidup Variabel

- Variabel `i` hanya memiliki masa hidup di dalam lingkungan for.
- Di luar for berikut { dan }, variabel tersebut sudah tidak ada.
- Anda boleh saja mendeklarasikan variabel `i` lagi, tanpa mendapatkan *error* bahwa nama variabel telah terdefinisi.
- Variabel yang hanya hidup dalam suatu cakupan { } disebut dengan variabel lokal.

Contoh Program: fordownto.cpp

- Struktur for cukup luwes untuk kasus-kasus lainnya.
- Berikut ini contoh dari penggunaan for yang bekerja secara terbalik:

```
#include <cstdio>

int main() {
  int N;
  scanf("%d", &N);

  for (int i = N-1; i >= 0; i--) {
    printf("tulisan ini dicetak saat i = %d\n", i);
  }
  printf("akhir dari program\n");
}
```

Contoh Program: forskip.cpp

- Sementara berikut contoh dari penggunaan for yang lompatannya 2:

```
#include <cstdio>

int main() {
  int N;
  scanf("%d", &N);

  for (int i = 0; i < N; i += 2) {
    printf("tulisan ini dicetak saat i = %d\n", i);
  }
  printf("akhir dari program\n");
}
```

- Ekspresi `i += 2` setara dengan `i = i + 2`, yang mengisikan `i` dengan dirinya ditambah 2.

Contoh Program: forskip2.cpp

- Tidak terbatas pada penjumlahan, hal semacam ini pun bisa dilakukan:

```
#include <cstdio>

int main() {
  int N;
  scanf("%d", &N);

  for (int i = 0; i < N; i *= 2) {
    printf("tulisan ini dicetak saat i = %d\n", i);
  }
  printf("akhir dari program\n");
}
```

- Ekspresi `i *= 2` setara dengan `i = i * 2`, yang mengisikan `i` dengan dirinya dikali 2.

Contoh Program: forsum.cpp

- Berikut contoh program untuk menjumlahkan seluruh bilangan di antara dua bilangan:

```
#include <cstdio>

int main() {
  int awal, akhir;
  scanf("%d %d", &awal, &akhir);

  int jumlah = 0;
  for (int i = awal; i <= akhir; i++) {
    jumlah += i;
  }
  printf("jumlah bilangan bulat di antara %d dan %d
    (inklusif) adalah %d\n", awal, akhir, jumlah);
}
```

Perulangan: while

- Selain for, terdapat pula struktur while.
- Biasa digunakan ketika tidak diketahui harus berapa kali serangkaian perintah dilaksanakan, tetapi diketahui perintah-perintah itu perlu dilaksanakan selama suatu kondisi terpenuhi.
- Pada C++, strukturnya:

```
while (<kondisi>) {
  <perintah 1>;
  <perintah 2>;
  ...
}
```

- Seperti pada if, <kondisi> adalah suatu nilai boolean. Selama nilainya **TRUE**, seluruh <perintah x> di dalamnya akan dieksekusi secara berurutan.

Contoh Program: while.cpp

- Berikut adalah contoh penggunaan **while** untuk kasus yang sama dengan for.cpp:

```
#include <cstdio>

int main() {
  int N;
  scanf("%d", &N);

  int i = 0;
  while (i < N) {
    printf("tulisan ini dicetak saat i = %d\n", i);
    i++;
  }
  printf("akhir dari program\n");
}
```

Penjelasan Program: while.cpp

- Alur programmnya sangat mirip dengan for.
- Perbedaannya hanya pada gaya penulisan.

```
// Bentuk for
for (<kondisi_awal>; <kondisi_ulang>; <perubahan>) {
  <perintah 1>;
  <perintah 2>;
  ...
}
// Bentuk while
<kondisi_awal>;
while (<kondisi_ulang>) {
  <perintah 1>;
  <perintah 2>;
  ...
  <perubahan>;
}
```

Perulangan: while (lanj.)

- Perhatikan bahwa perintah "`i++`" diperlukan, supaya suatu saat nanti <kondisi> pada while akan tidak dipenuhi.
- Sekarang coba hapus perintah "`i++`" pada while.cpp, dan jalankan kembali programnya.
- Apa yang terjadi? Program akan terjebak dalam *infinite loop*, atau perulangan yang tidak akan pernah berhenti! Gunakan tombol CTRL+C pada *keyboard* untuk memberhentikan program secara paksa.
- Dengan demikian, pastikan suatu saat "kondisi pada while" tidak dipenuhi, atau program tidak akan pernah berhenti :)

Contoh Program: whilesum.cpp

- Berikut ini contoh program dengan while yang melakukan hal serupa dengan forsum.cpp:

```
#include <cstdio>

int main() {
  int awal, akhir;
  scanf("%d %d", &awal, &akhir);

  int jumlah = 0;
  int i = awal;
  while (i <= akhir) {
    jumlah += i;
    i++;
  }
  printf("jumlah bilangan bulat di antara %d dan %d
    (inklusif) adalah %d\n", awal, akhir, jumlah);
}
```

Perulangan: while (lanj.)

- Terdapat variasi lain dari while, yang biasa disebut "do ... while".
- Pada C++, strukturnya:

```
do {
  <perintah 1>;
  <perintah 2>;
  ...
} while (<kondisi>);
```

- Perbedaannya adalah, seluruh perintah akan dilakukan dulu, baru diperiksa apakah kondisi masih terpenuhi. Bila ya, maka seluruh perintah akan diulang.
- Hal ini menjamin seluruh perintah dijalankan paling sedikit satu kali.

Contoh Program: dowhile.cpp

- Berikut ini adalah contoh program dengan **do ... while** yang menjalankan tugas serupa dengan for.cpp dan while.cpp.

```
#include <cstdio>

int main() {
  int N;
  scanf("%d", &N);

  int i = 0;
  do {
    printf("tulisan ini dicetak saat i = %d\n", i);
    i++;
  } while (i < N);
  printf("akhir dari program\n");
}
```

Sejauh ini...

Kalian sudah belajar tentang:

- Konsep perulangan pada pemrograman.
- Struktur for dan while, beserta kegunaan dan perbedaannya.

Selanjutnya kita akan memasuki tentang:

- Penggunaan perulangan yang lebih kompleks, yaitu perulangan bersarang.
- Membuat program dengan apa yang telah dipelajari sejauh ini.