Perulangan

Tim Olimpiade Komputer Indonesia

Pendahuluan

Melalui dokumen ini, kalian akan:

- Memahami konsep perulangan.
- Mempelajari struktur **for**, **while**, dan **repeat** pada Pascal.

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) then <cetak satu kali>, else if (N = 2) then <cetak dua kali>, ..." tidak mungkin digunakan, karena **N** bisa jadi sangat besar.
- Kita membutuhkan suatu struktur yang memungkinkan untuk mengulangi serangkaian pekerjaan!

Perulangan

- Umumnya setiap bahasa pemrograman memberikan struktur perulangan yang bentuknya bisa berupa:
  - Perulangan dengan pencacah. Contoh: "untuk i dari 1 sampai N, cetak 'halo dunia!'".
  - Perulangan selama dicapai suatu kondisi. Contoh: "selama i $<$ N, cetak 'halo dunia!' dan tambah i dengan 1".
  - Perulangan sampai dicapai suatu kondisi. Contoh: "cetak 'halo dunia!' dan tambah i dengan 1, hingga tercapai i $=$ N".
- Pada Pascal, ada ketiga struktur itu yang masing-masing diwakili dengan **for**, **while**, dan **repeat**.

Perulangan: for

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

```
for <pencacah> := <awal> to <akhir> do begin
  <perintah 1>;
  <perintah 2>;
  ...
end;
```

- Dengan <pencacah> adalah suatu variabel bertipe data ordinal, <awal> dan <akhir> adalah nilai awal dan akhir untuk pencacah, dan <perintah x> adalah perintah yang akan diulang.
- "Pencacah" mungkin istilah yang asing bagi kalian. Penjelasan berikutnya dengan contoh akan meningkatkan pemahaman kalian.

Contoh Program: for.pas

- Ketikkan program berikut dan coba jalankan:

```
var
  N, i: longint;
begin
  write('Masukkan nilai N: ');
  readln(N);

  for i := 1 to N do begin
    writeln('tulisan ini dicetak saat i = ', i);
  end;

  writeln('akhir dari program');
end.
```

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

Penjelasan Program: for.pas

- Pada contoh tersebut, **i** merupakan variabel pencacah yang bertipe **longint**.
- Pertama kali dijalankan, **i** akan bernilai 1 dan tulisan dicetak saat **i** = 1.
- Setelah itu, **end** dari struktur **for** ditemukan. Pascal akan menambahkan **i** dengan 1, lalu kembali ke awal dari **for**. Jika **i** masih belum lebih dari **N**, maka perintah di dalamnya akan kembali dilaksanakan.
- Dengan demikian, tercetaklah tulisan saat **i** = 2, 3, dan seterusnya hingga **N**.

Penjelasan Program: for.pas (lanj.)

- Jika **i** sudah lebih dari **N**, perulangan akan berhenti dan Pascal akan menjalankan perintah selanjutnya.
- Pada contoh ini, mencetak tulisan "akhir dari program".
- Dalam kasus ini, variabel **i** mencacah 1, 2, 3, ..., **N**.

Perulangan: for (lanj.)

- Struktur **for** hanya bisa mencacah dari suatu nilai awal ke suatu nilai akhir, dan pencacah akan "bertambah satu tingkat" setiap waktunya.
- Untuk keperluan pencacah "turun satu tingkat" setiap waktunya, terdapat sebuah variasi lain dari **for**, yaitu **for downto** dengan struktur:

```
for <pencacah> := <awal> downto <akhir> do begin
  <perintah 1>;
  <perintah 2>;
  ...
end;
```

- Ya, cukup ubah **to** menjadi **downto**. Kini **for** akan bekerja secara menurun. Tentu saja jika <nilai awal> dan <nilai akhir> nilainya disesuaikan pula.

Contoh Program: fordownto.pas

- Berikut ini contoh dari penggunaan **for downto**:

```
var
  N, i: longint;
begin
  write('Masukkan nilai N: ');
  readln(N);

  for i := N downto 1 do begin
    writeln('tulisan ini dicetak saat i = ', i);
  end;

  writeln('akhir dari program');
end.
```

Hal Penting untuk Perulangan for

Penting untuk diketahui:

- Tipe data dari pencacah, nilai awal, dan nilai akhir harus sama dan bersifat ordinal.
- Karena tipe data pencacah yang penting merupakan tipe data ordinal, artinya pencacah boleh saja berupa **char**.
- Nilai pencacah tidak boleh diubah saat perulangan dikerjakan.
- Pascal tidak bisa melayani **for** yang pencacahnya "bertambah dua tingkat" pada setiap perulangan.

Contoh Program: jumlahfor.pas

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

```
var
  awal, akhir, i: longint;
  jumlah: longint;
begin
  write('Nilai awal: '); readln(awal);
  write('Nilai akhir: '); readln(akhir);

  jumlah := 0;
  for i := awal to akhir do begin
    jumlah := jumlah + i;
  end;

  writeln('jumlah bilangan bulat di antara awal dan
    akhir (inklusif) adalah ', jumlah);
end.
```

Perulangan: while

- Biasa digunakan ketika tidak diketahui harus berapa kali serangkaian perintah dilaksanakan, tetapi diketahui perintah-perintah itu perlu dilaksanakan selama suatu kondisi terpenuhi.
- Pada Pascal, strukturnya:

```
while <kondisi> do begin
  <perintah 1>;
  <perintah 2>;
  ...
end;
```

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

Contoh Program: while.pas

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

```
var
  N, i: longint;
begin
  write('Masukkan nilai N: ');
  readln(N);

  i := 1;
  while (i <= N) do begin
    writeln('tulisan ini dicetak saat i = ', i);
    i := i + 1;
  end;

  writeln('akhir dari program');
end.
```

Penjelasan Program: while.pas

- Misalkan dimasukkan nilai **N** = 5.
- Pada awalnya, nilai **i** diinisialisasi dengan 1.
- Kemudian diperiksa apakah dipenuhi **i** <= **N**. Karena dipenuhi, perintah mencetak saat **i** = 1 dilaksanakan. Demikian pula dengan perintah "i := i + 1". Kini nilai **i** = 2.
- Selanjutnya ditemukan **end** dari **while**. Pascal akan kembali ke awal dari **while** dan memeriksa apakah masih dipenuhi kondisi yang diberikan.
- Jika masih, maka perintah di dalam while akan kembali dilaksanakan.
- Jika sudah tidak dipenuhi, maka Pascal akan mengeksekusi perintah-perintah di bawah **end** dari **while**, yakni mencetak tulisan "akhir dari program".

Perulangan: while (lanj.)

- Perhatikan bahwa perintah "i := i + 1" diperlukan, supaya suatu saat nanti <kondisi> pada while akan tidak dipenuhi.
- Sekarang coba hapus perintah "i := i + 1" pada while.pas, 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.
- Pastikan suatu ketika "kondisi pada **while**" tidak dipenuhi, atau program tidak akan pernah berhenti :)

Contoh Program: jumlahwhile.pas

- Berikut ini contoh program dengan while yang melakukan hal serupa dengan jumlahfor.pas:

```
var
  awal, akhir, i, jumlah: longint;
begin
  write('Nilai awal: '); readln(awal);
  write('Nilai akhir: '); readln(akhir);

  jumlah := 0;
  i := awal;
  while (i <= akhir) do begin
    jumlah := jumlah + i;
    i := i + 1;
  end;

  writeln('jumlah bilangan bulat di antara awal dan
    akhir (inklusif) adalah ', jumlah);
end.
```

Perulangan: while (lanj.)

- Struktur **while** bisa jadi lebih fleksibel dari struktur **for**, karena tipe datanya tidak harus ordinal.
- Kalian bisa membuat pencacah yang tidak sekedar "naik satu tingkat". Contoh:

```
i := 1;
while (i <= N) do begin
  writeln('i = ', i);
  i := i + 2;
end;
```

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

```
i := 1;
while (i <= N) do begin
  writeln('i = ', i);
  i := i * 5;
end;
```

Perulangan: while (lanj.)

- Selain itu, dengan menggunakan **while**, kalian tidak perlu tahu berapa kali suatu perulangan perlu dilakukan.
- Contoh soal: Pak Dengklek memberikan serangkaian bilangan, satu pada setiap barisnya, dan cetak bilangan itu. Ketika menemukan bilangan negatif, berhenti membaca dan akhiri program.
- Hal ini tidak dapat dilakukan dengan **for**, karena kita tidak tahu berapa nilai akhir dari **for** yang harus dilakukan.
- Dengan **while**, hal ini dapat dilakukan dengan mudah:

```
readln(x);
while (x >= 0) do begin
  writeln('bilangan yang dibaca: ', x);
  readln(x);
end;
```

Perulangan: repeat

- Struktur perulangan yang lain adalah **repeat**.
- Mirip dengan **while**, tetapi kali ini bentuknya adalah "lakukan perintah berikut, sampai dicapai suatu kondisi".
- Perhatikan bedanya:
  - **while**: selama suatu kondisi terpenuhi
  - **repeat**: sampai suatu kondisi terpenuhi
- Pada Pascal, strukturnya:

```
repeat
  <perintah 1>;
  <perintah 2>;
  ...
until <kondisi>;
```

Contoh Program: repeat.pas

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

```
var
  N, i: longint;
begin
  write('Masukkan nilai N: ');
  readln(N);

  i := 1;
  repeat
    writeln('tulisan ini dicetak saat i = ', i);
    i := i + 1;
  until (i > N);

  writeln('akhir dari program');
end.
```

Penjelasan Program: repeat.pas

- Misalkan **N** = 5.
- Awalnya **i** diinisialisasi dengan 1.
- Memasuki struktur repeat, perintah mencetak saat **i** = 1 dilaksanakan. Kemudian setelah dilaksanakan "i := i + 1", **i** bernilai 2.
- Memasuki **until**, diperiksa apakah **i** > **N**. Karena belum dipenuhi, maka seluruh perintah sesudah kata kunci **repeat** akan kembali dilaksanakan.
- Ketika **i** = 5, perintah mencetak saat **i** = 5 akan dilaksanakan. Setelah itu, **i** ditambah 1 menjadi 6.
- Karena sudah dipenuhi **i** > **N**, maka perulangan berhenti dan dilaksanakan perintah di bawah kata **until**. Dalam contoh ini, mencetak tulisan "akhir dari program".

Perulangan: repeat (lanj.)

- Seperti pada **while**, pastikan suatu saat nanti kondisi berhenti dicapai supaya tidak terjebak pada *infinite loop*.
- Perhatikan bahwa seluruh perintah di dalam "repeat ... until" pasti dilaksanakan setidaknya satu kali. Karena pemeriksaan pada kondisi baru di lakukan setelah seluruh perintah di dalamnya dilaksanakan.

Contoh Program Lagi: jumlahrepeat.pas

- Berikut adalah contoh yang serupa dengan jumlahfor.pas.

```
var
  awal, akhir, i: longint;
  jumlah: longint;
begin
  write('Nilai awal: '); readln(awal);
  write('Nilai akhir: '); readln(akhir);

  jumlah := 0;
  i := awal;
  repeat
    jumlah := jumlah + i;
    i := i + 1;
  until (i > akhir);

  writeln('jumlah bilangan bulat di antara awal dan
    akhir (inklusif) adalah ', jumlah);
end.
```

Sejauh ini...

Kalian sudah belajar tentang:

- Konsep perulangan pada pemrograman.
- Struktur **for**, **while**, dan **repeat** 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.