Percabangan

Tim Olimpiade Komputer Indonesia

Pendahuluan

Melalui dokumen ini, kalian akan:

- Mengenal percabangan.
- Memahami analisa kasus dan mengimplementasikannya pada Pascal.

Motivasi

- Bebek-bebek Pak Dengklek sedang belajar tentang membedakan bilangan positif, nol, atau negatif.
- Karena bebek-bebek kebingungan, mereka memberikan kalian sebuah bilangan dan meminta kalian menentukan apakah bilangan itu positif atau bukan positif!
- Jika positif, cetak "positif". Jika tidak, jangan cetak apa-apa.

Motivasi (lanj.)

- Sebuah bilangan dinyatakan positif apabila bilangan tersebut lebih dari nol.
- Dengan begitu, kita memerlukan suatu struktur yang memungkinkan "jika bilangan itu lebih dari 0, maka cetak positif".
- Pada Pascal, hal ini bisa diwujudkan dengan struktur kondisional **if**.

Struktur "if ... then ..."

- Struktur dari penulisan "if ... then ..." adalah:

```
if <kondisi> then begin
  <perintah 1>;
  <perintah 2>;
  ...
end;
```

- Dengan <kondisi> adalah suatu boolean.
- Jika nilai <kondisi> adalah **TRUE**, seluruh perintah yang ada di antara blok "begin" dan "end" akan dilaksanakan.
- Jika **FALSE**, seluruh perintah yang ada di antara blok "begin" dan "end" akan dilewati.

Blok "begin ... end"

- Struktur yang sebenarnya dari penulisan "if ... then ..." adalah:

```
if <kondisi> then
  <perintah>;
```

- Jika nilai <kondisi> adalah **TRUE**, <perintah> akan dilaksanakan.
- Jika nilai <kondisi> adalah **FALSE**, <perintah> tidak dilaksanakan.
- Lalu di mana bedanya?

Blok "begin ... end" (lanj.)

- Setelah kata kunci **then**, sebenarnya hanya satu perintah yang dapat dieksekusi jika <kondisi> bernilai **TRUE**.
- Perhatikan contoh berikut ini:

```
if (nilai = 10) then
   writeln('masuk');
   writeln('lagi');
```

Blok "begin ... end" (lanj.)

- Meskipun perintah **writeln('masuk')** hanya dieksekusi ketika **nilai** sama dengan 10, **writeln('lagi')** akan selalu dilaksanakan tanpa peduli isi variabel **nilai**.
- Blok "begin ... end", akan berperan sebagai "pembungkus" beberapa perintah menjadi "satu" perintah, sehingga berapapun perintah di dalam blok tersebut, akan dilihat oleh Pascal sebagai "satu" perintah.

Blok "begin ... end" (lanj.)

- Menulis "begin ... end" setiap sesudah **if** merupakan kebiasaan yang bagus, meskipun isi dari **if** itu hanya satu perintah.
- Dengan cara ini, ketika ada tambahan perintah yang perlu dimasukkan ke dalam **if**, kalian tidak perlu menuliskan lagi "begin ... end".
- Konsisten dengan selalu menulis "begin ... end" juga menjaga program tetap rapi.
- Sebagai informasi, seluruh kode yang memuat **if** pada tulisan ini akan selalu menggunakan "begin ... end".

Contoh Program: kondisi.pas

- Ketikkan dan jalankan program berikut:

```
var
  x: longint;
begin
  readln(x);

  if (x > 0) then begin
    writeln('positif');
  end;
end.
```

- Perhatikan bahwa ekspresi "x $>$ 0" akan merupakan operasi relasional yang menghasilkan nilai boolean. Sehingga tepat untuk digunakan pada if.
- Bagaimana jika ingin dibuat jika bilangan itu bukan positif, cetak "non-positif"?

Struktur "if ... then ... else ..."

- Kita juga bisa membuat percabangan jika nilai pada <kondisi> adalah **FALSE**, yaitu dengan kata kunci **else**.
- Struktur dari penulisan "if ... then ... else ..." adalah:

```
if <kondisi> then begin
  <perintah 1>;
  <perintah 2>;
  ...
end else begin
  <perintah a>;
  <perintah b>;
  ...
end;
```

- Jika nilai <kondisi> adalah **TRUE**, <perintah 1>, <perintah 2>, ..., akan dilaksanakan.
- Jika **FALSE**, <perintah a>, <perintah b>, ..., akan dilaksanakan.

Contoh Program: kondisi2.pas

- Dengan "if ... then ... else ...", kita bisa memodifikasi kondisi.pas menjadi kondisi2.pas:

```
var
  x: longint;
begin
  readln(x);

  if (x > 0) then begin
    writeln('positif');
  end else begin
    writeln('non-positif');
  end;
end.
```

Persoalan Sebenarnya

Ketika bebek-bebek memberikan kalian sebuah bilangan, sebut saja **x**, mereka ingin tahu:

- Jika **x** positif, cetak "positif".
- Jika **x** sama dengan nol, cetak "nol".
- Jika **x** negatif, cetak "negatif".

Pada kasus ini, diperlukan struktur **if** yang lebih dari dua cabang!

Struktur "if ... then ... else if ..."

- Pascal menyediakan struktur yang memungkinkan kita memilah-milah untuk cabang yang lebih dari dua, yaitu dengan struktur "if ... then ... else if ...".
- Struktur dari penulisan "if ... then ... else if ..." adalah:

```
if <kondisi 1> then begin
  <perintah 1>;
  <perintah 2>;
  ...
end else if <kondisi 2> then begin
  <perintah a>;
  <perintah b>;
  ...
end else if <kondisi 3> then begin
  ...
end
```

Struktur "if ... then ... else if ..." (lanj.)

- Jika nilai <kondisi 1> **TRUE**, <perintah 1>, <perintah 2>, ..., akan dilaksanakan.
- Jika nilai <kondisi 1> **FALSE**, diperiksa apakah <kondisi 2> bernilai **TRUE**. Jika ya, <perintah a>, <perintah b>, ..., akan dilaksanakan.
- Jika nilai <kondisi 2> **FALSE**, diperiksa apakah <kondisi 3> bernilai **TRUE**. Hal ini akan terus diulang sampai seluruh percabangan habis.
- Kalian juga bisa mengakhiri struktur ini dengan "else ...", yaitu ketika seluruh kondisi yang diberikan tidak terpenuhi, maka perintah-perintah di bawah **else** ini yang akan dilaksanakan.

Contoh Program: kondisi3.pas

- Dengan "if ... then ... else if ...", kita bisa memodifikasi kondisi2.pas menjadi kondisi3.pas:

```
var
  x: longint;
begin
  readln(x);

  if (x > 0) then begin
    writeln('positif');
  end else if (x = 0) then begin
    writeln('nol');
  end else if (x < 0) then begin
    writeln('negatif');
  end;
end.
```

Contoh Program: kondisi4.pas

- Pada kondisi3.pas, sebenarnya "else if ..." yang terakhir tidak diperlukan.
- Ketika suatu bilangan bukan positif dan bukan nol, sudah pasti bilangan itu negatif. Sehingga bisa didapatkan kondisi4.pas:

```
var
  x: longint;
begin
  readln(x);
  if (x > 0) then begin
    writeln('positif');
  end else if (x = 0) then begin
    writeln('nol');
  end else begin
    writeln('negatif'); (* sudah pasti negatif *)
  end;
end.
```

Kombinasi dengan Ekspresi Boolean

- Kalian juga bisa menggabungkan struktur if dengan ekspresi **boolean**:

```
if ((x > 0) and (x mod 2 = 1)) then begin
  writeln('positif dan ganjil');

end else if ((x > 0) and (x mod 2 = 0)) then begin
  writeln('positif dan genap');

end else if ((x < 0) and (x mod 2 = 1)) then begin
  writeln('negatif dan ganjil');

end else if ((x < 0) and (x mod 2 = 0)) then begin
  ...
```

If Bersarang

- Solusi yang lebih rapi dicapai dengan menggunakan if secara bersarang:

```
if (x > 0) then begin
  if (x mod 2 = 1) then begin
    writeln('positif dan ganjil');
  end else begin
    writeln('positif dan genap');
  end;
end else if (x < 0) then begin
  ...
```

Contoh Lainnya

- Misalkan diberikan sebuah karakter yang merupakan salah satu dari 'A', 'B', 'C', atau 'D'.
- Cetak frase berikut sesuai karakter yang diberikan:
  - 'A': Sempurna
  - 'B': Bagus
  - 'C': Cukup
  - 'D': Kurang

Contoh Lainnya (lanj.)

- Tentu saja kita bisa menggunakan "if ... then ... else if ..." untuk kasus ini.
- Namun, menuliskan "if ... then ... else if ..." untuk kasus sederhana ini bisa cukup panjang.
- Bayangkan, untuk setiap kasus, kalian perlu menulis:

```
if (nilai = 'A') then begin
  ...
end else begin if (nilai = 'B') then begin
  ...
```

- Untuk kasus sejenis ini, tersedia struktur kondisional lain yang ditawarkan Pascal dan lebih cocok, yaitu **case**.

Struktur "case ... of ..."

- Struktur dari penulisan "case ... of ..." adalah:

```
case <ekspresi> of
  <nilai 1>: begin <perintah 1>; end;
  <nilai 2>: begin <perintah 2>; end;
  <nilai 3>: begin <perintah 3>; end;
  ...
end;
```

- Dengan <ekspresi> adalah suatu ekspresi yang menghasilkan nilai ordinal.
- Ketika nilai yang dihasilkan <ekspresi> sama dengan <nilai 1>, maka seluruh <perintah 1> akan dilaksanakan.
- Ketika nilai yang dihasilkan <ekspresi> sama dengan <nilai 2>, maka seluruh <perintah 2> akan dilaksanakan.
- ... dan seterusnya.

Contoh Program: "case.pas"

- Berikut ini adalah solusi dengan menggunakan "case ... of ...":

```
var
  nilai: char;
begin
  readln(nilai);

  case (nilai) of
    'A': begin writeln('Sempurna'); end;
    'B': begin writeln('Bagus'); end;
    'C': begin writeln('Cukup'); end;
    'D': begin writeln('Kurang'); end;
  end;
end.
```

- Seperti pada **if**, blok "begin ... end" sebenarnya tidak harus dituliskan. Untuk menjaga konsistensi dan mempermudah jika ada penambahan kode, ada baiknya tetap dituliskan.

Struktur "case ... of ..." dengan Rentang

- Struktur dari penulisan "case ... of ..." juga dapat ditulis untuk suatu rentang:

```
case <ekspresi> of
  <nilai 1a>..<nilai 1b>: begin <perintah 1>; end;
  <nilai 2a>..<nilai 2b>: begin <perintah 2>; end;
  <nilai 3a>..<nilai 3b>: begin <perintah 3>; end;
  ...
end;
```

- Perhatikan bahwa <ekspresi> selalu menghasilkan tipe data ordinal, sehingga pemeriksaan rentang dapat dilakukan.

Contoh Lagi

- Misalkan diberikan sebuah bilangan bulat yang menyatakan nilai.
- Cetak frase berikut sesuai nilai yang diberikan:
  - Antara 0 sampai 59: Kurang
  - Antara 60 sampai 74: Cukup
  - Antara 75 sampai 89: Baik
  - Antara 90 sampai 99: Sangat Baik
  - Nilai 100: Sempurna

Contoh Program: "case-2.pas"

- Berikut ini adalah contoh program "case ... of ..." untuk suatu rentang:

```
var
  nilai: longint;
begin
  readln(nilai);

  case (nilai) of
    0..59: begin writeln('Kurang'); end;
    60..74: begin writeln('Cukup'); end;
    75..89: begin writeln('Baik'); end;
    90..99: begin writeln('Sangat Baik'); end;
    100..100: begin writeln('Sempurna'); end;
  end;
end.
```

Contoh Program: "case-3.pas"

- Penulisan "case ... of ..." juga bisa ditulis secara campuran, antara dengan atau tanpa rentang.
- Dengan demikian, kasus untuk nilai 100 dapat dipersingkat:

```
var
  nilai: longint;
begin
  readln(nilai);

  case (nilai) of
    0..59: begin writeln('Kurang'); end;
    60..74: begin writeln('Cukup'); end;
    75..89: begin writeln('Baik'); end;
    90..99: begin writeln('Sangat Baik'); end;
    100: begin writeln('Sempurna'); end;
  end;
end.
```

Selanjutnya...

- Ke bagian yang lebih menarik lagi, yaitu perulangan!
- Pastikan kalian menguasai materi percabangan terlebih dahulu.