Contoh Materi Pelatihan Jarak Jauh
Pra OSN
BIDANG INFORMATIKA/KOMPUTER

Disusun Oleh:
Tim Pembina Olimpiade Sains Bidang Komputer
dan Alumni TOKI

DAFTAR ISI

MATERI PJJ PRA OSN 2008

Bagian Pertama[1]

1. Readln dan Writeln

Nama Program:	pjj0101.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Sebagai perkenalan pertama dengan program Pascal, ketikanlah perintah-perintah program berikut ini lalu simpan sebagai 'pjj0101.pas'.

```
Program Pjj0101;
var
    brs: string;
begin
    readln(brs);
    writeln(brs);
end.
```

Program ini akan membaca satu baris teks masukan (dari standard input) dan mencetak keluaran (ke standard output) yang persis sama dengan masukan. Pada bagian awal program terdapat pernyataan (deklarasi) yang menyebutkan digunakannya suatu variabel dengan nama brs. Deklarasi ditunjukkan dengan adanya notasi var. Baris-baris berikutnya setelah notasi var adalah tempat menuliskan deklarasi variabel-variabel.

Variabel adalah tempat menyimpan suatu harga dalam program, dan selama berjalannya program, harga itu dapat berubah-ubah. Setiap variabel dideklarasikan dengan menyebutkan jenis dari harga yang dapat disimpannya. brs dideklarasikan sebagai variabel berjenis string, berarti brs dapat menyimpan string yang panjangnya maksimum 255 karakter.

Badan program dinyatakan dengan perintah begin dan diakhiri dengan perintah end. (yaitu end dengan tanda titik). readln(brs) berguna untuk membaca satu baris string masukan dan hasil pembacaannya disimpan dalam variabel brs. Perintah writeln(brs) berguna untuk menuliskan isi variabel brs ke output. Dalam program

[1] Materi ini dikembangkan oleh Brian Marshal dan Derianto Kusuma berdasarkan Materi PJJ OSN 2007 yang disusun oleh Suryana Setiawan, M.Sc.

setiap perintah dipisahkan dengan tanda ";" (titik koma). Sebagai kebiasaan baik, akhiri setiap baris perintah dengan tanda titik koma seperti pada contoh di atas.

Untuk menguji program Anda, bukalah editor ('edit.exe' atau 'notepad.exe') lalu ketikan suatu teks sesuka anda dalam satu baris dengan panjang kurang dari 255 karakter. Simpanlah teks tersebut dalam file teks, misalnya dengan nama 'uji1.txt'. Kompilasi program tersebut dengan compiler yang anda gunakan, menjadi 'pjj0101.exe', lalu jalankan perintah pada command prompt.

```
pjj0101 < uji.txt
```

Jika program mengeluarkan keluaran yang sama dengan isi teks pada input, maka program Anda sudah berjalan dengan benar.

Sebagai latihan menggunakan penguji otomatis, tentunya jika anda memiliki akses internet, akses alamat server penguji, login sesuai dengan UserId dan Password yang telah Anda miliki, lalu submit program 'pjj0101.pas' tersebut. Jika Anda tidak memiliki akses internet dan hendak mengirimkannya melalui pos, maka simpanlah ke dalam disket atau cd (beserta latihan-latihan lainnya, demi menghemat biaya pos Anda!) lalu kirimkan kepada pembina TOKI.

Contoh Masukan

```
abc
```

Contoh Keluaran

```
abc
```

Jawaban Anda atas soal-soal yang terdapat dalam materi PJJ ini dapat di-*submit* langsung melalui **TOKI Learning Center** (http://www.toki.or.id/) dengan User ID dan password yang akan diberikan kepada Anda via E-mail, telepon, atau media lainnya, atau melalui pos ke:

Tim Pembina TOKI
Jln. Kyai Gede Utama no. 12
Bandung 40132

Jangan lupa menyertakan identitas lengkap Anda (minimal: nama, alamat surat, dan asal sekolah).

2. While Loop

Nama Program:	pjj0102.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Melanjutkan latihan pertama gantilah kedua baris readln(brs) dan writeln(brs) dengan deretan perintah berikut ini, yang berfungsi untuk membaca beberapa baris masukan dan menulis baris yang baru dibaca, satu demi satu baris. Jangan mengganti bagian lain dari program, kecuali nama program 'pjj0102'. Simpan sebagai 'pjj0102.PAS'.

```
while not eof(input) do
begin
    readln(brs);
    writeln(brs);
end;
```

Dalam deretan perintah di atas terdapat struktur loop while

```
while <kondisi> do
begin
    <perintah-perintah>
end;
```

Untuk menguji program anda, buatlah file teks input 'uji2.txt' (seperti 'uji1.txt' namun dituliskan dalam beberapa baris). Setelah dikompilasi, jalankan dengan perintah

```
pjj0102 < uji2.txt
```

Jika keluaran sama dengan yang dituliskan dalam file 'uji2.txt' maka program Anda sudah berjalan dengan benar. Ujilah dengan penguji otomatis seperti dijelaskan pada 'pjj0101.PAS'.

Contoh Masukan

```
abc
123
```

Contoh Keluaran

```
abc
123
```

3. While Loop + Counter

Nama Program:	pjj0103.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Jika pada latihan sebelumnya program membaca string demi string masukan, kini program Anda harus membaca bilangan-bilangan (satu bilangan dalam satu baris), menjumlahkan bilangan-bilangan tersebut, dan menuliskan jumlah total bilangan-bilangan tersebut setelah bilangan terakhir dibaca. Pembacaan dilakukan dengan cara yang sama, tetapi variabel yang digunakan haruslah variabel bertipe integer (bilangan bulat). Gantilah nama variabel brs dengan nama baru, misalnya bil. Tentu saja setiap perintah readln(brs) juga diganti dengan perintah readln(bil). Untuk menyimpan total jumlah bilangan yang dibaca, diperlukan sebuah variabel berjenis integer seperti halnya variabel bil. Mari beri nama variabel ini jml. Jadi deklarasi dituliskan

```
var
    bil: integer;
    jml: integer;
```

Karena selama pembacaan jml digunakan untuk mencatat jumlah hingga bilangan terakhir dibaca, maka di awal program variabel jml harus diberi harga awal (diinisialisasi) 0. Di dalam loop nilai jml harus ditambahkan dengan harga yang dibaca ke dalam variabel bil. Setelah loop selesai, di bagian bawah program isi jml dituliskan ke output. Untuk itu badan program dapat dituliskan sebagai

```
jml := 0;
while not eof(input) do
begin
   readln(bil);
   jml := jml + bil;
end;
writeln(jml);
```

Notasi ‘:=’ menyatakan bahwa hasil ekspresi di sebelah kanan tanda ':=' akan disimpan pada variabel yang tertulis di sebelah kiri ‘:=’. Beri nama program ini 'pjj0103' dan simpanlah dengan nama 'pjj0103.PAS'. Untuk menguji program Anda buatlah file teks input 'uji3.txt' yang setiap barisnya berisikan satu bilangan bulat (boleh negatif) yang panjangnya tidak lebih dari 4 digit.

```
pjj0103 < uji3.txt
```

Ujilah dengan penguji otomatis seperti dijelaskan pada latihan-latihan sebelumnya.

Contoh Masukan

```
1
2
3
```

Contoh Keluaran

```
6
```

4. Menjumlah per Kolom

Nama Program:	pjj0104.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Jika pada latihan ketiga masukan terdiri dari baris-baris yang setiap barisnya hanya satu bilangan bulat, pada latihan ini Anda mencoba untuk membaca baris-baris yang per barisnya berisi 3 bilangan bulat dan masing-masing dipisahkan satu spasi. Lalu Anda diminta menjumlahkan bilangan-bilangan pada setiap kolom dengan variabel-variabel penjumlah yang berbeda. Kolom pertama dengan penjumlah pertama, kolom kedua dengan penjumlah kedua, dan kolom ketiga dengan penjumlah ketiga. Ketiga hasil penjumlahan tersebut dicetak dalam satu baris yang sama yang dipisahkan dengan satu spasi.

Untuk itu Anda perlu menggunakan 3 variabel pembaca dan 3 variabel penjumlah. Kita namai ketiga variabel pembaca itu bil1, bil2, dan bil3 sementara ketiga variabel penjumlah diberi nama jml1, jml2 dan jml3. Dalam Pascal, deklarasi beberapa variabel sejenis dapat dituliskan dalam satu baris seperti

```
var
    bil1, bil2, bil3, jml1, jml2, jml3: integer;
```

Namun, demi memudahkan pembacaan program kembali, sebaiknya variabel-variabel yang memiliki fungsi yang berbeda dideklarasi secara terpisah:

```
var
    bil1, bil2, bil3: integer;
    jml1, jml2, jml3: integer;
```

Ketiga bilangan dalam satu baris dapat sekaligus dibaca sesuai dengan urutannya serta spasinya. Perintah

```
    readln(bil1, bil2, bil3);
```

akan membaca ketiga bilangan sekaligus dari satu baris masukan. Penanda batas antar bilangan saat pembacaan dalam Pascal adalah tanda spasi.

Penjumlahaan masing-masing bilangan tersebut tentu saja harus dilakukan terhadap pejumlah masing-masing

```
jml1 := jml1 + bil1;
jml2 := jml2 + bil2;
jml3 := jml3 + bil3;
```

Pencetakan jml1, jml2, dan jml3 dalam satu baris yang sama (dengan pemisah spasi yang harus dituliskan juga karena kalau tidak maka bilangan-bilangan dituliskan bersambungan!) dapat dilakukan dengan perintah

```
writeln(jml1,' ', jml2,' ', jml3);
```

Lakukan juga pengujian seperti pada latihan-latihan sebelumnya.

Contoh Masukan

```
1 2 3
2 3 4
```

Contoh Keluaran

```
3 5 7
```

5. Menjumlah dalam Satu Baris

Nama Program:	pjj0105.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Jika pada latihan ketiga bilangan-bilangan dituliskan pada masing-masing baris, maka kali ini bilangan-bilangan dituliskan pada satu baris yang sama. Untuk membantu program Anda, bilangan pada baris pertama menunjukkan berapa banyak bilangan yang akan Anda jumlahkan. Jadi, program Anda harus membaca bilangan ini di awal, kemudian membaca bilangan-bilangan sebanyak nilai bilangan tadi.

Untuk perulangan (loop) dengan jumlah yang pasti/tertentu dalam Pascal, Anda dapat menggunakan struktur loop for berikut

```
for <iterator> := <harga-awal> to <harga-akhir> do
begin
    <perintah-perintah>
end;
```

<iterator> adalah variabel yang akan berubah harganya setiap loop dilakukan, dimulai dari harga <harga-awal>. Setiap perulangan, harga variabel tersebut bertambah satu. Perulangan demi perulangan dilakukan hingga variabel berharga <harga-akhir>. Tentu saja <harga-awal> harus lebih kecil atau sama dengan <harga-akhir>, karena kalau tidak maka program akan terus-menerus melakukan loop. Kita perlu dua variabel baru, untuk mencatat jumlah bilangan yang akan dibaca, misalnya jbil dan untuk <iterator> misalnya i.

Jadi pada bagian deklarasi ditambah dengan pernyataan

```
jbil, i: integer;
```

Dan bagian badan program diganti dengan

```
jml := 0;
read(jbil);

for i := 1 to jbil do
begin
   read(bil);
   jml := jml + bil;
end;
```

```
    writeln(jml);
```

Catatan: perintah readln diganti dengan perintah read agar pembacaan berikutnya tetap membaca pada baris yang sama.

Lakukan juga pengujian seperti pada latihan-latihan sebelumnya. Namai program tersebut dengan nama pjj0105 dan simpanlah dengan nama 'pjj0105.PAS'.

Contoh Masukan

```
5 1 2 3 4 5
```

Contoh Keluaran

```
15
```

6. If Then

Nama Program: pjj0106.PAS / C / CPP

Batas *Run-time*: 1 detik / test-case

Batas Memori: 16 MB

Nama Berkas Masukan: Standard input (keyboard)

Nama Berkas Keluaran: Standard output (layar)

Program Anda harus dapat membaca setiap bilangan bulat pada masukan dan memeriksa apakah bilangan tersebut positif. Jika positif maka bilangan itu dituliskan ke output, sementara jika bilangan negatif atau nol tidak melakukan apa-apa. Program mirip dengan pada latihan ketiga, tetapi keluaran dicetak di dalam loop sementara pencetakan di akhir ditiadakan. Karena bilangan positif saja yang dicetak maka diperlukan suatu struktur if-then berikut

```
if <kondisi> then
begin
    <perintah-perintah>
end;
```

<kondisi> yang diperiksa setelah notasi 'if' adalah 'apakah bil berharga positif'. <perintah-perintah> adalah yang akan dilakukan jika kondisi bernilai benar. Jadi Anda menambahkan

```
if bil > 0 then
begin
    writeln(bil);
end;
```

di dalam struktur loop while di atas (tentunya sebelum pembacaan berikutnya).

Namai program tsb dengan nama pjj0106 dan simpanlah dengan nama pjj0106.PAS.

Contoh Masukan 1

```
4
```

Contoh Keluaran 1

```
4
```

Contoh Masukan 2

```
0
```

Contoh Keluaran 2

Contoh Masukan 3

```
-1
```

Contoh Keluaran 3

7. If Then, Multi Condition

Nama Program:	pjj0107.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Program Anda harus membaca setiap bilangan bulat masukan dan memeriksa apakah bilangan tersebut positif dan genap. Jika positif dan genap maka bilangan itu dituliskan ke output, dan jika tidak maka tidak melakukan apa-apa. Jadi Anda dapat langsung mengubah program untuk latihan ke enam dengan menambahkan kondisi kedua (apakah bilangan tersebut genap) yang perlu diperiksa. Jika <kondisi> berisi dua kondisi yang keduanya harus benar maka Anda menuliskan kedua kondisi tersebut berurutan diperantarai oleh notasi 'and' sebagai berikut

```
if  (<kondisi pertama>) and (<kondisi kedua>) then
begin
    <perintah-perintah>
end;
```

Tentu, <kondisi pertama> adalah 'apakah bil positif' dan <kondisi kedua> adalah 'apakah bil bilangan genap', yang dapat diperiksa dengan memeriksa harga modulus (sisa pembagian) dari bil jika bil dibagi dengan 2. Dalam Pascal, dituliskan bil mod 2. Jika modulus itu berharga 0, bil habis dibagi (tanpa sisa), sementara jika bukan 0, maka bil tidak habis terbagi (ada sisa). Bilangan genap selalu habis dibagi 2 sehingga modulusnya harus 0. Jadi <kondisi kedua> memeriksa apakah 'bil mod 2 = 0' dan pemeriksaan kedua kondisi tersebut dalam struktur if-then menjadi

```
if (bil > 0) and (bil mod 2 = 0) then
begin
    writeln(bil);
end;
```

Apakah posisi kedua kondisi bisa ditukar? Tentu bisa karena 'and' sebagai operator logika tidak mementingkan urutan (kecuali pada kasus-kasus tertentu, yang akan dijelaskan kemudian).

Beri nama program pjj0107 dan simpanlah dengan nama pjj0107.PAS.

Contoh Masukan 1

```
12
```

Contoh Keluaran 1

```
12
```

Contoh Masukan 2

```
11
```

Contoh Keluaran 2

8. If Then Else

Nama Program:	pjj0108.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Program Anda harus dapat membaca setiap bilangan bulat masukan dan memeriksa apakah bilangan tersebut bilangan positif, negatif atau nol. Jika merupakan bilangan positif maka program akan menuliskan string 'positif', jika bilangan negatif maka program akan menuliskan 'negatif' dan jika bilangan nol maka proram akan menuliskan 'nol'. Untuk itu Anda perlu mempelajari struktur if-then-else yang sedikit berbeda dari struktur if-then sebelumnya. Perhatikan struktur if-then berikut

```
if <kondisi> then
begin
    <perintah-perintah>
end;
<perintah-perintah selanjutnya>
```

Jika pada struktur if-then saat kondisi yang diperiksa tidak benar maka komputer hanya melompati <perintah-perintah> untuk langsung menjalankan <perintah-perintah selanjutnya>. Sementara pada struktur if-then-else sebagai berikut.

```
if <kondisi> then
begin
    <perintah-perintah 1>
end
else
begin
    <perintah-perintah 2>
end;
<perintah-perintah selanjutnya>
```

Jika <kondisi> benar maka komputer akan menjalankan <perintah-perintah 1> lalu lompat ke <perintah-perintah selanjutnya> dan jika <kondisi> tidak benar maka komputer akan menjalankan <perintah-perintah 2> lalu ke <perintah-perintah selanjutnya>.

Jadi dalam latihan ini jika kondisi yang diperiksa adalah bil > 0 maka <perintah-perintah 1> adalah mencetak string 'positif'. Sementara itu karena kondisi tidak benar masih harus dibedakan antara negatif atau nol untuk mencetak 'negatif' atau 'nol', maka di dalam else-begin-end dibuat kembali pemeriksaan if-then-else yang mana kondisi yang diperiksa adalah apakah bil = 0 sebagai berikut.

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

Adanya satu struktur di dalam struktur yang sama dikenal dengan istilah 'nested structure' (struktur bersarang), dalam hal ini adalah nested if-then-else. Dalam Pascal seberapa dalam struktur nested tidak dibatasi, namun akan menyulitkan kita sendiri dalam membaca program itu. Untuk mempermudah pembacaan maka biasanya struktur yang berada lebih dalam dituliskan dengan indentasi seperti di atas. Namun compiler Pascal akan mengabaikan identasi tersebut, jadi indentasi sepenuhnya untuk kerapihan penulisan program demi kemudahan membacanya kembali.

Beri nama program tersebut pjj0108 dan simpanlah dengan nama 'pjj0108.PAS'.

Contoh Masukan 1

```
2
```

Contoh Keluaran 1

```
positif
```

Contoh Masukan 2

```
0
```

Contoh Keluaran 2

```
nol
```

Contoh Masukan 3

```
-2
```

Contoh Keluaran 3

```
negatif
```

9. Case

Nama Program:	pjj0109.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Program untuk latihan ini harus membaca setiap bilangan bulat masukan yang dipastikan berharga dari antara 1 sampai dengan 30000. Program akan mengenali apakah bilangan itu merupakan satuan (1 s.d. 9) atau puluhan (10 s.d. 99) atau ratusan (100 s.d. 999) atau ribuan (1000 s.d. 9999) atau puluh ribuan (10000 s.d. 30000). Jika satuan maka program akan memberikan keluaran string 'satuan', jika puluhan maka akan memberikan keluaran 'puluhan', dan seterusnya.

Anda dapat menggunakan struktur nested if-then-else seperti sebelumnya sbb.

```
if bil < 10 then
begin
    writeln('satuan');
end
else
begin
    if bil < 100 then
    begin
        writeln('puluhan');
    end
    else
    begin
        if bil < 1000 then
        begin
            writeln('ratusan');
        end
        else
        begin
            if bil < 10000 then
            begin
                writeln('ribuan');
            end
            else
            begin
                writeln('puluhribuan');
            end;
        end;
    end;
end;
```

Namun, karena penulisan nested structure dalam program yang terlalu banyak nest-nya tampak kurang rapi maka kita dapat menggunakan struktur alternatif yang disebut struktur case sbb.

```
case <variabel> of
    <harga atau harga-harga 1> : begin <perintah-perintah 1> end;
    <harga atau harga-harga 2> : begin <perintah-perintah 2> end;
    dan seterusnya...
end;
```

Dengan struktur ini maka <harga atau harga-harga> dapat berupa satu harga tunggal atau suatu jangkauan harga atau beberapa harga. Jangkauan harga, misalnya 'dari 10 s.d. 99' dituliskan '10..99' (kedua batas bilangan dengan dua titik di antaranya). Beberapa harga dituliskan dengan tanda koma misalnya '10, 100, 1000'. Jika ada beberapa <harga atau harga-harga> yang jangkauan bilangannya saling tumpang tindih, program akan menjalankan <perintah-perintah> yang berada pada urutan <harga atau harga-harga> yang lebih awal.

Jika jika menggunakan struktur case ini pemeriksaan menjadi lebih kompak dan sederhana sbb.

```
case bil of
    1..9: begin writeln('satuan'); end;
    10..99: begin writeln('puluhan'); end;
    100..999: begin writeln('ratusan'); end;
    1000..9999: begin writeln('ribuan'); end;
    10000..30000: begin writeln('puluhribuan'); end;
end;
```

Namai program tersebut dengan nama pjj0109 dan simpanlah dengan nama 'pjj0109.PAS'.

Contoh Masukan 1

```
4
```

Contoh Keluaran 1

```
satuan
```

Contoh Masukan 2

```
12345
```

Contoh Keluaran 2

```
puluhribuan
```

10. Procedure

Nama Program:	pjj0110.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Program ini harus membaca beberapa bilangan bulat, satu bilangan per baris, dan menuliskan keluaran 'satuan' atau 'puluhan' atau 'ratusan' atau 'ribuan' atau 'puluhribuan' untuk setiap bilangan yang dibaca.

Berbeda dengan latihan sebelumnya, pada latihan ini Anda membaca banyak bilangan bulat, tidak hanya satu.

Pada latihan ini akan diperkenalkan konsep procedure. Sebuah procedure (prosedur) adalah deretan perintah-perintah yang dapat dieksekusi dengan cara memanggil namanya. Bentuk umum deklarasi prosedur adalah:

```
procedure <nama>(<daftar parameter>);
<daftar deklarasi>
begin
  <perintah-perintah>
end;
```

Contohnya, kita dapat membuat sebuah procedure bernama 'TulisJawaban':

```
procedure TulisJawaban(x: integer);
begin
    case x of
        1..9: begin writeln('satuan'); end;
        10..99: begin writeln('puluhan'); end;
        100..999: begin writeln('ratusan'); end;
        1000..9999: begin writeln('ribuan'); end;
        10000..30000: begin writeln('puluhribuan'); end;
    end;
end;
```

Setelah itu procedure TulisJawaban ini dapat kita gunakan untuk memecahkan masalah awal:

```
while not eof(input) do
begin
   readln(bil);
   TulisJawaban(bil);
end;
```

Untuk setiap bil yang dibaca, nilai variabel bil akan "dimasukkan" ke dalam variabel x di dalam procedure TulisJawaban. Lalu prosedur tersebut akan mengeluarkan

'satuan', 'puluhan', 'ratusan', 'ribuan', atau 'puluhribuan' tergantung pada nilai x. Secara keseluruhan, program ini akan mengeluarkan jenis bilangan untuk setiap bilangan yang dibaca.

Perhatikan kesamaan penulisan readln(bil); dan TulisJawaban(bil);. Sesungguhnya, readln dan writeln juga adalah prosedur, tetapi prosedur-prosedur tersebut sudah dibuatkan untuk kita, sehingga kita tinggal menggunakannya saja.

Namai program tersebut dengan nama pjj0110 dan simpanlah dengan nama pjj0110.PAS'.

Contoh Masukan

```
1
12
123
```

Contoh Keluaran

```
satuan
puluhan
ratusan
```

11. Function

Nama Program:	pjj0111.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Program untuk latihan ini harus membaca sebuah bilangan N dan mengeluarkan nilai N! (N faktorial). Jika N berada dalam jangkauan 0 hingga 10, keluarkan nilai N!. Jika N negatif atau lebih besar dari 10, keluarkan 'ditolak'. Catatan: 0! = 1.

Kali ini akan diperkenalkan konsep function atau fungsi. Struktur fungsi mirip dengan prosedur, tetapi fungsi dapat mengembalikan suatu nilai untuk si pemanggil fungsi tersebut. Struktur fungsi secara umum adalah seperti berikut ini:

```
function <nama>(<daftar parameter>): <return type>;
<daftar deklarasi>
begin
  <perintah-perintah>
end;
```

Contohnya, untuk menyelesaikan latihan ini, kita dapat membuat fungsi seperti berikut ini:

```
function Faktorial(n: integer): longint;
var
    i: integer;
    bil: longint;
begin
    bil := 1;
    for i := 1 to n do
        bil := bil * i;
    Faktorial := bil;
end;
```

Dan kode program yang memanggilnya sebagai berikut:

```
var
    bil: integer;
begin
    readln(bil);
    if (n >= 0) and (n <= 10) then
        writeln(Faktorial(bil))
    else
        writeln('ditolak');
end.
```

Program ini akan membaca sebuah integer dari input dan menyimpannya di dalam variabel bil. Setelah itu, jika bil ada di dalam jangkauan 1 hingga 10, Faktorial dari bil akan dituliskan ke layar. Sudah dikatakan bahwa sebuah fungsi akan mengembalikan suatu nilai. Karena <return type> adalah integer, maka nilai yang dikembalikan oleh fungsi Faktorial bertipe integer.

Ketika fungsi Faktorial dipanggil oleh kode di atas, nilai variabel bil "dimasukkan" ke dalam variabel n di dalam fungsi Faktorial. Setelah itu, fungsi Faktorial akan menghitung nilai faktorial dari n dengan menggunakan sebuah variabel pembantu bernama bil lagi.

Namun, sekarang ada dua variabel bil, satu di dalam fungsi Faktorial dan satu di dalam program utama. Tetapi kita tidak perlu kuatir, karena meskipun memiliki nama yang sama, kedua variabel ini dianggap sebagai dua variabel yang berbeda. Variabel yang dideklarasikan di program utama disebut "global variable", dan yang dideklarasikan di dalam fungsi / prosedur disebut "local variable". Lebih dari itu, beberapa fungsi dan prosedur yang berbeda bisa memiliki local variable-nya sendiri-sendiri dan bisa memiliki nama yang sama, tetapi dua variabel akan dianggap berbeda jika dideklarasikan pada scope (tempat) yang berbeda.

Perhatikan juga perintah Faktorial := bil; yang berada di akhir fungsi Faktorial. Perintah ini menyatakan bahwa nilai variabel bil menjadi nilai yang akan dikembalikan oleh fungsi Faktorial setelah fungsi tersebut selesai.

Jadi, misalnya perintah writeln(Faktorial(4)); akan mengeluarkan hasil yang sama dengan perintah writeln(24);. Secara umum, fungsi biasanya digunakan untuk membuat program lebih terstruktur, atau untuk menghindari menuliskan kode berulang kali.

Kita juga dapat membuat sebuah fungsi lain yang bernama Valid sebagai berikut:

```
function Valid(n: integer): boolean;
begin
    Valid := (n >= 0) and (n <= 10);
end;
```

dan digunakan di program utama sebagai berikut:

```
var
    bil: integer;
begin
    readln(bil);
    if (Valid(bil)) then
        writeln(Faktorial(bil))
    else
        writeln('ditolak');
end.
```

Jadi fungsi Valid akan mengembalikan nilai true atau false, tergantung nilai n (yang diperoleh dari nilai bil). Program ini akan mengeluarkan keluaran yang sama dengan program sebelumnya.

Ada satu konsep lagi yang akan dikenalkan, yaitu konsep fungsi rekursif (ada juga prosedur rekursif). Sebuah fungsi rekursif adalah fungsi yang memanggil dirinya sendiri, dan struktur ini sering dipakai dan memiliki banyak kegunaan. Contoh fungsi rekursif adalah sebagai berikut:

```
function Faktorial(n: integer): longint;
begin
    if (n = 0)
        Faktorial := 1
    else
        Faktorial := n * Faktorial (n - 1);
end;
```

Tentu Faktorial(0) akan mengembalikan hasil 1, seperti yang diharapkan. Bagaimana jika Faktorial(1) dipanggil? Fungsi Faktorial akan melakukan perintah Faktorial := 1 * Faktorial(0);, sehingga pada akhirnya fungsi ini akan mengembalikan hasil 1. Jika Faktorial(n) dipanggil, fungsi ini akan melakukan perintah Faktorial := n * Faktorial(n - 1) dan setelah itu, fungsi ini akan dipanggil lagi dengan parameter n yang lebih kecil daripada sebelumnya. Fungsi Faktorial dipanggil terus-menerus oleh dirinya sendiri sampai nilai n menjadi 0 dan fungsi ini berhenti memanggil dirinya sendiri. Pada akhirnya, nilai yang dikeluarkan oleh Faktorial(n) adalah n * (n - 1) * (n - 2) * ... * 1, yang merupakan hasil yang benar.

Namai program tersebut dengan nama pjj0111 dan simpanlah dengan nama 'pjj0111.PAS'.

Contoh Masukan 1

```
4
```

Contoh Keluaran 1

```
24
```

Contoh Masukan 2

```
11
```

Contoh Keluaran 2

```
ditolak
```

12. Var Parameter

Nama Program:	pjj0112.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Program untuk latihan ini harus membaca dua buah bilangan bulat dalam satu baris, A dan B, lalu mengeluarkan kedua bilangan itu tetapi dengan posisi ditukar, menjadi B dan A, dalam satu baris.

Soal ini sebetulnya dapat dipecahkan dengan program yang sangat pendek seperti ini:

```
var
    a, b: integer;
begin
    readln(a, b);
    writeln(b, ' ', a);
end.
```

Namun, kali ini cobalah membuat sebuah prosedur Swap yang memiliki dua buah parameter integer, yang akan menukarkan nilai a dan b, seperti berikut ini:

```
var
    a, b: integer;

// menukar nilai a dengan b
procedure Swap(a, b: integer);
var
    temp: integer;
begin
    temp := a;
    a := b;
    b := temp;
end;


// program utama
begin
    readln(a, b);
    Swap(a, b);
    writeln(a, ' ', b);
end.
```

Coba ujilah program itu dengan sebuah file input yang berisi dua buah bilangan bulat. Apakah program ini mengeluarkan hasil yang benar? Ternyata program ini tidak berjalan dengan benar. Mengapa demikian? Dalam prosedur Swap di atas,

parameter a dan b sebetulnya adalah "local variable" yang dideklarasikan di dalam prosedur Swap, sehingga a dan b ini sama sekali bukan variabel a dan b yang dideklarasikan di awal program. Dengan demikian, menukarkan nilai a dan b yang berada di dalam prosedur Swap tidak akan berpengaruh apa-apa.

Oleh karena itu, kita harus mengubah sedikit prosedur Swap kita menjadi:

```
procedure Swap(var a: integer; var b: integer);
var
    temp: integer;
begin
    temp := a;
    a := b;
    b := temp;
end;
```

atau

```
procedure Swap(var a, b: integer);
var
    temp: integer;
begin
    temp := a;
    a := b;
    b := temp;
end;
```

"Modifier" var menunjukkan bahwa parameter a dan b bukanlah "local variable" di dalam prosedur tersebut, tetapi adalah referensi ke sebuah variabel yang nyata di luar prosedur tersebut. Karena Swap dipanggil dari program utama dengan parameter a dan b pada program utama, maka variabel-variabel global inilah yang direferensi oleh parameter prosedur Swap. Sehingga, jika nilai a dan b ditukarkan di dalam prosedur, sebetulnya nilai yang ditukarkan adalah nilai variabel global a dan variabel global b.

Untuk lebih jelasnya, meskipun kode prosedur Swap kita ganti menjadi:

```
procedure Swap(var c, d: integer);
var
    temp: integer;
begin
    temp := c;
    c := d;
    c := temp;
end;
```

program tetap berjalan dengan benar karena sekarang c mengacu pada variabel a global, dan d mengacu pada variabel b global. Sebuah prosedur atau fungsi dapat memiliki beberapa variabel "dengan modifier var" dan beberapa variabel "tanpa modifier var" sekaligus.

Namai program tersebut dengan nama pjj0112 dan simpanlah dengan nama 'pjj0112.PAS'.

Contoh Masukan

```
4 5
```

Contoh Keluaran

```
5 4
```

13. Break, Continue, Exit

Nama Program:	pjj0113.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Pada latihan ini, program Anda harus membaca sebuah bilangan bulat N ($1 \le N \le 100$), dan harus mengeluarkan bilangan-bilangan dari 1 sampai dengan N secara berurutan, satu per baris, dengan aturan sebagai berikut:

- Lompati bilangan kelipatan 10
- Jika program akan mengeluarkan bilangan 93, jangan keluarkan 93, tetapi keluarkan 'ERROR' dan jangan keluarkan apa-apa lagi.

Aturan tersebut kesannya dibuat-buat, karena masalah ini hanya sebagai contoh untuk mengilustrasikan kegunaan break, continue, dan exit.

Untuk menyelesaikan masalah di atas, Anda dapat membuat program seperti ini:

```
var
    n: integer;
    i: integer;
    error: boolean;
begin
    readln(n);

    error := false;

    for i := 1 to n do
    begin
        if (i = 93) then
            error := true;
        if (not error) and (i mod 10 <> 0) then
            writeln(i);
    end;

    if (error) then
        writeln('ERROR');
end.
```

Pada program di atas, variabel boolean error hanya berfungsi sebagai variabel pembantu. Pada mulanya, error diinisialisasi dengan false. Setelah itu, di dalam loop for, jika i bukan kelipatan 10, i dituliskan ke layar. Namun jika i bernilai 93, maka variabel error diberi nilai true, sehingga sejak saat itu tidak ada lagi yang ditulis ke layar kecuali 'ERROR' di akhir program.

Ada cara lain menuliskan program tersebut, yaitu seperti berikut:

```
var
    n: integer;
    i: integer;
begin
    readln(n);

    for i := 1 to n do
    begin
        if (i = 93) then
        begin
            writeln('ERROR');
            break;
        end;
        if (i mod 10 = 0) then
            continue;

        writeln(i);
    end;
end.
```

Pada program ini, break berfungsi untuk keluar secara paksa dari loop for. Jika i = 93, 'ERROR' dituliskan ke layar dan loop for dihentikan secara paksa, dan program berlanjut ke perintah berikutnya setelah loop for. Karena tidak ada perintah lagi, program selesai. Perintah break sebetulnya juga dapat dipakai untuk menghentikan loop while secara paksa.

Perintah continue berfungsi untuk menghentikan aliran program dan kembali ke baris for i := 1 to n do dengan nilai i selanjutnya. Jadi jika i adalah kelipatan 10, perintah writeln(i) tidak dijalankan dan loop for dilanjutkan dengan nilai i berikutnya.

Perintah break di atas juga dapat diganti dengan perintah exit. Perintah ini akan menghentikan sebuah prosedur, fungsi, atau program secara paksa. Karena pada kasus di atas aliran program berada di program utama, perintah exit akan menghentikan program seketika.

Namai program tersebut dengan nama pjj0113 dan simpanlah dengan nama 'pjj0113.PAS'.

Contoh Masukan 1

```
12
```

Contoh Keluaran 1

```
1
2
3
4
5
6
7
8
```

```
9                                       12
11
```

Contoh Masukan 2

```
94
```

Contoh Keluaran 2

1	24	47	71
2	25	48	72
3	26	49	73
4	27	51	74
5	28	52	75
6	29	53	76
7	31	54	77
8	32	55	78
9	33	56	79
11	34	57	81
12	35	58	82
13	36	59	83
14	37	61	84
15	38	62	85
16	39	63	86
17	41	64	87
18	42	65	88
19	43	66	89
21	44	67	91
22	45	68	92
23	46	69	ERROR
→	→	→	

14. Operasi String

Nama Program:	pjj0114.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Pada latihan ini, program Anda harus membaca sebuah empat buah string yang kita beri nama S1, S2, S3, dan S4. Misalnya program Anda mendapat input seperti ini:

```
abcdehalofghi
bcd
halo
semua
```

Dijamin bahwa string S1 mengandung sebuah string S2 di dalamnya. Buang string S2 yang ditemukan di string S1 (dijamin ada, dan hanya satu). Kemudian sisipkan string S4 pada posisi setelah string S3 yang ditemukan di string S1 (dijamin ada, dan hanya satu). Jadi pada contoh di atas, abcdehalofghi diubah menjadi aehalofghi, lalu menjadi aehalosemuafghi. Keluarkan string hasil akhir, yang pada contoh ini adalah aehalosemuafghi.

Untuk menyelesaikan masalah tersebut, kita perlu mengenal berbagai fungsi-fungsi dan prosedur-prosedur penanganan string yang disediakan oleh library Pascal yang bisa kita gunakan.

(Dapat dilihat di
http://community.freepascal.org:10000/docs-html/rtl/system/stringfunctions.html)

Fungsi-fungsi dan prosedur-prosedur yang akan kita gunakan adalah:

```
function length(s: string): integer;
function pos(substr: string; s: string): integer;
procedure delete(var s: string; index: integer; count: integer);
procedure insert(var source: string; s: string; index: integer);
```

Fungsi length akan mengembalikan panjang dari s.

Jika string s mengandung string substr, fungsi pos akan mengembalikan index pertama dari kemunculan pertama substr di dalam s (karakter pertama diberi index 1). Jika tidak ada, fungsi ini akan mengembalikan 0.

Prosedur delete akan membuang sebanyak count karakter pada string s, dimulai dengan index ke-index. Misalnya, jika s pada mulanya adalah 'Halo', delete(s, 1, 2) akan mengubah isi s menjadi 'lo'.

Prosedur insert akan memasukkan string s ke dalam string source, dimulai pada posisi index ke-index.

Untuk menyelesaikan masalah awal, kita dapat membuat program sebagai berikut:

```
var
    S1, S2, S3, S4: string;
begin
    readln(S1);
    readln(S2);
    readln(S3);
    readln(S4);

    delete(S1, pos(S2, S1), length(S2));
    insert(S4, S1, pos(S3, S1) + length(S3));

    writeln(S1);
end.
```

Pada program di atas, S2 di dalam S1 akan dibuang dari S1, dan selanjutnya S4 akan dimasukkan ke dalam S1 tepat pada posisi pos(S3, S1) + length(S3), yaitu posisi karakter pertama yang tidak termasuk S3, setelah kemunculan seluruh karakter S3 di dalam S1.

Namai program tersebut dengan nama pjj0114 dan simpanlah dengan nama 'pjj0114.PAS'.

Contoh Masukan

```
abcdehalofghi
bcd
halo
semua
```

Contoh Keluaran

```
aehalosemuafghi
```

Masih banyak fungsi-fungsi dan prosedur-prosedur lain yang mungkin berguna, yang bisa dipelajari di

http://community.freepascal.org:10000/docs-html/rtl/system/index.html

Berikut ini adalah ringkasan dari beberapa fungsi yang mungkin berguna:

```
procedure str(x: integer; var s: string);
```

Prosedur ini akan mengubah nilai integer x menjadi string, lalu dimasukkan ke dalam variabel s. Misalnya, jika x bernilai 10, s akan menjadi bernilai '10'.

```
function lowerCase(s: string): string;
```

Fungsi ini akan menghasilkan sebuah string seperti s tetapi semua karakternya diubah ke huruf kecil.

```
function upCase(s: string): string;
```

Fungsi ini akan menghasilkan sebuah string seperti s tetapi semua karakternya diubah ke huruf besar.

```
function chr(b: byte): char;
```

Fungsi ini akan mengembalikan sebuah karakter yang memiliki kode ASCII b. Misalnya chr(65) akan mengembalikan "A".

```
function ord(c: char): longint;
```

Fungsi ini akan mengembalikan nilai bilangan bulat dari sebuah tipe ordinal. Biasanya fungsi ini digunakan untuk menentukan kode ASCII dari karakter c. Misalnya, ord("A") akan mengembalikan 65.

Kombinasi chr dan ord dapat digunakan untuk mengubah sebuah karakter dari huruf kecil menjadi huruf besar, atau sebaliknya. Misalnya, chr(ord("x") + (ord("A") - ord("a"))) akan menghasilkan "X".

```
function abs(l: longint): longint;
```

Fungsi ini akan mengembalikan nilai absolut dari l. Misalnya, abs(-3) akan mengembalikan 3, dan abs(3) juga mengembalikan 3.

```
procedure dec(var x: integer);
procedure dec(var x: integer; decrement: integer);
```

Prosedur ini akan mengurangi nilai x dengan 1, atau dengan nilai decrement jika diberikan.

```
procedure inc(var x: integer);
procedure inc(var x: integer; increment: integer);
```

Prosedur ini akan menambah nilai x dengan 1, atau dengan nilai increment jika diberikan.

```
function sqr(x: longint): longint;
function sqr(x: real): real;
```

Fungsi ini akan mengembalikan kuadrat dari x.

```
function sqrt(x: real): real;
```

Fungsi ini akan mengembalikan akar kuadrat dari x.

```
function trunc(x: real): integer;
```

Fungsi ini akan mengembalikan bagian bilangan bulat dari sebuah bilangan real x. Misalnya, trunc(3.456) akan menghasilkan 3.

```
function round(x: real): integer;
```

Fungsi ini akan menghasilkan pembulatan dari sebuah bilangan real x. Pada Pascal, aturan pembulatan untuk 0.5 adalah ke arah bilangan genap. Jadi, round(1.5) akan menghasilkan 2, tetapi round(2.5) juga akan menghasilkan 2.

```
function pi: real;
```

Fungsi ini mengembalikan nilai pi (3.14159...).

15. Manhattan Distance

Nama Program:	pjj0115.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Manhattan distance adalah jarak dari suatu titik menuju titik lainnya di bidang kartesian dengan menyusuri bagian vertikal dan horizontal, tanpa pernah kembali. Secara sederhana sama dengan jumlah dari selisih absis dan selisih ordinat (distance = |x1-x2| + |y1-y2|). Pada soal ini, Anda akan membaca masukan yang berisi empat buah bilangan bulat yang merupakan koordinat dari dua buah titik, x1 y1 x2 y2 (semuanya berada dalam jangkauan -1000000000..1000000000) secara berturutan dalam 1 baris. Hitung berapa manhattan distance untuk kedua buah titik tersebut.

Format Masukan

Sebuah baris berisi empat buah bilangan bulat x1 y1 x2 y2.

Format Keluaran

Sebuah bilangan bulat yang merupakan manhattan distance untuk kedua buah titik yang diberikan.

Contoh Masukan

```
-1 -1 1 1
```

Contoh Keluaran

```
4
```

16. Floor and Ceiling

Nama Program:	pjj0116.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Nilai floor dari sebuah bilangan adalah bilangan bulat terbesar yang masih lebih kecil atau sama dengan bilangan tersebut, sebaliknya nilai ceiling dari sebuah bilangan adalah bilangan bulat terkecil yang masih lebih besar atau sama dengan bilangan tersebut.

Format Masukan

Sebuah bilangan real (dalam jangkauan -1000000 .. 1000000).

Format Keluaran

Dua buah bilangan bulat berturutan dalam satu baris dipisahkan oleh spasi, yakni nilai floornya dan nilai ceilingnya.

Contoh Masukan

```
-256.652
```

Contoh Keluaran

```
-257 -256
```

17. Dua Pangkat

Nama Program:	pjj0117.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Bilangan "dua pangkat" dalam soal ini adalah bilangan bulat yang dapat dituliskan dalam bentuk 2^K dimana K adalah sebuah bilangan bulat.

Format Masukan

Sebuah bilangan bulat dalam jangkauan 1 sampai 2^20.

Format Keluaran

"TRUE" jika bilangan yang diberikan adalah bilangan "dua pangkat", dan "FALSE" jika sebaliknya.

Contoh Masukan 1

```
8
```

Contoh Keluaran 1

```
TRUE
```

Contoh Masukan 2

```
6
```

Contoh Keluaran 2

```
FALSE
```

18. POLA 1

Nama Program:	pjj0118.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Perhatikan contoh masukan dan keluaran yang diberikan, temukan polanya, lalu buatlah programnya.

Format Masukan

Sebuah bilangan bulat N (0<N<10)

Format Keluaran

Pola berukuran N, seperti pada contoh keluaran.

Contoh Masukan

```
5
```

Contoh Keluaran

```
    *
   **
  ***
 ****
*****
```

19. POLA 2

Nama Program:	pjj0119.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Perhatikan contoh masukan dan keluaran yang diberikan, temukan polanya, lalu buatlah programnya.

Format Masukan

Sebuah bilangan bulat N (0<N<10)

Format Keluaran

Pola berukuran N, seperti pada contoh keluaran.

Contoh Masukan 1

```
5
```

Contoh Keluaran 1

```
0
12
345
6789
01234
```

Contoh Masukan 2

```
7
```

Contoh Keluaran 2

```
0
12
345
6789
01234
567890
1234567
```

20. POLA 3

Nama Program:	pjj0120.PAS / C / CPP
Batas *Run-time*:	1 detik / test-case
Batas Memori:	16 MB
Nama Berkas Masukan:	Standard input (keyboard)
Nama Berkas Keluaran:	Standard output (layar)

Perhatikan contoh masukan dan keluaran yang diberikan, temukan polanya, lalu buatlah programnya.

Format Masukan

Dua buah bilangan bulat N dan K ($0 < N < 100$, $1 < K < 10$).

Format Keluaran

Pola berukuran N, seperti pada contoh keluaran.

Contoh Masukan 1

```
11 3
```

Contoh Keluaran 1

```
1 2 * 4 5 * 7 8 * 10 11
```

Contoh Masukan 2

```
11 2
```

Contoh Keluaran 2

```
1 * 3 * 5 * 7 * 9 * 11
```

Contoh Masukan 3

```
11 5
```

Contoh Keluaran 3

```
1 2 3 4 * 6 7 8 9 * 11
```