PERANCANGAN FUNGSI TERBILANG GENERIK MENGGUNAKAN TEKNIK REKURSI

Choerun Asnawi
Chanief Budi Setiawan

STMIK Jenderal Achmad Yani Yogyakarta

c.asnawi@gmail.com; chanief.b.s@gmail.com

ABSTRAK

Angka telah menjadi bagian yang penting dalam kegiatan manusia. Dalam keadaan tertentu, seringkali diperlukan penyebutan nilai dari suatu angka, misal dalam transaksi keuangan. Penyebutan nilai angka mengikuti aturan tertentu sesuai bahasa yang digunakan, dengan beberapa pengecualian untuk sejumlah angka tertentu. Untuk keperluan penyebutan nilai angka ini, sebuah fungsi dalam bahasa pemrograman dapat didefinisikan. Algoritma rekursif digunakan dalam fungsi ini, untuk mendapatkan hasil penyebutan angka yang teratur sesuai posisi digit masing-masing.

Kata kunci: angka, terbilang, fungsi, rekursi, algoritma

PENDAHULUAN

Banyak kegiatan manusia yang di dalamnya memerlukan penyebutan nilai dari suatu angka, terutama kegiatan yang berhubungan dengan bisnis dan keuangan. Contoh yang paling umum adalah penulisan ejaan nama angka pada kuitansi, faktur atau struk pembayaran, seperti terlihat pada gambar 1. Contoh lainnya adalah untuk menampilkan banyaknya pengunjung suatu situs *web* bukan dalam bentuk angka, melainkan dalam bentuk kata-kata.


STRUK PEMBAYARAN TAGIHAN PLN*

No. Ref: 96788 — Tanggal Bayar : 09/07/2009

Id. Pelanggan : 538314145920 — Nama : Hasan — Tarif/Daya : R1/900

Bulan Pembayaran : Jul-2009 — Besar Tagihan : 73.591 — Biaya Adm. : 2.000

Total : 75.591 TUJUH PULUH LIMA RIBU LIMA RATUS SEMBILAN PULUH SATU Rupiah

Terima kasih, Bukti pembayaran ini mohon di simpan

Hormat kami,

RAKA PULSA

Gambar 1: Contoh penulisan ejaan nama angka pada struk pembayaran

Keberadaan komputer dan *software* memudahkan pengguna untuk mengotomasikan proses tersebut ke dalam sebuah kode program. Tugas utama program tersebut adalah mengkonversi suatu bilangan menjadi penyebutan namanya. Bentuk implementasi yang paling umum adalah subrutin atau fungsi untuk memetakan bilangan ke nama bilangan tersebut, atau sering disebut sebagai fungsi **terbilang**.

Tulisan ini membahas langkah-langkah untuk merancang sebuah fungsi yang menghasilkan bentuk terbilang dari suatu bilangan bulat (*integer*) dalam

suatu bahasa tertentu sesuai dengan *setting* yang diberikan. Pembahasan dimulai dari penjelasan algoritma secara umum, diikuti dengan berbagai perkecualian aturan penamaan untuk bilangan-bilangan tertentu di setiap bahasa yang dibahas sekaligus cara mengatasinya, dan dilanjutkan dengan penulisan rancangan fungsi serta penjelasannya.

DASAR TEORI

Subrutin dan Fungsi

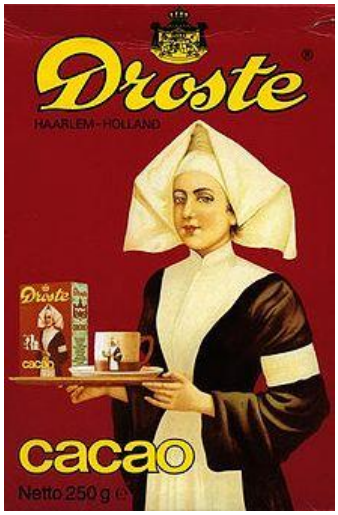
Dalam ilmu komputer, sebuah subrutin merupakan sebagian kode program yang melakukan suatu tugas spesifik dan relatif tidak tergantung pada kode program selainnya. Subrutin juga dikenal dengan nama prosedur, fungsi, rutin, subprogram, atau metode, sesuai bahasa pemrograman yang digunakan dan konteksnya. Sebuah subrutin didefinisikan dalam suatu program untuk dapat digunakan dalam program tersebut, baik oleh program itu sendiri atau subrutin lain yang terdapat dalam program tersebut. Proses penggunaan sebuah subrutin di dalam suatu program dinamakan sebagai pemanggilan subrutin atau terkenal dengan istilah *procedure call* atau *function call*. Program atau subrutin yang menggunakan suatu subrutin dinamakan sebagai **pemanggil** atau *caller*, sedangkan subrutin yang digunakan dinamakan **yang dipanggil** atau *calee*.

Secara garis besar, subrutin dapat dibagi menjadi 2 (dua) bentuk. Bentuk pertama adalah subrutin yang melakukan pekerjaan tertentu tanpa melaporkan hasilnya pada pemanggilnya, yang dalam bahasa pemrograman Pascal dikenal dengan sebutan prosedur (*procedure*). Bentuk kedua adalah subrutin yang saat selesai melakukan pekerjaannya akan melaporkan hasilnya dalam bentuk suatu nilai tertentu ke pemanggilnya. Bentuk kedua ini dalam Pascal dikenal dengan sebutan fungsi (*function*). Proses pelaporan hasil ini dalam dunia pemrograman dikenal dengan istilah pengembalian nilai (*returning value*) dan nilai yang dihasilkan disebut sebagai nilai kembalian (*return value*).

Rekursi dan Fungsi Rekursif

Rekursi merupakan suatu istilah yang digunakan untuk mendefinisikan penggunaan suatu objek yang sama di dalam objek itu sendiri secara berulang-ulang. Sebagai contoh, jika permukaan dua buah cermin dihadapkan satu sama lain maka gambar yang terbentuk pada cermin merupakan salah satu bentuk rekursi tak berhingga (*infinite*). Gambar 2 memperlihatkan suatu bentuk visual dari rekursi yang dikenal sebagai efek Droste. Wanita pada gambar tersebut

terlihat memegang suatu objek berisi gambarnya sendiri yang memegang objek yang sama namun lebih kecil, yang masih berisi gambar wanita tersebut sedang memegang objek yang sama namun lebih kecil lagi, dan seterusnya.



Gambar 2: Efek Droste, suatu contoh rekursi

Dalam ilmu komputer, rekursi adalah metode pemecahan masalah dengan solusi permasalahan tergantung pada solusi pada masalah-masalah yang sama namun lebih kecil. Hampir semua bahasa pemrograman komputer tingkat tinggi saat ini mendukung konsep rekursi dengan memperbolehkan suatu subrutin untuk memanggil dirinya sendiri dalam kode program. Fungsi yang memanggil dirinya sendiri inilah yang dinamakan sebagai fungsi rekursif. Contoh fungsi rekursif yang terkenal adalah fungsi untuk menampilkan elemen tertentu dalam deret *fibonacci*.

Suatu objek dikatakan memiliki sifat rekursif jika dapat didefinisikan menggunakan dua properti berikut:

1. Satu atau lebih **bentuk sederhana** yang tidak mengandung rekursi.
2. Sekelompok **aturan yang menggunakan rekursi** untuk menyederhana-kan semua bentuk lainnya ke bentuk sederhana pada nomor 1 di atas.

Bilangan dan Bentuk Terbilang

Istilah terbilang yang digunakan dalam tulisan ini mengacu pada penyebutan nama sebuah nilai bilangan. Sebagai contoh, bilangan 12345 memiliki bentuk terbilang dalam Bahasa Indonesia sebagai "dua belas ribu tiga ratus empat puluh lima" dan dalam Bahasa Inggris sebagai "*twelve thousand three hundred and forty-five*".

Berdasarkan aturan bentuk terbilangnya, bilangan dapat dibagi menjadi 3 (tiga) kelompok, yaitu:

1. Bilangan dengan nama teratur atau *reguler*

   Kelompok ini memiliki nama yang mengikuti aturan yang pasti dan sangat umum digunakan di hampir semua bahasa. Perbedaan yang ada biasanya ditemukan pada penamaan digit, penyebutan akhiran dan pengelompokan digit. Sebagian besar bilangan yang ada masuk ke dalam kelompok ini.
2. Bilangan dengan nama semi teratur atau *semi reguler*

   Kelompok ini namanya tidak disusun memakai aturan yang berlaku umum pada bilangan *reguler*, tetapi menggunakan aturan khusus. Biasanya terdiri atas kelompok-kelompok yang memiliki aturan-aturan yang berbeda satu sama lain. Misalkan bilangan-bilangan di antara 10 dan 20 (belasan) dalam Bahasa Indonesia kebanyakan adalah bilangan *semi reguler*.
3. Bilangan dengan nama tak teratur atau *irreguler*

   Kelompok ini terdiri atas bilangan-bilangan yang namanya benar-benar tidak mengikuti aturan tertentu dan jumlahnya hanya sedikit. Sebagai contoh, bilangan 10, 11 dan 12 dalam Bahasa Inggris merupakan bilangan *irreguler*.

Perlu diperhatikan bahwa pengelompokan ini sangat relatif terhadap bahasa yang digunakan dan bisa jadi berbeda antara satu bahasa dengan bahasa yang lain. Sebagai contoh, bilangan 12 yang dalam Bahasa Inggris (*twelve*) merupakan bilangan *irreguler*, dalam Bahasa Indonesia (dua belas) merupakan bilangan *semi reguler*. Contoh lainnya, bilangan 25 yang dalam Bahasa Indonesia (dua puluh lima) dan Bahasa Inggris (*twenty-five*) merupakan bilangan reguler, dalam Bahasa Jawa (*selawe* atau *selangkung*) adalah bilangan *irreguler*.

PEMBAHASAN

Dasar Pemikiran

Nama suatu bilangan tentu saja berbeda antara satu bahasa dengan bahasa yang lainnya. Bahkan sistem bilangannya pun dapat jauh berbeda. Meskipun demikian, pada bahasa-bahasa yang menggunakan sistem yang sama terdapat kemiripan tata cara pemberian nama untuk sebagian besar bilangan.

Kemiripan tersebut dapat dilihat pada tabel 1 yang berisi penamaan untuk bilangan 4536 dalam berbagai bahasa.

Tabel 1: Kemiripan pemberian nama bilangan dalam berbagai bahasa

Bahasa	Nama Lengkap	Ribuan	Ratusan	Puluhan	Satu-an
Inggris	four thousand five hundred and thirty-six	four thousand	five hundred	thirty	six
Perancis	quatre mille cinq cent trente-six	quatre mille	cinq cent	trente	six
Indonesia	empat ribu lima ratus tiga puluh enam	empat ribu	lima ratus	tiga puluh	enam
Jawa Ngoko	patangewu limangatus telungpuluh enem	patangewu	limangatus	telungpuluh	enem
Sino Korea	sa-chon o-bek sam-sip yuk	sa-chon	o-bek	sam-sip	yuk

Bentuk terbilang dapat dicari menggunakan cara rekursif maupun non-rekursif. Cara non-rekursif dilakukan dengan memecah bilangan ke dalam digit-digitnya dan setiap digit dicatat nilai posisinya (satuan, puluhan, ratusan, dst). Digit dipisahkan mulai dari yang paling tinggi nilai posisinya, lalu jika digit itu bukan 0 (nol) maka diambil nama untuk digit tersebut, diikuti dengan akhiran sesuai dengan posisinya. Setelah itu disambung dengan hasil proses yang sama untuk digit pada posisi di bawahnya hingga digit terakhir yang nilai posisinya paling rendah.

Sedangkan dengan cara rekursif, secara umum bentuk terbilang dari suatu bilangan dapat didefinisikan sebagai berikut:

1. Untuk bilangan satu digit, langsung disebut namanya (bentuk dasar).
2. Untuk bilangan yang lebih dari satu digit, bilangan dapat dipecah menjadi 2 (dua) bilangan yaitu kepala dan sisanya, lalu bentuk terbilang dari bilangan dapat dibentuk dari bentuk terbilang dari bilangan kepala disambung dengan suatu akhiran yang sesuai dengan nilai posisi kepala dan jika sisanya tidak nol maka akan diikuti dengan bentuk terbilang dari bilangan sisanya (aturan rekursif).

Berdasarkan definisi di atas, maka bentuk terbilang dari bilangan 5340 dalam Bahasa Indonesia misalnya, dapat dicari dengan langkah-langkah berikut:

1. Bilangan 5340 memiliki kepala 5 yang berupa ribuan dan sisa 340. Untuk kepala ribuan akhirannya adalah “ribu”, sehingga didapatkan:
   $terbilang(5340) = terbilang(5) + \text{'ribu'} + terbilang(340)$..... (1)
2. Bilangan 5 sudah memiliki bentuk dasar, sehingga cukup disebutkan namanya saja yaitu “lima”. Setelah digabungkan maka didapatkan:
   $terbilang(5340) = \text{'lima ribu'} + terbilang(340)$..... (2)
3. Bilangan 340 memiliki kepala 3 yang berupa ratusan dan sisa 40. Untuk kepala ratusan akhirannya adalah “ratus”, sehingga didapatkan:
   $terbilang(5340) = \text{'lima ribu'} + terbilang(3) + \text{'ratus'} + terbilang(40)$ (3)
4. Bilangan 3 sudah memiliki bentuk dasar, sehingga cukup disebutkan namanya saja yaitu “tiga”. Setelah digabungkan akan menjadi:
   $terbilang(5340) = \text{'lima ribu tiga ratus'} + terbilang(40)$ ..... (4)
5. Bilangan 40 memiliki kepala 4 yang berupa puluhan dan sisa 0. Untuk kepala puluhan akhirannya adalah “puluh”. Karena sisanya 0 maka sesuai dengan definisi di atas tidak perlu disebutkan bentuk terbilang sisanya, sehingga pada langkah ini didapat:
   $terbilang(5340) = \text{'lima ribu tiga ratus'} + terbilang(4) + \text{'puluh'}$ ..... (5)
6. Bilangan 4 sudah dalam bentuk dasar dan memiliki nama “empat”, sehingga didapatkan hasil akhirnya adalah:
   $terbilang(5340) = \text{'lima ribu tiga ratus empat puluh'}$..... (6)

Definisi di atas berlaku hanya untuk bilangan-bilangan yang memiliki nama teratur atau *reguler* dan tidak tepat diberlakukan untuk bilangan *non-reguler*. Misalkan bilangan 11 dalam Bahasa Indonesia (sebelas) jika mengikuti definisi di atas akan memiliki bentuk terbilang “satu puluh satu”. Oleh karena itu untuk bilangan *non-reguler* akan ada pengecualian. Untuk bilangan *semi reguler* akan diberlakukan pengecualian definisi sedangkan untuk bilangan *irreguler* akan diberlakukan pengecualian penamaan.

Selain itu secara umum penamaan bilangan memiliki aturan pengelompokan digit, yaitu membagi digit-digit yang menyusun bilangan ke dalam kelompok-kelompok yang terdiri dari jumlah digit tertentu. Penyusunan kelompok dimulai dari digit yang nilai posisinya paling rendah. Adanya pengelompokan digit tersebut membuat cara rekursif lebih mudah dan lebih sederhana untuk diimplementasikan daripada cara non-rekursif. Hal tersebut

karena tiap kelompok digit memiliki metode yang sama dalam pencarian bentuk terbilangnya, hanya berbeda nilai posisinya.

Sebagai contoh dalam Bahasa Indonesia berlaku pengelompokan yang terdiri atas 3 (tiga) digit, sehingga bilangan 23456542789 akan dibagi dalam kelompok-kelompok digit 789, 542, 456, dan 23. Kelompok digit 789 memiliki nilai posisi terendah, yaitu satuan. Kelompok digit 542 memiliki nilai posisi ribuan, kelompok digit 456 memiliki nilai posisi jutaan, dan kelompok digit 23 memiliki nilai posisi milyaran. Pada contoh tersebut dapat dilihat bahwa kelompok digit terakhir terdiri atas digit yang tersisa dan bisa jadi memiliki jumlah digit yang kurang dari lainnya.

Algoritma

Dengan menyertakan bilangan negatif, maka definisi rekursif dari bentuk terbilang dari bilangan *reguler* yang sudah tertulis di atas dapat dimodifikasi dan dilengkapi lagi menjadi sebagai berikut:

1. Bentuk dasar: untuk bilangan satu digit (dari nol hingga sembilan), tinggal disebutkan saja namanya.
2. Aturan rekursif:
    a. Untuk bilangan negatif (nilainya kurang dari nol), bentuk terbilangnya diawali dengan kata yang menunjukkan kenegatifan (kata negatif, *negative*, *minus*, min, atau yang lainnya sesuai dengan bahasanya) dan diikuti dengan bentuk terbilang dari nilai positif dari bilangan tersebut.
    b. Untuk bilangan yang lebih dari satu digit (sepuluh ke atas), bilangan dapat dipecah menjadi 2 (dua) bilangan yaitu kepala dan sisanya, lalu bentuk terbilangnya dapat dibentuk dari bentuk terbilang dari bilangan kepala disambung dengan suatu akhiran yang sesuai dengan nilai posisi kepala dan jika sisanya tidak nol maka akan diikuti dengan bentuk terbilang dari bilangan sisanya.

Berdasarkan definisi di atas, maka untuk bilangan negatif dan bilangan satu digit tidak memerlukan kalkulasi maupun algoritma yang kompleks. Untuk bilangan satu digit bisa dengan menggunakan konstruksi percabangan bertingkat atau pemilihan kondisi sederhana, atau bisa juga dengan membangun sebuah tabel yang berisi nama-nama angka lalu tinggal mengambil nama yang sesuai.

Sementara itu untuk bilangan positif multidigit menjadi tidak sederhana karena adanya berbagai kalkulasi untuk menentukan komponen kepala dan

sisanya serta komponen akhiran yang sesuai untuk nilai posisi kepala. Algoritma yang digunakan untuk memperoleh komponen kepala dan sisanya adalah:

1. Menentukan banyaknya digit dalam pengelompokan digit
2. Menghitung banyaknya digit yang menyusun bilangan
3. Menghitung banyaknya kelompok digit
4. Jika kelompok digit hanya satu maka komponen kepalanya adalah satu digit dengan nilai posisi yang tertinggi
5. Selainnya itu (jika kelompok digit lebih dari satu) maka komponen kepalanya adalah kelompok digit dengan nilai posisi tertinggi
6. Komponen sisa diambil dari digit-digit yang tersisa

Algoritma di atas akan lebih jelas dengan melihat contoh-contoh pada ke tabel 2.

Tabel 2: Contoh penentuan komponen kepala dan sisa

Bilangan	Digit Pengelom-pokan	Banyaknya Digit	Banyaknya Kelompok Digit	Kepala	Sisa
25	3	2	1	2	5
123	3	3	1	1	23
1234	3	4	2	1	234
54321	3	5	2	54	321
654321	3	6	2	654	321
9876543	3	7	3	9	876543
12345678	3	8	3	12	345678

Akhiran yang menyatakan nilai posisi kepala yang sesuai dapat dicari dengan melihat banyaknya digit atau kelompok digit pada komponen sisa. Sebagai contoh, tabel akhiran dapat dilihat pada tabel 3.

Tabel 3: Akhiran nilai posisi dalam beberapa bahasa

Banyaknya Digit Sisa	Akhiran			
	Indonesia	Jawa Krama	Inggris	Perancis
1	puluh	dasa	ty	ante
2	ratus	atus	hundred	cents
3	ribu	ewu	thousand	mille
6	juta	yuta	million	millions

Banyaknya digit dalam suatu bilangan positif dapat dicari menggunakan beberapa cara yang berbeda. Salah satunya adalah dengan melihat posisi

bilangan tersebut pada kisaran-kisaran nilai berdigit sama. Misalnya jika nilainya dari 10 hingga di bawah 100 berarti terdiri atas dua digit, sedangkan nilai di atasnya hingga di bawah 1000 memiliki tiga digit, dan seterusnya. Cara lainnya adalah dengan menghitung banyaknya pembagian dengan 10 yang harus dilakukan terhadap bilangan agar nilainya menjadi kurang dari 10. Cara lain yang lebih singkat melibatkan adanya konversi tipe data, yaitu dengan melakukan konversi ke tipe data *string* lalu menghitung panjangnya.

Untuk membagi bilangan ke dalam komponen kepala dan komponen sisanya juga dapat dilakukan dengan beberapa cara. Salah satunya adalah dengan mencari suatu bilangan pembagi yang didapat dari 10 pangkat banyaknya digit sisa. Komponen kepala didapat dari bilangan dibagi dengan pembagi tersebut lalu dipotong bagian pecahannya, sedangkan komponen sisanya didapat dari sisa hasil baginya. Cara lainnya yang lebih singkat adalah dengan melakukan konversi tipe data ke *string* lalu menggunakan operasi atau fungsi untuk mengambil *substring*-nya.

Selengkapnya algoritma untuk menentukan bentuk terbilang untuk bilangan *reguler* multidigit dapat disajikan dalam *pseudocode* 1.

Pseudocode 1: Algoritma untuk menangani bilangan *reguler* multidigit

```
input bilangan, digit_per_kelompok;

digit := jumlah_digit(bilangan);
if digit > digit_per_kelompok then begin
   digit_kepala := digit mod digit_per_kelompok;
   if digit_kepala = 0 then digit_kepala := digit_per_kelompok;
end
else digit_kepala := 1;
digit_sisa := digit - digit_kepala;
pembagi := 10 ^ digit_sisa;
kepala := bilangan div pembagi;
sisa := bilangan mod pembagi;
case digit_sisa of
   1: akhiran := 'akhiran puluhan';
   2: akhiran := 'akhiran ratusan';
   3: akhiran := 'akhiran ribuan';
   6: akhiran := 'akhiran jutaan';
   9: akhiran := 'akhiran milyaran';
end;
hasil := terbilang(kepala) + akhiran;
if sisa > 0 then hasil := hasil + terbilang(sisa);

output hasil;
```

Untuk bilangan *irreguler* penanganannya harus dilakukan sebelum menangani bilangan-bilangan *reguler* dengan langsung menyebutkan namanya. Untuk membedakannya dengan jenis bilangan yang lainnya maka pengecekan bilangan *irreguler* dilakukan dengan langsung membandingkan nilainya di dalam konstruksi `if` bertingkat. Algoritma untuk menangani bilangan *irreguler* tersebut dapat dilihat pada *pseudocode* 2.

Pseudocode 2: Bagian algoritma untuk menangani bilangan *irreguler*

```
// misalkan bilangan irreguler-nya adalah x, y, dan z.
if bilangan = x then hasil := 'nama bilangan x'
else if bilangan = y then hasil := 'nama bilangan y'
else if bilangan = z then hasil := 'nama bilangan z'
else begin
   // bagian untuk menangani bilangan reguler & semi reguler.
end;
```

Penanganan bilangan *semi reguler* dapat menggunakan cara yang sama dengan penanganan bilangan *irreguler*, tapi bisa juga digabung dalam penanganan bilangan *reguler*. Hal ini dikarenakan aturan penamaan bilangan semi *reguler* secara umum mengikuti pola yang sama dengan aturan pada bilangan *reguler*, yaitu terbentuk dari suatu komponen kepala, komponen akhiran, dan komponen sisa. Perbedaan yang muncul bisa jadi dikarenakan oleh perubahan komponen akhiran, perubahan nama komponen kepala, atau bahkan karena perubahan nilai komponen kepala dan sisanya. Berikut ini beberapa contohnya dalam beberapa bahasa yang berbeda:

- Untuk bilangan 11 sampai 19 dalam Bahasa Indonesia, akhiran “puluh” diganti menjadi “belas”, komponen kepalanya justru diisi dengan bilangan pada komponen sisa, dan sisanya seakan-akan menjadi nol. Selain itu untuk bilangan 11, nama komponen kepala yang harusnya “satu” berubah jadi “se”.
- Untuk bilangan dari 60 sampai 69 dalam Bahasa Jawa Ngoko, akhirannya berubah dari “puluh” menjadi “widak”, komponen kepalanya berubah dari 6 menjadi 1, dan namanya berubah dari “satu” menjadi “se”.
- Untuk bilangan dari 13 hingga 19 dalam Bahasa Inggris, akhiran “ty” diganti menjadi “teen”, komponen kepalanya diisi bilangan pada komponen sisa, dan banyak yang mengalami perubahan nama (“three” jadi “thir”, “five” jadi “fif”, dan “eight” jadi “eigh”).

- Untuk bilangan dari 70 hingga 79 dalam Bahasa Perancis, akhiran tetap "ante", namun komponen kepalanya berubah dari 7 menjadi 6, namanya berubah dari "six" menjadi "soix", dan komponen sisanya ditambah 10 (dari 0 jadi 10, dari 1 jadi 11, dst.).

Perancangan

Berdasarkan algoritma yang dijabarkan di atas, pada bagian ini akan dibuat rancangan sebuah implementasi fungsi yang bersifat generik. Artinya fungsi tersebut dirancang akan dapat diterapkan untuk memperoleh bentuk terbilang dari bahasa apapun tergantung dari input dan *setting* yang diberikan.

Secara garis besar rancangan fungsi tersebut akan dibagi ke dalam bagian-bagian yang disusun secara berurutan berikut:

1. Bagian inisialisasi

   Bagian ini berisi berbagai inisialisasi data dan setting yang dibutuhkan, semacam nama-nama angka, banyaknya digit per kelompok digit, akhiran-akhiran nilai posisi yang umum digunakan, dan daftar bilangan-bilangan *irreguler* (bisa juga dengan disertai namanya).
2. Bagian untuk menangani bilangan negatif

   Di dalam bagian ini intinya hanya melakukan konkatenasi teks yang menyatakan kata "negatif" dalam bahasa yang ditangani dengan bentuk terbilang dari nilai mutlak bilangan awal.
3. Bagian untuk menangani bilangan satu digit

   Inti bagian ini hanyalah untuk menghasilkan nama angka yang sesuai.
4. Bagian untuk menangani bilangan irreguler

   Bagian ini juga digunakan untuk menghasilkan nama yang sesuai dengan bilangan *irreguler* yang terkait.
5. Bagian untuk menangani bilangan *semi reguler* dan *reguler*

   Bagian ini dapat dibagi menjadi berikut:

   a. Penangan bilangan *semi reguler* tahap pertama

      Bagian ini hanya perlu diadakan jika terdapat bilangan *semi reguler* yang namanya melibatkan perubahan pada banyaknya digit per kelompok digit. Sebagai contoh bagian ini berguna untuk menangani kelompok "belasan ratus" pada Bahasa Inggris.

   b. Inisialisasi awal bentuk terbilang

      Bagian ini digunakan untuk menentukan nilai awal komponen kepala, komponen sisa, dan komponen akhiran yang digunakan.

c. Penangan bilangan *semi reguler* tahap kedua

   Setelah pada bagian sebelumnya didapatkan ketiga komponen penyusunan bentuk terbilang, maka pada bagian ini dilakukan berbagai pengubahan dan atau perhitungan yang khusus ditujukan untuk menangani kelompok bilangan *semi reguler*. Misalkan pengubahan nilai kepala atau sisa serta perubahan nama angka atau akhiran.

d. Penyusunan akhir

   Bagian ini berisi penyusunan semua komponen yang terakhir dihasilkan menjadi bentuk akhir terbilang.

PENUTUP

Penyebutan nilai dari suatu angka dapat dilakukan melalui sebuah fungsi dalam bahasa pemrograman. Fungsi ini bekerja mengikuti aturan bahasa untuk penyebutan nilai-nilai angka secara *reguler*, *semi reguler*, maupun *irreguler*. Algoritma yang digunakan bersifat rekursif, dan memberikan penyebutan pada suatu nilai angka sesuai dengan posisi dan nilai satuan berdasar digitnya.

DAFTAR PUSTAKA

Coffey, Neil, *Number in French*, http://www.french-linguistics.co.uk/tutorials/numbers/, diakses pada tanggal 2 Nopember 2010.

Dahl, O.J., Dijkstra, E.W., Hoare, C.A.R., 1972, *Structured Programming*, Academic Press.

Menninger, K., Broneer, P., 1992, *Number Words and Number Symbols*, Courier Dover Publications.

Wirth, Niklaus, 1976, *Algorithms + Data Structures = Programs*, Prentice-Hall.

_____, *English Numerals*, http://en.wikipedia.org/wiki/English_numerals, diakses pada tanggal 29 Oktober 2010.

_____, *Korean Numerals*, http://en.wikipedia.org/wiki/Korean_numerals, diakses pada tanggal 10 Nopember 2010.

_____, *Number Names*, http://en.wikipedia.org/wiki/Number_names, diakses pada tanggal 1 Nopember 2010.

_____, *Recursion*, http://en.wikipedia.org/wiki/Recursion, diakses pada tanggal 20 Oktober 2010.