BAB II

TEORI PENUNJANG

2.1. Sistem Bilangan.

Sistem bilangan menggunakan suatu bilangan dasar atau basis tertentu. Basis yang dipergunakan pada setiap sistem bilangan tergantung dari jumlah nilai bilangan yang dipergunakan. Sistem bilangan yang banyak digunakan oleh manusia adalah bilangan desimal (0,…,9). Akan tetapi selain bilangan desimal juga terdapat bilangan biner (0,1), bilangan oktal (0,…,7), dan heksa desimal (0,…,9,A,…,F). Dan karena dalam tugas akhir ini hanya berhubungan dengan bilangan desimal dan bilangan biner, maka hanya kedua bilangan tersebut saja yang akan dibahas.

Untuk membedakan bilangan pada sistem yang berbeda digunakan subskrip. Sebagai contoh : 9_{10} menyatakan bilangan sembilan pada bilangan desimal, sedangkan 101_2 menunjukkan bilangan biner 101. Akan tetapi subskrip tersebut sering diabaikan jika bilangan yang dipakai sudah jelas.

2.1.1. Bilangan Desimal

Bilangan denary atau yang lebih dikenal dengan bilangan desimal merupakan bilangan dengan basis 10 yang mempunyai 10 simbol yaitu 0,1,2,3,4,5,6,7,8,9. Setiap posisi digit dalam bilangan desimal mempunyai bobot atau harga tersendiri yang menunjukkan kenaikan pada pangkat dengan basis 10, yaitu 10^0=1, 10^1=10, 10^2=100, dan sebagainya. Digit paling kanan mempunyai bobot 10^0 (satuan), digit kedua mempunyai bobot 10^1 (puluhan), digit ketiga mempunyai bobot 10^2 (ratusan),

demikian seterusnya. Jumlah dari semua digit yang dikalikan dengan bobot masing-masing memberikan harga dari bilangan yang bersangkutan.

Jadi aturan umum untuk mempresentasikan bilangan desimal dengan notasi kedudukan $a_{n-1}10^{n-1}+ a_{n-2}10^{n-2}+ \ldots +a_0$ adalah $a_{n-1}\, a_{n-2} \ldots a_0$,

dengan n : banyaknya digit di sebelah kiri tanda koma dalam sistem desimal.

Contoh : Bilangan desimal 2453 mempunyai notasi kedudukan :

$(2x10^3) + (4x10^2) + (5x10^1) + (3x10^0) = 2000 + 400 + 50 + 3 = 2453$

2.1.2. Bilangan Biner

Biner berasal dari kata bi yang artinya dua. Jadi bilangan biner merupakan bilangan yang hanya menggunakan 2 simbol yaitu 0 dan 1. Seperti pada bilangan desimal, setiap posisi digit dalam bilangan biner juga mempunyai bobot tertentu. Sistem biner hanya menggunakan 2 macam digit sehingga bobot yang bersangkutan merupakan pangkat dari 2 dan bukan dari 10.

Bobot-bobot dalam biner tersebut dimulai dari angka yang paling kecil (paling kanan) adalah 2^0 (satuan), 2^1 (duaan), 2^2 (empatan), dan seterusnya. Untuk bilagnan biner yang lebih besar, bobot ini dapat dilanjutkan dengan pangkat yang lebih besar.

Jadi untuk menyatakan sebuah bilangan biner dengan notasi kedudukan

$a_{n-1}2^{n-1}+ a_{n-2}2^{n-2}+ \ldots +a_0$ dipresentasikan sebagai $a_{n-1}\, a_{n-2} \ldots a_0$,

dengan a_i : berharga 0 atau 1

n : banyaknya digit disebelah kiri tanda koma biner.

Contoh : Bilangan biner 101 mempunyai notasi kedudukan

$(1x2^2) + (0x2^1) + (1x2^0)$

Bit (dari : binary digit) dipakai untuk menyatakan sesuatu angka biner. Contoh: bilangan biner 1011 terdiri atas 4 bit (= 4 angka biner).

2.1.3. Aritmatika Biner

a. Penjumlahan Biner.

Penjumlahan bilangan biner dilakukan dengan cara yang sama seperti penjumlahan desimal. Dua bilangan yang akan dijumlahkan disusun secara vertikal dan digit-digit yang mempunyai bobot sama ditempatkan pada kolom yang sama. Digit-digit ini kemudian dijumlahkan dan karena 1 merupakan digit terbesar dalam sistem biner, maka setiap jumlahan yang lebih besar dari satu ada bilangan yang dibawa. Bilangan yang dibawa itu kemudian dijumlahkan dengan digit disebelah kirinya, dan seterusnya.

Berikut adalah aturan dasar untuk penjumlahan biner :

Aturan 1 : 0 + 0 = 0 Aturan 3 : 1 + 0 = 1

Aturan 2 : 0 + 1 = 1 Aturan 4 : 1 + 1 = 0 , dengan bawaan 1

Contoh : menjumlah 1 0 0 1 dengan 0 1 0 1

Penyelesaian : bilangan disusun dalam kolom-kolom,

1	bawaan
1 0 0 1	bit atas
0 1 0 1	bit bawah
1 1 1 0	
DCBA	

Penjelasan proses penjumlahan kolom demi kolom :

Kolom A : 1+1=0, dengan bawaan 1 dan bawaan 1 ditambahkan ke atas kolom B (aturan 4).

Kolom B : Penjumlahan pertama yang ditunjukkan dalam kolom B adalah bawaan 1 ditambah bit atas 0, maka 1+0=1 (aturan 3). Sekarang hasil penjumlahan ditambahkan bit bawah, maka 1+0=1 (kembali menggunakan aturan 3).

Kolom C : 0+1=1 (aturan 2)

kolam D : 1+0=1 (aturan 3)

Jadi penjumlahan 1001 dengan 0101 adalah 1110.

b. Pengurangan Biner.

Pengurangan caranya sama dengan penjumlahan, hanya operasinya merupakan kebalikan dari penjumlahan. Untuk mengurangi, kita perlu menegakkan prosedur untuk mengurangi suatu digit lebih besar dari digit lebih kecil. Satu-satunya kasus terjadinya hal ini dalam bilangan biner ialah ketika mengurangi 1 dari 0. Sisanya 1, tetapi harus meminjam 1 dari kolom berikutnya yang ada disebelah kiri.

Aturan dasar untuk pengurangan biner :

Aturan 1 : 0-0=0 Aturan 3 : 1-1=0

Aturan 2 : 1-0=1 Aturan 4 : 0-1=1 pinjam 1 atau 10-1=1

Contoh : mengurangi 011011 dari 110101

Penyelesaian :

```
  0 10 10  0 10
  1  1  0  1  0  1
  0  1  1  0  1  1
  ---------------
  0  1  1  0  1  0
  F  E  D  C  B  A
```

Penjelasan proses pengurangan kolom demi kolom :

Kolom A : 1-1=0 (aturan 2).

Kolom B : 0-1=1 , dengan pinjam 1 (aturan 4).

Kolom C : 0-0=0 (aturan 1).

Ingat bahwa dirubah dari 1 ke 0 karena dipinjam kolom B.

Kolom D : 0-1=1 , dengan pinjam 1 (aturan 4)

Ingat bahwa dirubah dari 0 menjadi 10.

Kolom E : 0-1= 1 , dengan pinjam 1 (aturan 4).

Ingat bahwa dirubah dari 0 ketika dipinjam oleh kolom D, menjadi 10 dengan meminjam dari kolom F.

Kolom F : 0-0=0 (aturan 1).

Jadi pengurangan 011011 dari 110101 adalah 011010.

2.1.4. Konversi Desimal ke Biner

Agar dapat membuat program untuk mengubah bilangan desimal menjadi bilangan biner, maka perlu diketahui cara mengubah bilangan desimal menjadi biner.

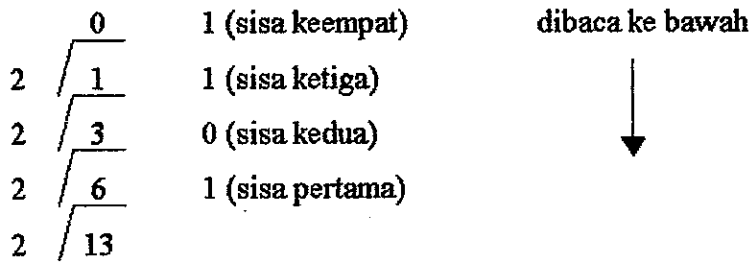
Suatu cara untuk mengubah bilangan desimal menjadi bilangan biner adalah dengan metode double-dabble. Metode ini berlaku untuk semua bilangan desimal.

Adapun metode double-dabble ini adalah sebagai berikut :

1. Bagilah bilangan desimal yang akan dirubah dengan bilangan 2 secara berulang-ulang dan tuliskan hasil bagi dan sisanya setiap kali pembagian dilakukan.
2. Bila sudah diperoleh hasil bagi 0 dan sisa pembagian 1, maka selesailah proses konversi tersebut.
3. Bilangan biner yang dihasilkan dapat diketahui dengan membaca sisa-sisa pembagian dari atas ke bawah (mulai dari sisa pertama sampai sisa terakhir).

Contoh :

Akan ditunjukkan cara mengubah bilangan desimal 13 ke dalam bilangan biner.



Keterangan gambar diatas :

Bagilah 13 dengan 2, hasil baginya 6 dan 1 merupakan sisa pembagian. Lalu bagi lagi 6 dengan 2, diperoleh hasil 3 dengan sisa 0. Bagi lagi 3 dengan 2, didapatkan 1 sebagai hasil dan sisanya 1. Sekali lagi dibagi dengan 2, dan ternyata 2 tidak dapat membagi 1, karena itu hasil baginya 0 dan sisanya 1. Dengan membaca sisa-sisa hasil pembagian dari atas ke bawah diperoleh bahwa bilangan biner dari 13 adalah 1101.

2.1.5. Konversi Biner ke Desimal.

Sudah diketahui cara untuk mengubah bilangan desimal ke bilangan biner. Selanjutnya yang harus dipelajari adalah cara mengubah bilangan biner menjadi bilangan desimal ekuivalennya.

Dengan menggunakan metode lurusan (streamlined method), dapat meluruskan perubahan biner ke desimal. Berikut merupakan langkah-langkahnya :

1. Tuliskan bilangan biner yang dimaksud.
2. Tuliskan bobotnya (1,2,4,8,...) di bawah masing-masing digit yang bersangkutan.
3. Coretlah setiap bobot yang berada di bawah digit 0.

4. Jumlahkan bobot-bobot yang tersisa.

Contoh :

Dilakukan konversi biner 1101 ke dalam desimal sebagai berikut :

1. 1 1 0 1 (tulis bilangan binernya)
2. 8 4 2 1 (tulis bobot masing-masing)
3. 8 4 0 1 (coret bobot di bawah digit 0)
4. 8 + 4 + 0 + 1 = 13 (jumlahkan semua bobot yang tidak dicoret)

Jadi bilangan biner 1101 ekuivalen dengan bilangan desimal 13.

2.2. Program dan Algoritma.

2.2.1. Program.

Program adalah seperangkat perintah atau langkah-langkah yang memerintahkan komputer secara tepat untuk menangani suatu masalah lengkap dengan penyelesaiannya, yang ditulis dalam bahasa pemrograman komputer.

Tahapan-tahapan dalam pembuatan program adalah sebagai berikut :

a. Mengerti masalah.

Pemrogram harus tahu persis untuk apa program dibuat. Biasanya diketahui dari spesifikasi program. Dengan mengetahui spesifikasi program, dapat didefinisikan masukan, pemrosesan dan keluaran yang diinginkan.

b. Merencanakan metode yang digunakan untuk menyelesaikan masalah.

Perencanaan bergantung pada besar kecilnya suatu permasalahan. Program besar dapat dibuat oleh beberapa pemrogram dan setiap pemrogram membuat sebagian program secara terpisah. Bagian yang terpisah ini sering disebut dengan modul atau segmen. Modul-modul disiapkan dan dites secara terpisah.

Setelah itu disatukan dan dites secara keseluruhan. Proses ini dinamakan dengan integrasi.

c. Mengembangkan metode dengan menggunakan pertolongan yang mudah, seperti misalnya algoritma.

Pendekatan modern dalam pemrograman ialah menyelesaikan masalah yang sulit dengan membagi-bagi masalah yang lebih sederhana dan teratur dalam model langkah demi langkah, yang selanjutnya disebut sebagai algoritma. Kemudian diadakan pengetesan algoritma, jika masih terdapat kesalahan, ulangi kembali mulai tahapan a, jika tidak lanjutkan ke tahapan d.

d. Menulis instruksi-instruksi kedalam bahasa pemrograman tertentu.

Instruksi-instruksi yang ditulis dalam algoritma ditulis kembali kedalam bahasa pemrograman tertentu.

e. Mengetes program.

Kebenaran dari suatu program harus dites terlebih dahulu sebelum dieksekusi. Pengetesan terhadap kebenaran sintak penulisan akan dilaksanakan oleh komputer itu sendiri. Sedangkan pengetesan terhadap kebenaran logikanya dapat dilihar dari output program tersebut. Rancangan yang sangat berhati-hati dalam tahap awal pemrograman akan dapat membantu meminimalkan kesalahan.

f. Dokumentasi.

Sangat penting kiranya hasil kerja pemrogram dalam menghasilkan suatu program didokumentasikan secara lengkap. Dokumentasi meliputi pernyataan masalah, algoritma, data fungsi dan hasil, rincian teknis dan instruksi untuk pemakai.

2.2.2. Algoritma.

Algoritma adalah sekumpulan instruksi yang menjelaskan langkah-langkah yang teratur dan menggambarkan aktifitas penyelesaian suatu masalah. Dengan langkah-langkah yang terperinci dan berturut diharapkan akan mempermudah pembuatan logika program sebagai dasar penulisan program, dan lebih mudah dibaca dan dikomunikasikan dengan orang lain.

Syarat-syarat algoritma :

a. Setiap langkah berupa instruksi yang harus dapat dilaksanakan.

b. Langkah harus tertentu, jelas dan berurutan.

c. Harus mempunyai akhir.

d. Tidak terikat pada bahasa pemrograman manapun.

Cara menyatakan algoritma :

a. Dengan bahasa ilmiah (natural language).

Algoritma ditulis dengan ungkapan bahasa sehari-hari.

b. Dengan sandi semu (pseudocode).

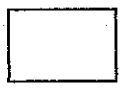
Algoritma yang telah ditulis dengan menggunakan perintah bahasa pemrograman tertentu ditambah dengan bahasa alamiah.

c. Dengan flowchart (bagan alir).

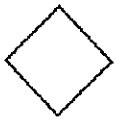
Algoritma ditulis dalam bentuk simbol atau bentuk diagram.

Simbol-simbol yang biasa digunakan adalah :

Terminal : digunakan sebagai simbol awal atau akhir program. 

Proses : suatu simbol yang melambangkan diprosesnya suatu data. Jenis proses dituliskan didalamnya 

Decision/pengambilan keputusan: digunakan untuk menentukan arah selanjutnya dengan jalan pengecekan terhadap suatu kondisi/logika. Pada simbol ini hanya ada 1 titik masuk dan paling sedikit 2 titik keluar.



Off page connector : digunakan untuk menandakan sambungan aliran program dalam halaman yang berbeda.

Flow lines : digunakan untuk menandakan arah aliran program.

2.3. Bahasa Pemrograman Pascal.

Pascal adalah bahasa tingkat tinggi yang orientasinya pada segala tujuan. Dirancang oleh Prof. Niclaus Wirth dari Technical University dari Zurich, Zwitzerland. Nama Pascal diambil sebagai penghargaan terhadap Blaise Pascal, ahli matematika dan philosophi terkenal abad 17 dari Perancis.

Pascal dipublikasikan pada tahun 1971 dengan tujuan untuk membantu mengajar program komputer secara sistematis, khususnya untuk memperkenalkan pemrograman yang terstruktur. Jadi Pascal adalah bahasa yang ditujukan untuk membuat program terstruktur.

Karena antara satu bahasa pemrograman dengan bahasa pemrograman lain mempunyai bentuk program dan aturan-aturan yang saling berbeda, maka dirasa perlu untuk dibahas dan dipelajari mengenai aturan-aturan bahasa pemrograman yang akan digunakan, dalam hal ini adalah Pascal. Dan dengan memahami aturan-aturan tersebut, diharapkan program yang tersusun akan bisa dijalankan dengan baik dan memberikan hasil seperti yang diharapkan.

2.3.1. Struktur Program Pascal.

Program dalam bahasa Pascal mempunyai bentuk atau struktur tertentu. Bentuk ini harus dipenuhi agar program tidak dapat disalahkan oleh kompiler. Kompiler yang dimaksud adalah penterjemah program Pascal bahasa mesin, yaitu sistem Turbo Pascalnya itu sendiri.

Secara umum struktur program Pascal dibagi menjadi 3 bagian, yaitu :

1. Kepala program, yang terdiri dari kata **program** dan judul program.

   Program ... {Kepala Program}

2. Bagian deklarasi, yang terdiri dari deklarasi-deklarasi.

   Uses ... {Deklarasi Piranti}
   Const ... {Deklarasi Konstanta}
   Type ... {Deklarasi tipe Data}
   Var ... {Deklarasi Peubah}
   Prosedure ... {Deklarasi Procedure}
   Function ... {Deklarasi Function}

3. Bagian pernyataan (statement), yang dimulai dengan kata **begin** dan diakhiri dengan kata **end**.

   Begin {Awal Program Utama}
   ... {Statement-statement}
   End {Akhir Program Utama}

2.3.2. Procedure dan Function.

Sebagai salah satu alat pemrograman terstruktur, subprogram merupakan bagian yang sangat bermanfaat dan produktif, kegunaannya antara lain :

1. Untuk memecah suatu program yang besar dan rumit menjadi bagian-bagian yang lebih kecil dan sederhana.

2. Untuk menuliskan sekumpulan instruksi yang sering dilakukan berkali-kali hanya dengan sekali tulis saja.

Sehingga kegunaan subprogram kiranya tidak bisa diabaikan. Dalam Pascal, dikenal 2 (dua) subprogram, yaitu procedure dan function.

Procedure

Procedure mempunyai struktur yang sama dengan struktur program. Di dalam procedure juga dimungkinkan ada procedure lain. Bentuk ini dinamakan dengan procedure tersarang (nested procedure).

Semua deklarasi dalam procedure dikatakan sebagai deklarasi lokal, sehingga hanya dapat digunakan dalam procedure itu saja dan tidak dikenal diluar procedure. Supaya nilai variabel dalam suatu procedure bersifat global, maka variabel tersebut harus dideklarasikan di atas procedure yang akan menggunakan.

Bentuk umum deklarasi procedure adalah :

Procedure *pengenal* (*parameter_formal* : *tipe_data*) ;

Sedangkan bentuk umum pemanggil procedure adalah :

Pengenal (*parameter_aktual*) ;

dengan :

pengenal : nama procedure

parameter_formal : parameter yang ada dan dituliskan pada deklarasi procedure

parameter_aktual : parameter yang dikirimkan ke modul procedure

tipe_data : tipe data parameter

Function

Secara umum function hampir sama dengan procedure, dengan sedikit perbedaan :

1. function hanya memberikan satu hasil
2. nama function, selain digunakan untuk memanggil function tersebut, juga dianggap sebagai satu perubah sehingga bisa digunakan sebagai bagian dari ungkapan.
3. dalam function tidak ada istilah parameter perubah, semua parameter dianggap sebagai parameter nilai,
4. semua parameter diperlakukan sebagai masukan pada function tersebut dengan keluarannya adalah nama function yang bertindak sebagai satu peubah.

Bentuk umum deklarasi function adalah :

Function *pengenal* (*parameter_formal* : *tipe_data*) : tipe_fungsi ;

Sedangkan bentuk umum pemanggil function adalah :

Pengenal (*parameter_aktual*) ;

dengan

pengenal : nama function

parameter_formal : parameter yang ada dan dituliskan pada deklarasi function

parameter_aktual : parameter yang dikirimkan ke modul function

tipe_data : tipe data parameter

tipe_fungsi : tipe hasil dari fungsi

2.4. Struktur Data.

Sehubungan dengan struktur data, maka data di dalam program Pascal dapat diorganisir sedemikian rupa sehingga bisa dilihat dengan jelas bahwa data tersebut adalah data yang terstruktur. Salah satu tipe data terstruktur adalah larik (array).

Larik

Larik (array) adalah sekumpulan nilai data (masing-masing nilai data tersebut disebut dengan komponen) dalam jumlah yang tetap dan setiap komponen mempunyai tipe data yang sama. Posisi masing-masing komponen dalam larik dinyatakan dengan nomor index.

Bentuk umum deklarasi larik dimensi satu adalah :

Var *pengenal* **: array** [*tipe_index*] **of** *tipe_data* ;

dengan pengenal : nama larik yang akan dideklarasikan

tipe_index : banyak komponen dalam suatu larik

tipe_data : tipe data dari komponen dalam larik

Tipe dari larik juga bisa dideklarasikan pada bagian deklarasi tipe, dengan bentuk umum :

Type *pengenal* = **array** [*tipe_index*] **of** *tipe_data* ;

Deklarasi tipe data larik pada type kadang-kadang diperlukan jika pada program ada sejumlah larik yang mempunyai banyak elemen dan tipe data yang sama.

Selain deklarasi larik dimensi satu, bisa juga dibuat larik dimensi banyak (dimensi n). Dengan bentuk umum :

Var *pengenal* **: array** [*tipe_index_1, tipe_index_2, ... , tipe_index_n*] **of** *tipe_data* ;

atau

Type *pengenal* **: array** [*tipe_index_1, tipe_index_2, ... , tipe_index_n*] **of** *tipe_data*;

2.5. Operasi OR-eksklusif

Istilah OR-eksklusif seringkali disingkat sebagai XOR. Simbol tanda tambah di dalam lingkaran (⊕) menandakan fungsi XOR dalam aljabar boolean yang menyatakan bahwa masukan di OR-kan dengan eksklusif satu sama lain. XOR merupakan relasi antara OR, AND, dan NOT.

Dan XOR mempunyai ketentuan sebagai berikut :

'Output akan bernilai 1 bila input-inputnya yang berlogika 1 berjumlah ganjil.'

Oleh karena itu operasi XOR dapat disebut sebagai operasi pemeriksa bit ganjil.

Operasi XOR (⊕) yang digunakan pada tugas akhir ini caranya sama dengan penjumlahan biner biasa, hanya aturan untuk operasinya saja yang berbeda.

Aturan dasar untuk operasi XOR :

Aturan 1 : 0+0=0

Aturan 2 : 1+0=1

Aturan 3 : 0+1=1

Aturan 4 : 1+1=0

Contoh : mengXORkan 1001 dengan 0101

Penyelesaian : bilangan disusun dalam kolom-kolom,

1 0 0 1	bit atas
0 1 0 1	bit bawah
1 1 0 0	
DCBA	

Penjelasan proses penjumlahan kolom demi kolom :

Kolom A : 1+1=0 (aturan 4)

Kolom B : 0+0=1 (aturan 1)

Kolom C : 0+1=1 (aturan 3)

Kolom D : 1+0=1 (aturan 2)