BAB II

DASAR PEMROGRAMAN GRAFIK

II.1. Pengertian Pemrograman Grafik

II.1.1. Program

Sebelum sampai pada pengertian pemrograman grafik perlu dibahas lebih dahulu pengertian dasar dari pemrograman. Suatu perangkat komputer terdiri dari dua unsur utama yaitu perangkat keras dan perangkat lunak. Perangkat keras hanya dapat bekerja sesuai keinginan jika dikendalikan oleh suatu perangkat lunak. Perangkat lunak itu sendiri ada yang telah menyatu dengan perangkat kerasnya seperti sistem BIOS, ada pula yang terpisah misalnya sistem operasi, program aplikasi dan lain-lain.

Program adalah kumpulan instruksi yang terstruktur dan terintegrasi yang berfungsi mengendalikan dan memanfaatkan perangkat keras komputer untuk melaksanakan instruksi yang ada dalam program. Dengan demikian perangkat keras yang ada haruslah difungsikan secara optimal untuk mencapai maksud dari pemrogram. Misalnya untuk menampilkan warna-warna tertentu dibutuhkan instruksi untuk maksud tersebut sehingga pemakaian monitor berwarna dapat optimal.

II.1.2. Grafik

Untuk mempresentasikan suatu data digunakan berbagai cara. Dapat ditempuh dengan menampilkan angka-angka yang menerangkan tentang data

yang bersangkutan, atau menggunakan bagan-bagan yang menggambarkan hubungan antar data. Hal ini dilakukan dengan menghubungkan data-data yang ada yang berakibat tampilnya kaitan satu data dengan lainnya.

Cara yang terakhir disebutkan lebih diminati karena merupakan upaya memvisualisasikan data yang ada kedalam bentuk-bentuk diagram. Bentuk diagram dapat berupa grafik 2 dimensi ataupun 3 dimensi (grafik 2D dan 3D). Visualisasi deretan data angka ke dalam bentuk bagan / diagram dikenal dengan grafik. Dalam perkembangannya grafik tidak hanya terbatas pada diagram saja melainkan bentuk bentuk geometris lainnya seperti poligon, trapesium, bahkan gambar-gambar yang rumit misalnya objek bangunan, binatang, logo dan sebagainya. Untuk membedakan pengertian diagram dengan gambar yang lain dikenal istilah *chart* untuk diagram-diagram dan grafik untuk gambar-gambar.

Perkembangan lebih lanjut di bidang visualisasi data memungkinkan adanya interaksi antar pemakai atau operator dengan bagan atau grafik yang ada menggunakan peranti masukan yang tersedia. Interaksi yang terjadi adalah operator melakukan perubahan atau masukan, langsung pada layar komputer. Sistem yang memungkinkan operator untuk berdialog langsung dengan apa yang terlihat pada layar komputer disebut dengan sistem grafik komputer interaktif (*interactive computer graphic*). (P. Insap Santosa, 1996:2)

II.1.3. Pemrograman Grafik

Dari pengertian-pengertian unsur pemrograman grafik di atas dapat

disimpulkan bahwa pemrograman grafik adalah pemrograman yang mengoptimalkan perangkat-perangkat keras grafis dari komputer dengan menggunakan kumpulan instruksi yang berorientasi grafis untuk menciptakan, memanipulasi, dan menyimpan gambar model suatu objek yang bertujuan memudahkan komunikasi antara pemakai dengan komputer.

Menilik orientasi objek yang dibangkitkan hasil program grafik maka dapat dilihat bahwa pemrograman ini sepenuhnya menggunakan media yang dibangkitkan oleh penggerak grafis dari bahasa pemrograman yang digunakan. Turbo Pascal 7.0 digunakan untuk pembuatan program aplikasi ini. Banyak kemampuan yang ditawarkan untuk mengoptimalkan fungsi pemrograman grafik dari Turbo Pascal.

II.2 Komponen Pemrograman Grafik

II.2.1. Perangkat Keras dan Perangkat Lunak

Komponen yang terbilang penting dalam pemrograman grafik adalah layar penampil komputer, dan penggerak tampilan secara perangkat keras. Selain itu dibutuhkan pula pengolah tampilan yang ada. Keterkaitan perangkat keras dan perangkat lunak dalam pengelolaan pemrograman grafik berjalan secara bersamaan. Karena tidak mungkin perangkat keras yang ada diaktifkan tanpa adanya pengolah tampilan yang dalam hal ini bersifat perangkat lunak.

Pengolah tampilan (*display processor*) atau *video display adapter* adalah bagian yang mengubah pola bit dari pengingat digital menjadi

tegangan analog, yang selanjutnya akan membangkitkan elektron yang digunakan untuk menembak fosfor pada layar tampilan. (P. Insap Santosa, 1996:8)

Sedangkan untuk menampilkan grafik dibutuhkan suatu layar tampilan yang sesuai dengan pengolah tampilan. Terdapat lima tipe layar tampilan sebagai berikut:

Direct Monochrome Monitor. Tipe layar tampilan ini biasanya digunakan untuk adapter dari jenis MDA atau EGA. Layar tampilan jenis ini hanya menyajikan warna latar depan (*foreground*) dan warna latar belakang (*background*).

Composite Monochrome Monitor. Tipe layar ini digunakan bersama-sama dengan adapter dari jenis CGA. Tipe layar ini hanya bisa menyajikan sebuah warna latar depan, dan hanya bisa digunakan bersama-sama dengan adapter jenis CGA.

Composite Color Monitor. Tipe layar ini bisa menghasilkan teks dan grafik berwarna (*color*). Meskipun demikian tipe layar ini mempunyai resolusi yang jelek sehingga gambar yang dihasilkan tidak bagus. Tipe layar tampilan ini harus digunakan bersama-sama dengan adapter dari jenis CGA.

Red-Green-Blue Monitor. Tipe layar ini lebih dikenal dengan sebutan RGB Monitor (RGB = *Red Green Blue*). Tipe layar RGB lebih baik dibanding dengan *composite color monitor* karena layar tampilan ini memproses isyarat warna merah, hijau dan biru secara terpisah. Dengan demikian, teks dan grafik yang dihasilkan juga lebih halus.

Variable-Frequency Monitor. Adapter tampilan yang berbeda seringkali membangkitkan isyarat yang berbeda pula, sehingga ada beberapa layar tampilan yang tidak bisa dipasang dengan adapter tertentu. Layar tampilan ini memungkinkan kita untuk menggunakan adapter tampilan yang berbeda, sehingga apabila ada teknologi adapter tampilan yang lebih baru, kita tidak perlu membeli layar tampilan yang baru pula. (P. Insap Santosa, 1996:11).

Perkembangan teknologi pengolah tampilan hingga saat ini telah memungkinkan suatu layar tampilan untuk menampilkan hingga 1 juta warna. Kemampuan ini memungkinkan menampilkan gambar dengan kualitas foto. Pemrograman yang memanfaatkan kemampuan layar tampilan ini pun berkembang pula terutama dibidang simulasi dan animasi, maupun publishing.

II.2.2. Penggerak Grafik

Penggerak grafik atau *video display adapter* yang dibutuhkan dalam pemrograman grafik disesuaikan dengan jenis layar tampilan yang digunakan. Tuntutan kualitas grafik yang lebih baik dengan resolusi yang tinggi menyebabkan pembuat perangkat lunak pemrograman berusaha menghasilkan penggerak grafik yang lebih baik. Akibat langsung yang muncul adalah membuat program grafik menjadi lebih mudah dengan kualitas yang semakin lebih baik.

Turbo Pascal sebagai salah satu bahasa pemrograman, sampai dengan versi 7.0, menyediakan berbagai penggerak grafik untuk memanfaatkan layar tampilan yang ada di pasaran. Penggerak grafik yang sesuai dengan layar

tampilan yang ada dapat dilihat pada tabel berikut.

Tabel II.1. Daftar Penggerak Grafik Turbo Pascal

Penggerak Grafik		Mode		Resolusi	Palette
Nama	Nilai	Nama	Nilai	kolom x baris	(warna)
CGA	1	CGAC0	0	320 x 200	C0
		CGAC1	1	320 x 200	C1
		CGAC2	2	320 x 200	C2
		CGAC3	3	320 x 200	C3
		CGAHi	4	640 x 200	2 warna
MCGA	2	MCGAC0	0	320 x 200	C0
		MCGAC1	1	320 x 200	C1
		MCGAC2	2	320 x 200	C2
		MCGAC3	3	320 x 200	C3
		MCGAMed	4	640 x 200	2 warna
		MCGAHi	5	640 x 480	2 warna
EGA	3	EGALo	0	640 x 200	16 warna
		EGAHi	1	640 x 350	16 warna
EGA64	4	EGA64Lo	0	640 x 200	16 warna
		EGA64Hi	1	640 x 350	4 warna
EGAMONO	5	EGAMono-Hi	2	640 x 350	2 warna*
		EGAMono-Hi	3	640 x 350	2 warna**
IBM8514	6	IBM8514Lo	0	640 x 480	256 warna
		IBM8514Hi	1	1024 x 768	256 warna
HERCMono	7	HercMonoHi	0	720 x 348	2 warna
ATT400	8	ATT400C0	0	320 x 200	C0
		ATT400C1	1	320 x 200	C1
		ATT400C2	2	320 x 200	C2
		ATT400C3	3	320 x 200	C3
		ATT400Med	4	640 x 200	2 warna
		ATT400Hi	5	640 x 400	2 warna
VGA	9	VGALo	0	640 x 200	16 warna
		VGAMed	1	640 x 350	16 warna
		VGAHi	2	640 x 480	16 warna
PCS3270	10	PCS3270Hi	0	720 x 350	2 warna

* Kartu EGAMONO 64K
** Kartu EGAMONO 256K

Penggerak grafik yang tercantum dalam kolom pertama dan kedua dari tabel di atas, merupakan penggerak grafik yang digunakan di pasaran. Sehingga

dengan kemampuan untuk menyesuaikan penggerak grafik (*video display adapter*) dengan jenis layar tampilan, Turbo Pascal menjanjikan fleksibilitas pemakaian secara optimal penggerak tampilan yang ada.

II.2.3. Peranti Masukan

Instruksi-instruksi yang akan dikirimkan ke komputer membutuhkan peralatan masukan. Instruksi tersebut bisa berupa perintah untuk melakukan sesuatu yang dapat dilaksanakan komputer tergantung pada program/perangkat lunak yang digunakan. Hingga saat ini telah berkembang berbagai peranti masukan untuk maksud-maksud tertentu.

Dapat disebutkan disini misalnya *keyboard* atau papan ketik, *mouse*, *joystick, trackball, digitizer, scanner, touch screen* dan sebagainya. Berbagai peranti masukan di atas memiliki fungsi dan karakteristik sendiri tergantung penggunaannya. Sehubungan dengan pembahasan topik dalam Tugas Akhir ini, maka akan dibatasi pembahasan tentang *mouse* sebagai peranti masukan.

Jika pemakai komputer ingin memberikan instruksi kepada komputer untuk melaksanakan suatu aktivitas, maka umumnya instruksi tersebut perlu ditulis artinya diketikkan lewat papan ketik atau *keyboard*. Perkembangan lebih lanjut menunjukkan semakin kompleksnya keinginan para pemakai komputer dengan dipicu perkembangan pemrograman aplikasi grafis. Karenanya dibutuhkan suatu peranti masukan yang lebih interaktif. Maka *mouse* pun dikembangkan untuk memenuhi kebutuhan tersebut. Dengan *mouse*, rangkaian perintah atau instruksi kepada komputer dapat dilakukan

dengan sekali menekan tombol pada *mouse* sembari mengarahkan penunjuk *mouse* pada perintah yang diinginkan. Dengan kata lain terdapat efisiensi pemberian instruksi.

Dengan perkembangan konsep Antar Muka Grafik atau *Graphic User Interface (GUI)*, semakin didapatkan efisiensi dalam pemberian perintah untuk maksud tertentu. Efisiensi ini berpengaruh pada efektifitas kerja. Hal ini mungkin karena beberapa perintah yang pada waktu sebelumnya harus dituliskan/diketikkan dapat diringkas dalam bentuk gambar (*icon*). Misalnya untuk menyimpan file cukup dilakukan dengan menekan tombol *mouse* pada *icon* disket. Kehadiran *mouse* ini memberikan banyak kemudahan untuk mengembangkan antar muka grafik.

Sebuah perangkat *mouse* dapat diaktifkan untuk digunakan sebagai peranti masukan dengan memasang *driver* untuk *mouse*. Saat ini telah banyak dibuat *driver* untuk suatu *mouse*, tergantung pada pembuat *mouse*. Dengan demikian karakteristik *mouse* dapat dioptimalkan dengan menggunakan *driver* dari pembuat *mouse*. *Driver* ini dapat diaktifkan dalam lingkungan DOS maupun Windows. Khusus untuk Windows 3.xx maupun Windows 9x, telah memiliki kemampuan mengenali *mouse* tanpa harus memanggil *driver*nya.

II.3. Objek dalam Pemrograman Grafik

Pemrograman grafik sangat membutuhkan berbagai objek untuk ditampilkan guna berinteraksi dengan pemakai komputer. Objek dapat berupa gambar maupun

tulisan dengan bentuk yang lebih menarik. Banyak objek yang berupa gambar yang dapat dibuat dalam pemrograman grafik. Gambar ini dapat berupa bentuk geometris, maupun bentuk poligon yang mempunyai bentuk yang lebih beragam dibandingkan bentuk geometris. Berikut akan diuraikan tentang objek-objek dalam pemrograman grafik dengan Turbo Pascal.

II.3.1. Sistem Koordinat

Suatu objek dalam pemrograman grafik yang akan ditampilkan di monitor perlu diletakkan pada lokasi tertentu. Untuk ini dibutuhkan suatu sistem koordinat untuk menentukan pada posisi mana akan diletakkan. Sistem koordinat yang digunakan pada pemrograman grafik, sedikit berbeda dengan sistem koordinat Cartesius yang digunakan sehari-hari.

Pada sistem koordinat Cartesius, titik (0,0) terletak pada bagian kiri bawah suatu bidang gambar, sedangkan pada pemrograman grafik justru terletak pada posisi kiri atas bidang gambar (monitor). Sumbu X (absis) pada sistem Cartesius diletakkan mendatar ke kanan, sama dengan sistem koordinat pada pemrograman grafik sedangkan sumbu Y (ordinat) diletakkan tegak keatas, yang dalam pemrograman grafik tegak kebawah. Perbedaan ini perlu diketahui dan dipahami agar tidak membingungkan pada saat akan meletakkan suatu objek di layar tampilan.

Satuan ukur yang digunakan pada absis dan ordinat adalah titik dengan tingkat kerapatan tertentu (resolusi). Jumlah titik yang bisa ditampilkan lebih tergantung pada resolusi layar tampilan yang digunakan. Semakin tinggi resolusi suatu layar tampilan berarti semakin banyak titik yang bisa

ditampilkan dan berpengaruh pada kualitas objek yang akan ditampilkan. Kualitas tampilan di layar dapat diatur dengan menentukan *driver* yang dikehendaki, namun tetap tergantung pada kemampuan perangkat keras layar tampilan itu sendiri.

II.3.2. Titik dan Garis

II.3.2.1. Titik.

Titik merupakan elemen terkecil dari suatu objek grafik. Suatu layar penampil dapat membangkitkan suatu titik yang disebut sebagai piksel. Titik dalam pemrograman grafik dengan Turbo Pascal dibangkitkan berdasarkan data digital yang terdapat dalam pengingat digital. Nilai 0 berarti piksel dalam keadaan mati, dan nilai 1 menunjukkan bahwa piksel dalam keadaan hidup (Insap Santosa, 22).

Turbo Pascal menyediakan suatu prosedur yang dapat digunakan untuk membangkitkan suatu piksel atau titik pada layar penampil, yaitu prosedur `putPixel` yang selengkapnya sebagai berikut :

```
PutPixel(x, y : integer, warna : word);
```

dengan x : posisi mendatar (absis) piksel

y : posisi tegak (ordinat) piksel

warna: nomor atau nama warna pilihan untuk piksel yang dihidupkan.

Parameter (x,y) menunjukkan posisi piksel yang akan dihidupkan

pada layar penampil. Nilai x harus berada antara 0 hingga batas resolusi absis maksimum yang bisa diketahui dengan fungsi `getMaxX`, sedangkan nilai y harus berada antara 0 hingga batas maksimum ordinat yang dapat diketahui dengan fungsi `getMaxY`. Nilai maksimum untuk absis dan ordinat ini tergantung pada tingkat resolusi layar penampil yang digunakan.

II.3.2.2. Kursor Grafis

Untuk menggambar suatu objek grafik pada lokasi yang tertentu dibutuhkan suatu titik acuan. Titik acuan ini digunakan sebagai patokan yang dijadikan titik awal penggambaran/ pembuatan objek grafik. Karena merupakan titik acuan sehingga jika pembuatan objek grafik akan dilakukan di lokasi berbeda, maka perlu dilakukan pemindahan kursor grafis pada lokasi awal pembuatan objek. Pemindahan kursor grafis yang dikenal dengan istilah *current pointer* (CP) dilakukan dengan dua cara yaitu pemindahan secara absolut dan pemindahan secara relatif.

Pemindahan secara absolut dilakukan dengan langsung memberikan nilai koordinat untuk posisi CP yang baru tanpa terpengaruh oleh posisi CP sebelumnya. Pemindahan ini dilakukan

dengan prosedur **`MoveTo(x, y : integer).`**

dengan x dan y merupakan nilai untuk ordinat dan absis.

Jika pemindahan secara absolut tidak dipengaruhi oleh posisi awal CP maka pada pemindahan secara relatif dipengaruhi oleh posisi awal CP. Prosedur **`MoveRel(dx, dy : integer)`**. dx dan dy merupakan nilai tambahan koordinat pada koordinat CP semula, yang digunakan sebagai koordinat CP yang baru.

II.3.2.3. Garis

Suatu garis pada dasarnya tersusun dari kumpulan titik dengan jarak yang sangat dekat sehingga seolah-olah bukan tersusun dari kumpulan titik. Pembuatan garis dapat dilakukan dengan menggambar titik-titik yang berjarak sangat dekat. Masalah akan timbul adalah jika garis yang akan kita gambar letaknya membentuk sudut dengan sumbu x dan sumbu y. Hal ini dapat saja dilakukan tetapi garis yang dihasilkan tentulah tidak dapat rapi. Dalam Turbo Pascal tersedia tiga buah prosedur untuk membuat garis yaitu prosedur **`Line(x1,y1,x2,y2 : integer)`**, prosedur **`LineTo(x,y : integer)`**, dan prosedur **`LineRel(dx, dy : integer)`**.

Prosedur `Line`, membutuhkan masukan koordinat titik awal penggambaran yaitu (x1,y1) dan koordinat titik akhir penggambaran yaitu (x2,y2). Dengan prosedur ini maka dapat digambar sembarang garis yang memenuhi syarat diatas yaitu koordinat awal dan akhir penggambaran. Prosedur `Line` ini tidak akan mempengaruhi

keberadaan kursor grafis (CP).

Dalam prosedure `LineTo`, peranan kursor grafis sangat penting karena posisi CP berguna sebagai titik awal penggambaran. Sehingga prosedur `LineTo` tidak membutuhkan masukan koordinat titik awal penggambaran karena mengacu pada posisi CP. Yang dibutuhkan untuk prosedur `LineTo` hanyalah koordinat akhir penggambaran yaitu (x,y). Koordinat (x,y) ini akan memindahkan CP pada posisi (x,y) yang selanjutnya menjadi titik awal penggambaran berikutnya bila tidak ada pemindahan CP dengan menggunakan prosedur `MoveTo`.

Prosedur `LineRel` digunakan untuk menggambar garis dari posisi CP sekarang dan menambahkan dengan parameter perubahan pada sumbu X dan sumbu Y dihitung dari posisi CP. Dengan demikian maka parameter (dx, dy) bukan merupakan koordinat titik akhir penggambaran melainkan nilai yang ditambahkan pada posisi CP sekarang (x, y) untuk mendapatkan koordinat titik akhir penggambaran. Prosedur ini pun akan mengubah posisi CP ke posisinya yang baru dengan memperhatikan parameter (dx, dy) yang dipakai.

Ketebalan, corak dan pola penggambaran garis yang dihasilkan oleh salah satu prosedur penggambaran garis di atas dapat diatur dengan prosedur **`SetLineStyle(corak, pola,`**

`tebal : word);`. Prosedur ini diaktifkan sebelum prosedur penggambaran garis di atas.

Untuk parameter `Corak`, tersedia mulai 0 hingga 4, yang mempunyai arti; 0 berarti `SolidLn`, 1 berarti `DottedLn`, 2 berarti `CenterLn`, 3 berarti `DashedLn`, dan 4 berarti `UserLn`. `SolidLn` menggambar garis lurus, `DottedLn` menggambar garis titik-titik, `CenterLn` akan menghasilkan gambar garis dengan corak titik-garis, sedangkan `DashedLn` akan menggambar garis dengan corak garis putus-putus. Corak 4 merupakan corak yang dapat ditentukan sendiri.

Parameter Pola, hanya akan berpengaruh bila digunakan corak 4 yaitu `UserLn`. Jika corak yang digunakan bukan 4 maka pola akan dapat diisi dengan nilai 0. Pola garis dapat dibentuk dengan menentukan pola 16-bit, yaitu bilangan heksadesimal, yang dapat diperoleh setelah mengkonversi bilangan biner yang menjadi pola garis. Ketebalan garis ditentukan dengan menentukan nilai untuk parameter Tebal. Dalam Turbo Pascal dikenal 2 jenis ketebalan garis yaitu `NormalWidth` yang bernilai 1, dan `ThickWidth` yang bernilai 3.

Garis yang dihasilkan dapat pula diberi warna garis tertentu. Pengaturan warna garis dilakukan dengan menulis prosedur **`SetColor(warna : word);`** parameter warna pada prosedur

ini dapat dipilih dengan melihat tabel berikut:

Tabel. II.2. Nama dan nomor warna pada palet standar yang berlaku pada penggerak grafik VGA.

No.	Nama Warna	No.	Nama Warna
0	Black (Hitam)	8	DarkGray (Abu gelap)
1	Blue (Biru)	9	LightBlue(Biru terang)
2	Green (Hijau)	10	LightGreen(Hijau terang)
3	Cyan (Biru telur)	11	LightCyan (Biru telur terang)
4	Red (Merah)	12	LightRed (Merah terang)
5	Magenta (Ungu)	13	LightMagenta (Ungu terang)
6	Brown (Cokelat)	14	Yellow (Kuning)
7	LightGray(Abu terang)	15	White (Putih)

II.3.3. Bentuk Geometris

Dalam pemrograman grafik diperlukan berbagai bentuk geometris yang dapat digunakan untuk mencapai tujuan program yang dibuat. Bentuk geometris adalah areal yang tertutup oleh batas-batas tertentu. Batas disini dapat berbentuk garis lurus atau lengkungan/kurva. Bentuk-bentuk geometris tersebut antara lain; empat persegi panjang, lingkaran dan elips. Bentuk geometris ini dapat dicat dengan warna cat tertentu maupun pola arsiran yang akan diuraikan pula pada bagian ini.

II.3.3.1. Empat persegi panjang.

Bentuk geometris ini tergolong bentuk yang sederhana karena terbentuk hanya dari dua pasang garis yang saling sejajar satu sama lain. Sebenarnya empat persegi panjang dapat saja dibentuk dari

prosedur penggambaran garis yang telah diuraikan di atas. Namun hal ini akan membuat rumit proses yang harus dilalui, karena harus digambar empat garis, sedangkan dengan prosedur empat persegi panjang akan menjadi lebih mudah dan sederhana.

Kesederhanaan lainnya tampak dari prosedur untuk membentuk empat persegi panjang yaitu prosedur **`Rectangle(x1, y1, x2, y2 : integer);`**. Titik (x1, y1) adalah koordinat untuk titik sudut kiri atas dari empat persegi panjang yang akan dibuat. Sedangkan titik (x2, y2) adalah koordinat titik sudut kanan bawah dari empat persegi panjang tersebut.

Gambar yang terbentuk haruslah dapat ditampilkan oleh layar penampil. Untuk itu perlu diingat bahwa nilai x1, x2, y1, dan y2 harus memenuhi syarat berikut :

0 <= x1 < x2 < `GetMaxX` dan 0 <= y1 < y2 < `GetMaxY`.

Bentuk yang dihasilkan dari prosedur `Rectangle` ini hanyalah sebuah empat persegi panjang. Persoalan timbul bila objek yang akan digambar adalah belah ketupat, trapesium, jajaran genjang dan segi banyak. Jika menggunakan prosedur `Rectangle` tidak dapat menghasilkan bentuk-bentuk diatas. Untuk itu diperlukan suatu prosedur lain yaitu `DrawPoly`. Prosedur ini dapat menghasilkan bentuk geometris segi banyak.

```
DrawPoly(cacahtitik:word;var titiksudut);
```

dengan cacahtitik: banyaknya titik dari segi banyak yang akan dibuat.

titiksudut : koordinat titik-titik sudut segi banyak yang akan dibuat.

Prosedur `DrawPoly` ini tidak selalu menghasilkan segi banyak yang tertutup. Jika ingin dihasilkan segi banyak yang tertutup maka koordinat titik sudut akhir harus sama dengan koordinat titik sudut awal segi banyak tersebut.

II.3.3.2. Lingkaran

Untuk menggambar bentuk lingkaran Turbo Pascal menyediakan prosedur `Circle`, yang bentuk umumnya adalah

```
Circle(x, y : integer; jejari : word);
```

dengan (x, y) : koordinat titik pusat lingkaran

Jejari : jari-jari lingkaran yang akan dibuat.

Walaupun titik pusat lingkaran bernilai integer, namun tidak berarti bahwa dapat diisi dengan sembarang nilai negatif. Sebabnya jika titik pusat lingkaran yang akan dibuat disubtitusikan dengan nilai negatif maka titik pusat lingkaran akan berada diluar batas layar tampilan yang bernilai positif untuk sumbu x dan sumbu y. Memperhatikan keadaan tersebut maka parameter (x,y) diisi dengan bilangan bulat positif.

II.3.3.3. Elips

Elips adalah suatu bentuk geometris yang menyerupai

lingkaran yang pejal. Untuk menciptakan sebuah elips dibutuhkan dua buah jari-jari yaitu jari-jari tegak dan mendatar. Sedangkan lingkaran hanya membutuhkan satu jari-jari saja. Hal ini karena sebuah lingkaran dibentuk dengan jari-jari yang sama panjangnya untuk jari-jari mendatar maupun tegak. Maka dapat dikatakan sebuah lingkaran adalah elips yang mempunyai jari-jari mendatar dan tegak yang sama panjangnya.

Turbo Pascal menyediakan sebuah prosedur untuk membuat gambar elips. Bentuk umum prosedur tersebut adalah :

```
Ellipse(x,y : integer; SdtAwal, SdtAkhir :
     word; JariX, JariY : word);
```

dengan (x, y) : titik pusat elips

SdtAwal : sudut awal penggambaran elips

SdtAkhir : sudut akhir penggambaran elips

JariX : jari-jari mendatar sumbu x

JariY : jari-jari tegak sumbu y.

Sudut awal penggambaran dimulai dari sudut 0° yang terletak di sisi sebelah kanan sumbu mendatar ke arah kiri (berlawanan arah jarum jam) hingga 360°. Namun ini tidak berarti bahwa untuk mendapatkan bentuk elips yang utuh, sudut penggambaran harus dimulai dari 0°. Sudut penggambaran harus berjumlah 360°. Jika sudut awal dan sudut akhir penggambaran kurang dari 360° akan didapatkan bentuk busur dari suatu elips.

II.3.3.4. Mewarnai Gambar

Bentuk-bentuk geometris yang telah dibahas di atas masih berbentuk bidang kosong yang dibatasi oleh batas tertentu. Akan menjadi lebih menarik apabila diberi warna atau paling tidak arsiran yang menjadikan gambar lebih bervariasi. Jika pewarnaan yang diberikan pada bentuk-bentuk geometris tersebut memperhatikan sudut pencahayaan bisa didapatkan bentuk tiga dimensi.

Pemberian warna pada bentuk-bentuk geometris ini dapat dilakukan dengan menggunakan prosedur `SetColor` yang telah dibahas di atas. Selain menentukan warna yang akan digunakan untuk mewarnai bentuk geometris yang telah dibuat, perlu ditentukan pula pola atau model arsiran daerah tertutup tersebut.

Pola arsiran daerah tertutup dilakukan dengan menggunakan suatu prosedur yaitu prosedur `SetFillStyle` yang selengkapnya memiliki bentuk umum

`SetFillStyle(pola : word; warna :word);`.

dengan Pola : corak pola yang akan digunakan

Warna : warna yang akan dipakai pada pola

Pola yang disediakan dalam Turbo Pascal dapat disubtitusikan nilainya berupa teks maupun angka ekivalen pola yang selengkapnya adalah :

`EmptyFill`	= 0; (tak bercorak/polos)
`SolidFill`	= 1; (corak penuh)
`LineFill`	= 2; (corak garis datas)

```
LtSlashFill     = 3; (corak garis miring tipis)
SlashFill       = 4; (corak garis miring tebal)
BkSlashFill     = 5; (corak backslash tebal)
LtBkSlashFill   = 6; (corak backslash tipis)
HatchFill       = 7; (corak kotak-kotak)
XHatchFill      = 8; (corak garis miring bersilang)
InterLeaveFill = 9; (corak jaring/jala)
WideDotFill     =10; (corak titik-titik renggang)
CloseDotFill    =11; (corak titik-titik rapat)
UserFill        =12;(corak yang ditentukan );
```

(Insap Santosa, 1996:94)

Warna dan pola yang telah ditentukan kemudian dapat diaplikasikan dalam bentuk-bentuk geometris dengan menggunakan prosedur

`FloodFill(x, y : word; warnabatas : word);`

dengan (x, y) : koordinat suatu titik yang terletak di dalam areal tertutup bentuk geometris yang bersangkutan

warnabatas : warna pembatas daerah tertutup yang akan diwarnai.

Yang dimaksudkan sebagai daerah tertutup dalam Turbo Pascal adalah areal benar-benar yang tertutup oleh satu atau lebih bentuk geometris dengan warna batas yang sama. Artinya jika daerah/areal tertutup yang akan diwarnai tersebut merupakan interseksi/irisan dari dua atau lebih bentuk geometris, maka warna garis pembatas dari bentuk-bentuk tersebut harus sama. Jika tidak maka perpotongan yang terjadi dianggap sebagai areal yang tidak

tertutup karena tidak memiliki warna batas yang sama. Demikian pula jika bentuk geometris yang akan diwarnai hanya satu objek tertutup maka warna garis batas objek tersebut pun harus sama dengan warna batas yang didefinisikan pada prosedur `SetFillStyle`.

II.3.4. Teks Grafik

Dalam Turbo Pascal disediakan fasilitas untuk penulisan teks-teks pada modus grafik. Untuk penulisan atau pembangkitan teks pada modus grafik dikenal dua bentuk teks grafik yaitu karakter *bit-mapped* dan karakter goresan (*stroked*). Terdapat perbedaan yang mendasar antara kedua jenis karakter tersebut.

Karakter *bit-mapped* merupakan karakter standar yang digunakan oleh Turbo Pascal untuk mencetak karakter dalam modus teks. Karakter yang dibangkitkan oleh Turbo Pascal dengan model *bit-mapped* menggunakan ukuran dan pola yang sama untuk semua karakter, sehingga dapat ditampilkan dengan lebih cepat. Ukuran karakter *bit-mapped* (pemetaan bit) menggunakan pola bit 8x8 dan setiap bit menampilkan sebuah piksel. Dengan mengubah-ubah nilai pada bit-bit tersebut maka akan diperoleh pola huruf tertentu. Karakter *bit-mapped* disimpan dalam bentuk bit yang sudah tertentu. Karena pola bit yang berukuran 8x8 tadi maka ukuran untuk tiap karakter yang ditampilkan pun memiliki ukuran yang sama.

Karakter jenis kedua adalah karakter yang dibangkitkan dengan metode

goresan. Dikatakan demikian karena tiap karakter yang termasuk dalam jenis ini dibangun dari kumpulan segmen garis. Semakin rumit bentuk karakter maka diperlukan lebih banyak garis untuk membentuk karakter tersebut. Ukuran tiap karakter jenis goresan ini berbeda-beda, sehingga bisa didapatkan bentuk yang bervariasi. Karena dibangun dari kumpulan segmen garis maka karakter goresan ini akan lebih baik tampilannya jika ditampilkan pada skala yang besar. Hal ini mungkin karena terjadinya perubahan koordinat penghubung tiap garis.

Turbo Pascal menyediakan kedua jenis karakter ini untuk digunakan sesuai kebutuhan pemrogram. Untuk teks yang memerlukan kualitas yang baik dengan skala yang lebih besar dapat digunakan karakter jenis goresan (*stroked font*), sedangkan untuk penulisan standar dapat digunakan karakter jenis *bit-mapped*.

II.3.4.1. Penulisan Teks Grafik

Ada beberapa langkah yang harus dilakukan sebelum melakukan penulisan menggunakan fasilitas karakter dalam Turbo Pascal, yaitu:

1. Daftarkan sembarang *font* yang diperlukan
2. Tentukan *style text*, arah penulisan, dan ukuran
3. Atur justifikasi teks
4. Tuliskan teks yang dimaksud. (Insap Santosa, 1996, 116).

Langkah pertama dibutuhkan untuk menyiapkan huruf yang akan digunakan dalam penulisan modus grafik yaitu dengan

mendaftarkannya pada unit `Graph`, agar dikenali pada saat akan digunakan. Hal ini perlu dilakukan terutama untuk jenis karakter *stroked-font* yang memiliki cara penyajian yang berbeda satu sama lainnya. Untuk jenis karakter *bit-mapped* langkah ini tidak perlu dilakukan karena telah ditentukan sendiri oleh Turbo Pascal secara standar. Sedangkan untuk jenis karakter goresan diperlukan data karakter yang tersimpan dalam file berakhiran .**CHR**. Untuk dapat menggunakan karakter tersebut perlu didaftarkan dengan menggunakan prosedur

```
SetTextStyle(namahuruf,arah,ukuran:word);
```

dengan nama huruf : jenis font goresan yang akan digunakan
arah : arah penulisan teks (mendatar atau tegak).
ukuran : ukuran karakter teks yang akan ditampilkan.

Nama huruf yang digunakan adalah jenis-jenis yang dimiliki Jenis huruf goresan yang telah tersedia dalam Turbo Pascal yaitu, `GothicFont`, `SansSerifFont`, `TriplexFont` dan `SmallFont`. Secara standar Turbo Pascal menggunakan huruf standar berjenis *bit-mapped* yaitu `DefaultFont`.

Arah penulisan yang dimaksud dalam prosedur ini yaitu teks yang dituliskan atau ditampilkan akan ditulis secara mendatar (*Horizontal*) atau tegak (*Vertical*). Dengan demikian hanya terdapat dua arah penulisan yaitu tegak atau mendatar. Ukuran yang

digunakan untuk menuliskan karakter pada mode grafik Satuan ukurnya adalah bilangan bulat positif. Pengaruh ukuran ini akan semakin jelas dampaknya pada karakter yang termasuk dalam golongan karakter bit-mapped, yaitu `DefaultFont`.

Langkah selanjutnya adalah untuk melakukan penulisan teks grafis adalah menentukan arah justifikasi (perataan teks). Yang dimaksud perataan teks adalah bagian yang menjadi patokan awal arah penulisan teks grafik yang didasarkan pada posisi CP saat itu. Justifikasi dalam Turbo Pascal terdiri dari perataan kiri, perataan kanan, dan perataan tengah. Dalam keadaan normal perataan teks yang digunakan adalah perataan kiri. Perataan teks dilakukan dengan menggunakan prosedur

```
SetTextJustify(mendatar,tegak:word);
```

dengan mendatar : arah tulisan mendatar, rata kiri, kanan, atau tengah

tegak : posisi penulisan terhadap posisi CP

Parameter **mendatar** terdiri dari konstanta :

```
LeftText    = 0;
CenterText  = 1;
RightText   = 2;
```

Selain dapat menggunakan konstanta dengan kata `LeftText`, `CenterText`, `RightText`, dapat pula digunakan konstanta angka 0 - 2 untuk melengkapi parameter **mendatar**.

Parameter **tegak** digunakan untuk menentukan posisi penulisan teks

terhadap posisi CP saat itu yang terdiri dari konstanta :

```
BottomText  = 0;
CenterText  = 1;
TopText     = 2;
```

Jika dilihat terdapat dua buah konstanta yang sama antara kedua parameter diatas yaitu pada konstanta `CenterText`. Untuk itu konstanta `CenterText` bisa digunakan untuk parameter **tegak** maupun **mendatar**.

Langkah terakhir adalah menuliskan atau menampilkan teks grafik yang telah disiapkan. Tipe data yang akan ditampilkan lewat prosedur ini adalah tipe data *string*. Ada dua prosedur untuk menampilkan teks grafik ke layar tampilan. Cara pertama menggunakan prosedur

`OutText(kalimat : string);`

dengan kalimat : teks yang akan ditampilkan.

Prosedur ini adalah prosedur menampilkan teks grafik yang paling sederhana. Artinya tidak memperhatikan posisi penempatan teks grafik, sehingga untuk mengatur letak teks grafik perlu digunakan prosedur **`MoveTo`**. Prosedur `MoveTo` ini akan memindahkan kursor grafis (CP) yang menjadi titik awal penulisan.

Prosedur berikutnya untuk menampilkan teks grafik adalah prosedur

`OutTextXY(x, y:integer;kalimat:string);`

dengan x, y : koordinat awal penulisan teks

kalimat : teks yang akan ditampilkan

Dengan ditentukannya koordinat awal penulisan (x,y) maka teks yang akan ditampilkan/ditulis dapat ditempatkan secara lebih luwes pada koordinat layar tampilan. Namun yang perlu diperhatikan bahwa kedua jenis karakter baik *bit-mapped* maupun goresan akan dipotong pada batas *viewport*. Karakter jenis *bit-mapped* akan dipotong pada hurufnya sesuai batas *viewport* dan sisanya tidak ditampilkan sedangkan jenis goresan akan dipotong kalimatnya pada batas *viewport* dan sisanya akan ditampilkan.

II.3.5. Perekaman Objek Grafik

Pemrograman grafik kadangkala membutuhkan proses untuk merekam citra atau objek grafik yang tampil pada layar tampilan. Hasil perekaman ini dapat digunakan untuk berbagai keperluan. Perekaman objek grafik yang dilakukan dapat digunakan sebatas kebutuhan saat program aplikasi dijalankan maupun direkam secara permanen pada media perekam sehingga dapat dipergunakan kembali di waktu lain.

Untuk keperluan sesaat, perekaman objek grafik dapat dilakukan di memori komputer melalui beberapa tahap berikut :

1. Menentukan ukuran objek yang akan direkam. Ukuran objek grafik dapat diketahui dengan menentukan koordinat titik sudut kiri atas objek dan koordinat titik sudut kanan bawah objek yang bersangkutan. Prosedur yang digunakan adalah :

`ImageSize(x1, y1, x2, y2 : integer);`

dengan (x1, y1) : koordinat titik sudut kiri atas objek

(x2, y2) : koordinat titik sudut kanan bawah objek

Dengan demikian objek yang akan direkam berada dalam suatu empat persegi panjang yang akan menentukan besarnya ukuran memori yang akan digunakan. Hasil keluaran prosedur ini dapat disubtitusikan ke dalam suatu variabel ukuran yang dapat digunakan pada prosedur `GetMem`.

2. Mengalokasikan ukuran perubah dinamis di memori. Setelah menentukan ukuran objek kemudian mengalokasikan perubah dinamis yang akan menyimpan objek grafik. Untuk itu diperlukan suatu prosedur yang menyiapkan perubah dinamis yaitu prosedur

   `GetMem(var PDinamis : pointer; Ukuran : word);`

   dengan PDinamis : nama perubah dinamis yang akan dialokasikan

   Ukuran : besarnya perubah dinamis (dalam byte) yang akan dialokasikan

   Isi parameter **Ukuran** diperoleh dari prosedur `ImageSize` yang menentukan ukuran objek.

3. Merekam objek grafik ke memori. Perekaman objek grafik yang telah ditentukan ukuran objeknya maupun ukuran perubah dinamis dilakukan dengan prosedur

   `GetImage(x1,y1,x2,y2 : integer; var Pdinamis^);`

   dengan (x1, y1) : koordinat titik sudut kiri atas objek

(x2, y2) : koordinat titik sudut kanan bawah objek

PDinamis : nama perubah dinamis yang telah dialokasikan

Pada prosedur ini penulisan perubah dinamis diakhiri dengan tanda (^) sebagai pengenal perubah dinamis.

Jika objek grafik yang direkam akan digunakan lagi pada waktu yang akan datang maka objek grafik yang ada perlu direkam secara permanen pada media penyimpan. Objek gambar yang direkam memiliki format bitmap artinya memiliki ukuran objek yang tetap.

Perekaman objek gambar ke media penyimpan harus melalui tahap-tahap diatas yaitu penentuan ukuran objek, mengalokasikan variabel dinamis, dan merekam objek grafik ke memori. Kemudian dilanjutkan dengan menjalankan prosedur-prosedur perekaman file ke media penyimpan. Prosedur tersebut telah mengikuti aturan pembuatan file dalam Turbo Pascal.

1. Menentukan nama file rekaman. Hal ini dilakukan dengan menggunakan prosedur :

```
Assign(var JenisFile:tipe;NamaFile:string)
```

dengan JenisFile : Jenis file yang akan dibuka (Text atau File)

NamaFile : Nama file yang merekam data objek grafik

Jenis file rekaman adalah tipe perekaman yang akan dilakukan yaitu berupa Text atau File. Jenis file bertipe Text, direkam dalam bentuk untai data objek berupa kumpulan koordinat titik yang merupakan elemen gambar, sedangkan jenis file bertipe File akan merekam objek

gambar berupa byte-byte image segmen layar seluas bidang grafik yang telah ditentukan dalam prosedur `ImageSize`. Format file terakhir ini dikenal dengan format file *untyped*. (Yoyok Adisetio Laksono, 159).

2. Membuka File. Setelah menentukan nama file yang akan digunakan, prosedur selanjutnya adalah membuka file untuk ditulisi yaitu

   `Rewrite(var JenisFile:tipe;[UkuranRecord:Word]);`

   dengan JenisFile : Tipe file yang ditentukan dalam Prosedur `Assign`

   UkuranRecord : Ukuran rekaman data untuk tipe file untyped.

   UkuranRecord perlu ditambahkan yang akan digunakan pada saat transfer file untuk jenis file *untyped*. Diisikan dengan nilai "1".

3. Merekam Objek Grafik. Perekaman objek grafik berjenis *untyped* ke media penyimpan dilakukan menggunakan prosedur

   `BlockWrite(var JenisFile:tipe;Pdinamis^:`
   `pointer;Ukuran : byte);`

   dengan JenisFile: Tipe file yang ditentukan dalam Prosedur Assign

   Pdinamis^ : Nama variabel dinamis

   Ukuran : Variabel yang menampung data ukuran besar objek

   Penulisan file dengan metode `BlockWrite`, akan merekam objek per *byte*.

4. Menutup File. Setelah perekaman/penulisan file selesai file terbuka di memori akibat prosedur Rewrite, perlu ditutup dengan prosedur

   `Close(JenisFile : tipe);`

   dengan JenisFile : Tipe file yang ditentukan dalam Prosedur `Assign`

5. Mengubah file *bit-mapped* ke format OBJ. File yang berisi rekaman objek grafik perlu diubah ke format OBJ agar dapat digabungkan ke dalam tubuh program. Untuk itu dilakukan dengan menggunakan file BINOBJ.EXE dari Turbo Pascal yang diketikkan dari prompt DOS. Bentuk perintahnya adalah :

   `BINOBJ namafile_gambar nama_object nama_public`

   dengan Namafilegambar: Nama File rekaman objek grafik
   NamaObjek : Nama untuk file OBJ yang dibuat
   NamaPublik : Nama yang akan digunakan dalam tubuh program

II.3.6. Menampilkan Rekaman Objek Grafik

Data rekaman objek grafik yang telah direkam ke dalam media penyimpan dalam format OBJ, dapat dipanggil atau diaktifkan dalam program. Untuk itu yang perlu diperhatikan adalah Nama File OBJ dan Nama Publik yang telah ditentukan dalam perintah BINOBJ. Berikut langkah-langkah yang diperlukan untuk proses menampilkan rekaman objek grafik.

1. Menentukan variabel penampung data bertipe pointer. Hal ini ditentukan pada penentuan variabel di awal program dengan menyebutkan nama variabel yang digunakan.

```
Var  NamaVariabel,...    : pointer;
```

2. Mengaktifkan file data. File data rekaman objek diaktifkan atau dipanggil menggunakan suatu prosedur dengan deklarasi eksternal. Prosedur deklarasi eksternal ini berfungsi menghubungkan prosedur atau fungsi yang dikompilasi secara terpisah (lewat perintah BINOBJ) yang menghasilkan format *assembly*. Prosedur ini berisi instruksi untuk menghubungkan (link) nama file data rekaman dengan subprogram/unit yang sedang aktif dan akan disertakan dalam proses kompilasi. File data ini harus memiliki format .OBJ

```
Procedure NamaProsedur; external;
   {$L NamaFileOBJ};
```

3. Menampilkan rekaman data. Hal ini dilakukan melalui suatu prosedur yang dapat diciptakan untuk melakukan beberapa proses yaitu :

```
Procedure NamaProsedur;
Begin
   VariabelPointer := namafile_OBJ;
   PutImage(x,y:integer;VariabelPointer^;
            Operator);
End;
```

dengan

(x, y) : Koordinat letak objek grafik

VariabelPointer^ : Nama variabel yang digunakan menampung data

Operator : Jenis peletakan objek grafik di layar

Operator yang digunakan dalam Turbo Pascal dikenal 5 jenis operator peletakan objek grafik yang dapat dilihat pada tabel berikut :

Tabel II.3. Konstanta cara penampilan kembali citra.

Nilai Konstanta	Nama Konstanta	Keterangan
0	CopyPut	Menampilkan citra seperti apa adanya
1	XorPut	Setiap bit dari citra yang ada di layar dan citra yang akan ditampilkan ulang dioperasikan berdasar operasi logika XOR (exclusive OR)
2	OrPut	Setiap bit dari citra yang ada di layar dan citra yang akan ditampilkan ulang dioperasikan berdasar operasi logika OR
3	AndPut	Setiap bit dari citra yang ada di layar dan citra yang akan ditampilkan ulang dioperasikan berdasar operasi logika AND
4	NotPut	Citra yang akan ditampilkan ulang secara inverse video

(Insap Santosa, 224)

II.4. Interaksi Pemakai dengan Komputer

Aplikasi yang mengandalkan antarmuka grafis (Graphics User Interaface) selalu memperhitungkan interaksi dengan pemakai komputer. Interaksi ini terjadi karena program yang dibuat ditujukan untuk terciptanya komunikasi dua arah. Artinya komputer (program) menawarkan pilihan dan pemakai memberi masukan lewat berbagai media interaksi. Komunikasi yang terjadi tidak hanya berupa masukan perintah namun juga bisa berupa data yang harus diolah program yang bersangkutan sesuai tujuan program aplikasi yang dibuat.

II.4.1. Proses Interaksi

Dalam program pelatihan berbasis komputer interaksi dari pemakai

semakin nampak karena tujuannya memberi pengetahuan ataupun solusi tertentu terhadap permasalahan yang muncul. Proses interaksi yang terjadi bisa berujud pemilihan terhadap menu pilihan yang ditawarkan dalam program. Pemilihan ini dilakukan lewat media interaktif yang tersedia misalnya papan ketik, *mouse*, dan sebagainya. Proses yang lain misalnya memberi masukan ke program aplikasi untuk diolah oleh program yang bersangkutan.

Pemilihan menu menggunakan media interaktif ini muncul karena tersedianya menu pilihan untuk proses jalannya program yang dibuat. Dengan adanya pilihan-pilihan menu maka pemakai dapat menentukan keinginannya untuk mengikuti keseluruhan program dari awal sampai akhir program melalui pemilihan menu yang tersedia. Jika program memerlukan masukan data yang akan diolah, maka sekali lagi diperlukan interaksi dengan pemakai program.

Perintah atau menu yang ada umumnya diwakili oleh teks atau gambar yang mudah diingat dan dipahami. Penyajian menu ini sedapat-dapatnya menggunakan sedikit teks atau gambar sehingga mempersempit arti dari menu tersebut. Hal ini akan menghindarkan pemakai dari pengertian yang bias terhadap menu tersebut sehingga program yang dibuat dapat mencapai tujuannya.

II.4.2. Komponen Interaksi

Seperti telah diungkapkan di atas, proses interaksi memerlukan komponen interaksi yang mendukung proses interaksi. Dapat disebutkan

disini antara lain papan ketik, *mouse*, joystick, touch screen, digitizer, dan light pen. Agar pembahasan tidak meluas maka dibatasi hanya pada peranti interaksi yang mendukung program Tugas Akhir ini.

II.4.2.1. Papan Ketik.

Papan ketik digunakan untuk memasukkan data teks atau angka yang diperlukan oleh program aplikasi yang dibuat. Selanjutnya masukan tersebut akan diolah sesuai tujuan program. Selain itu papan ketik dapat pula digunakan untuk menentukan pilihan menu yang tersedia berupa huruf atau kode pilihan menu yang tersedia.

II.4.2.2. Mouse.

Media interaktif lain yang sering digunakan dalam program antarmuka grafis adalah *mouse*. Seperti papan ketik, *mouse* juga merupakan media masukan namun dalam wujud yang lebih khas sehingga masukan datanya pun memiliki kekhasan tersendiri. Wujud khas tersebut dari bentuk kursor, pengaktifan *mouse*, dan cara memberi masukan data. Selanjutnya akan diuraikan sebagai berikut.

4.2.2.1. Pengaktifan *mouse*

Untuk menggunakan/mengaktifkan *mouse* sebagai media masukan diperlukan suatu penggerak/pengendali *mouse*. Penggerak ini akan menghubungkan komputer dengan perangkat *mouse* yang ada. Hubungan ini dimungkinkan karena tersedianya suatu koneksi melalui terminal komunikasi (*communication port*).

Penggerak/pengendali mouse ini mempunyai kekhususan sendiri-sendiri tergantung pada pembuat perangkat keras *mouse*. Namun demikian telah terdapat standarisasi terhadap pemakaian alamat-alamat di memori (*interrupt*) untuk penggunaan peranti *mouse*. Penggerak *mouse* ini akan menetap di memori (*resident*) selama komputer mendapat aliran listrik dan akan memantau seluruh aktifitas *mouse*. Aktifitas *mouse* ini nampak dari perubahan status *mouse*, misalnya bergerak ke koordinat tertentu di layar atau terjadi penekanan tombol *mouse* tertentu. Karena penggerak *mouse* ini selalu memantau setiap aktifitas *mouse* maka rutin pengoperasian *mouse* diletakkan dalam suatu perulangan (looping) secara terus menerus hingga terdapat masukan dari pemakai.

Program pengendali *mouse*, misalnya Microsoft Mouse datang dengan dua versi yaitu MOUSE.SYS dan MOUSE.COM,...harus diinstall sebelum kita bisa memanfaatkan mouse. ...dengan menuliskan MOUSE.SYS pada berkas konfigurasi yaitu CONFIG.SYS, dengan cara menuliskan `DEVICE = MOUSE.SYS`. Cara yang kedua adalah dengan menuliskan `MOUSE.COM` pada berkas AUTOEXEC.BAT. (Insap Santosa, 381)

4.2.2.2. Parameter Mouse

Interrupt dapat diasumsikan sebagai sebuah prosedur *build-in*, dan mempunyai beberapa argumen yang diperlukan untuk mengaktifkan *interrupt* tersebut. Untuk pengoperasian *mouse* digunakan *interrupt* 51 (atau $33 heksadesimal) yang pengoperasiannya membutuhkan sejumlah argumen untuk inisialiasi sebelum dilaksanakan. Argumen-argumen tersebut ditempatkan dalam *register* yang sudah mempunyai nama tetap, yaitu AX, BX, CX, dan DX. Sehingga diperlukan suatu prosedur khusus yang akan selalu mengamati nilai keempat *register* diatas (Insap Santosa, 381).

Untuk mengetahui keadaan *mouse* pada suatu saat, diperlukan prosedur untuk mengamati keempat register untuk *mouse*, kemudian mengaktifkan interrupt 51. Hasilnya kemudian dapat digunakan untuk kepentingan program. Misalnya mengetahui posisi *mouse*, atau mendeteksi penekanan tombol *mouse*.

```
Procedure Mouse(var M1,M2,
                M3,M4: Integer);
Var Regs : Registers;
Begin
    {inisialisasi register}
    With Regs Do
     Begin
         AX := M1;
```

```
        BX := M2;
        CX := M3;
        DX := M4;
    End;
    {aktifkan interrupt 51}
    Intr(51, Regs);
    {mengambil kembali isi register
    untuk diolah}
    With Regs Do
        Begin
            M1 := AX;
            M2 := BX;
            M3 := CX;
            M4 := DX;
        End;
        End; (Insap Santosa, 382).
```

Dari program di atas bisa dilihat bahwa terdapat empat buah parameter yang diperlukan untuk mengendalikan *mouse*. Kegunaan masing-masing parameter adalah :

- M1, digunakan untuk mengatur status mouse, misalnya untuk menyembunyikan atau menampakkan kursor mouse atau menempatkan pada posisi tertentu.
- M2, digunakan untuk menentukan tombol mana yang ditekan. Jika mouse memiliki 2 tombol maka M2 = 0 untuk tombol kiri dan M2 = 1 untuk tombol kanan.
- M3, digunakan untuk menyatakan posisi horizontal kursor *mouse*
- M4, digunakan untuk menyatakan posisi vertikal kursor mouse.

4.2.2.3. Bentuk kursor mouse.

Ujud interaksi pengguna dengan komputer yang nampak adalah adanya penunjuk *mouse* (*mouse pointer*/kursor) berbentuk anak panah. Bentuk ini adalah bentuk standar yang dikenali secara umum oleh pengguna antar muka grafis. Kursor *mouse* ini dapat digerakkan mengitari seluruh areal layar. Sebenarnya hal ini merupakan suatu proses animasi yaitu menggerakkan objek grafik yang dalam hal ini adalah gambar panah.

II.4.2.2.3.1. Anatomi kursor mouse

Suatu kursor *mouse* apapun bentuknya tersusun dari dua bagian yaitu citra (*mask*) yang menunjukkan bentuk kursor *mouse* yang bisa digerakkan seluas layar, dan bagian dari citra yang disebut *hotspot* yang dipakai untuk mendeteksi posisi piksel saat tombol *mouse* ditekan. (Insap Santosa, 392).

Komponen-komponen kursor mouse bisa dirinci sebagai berikut :

- Topeng layar (*screen mask*)
- Topeng kursor (*cursor mask*)
- PosisiX, yang menunjukkan posisi bit pada arah mendatar dari kursor *mouse* yang dipakai

sebagai *hotspot* pada arah mendatar

- PosisiY, yang menunjukkan posisi bit pada arah tegak dari kursor *mouse* yang dipakai sebagai *hotspot* para arah tegak.

4.2.2.3.2. Mengubah bentuk kursor mouse

Seperti ditulis di atas bentuk standar kursor *mouse* adalah panah yang merupakan matriks piksel berukuran 16 x 16 piksel. Tiap piksel membutuhkan 1 bit untuk perekamannya sehingga setiap baris memerlukan 16 bit atau 2 byte, yang bisa dinyatakan sebagai kombinasi 4 buah digit heksadesimal. (Insap Santosa, 392).

Memperhatikan matriks piksel untuk kursor *mouse*, maka bisa dibuat suatu bentuk kursor *mouse* yang bisa disimpan dalam bentuk rekaman sebagai berikut :

```
Type NamaKursor = record
     TopengLayar : array[0..15] of
                   word;
     TopengKursor: array[0..15] of
                   word;
     PosisiX, PosisiY : integer;
End;
```

Elemen TopengLayar dan TopengKursor harus

bertipe word. Karena tipe `word` digunakan untuk menyimpan data bilangan bulat sepanjang 2 *byte*.

Setelah menyusun pola kursor *mouse*, maka dengan menggunakan struktur rekaman bentuk kursor **NamaKursor**, dapat dibuat bentuk kursor *mouse* yang dituliskan menggunakan format berikut :

```
Const BentukKursor : NamaKursor =
      (TopengLayar : ($nnnn, …);
       TopengKursor : ($nnnn,…);
      PosisiX : $nnnn;
      PosisiY : $nnnn);
```

dengan $nnnn : digit heksadesimal pola kursor *mouse*

PosisiX : nomor kolom hotspot

PosisiY : nomor baris hotspot