PEMBUATAN KURVA HORISONTAL DAN KURVA VERTIKAL GEOMETRI JALAN MENGGUNAKAN BAHASA PEMROGRAMAN VISUAL BASIC 6.0

TUGAS AKHIR

Disusun Oleh :

ADEN KUSNO W

00.25.019

JURUSAN TEKNIK GEODESI S-1

FAKULTAS TEKNIK SIPIL DAN PERENCANAAN

INSTITUT TEKNOLOGI NASIONAL

MALANG

2006

LEMBAR PENGESAHAN I

PEMBUATAN KURVA HORISONTAL DAN KURVA VERTIKAL GEOMETRI JALAN MENGGUNAKAN BAHASA PEMROGRAMAN VISUAL BASIC 6.0

TUGAS AKHIR

Diajukan sebagai Syarat Menyelesaikan Program Pendidikan Sarjana
Strata Satu (S1) Teknik Geodesi

Oleh :

Aden Kusno Wijoyo

00.25.019

Menyetujui

Dosen Pembimbing I	**Dosen Pembimbing II**
(Ir. Agus Darpono, MT)	**(Ir. Rinto Sasongko, MT)**

Mengetahui

Ketua Jurusan Teknik Geodesi S-1

INSTITUT TEKNOLOGI NASIONAL FAKULTAS TEKNIK SIPIL DAN PERENCANAAN JURUSAN TEKNIK GEODESI MALANG

26/9 '06

(Hery Purwanto, ST., MSc)

LEMBAR PERSEMBAHAN

TUHAN ITU BAIK KEPADA SEMUA ORANG , DAN PENUH RAHMAT TERHADAP SEGALA YANG DIJADIKANNYA

MAZMUR 145 : 9

10 HAL TERPENTING DALAM HIDUP :

1. ***Cinta***

   Perasaan khusus yang membuat kita merasa hangat dan indah.

2. ***Hormat***

   Perlakukan orang lain sebagaimana kita ingin diperlakukan orang lain

3. ***Penghargaan***

   Bersyukur akan segala sesuatu yang ada

4. ***Kebahagiaan***

   Selalu menikmati hidup dengan wajah penuh senyum

5. ***Maaf***

   Kemampuan melepaskan hal – hal tanpa kemarahan atau dendam

6. ***Berbagi***

   Senang berbagi tanpa mengharapkan imbalan

7. ***Kejujuran***

   Selalu berkata yang benar

8. ***Integritas***

   Selalu melakukan hal yang benar apapun resikonya

9. ***Kasih Sayang***

   Siap membantu dalam suka dan duka

10. ***Kedamaian***

    Hal yang akan kita capai bilamelakukan ke – 9 hal di atas

KATA PENGANTAR

Dengan mengucapkan puji syukur kepada Tuhan Yesus Kristus Anak Allah yang Tunggal yang telah melimpahkan kasih dan AnugerahNya sehingga penulis dapat menyelesaikan laboran Tugas Akhir ini. Tugas Akhir ini merupakan salah satu syarat dalam mencapai gelar Sarjana Teknik Geodesi Facultas Teknik Sipil dan Perencanaan Institut Teknologi Nasional Malang.

Pada kesempatan ini, penulis juga berterima kasih kepada semua pihak yang telah banyak membantu dalam hal bantuan, bimbingan, dorongan dan semangat, sehingga laboran Tugas Akhir ini dapat diselesaikan sesuai dengan tujuan yang telah diharapkan.

Oleh karena itu, penulis tak lupa ingin mengucapkan terima kasih yang sebesar – besarnya kepada :

1. Bapak Hery Purwanto, ST.,MSc selaku Ketua Jurusan Teknik Geodesi.
2. Bapak Ir. Agus Darpono, MT selaku dosen pembimbing I, yang telah banyak membantu dalam bimbingan dan penulisan skripsi.
3. Bapak Ir. Rinto Sasongko, MT selaku dosen pembimbing II yang telah membantu baik dalam pemberian judul maupun dalam pengarahan atau bimbingan.
4. Bapak Siswodarsono, yang telah memberikan penjelasan tentang gambaran dari penelitian ini di lapangan nantinya.

5. Para dosen – dosen teknik Geodesi ITN, yang telah memberikan materi perkuliahan dengan baik, sehingga bermanfaat dalam melaksanakan penelitian.
6. Teman – teman Teknik Geodesi yang telah banyak membantu baik dalam sarana dan prasarana ataupun masukan – masukan yang telah diberikan.

Penulis menyadari bahwa dalam penyusunan laboran tugas akhir ini banyak sekali terdapat kekurangan, untuk itu saran dan kritik sangat membantu demi sempurnanya laboran tugas akhir ini.

Malang, September 2006

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

I.1. Latar Belakang

Jaringan jalan raya yang merupakan prasarana transportasi darat memegang peranan yang sangat penting dalam sektor perhubungan terutama untuk kesinambungan distribusi barang dan jasa. Keberadaan jalan raya sangat diperlukan untuk menunjang laju pertumbuhan ekonomi seiring dengan meningkatnya kebutuhan sarana transportasi yang dapat menjangkau daerah – daerah terpencil.

Perkembangan kapasitas maupun kuantitas kendaraan yang menghubungkan kota – kota antar propinsi dan keterbatasan dana untuk pembangunan jalan raya serta belum optimalnya pengoperasian prasarana lalu – lintas yang ada, merupakan persoalan utama di Indonesia terutama negara – negara yang sedang berkembang.

Untuk membangun ruas jalan baru, maupun peningkatan yang diperlukan sehubungan dengan penambahan kapasitas jalan raya, tentu akan memerlukan metode efektif dalam perancangan maupun perencanaan agar diperoleh hasil yang terbaik dan ekonomis, tetapi memenuhi unsur keselamatan pengguna jalan.

Dalam perencanaan jalan raya, salah satu unsur yang sangat penting adalah perencanaan geometrik. Peraturan perencanaan geometrik ini dibuat atau disusun oleh panitia yang dibentuk berdasarkan Surat Perintah Direktur Jenderal Bina Marga tanggal 6 Mei 1970 nomor 64/SPRIN/BM/1970, dan telah ditetapkan menjadi peraturan yang resmi dengan Surat Putusan Direktur Jenderal Bina Marga No.25/Kpts/Bm/70, tanggal 31 Agustus 1970. Perencanaan geometrik jalan merupakan perencanaan route dari suatu ruas jalan secara lengkap, meliputi beberapa elemen yang disesuaikan dengan kelengkapan data dasar atau data yang telah dianalisis.

Dari uraian di atas, maka dengan memanfaatkan perkembangan teknologi komputer yang ada, terutama bahasa pemrograman Visual Basic 6.0 yang dapat dihubungkan dengan aplikasi – aplikasi yang lain, misalnya Autocad, Excel, dll,

maka dapat dibuat suatu program atau aplikasi yang mudah untuk dioperasikan, sehingga pengolahan data – data dapat diolah dengan mudah dan cepat.

I.2. Maksud dan Tujuan Penelitian

Maksud dari penelitian ini adalah :

1. Membuat perencanaan jalan secara digital.
2. Membuat suatu paket program untuk pengolahan dan penggambaran data – data perencanaan jalan raya.

Tujuan dari penelitian ini adalah :

Membuat program untuk pengolahan data – data perencanaan geometri jalan,desain badan jalan dan stacking out, dimana outputnya berupa tabel, dan gambar yang dapat diexpor ke program lain, seperti : Autocad, Excel, Acces.

Dengan adanya hasil penelitian pembuatan software ini diharapkan dapat lebih membantu para praktisi untuk menghitung (memproses data) dan membuat perencanaan desain kurva horisontal, kurva vertikal dan diagram superelevasi secara digital.

I.3. Identifikasi Masalah

Identifikasi masalah ini mengacu pada data – data input perencanaan geometrik, misalnya : jari – jari, kecepatan rencana, koordinat, jarak, jumlah staking out, kemudian bagaimana penggunaan bahasa pemrograman Visual Basic 6.0 dapat menghasilkan suatu program, dalam hal ini adalah program perencanaan geometrik jalan raya yaitu perhitungan stacking out yang meliputi : Kurva Horisontal (metode Selisih Busur, Polar, Poligon), Kurva Vertikal, Spiral, dan Diagram Superelevasi, kemudian hasil outputnya dapat diaplikasikan ke Autocad, Excel atau Acces.

I.4. Batasan Masalah

Untuk memberi kejelasan dari penelitian yang dilakukan, maka perlu adanya batasan – batasan masalah agar tidak menyimpang, yaitu :

1. Pengolahan data pengukuran antara lain :
   - Data Kurva Horisontal, meliputi : Full Circle dan Spiral Circle Spiral
   - Data Kurva Vertikal, meliputi : kurva cembung dan kurva cekung
2. Penentuan titik stacking out dari masing – masing perhitungan yaitu :
   - Busur Horisontal meliputi metode : Selisih Busur, Polar, Poligon pada pembuatan Full Circle dan Spiral Circle Spiral
   - Busur Vertikal
3. Bahasa pemrograman yang digunakan adalah Visual Basic 6.0 dan penggambarannya menggunakan Autocad 2004.
4. Hasil perhitungan dan penggambaran hanya menghasilkan output untuk satu lengkungan saja.
5. Paket program ini harus memiliki fasilitas Visual Basic 6.0 dan Autocad 2004.
6. Jalan yang akan direncanakan adalah jenis jalan tanpa median jalan.
7. Perencanaan geometri jalan raya yang dipergunakan pada penelitian ini menggunakan metode perencanaan di Dirjen Bina Marga Departemen Pekerjaan Umum Republik Indonesia.

I.5. Manfaat Penelitian

Manfaat hasil dari penelitian ini dapat digunakan para praktisi yaitu instansi terkait khususnya Dinas Pekerjaan Umum dapat membantu mempercepat dalam perencanaan geometri jalan raya dan para praktisi sebagai bahan acuan dalam melaksanakan praktikum.

Adapun manfaat lain dari hasil penelitian ini adalah kelebihannya, antara lain :

1. Menghasilkan program berbasis windows yang berfungsi untuk pengolahan data-data pengukuran di lapangan serta penggambaran pada Autocad 2004.
2. Berbasis windows sehingga mudah untuk dioperasikan oleh pemakainya.

3. Terintegrasi dengan program Autocad 2004 sehingga keuntungan, antara lain :
   a. Autocad 2004 adalah paket program umum yang sering digunakan dalam hal penggambaran hasil dari pengukuran di lapangan
   b. Hasil perhitungan dapat digambarkan langsung pada Autocad 2004 tanpa bantuan paket program lain dalam beberapa bentuk layer sehingga mudah untuk dilakukan perbaikan, layer tersebut antara lain :
      - Layer point, berisi gambar titik-titik hasil perhitungan
      - Layer ID, berisi gambar teks yang menggambarkan identifikasi dari titik hasil perhitungan
      - Layer garis, berisi gambar teks yang menggambarkan garis lengkung dan garis busur lingkaran dari titik hasil perhitungan
   c. Hasil penggambaran dapat dicetak dengan mudah.
4. Hasil dari perhitungan dapat dicetak langsung berupa tabel, ditampilkan pada monitor berupa tabel dan penggambaran pada Autocad 2004, serta dapat disimpan dalam bentuk file.
5. Input data sangat mudah dioperasikan serta sangat mudah untuk melakukan pengeditan dan penghapusan data maupun prosesing perhitungan ulang.

BAB II
DASAR TEORI

II.1. Alinemen

Pemanfaatan alinemen atau busur lapangan seringkali dijumpai pada proyek-proyek pembangunan atau pembuatan jalan raya, jalan kereta api, drainase, perencanaan jalur pipa dan sebagainya, digunakan untuk menghubungkan dua arah atau dua garis lurus yang berpotongan.

Ada dua macam alinemen yang berkaitan dengan perencanaan atau pekerjaan teknik sipil, yaitu :

1. Alinemen horisontal
2. Alinemen vertikal

Alinemen vertikal sangat berkaitan dengan daerah yang menurun dan tanjakan atau topografi wilayah yang dilalui jalur perencanaan. Sedangkan alinemen horisontal berkaitan dengan saluran yang menggunakan busur lingkaran sebagai belokan dari suatu pekerjaan konstruksi.

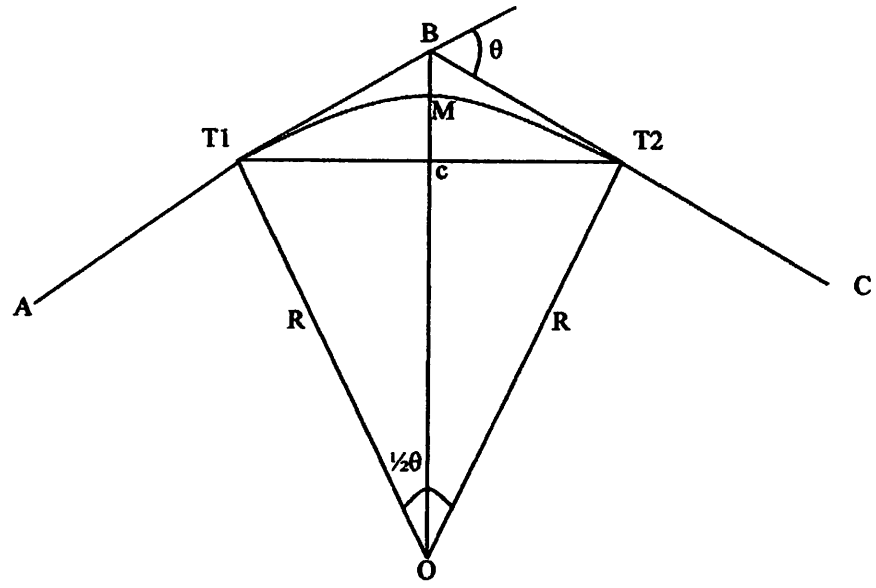
Bagian – bagian lurus tangen kebanyakan, jalur transportasi seperti jalan raya, baja, pipa, yang dihubungkan dengan lengkungan baik pada bidang horisontal maupun vertikal. Untuk lengkungan horisontal yang dipakai adalah busur lingkaran dan spiral.

Pemetaan ini bertujuan untuk meletakkan kembali semua rencana yang telah digambar di atas ke permukaan tanah. Jadi geometri lengkungan yang dibuat diatas peta dari daerah yang bersangkutan akan ditransformasikan kembali ke atas permukaan tanah. Pemetaan lengkungan ini banyak berkaitan dengan pekerjaan konstruksi jalan atau saluran, sehingga dengan sendirinya pelaksanaan pematokan (*stake out*) di lapangan sesuai dengan hasil penggambaran.

II.2. Alinemen Horisontal

Jenis alinemen yang sering digunakan dalam perencanaan dan mudah diterapkan dilapangan adalah busur lingkaran dan busur spiral. Diantara dua jenis busur tersebut, yang lebih sederhana dan yang paling sering digunakan adalah busur lingkaran, baik lengkungan tunggal maupun lengkungan majemuk. Jika

menggunakan busur lingkaran sebagai penghubung dua arah atau tangent, maka perlu diketahui adanya titik-titik utama yang sangat penting artinya bagi perencanaan atau pembuatan pematokan busur lingkaran tersebut.



Gambar 2.1.
Lengkung horizontal bentuk busur lingkaran

Keterangan gambar :

B = point of intersection

R = jari – jari

T_1, T_2 = titik tangent

O = titik pusat busur lingkaran

θ = sudut defleksi

M = titik tengah busur lingkaran

c = titik tengah tali busur lingkaran

Rumus yang digunakan untuk menghitung unsur – unsur parabola : (*Indr.S,1992*)

a. Busur lingkaran, $T_1 \cap T_2 = \frac{\theta}{360} \times 2\pi R$ ..... (1)

b. Panjang tangent, T1.B , T2B = R . tg ½ θ ..... (2)

c. Panjang tali busur, T1.T2 = 2 R sin ½ θ ..... (3)

d. Jarak B ke busur lingkaran, $BM = R\left[\frac{1}{\cos ½\theta} - 1\right]$ ..... (4)

Busur spiral adalah suatu busur peralihan yang menghubungkan dua busur lingkaran. Istilah peralihan dalam hal ini dimaksudkan untuk menyatakan perubahan jari – jari secara berangsur – angsur dari tak terhingga pada awal lengkungan sampai dengan jari – jari busur lingkaran yang bersangkutan.

II.3. Alinemen Vertikal

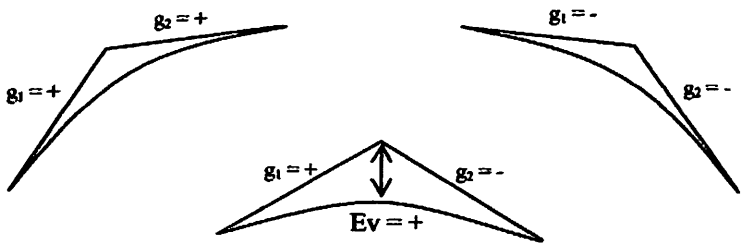
Alinemen Vertikal adalah bidang tegak yang melalui sumbu jalan atau proyeksi tegak lurus bidang gambar. Profil ini menggambarkan tinggi rendahnya jalan terhadap kemampuan kendaraan dalam keadaan naik dan dalam keadaan penuh.

Seperti halnya alinemen horizontal, fungsi alinemen vertikal ini untuk menghubungkan dua arah vertikal atau garis gradien (g) agar diperoleh perubahan yang smooth serta untuk menentukan elevasi titik pada lengkungan yang direncanakan. Pada setiap pergantian landai, harus dibuat suatu lengkung vertikal yang memenuhi syarat keamanan, kenyamanan, drainase dan keluwesan bentuk.

Ada dua jenis dasar lengkungan parabolik, yaitu cembung dan cekung. Jenis lengkungan cembung menurut definisi mengalami perubahan dalam gradien yang negatif yaitu lengkungan membelok ke bawah, sedangkan lengkungan jenis cekung mengalami perubahan positif serta lengkungan membelok ke atas.

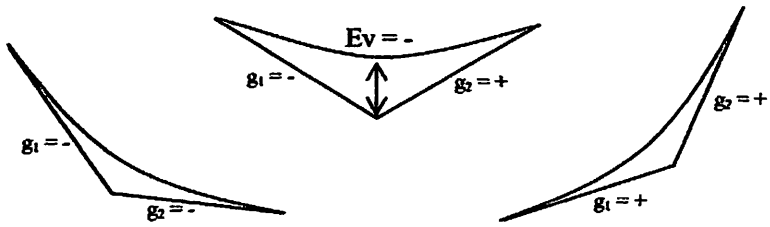
Macam – macam lengkung vertikal :

1. Lengkung Vertikal Cembung (bila PvI diatas permukaan)



Gambar 2.2.
Lengkungan vertikal cembung

2. Lengkung Vertikal Cekung (bila PvI di bawah jalan)



Gambar 2.3.
Lengkungan vertikal cekung

Alinemen vertikal dirancang untuk :

1. Menyesuaikan garis – garis gradien yang dihubungkan.
2. Mempunyai panjang yang cukup untuk memenuhi persyaratan – persyaratan yang meliputi angka perubahan gradien maksimum (yang mempengaruhi kenyamanan penumpang kendaraan)
3. Menghasilkan jarak pandang yang cukup untuk menjalankan kendaraan dengan aman.

Bentuk matematis lengkungan ini adalah :

$y = ax^2 + bx + c$ ..... (5)

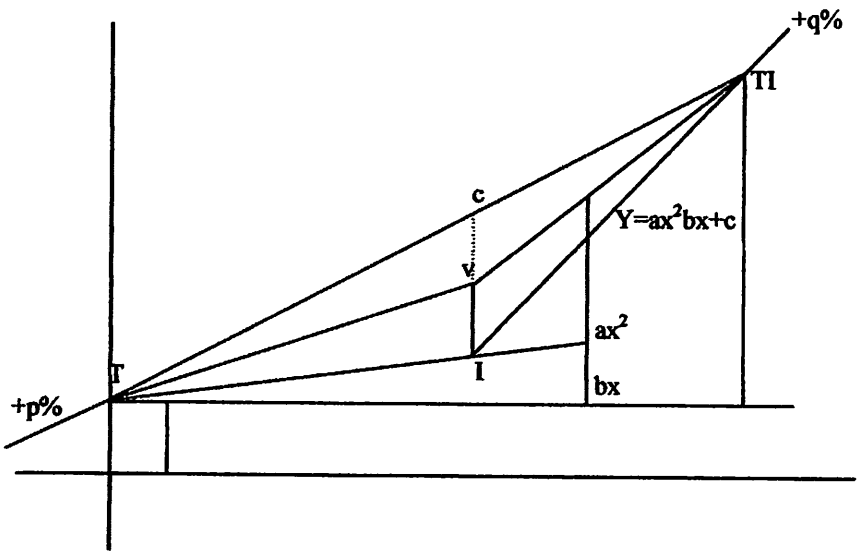
Keterangan :

y = Ketinggian (jarak vertikal) dari lengkungan di atas atau di bawah titik singgung pertama dari jarak x

x = Jarak yang bervariasi dan menyatakan jarak mendatar dari kedua titik singgung

a, b = Nilai koefisien dari x

c = Nilai konstanta



Gambar 2.4.
Lengkungan vertikal

Perhitungan untuk lengkung vertikal :

a. Elevasi tiap titik pada satu garis lengkung vertikal dicari dahulu data kemiringan / gradien garis tangent yang dinyatakan dalam % (persen)

$$g\% = \frac{H_{AB}}{D_{AB}} \times 100\% \quad \text{..... (6)}$$

dimana : g = kemiringan antara dua titik
H_{AB} = beda tinggi antara dua titik
D_{AB} = jarak antara dua titik

$$Ev = \frac{g_2 - g_1}{2L} \times (Xt)^2 \quad \text{..... (7)}$$

dimana : Ev = perubahan titik pada garis gradien ke elevasi lengkung vertikal
g = gradien
L = panjang lengkung vertikal
Xt = jarak titik detail yang dihitung dari titik awal lengkungan

b. Elevasi titik- titik detail setiap interval "X" (Hi)

$Hi = H_{awal} + (g_1.Xt) + E_v$ ..... (8)

Hi = Elevasi titik detail tiap interval x

H_{awal} = Elevasi awal lengkung parabola

g_1 = Gradien titik awal lengkungan

Xt = Jarak titik detail dari awal lengkungan

II.4. Metode Setting Out

Desain perencanaan pekerjaan konstruksi pada akhirnya ditentukan kembali posisinya di lapangan sesungguhnya (permukaan bumi fisis). Proses pemindahan dari gambar di atas peta ke permukaan bumi dinamakan setting out. Perhitungan untuk menentukan kembali posisi titik pada kurva horisontal dilakukan dengan menggunakan aljabar sederhana, ilmu ukur bidang datar dan trigonometri. Namun trigonometri yang digunakan hampir semuanya terbatas pada hubungan antara titik – titik dan sudut dari segitiga siku – siku.

Penentuan titik detail pada busur lingkaran dapat dilakukan dengan :

1. Metode selisih busur sama panjang
2. Metode koordinat polar
3. Metode poligon

Sedangkan untuk busur spiral, penentuan titik detail dapat dilakukan dengan metode sudut defleksi.

II.4.1. Penentuan Titik Detail Busur Lingkaran

Ada beberapa metode yang dapat digunakan untuk penentuan titik detail antara lain :

a. Metode Selisih Busur Sama Panjang

Keuntungan dari metode ini adalah titik – titik terletak teratur pada lengkungan. Dari perhitungan diperoleh unsur – unsur data lengkungan jari – jari (R), sudut defleksi (θ) dan panjang (L).

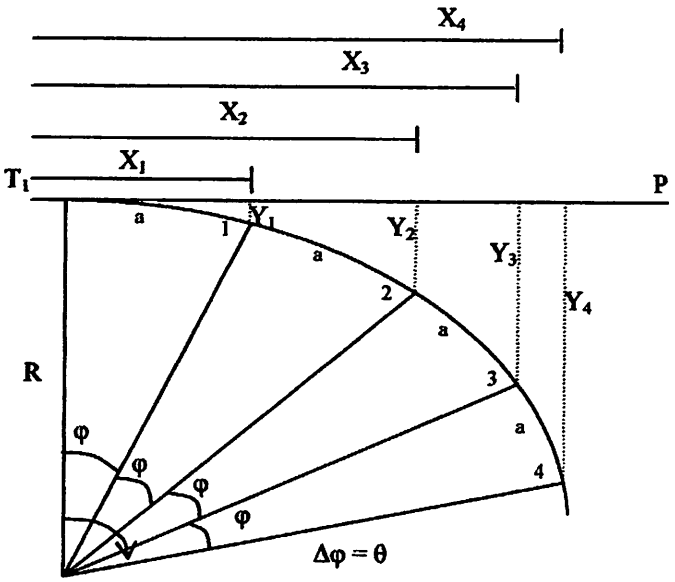
Misalkan : Banyaknya titik = n, maka panjang busur yang sama panjang (a) = $\frac{L}{n}$

Panjang busur a membentuk sudut φ sehingga : $\varphi = \frac{a}{R} \times \frac{360}{2\pi}$ .....(9)

Koordinat titik – titik pada salib sumbu garis tangent (T_1P) dengan garis yang tegak lurus pada (T_1P) adalah sebagai berikut :

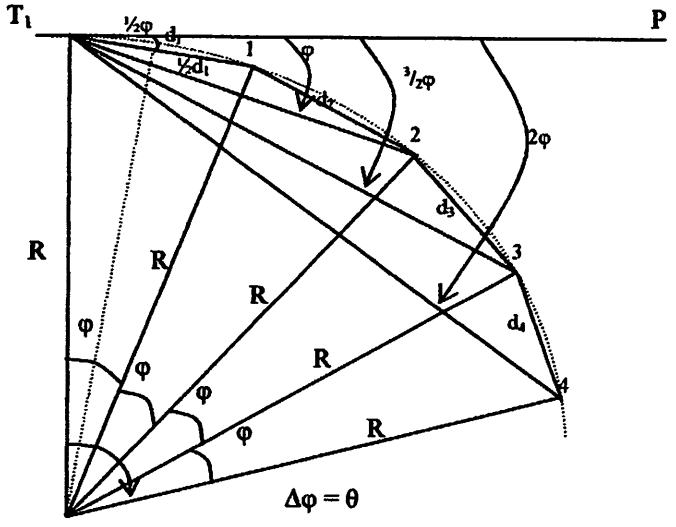
$$Xn = R \sin n\varphi \quad \text{..... (10)}$$

$$Yn = R - R \cos n\varphi = 2R \sin^2 \frac{n\varphi}{2} \quad \text{..... (11)}$$



Gambar 2.5.
Metode selisih busur sama panjang

b. Metode Koordinat Polar



Gambar 2.6.
Metode koordinat polar

Metode ini efisien untuk kurva yang berjari – jari besar. Penentuan titik dengan cara polar ini adalah pengukuran sudut ($\frac{\varphi}{2}$, φ, $3\frac{\varphi}{2}$, 2φ, ..., n$\frac{\varphi}{2}$) yang antara sudut satu dengan lainnya mempunyai selisih sama dimana : $\sin\frac{\varphi}{2} = \frac{k}{2R}$,

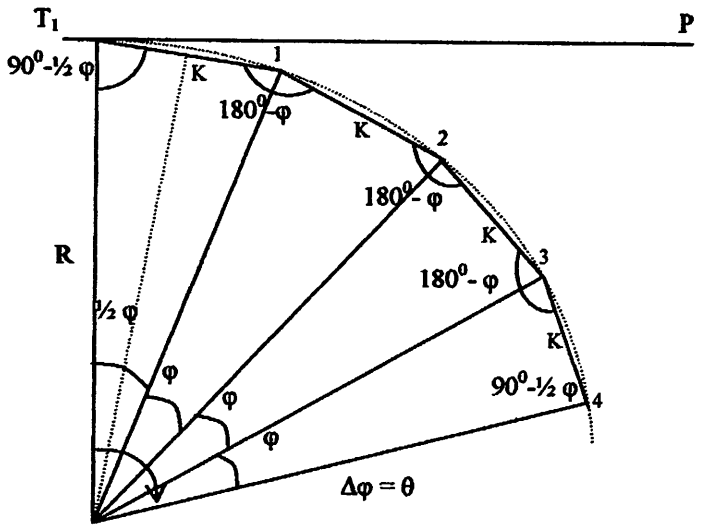
k = panjang tali busur atau jarak antara titik; atau $\varphi = \frac{\theta}{n}$.

Untuk n $\frac{\varphi}{2} = \frac{1}{2}\theta$ (merupakan syarat yang harus dipenuhi).

Dengan rumus jarak sebagai berikut : **dn = 2 R sin** $\frac{n\varphi}{2}$ ..... (12)

c. *Metode Poligon*

Metode ini digunakan apabila tempatnya sempit seperti terowongan dan cara mengukurnya disetiap titik pada busur lengkungan.



Gambar 2.7.

Metode Poligon

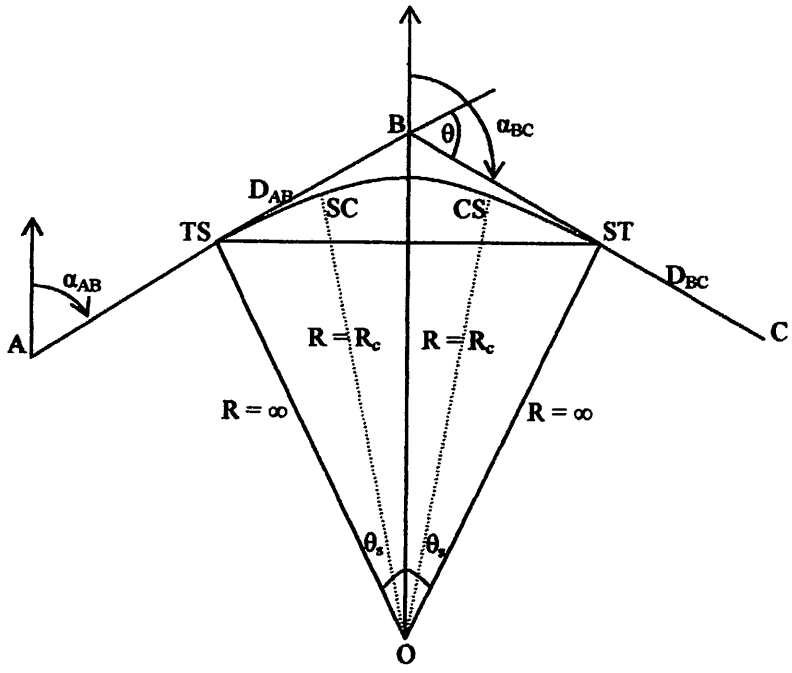
Dengan cara poligon, jarak antar titik – titiknya atau panjang tali busur adalah konstan (= k) dan sin $\frac{\varphi}{2} = \frac{k}{2R}$ (seperti pada metode koordinat polar).

Sudut – sudutnya adalah untuk sudut 1 = (90^0 - $\frac{\varphi}{2}$) dan sudut (2 .....n) = ($180^0 - \varphi$).

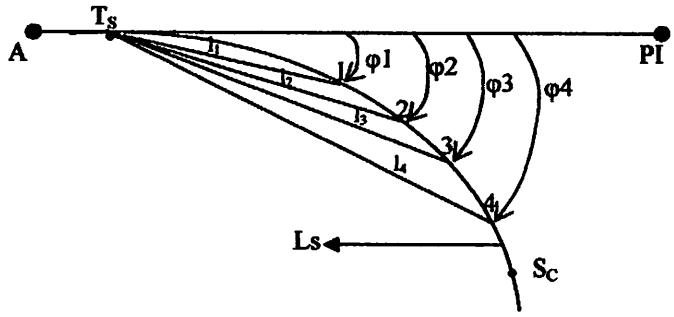
II.4.2. Penentuan Titik Detail Busur Spiral

Metode yang digunakan yaitu metode sudut defleksi. Dengan metode sudut defleksi ini, diperlukan data ukuran sudut dan jarak, dimana data tersebut harus dihitung dahulu dari data lengkungan yaitu Ls dan θs.

Lengkungan transisi lebih disukai terutama untuk jalan baja dan angkutan cepat, karena dapat mengurangi perubahan mendadak dalam lengkungan pada simpangan antara tangen (garis lurus dan lengkungan melingkar). Sebuah spiral merupakan lengkungan transisi yang amat baik karena jari – jarinya berkurang seragam dari tak terhingga pada tangen menjadi sebesar jari – jari lengkungan yang bersangkutan. Spiral dipakai untuk menghubungkan tangent dengan lengkungan melingkar, tangen dengan tangen (spiral ganda), dan lengkungan melingkar dengan lengkungan melingkar.



Gambar 2.8.
Bentuk busur Spiral



Gambar 2.9.
Penentuan titik detail menggunakan
Metode Sudut Defleksi

Data ukuran sudut dihitung sebagai berikut :

$$\Phi n = \frac{1}{3}\left[\frac{ln}{Ls}\right]^2 \theta s - Cs \quad \text{..... (13)}$$

Keterangan :

Φn = sudut lentur titik – titik detail = sudut defleksi spiral

ln = jarak titik Ts dengan titik detail

Ls = panjang busur spiral dari titik Ts ke titik Sc

θs = sudut spiral (dalam derajat); $\theta s = \frac{Ls}{2R} \times \frac{360}{2\pi}$ ..... (14)

Cs = koreksi spiral = $0.0031\ \theta s^3$ (dalam satuan detik)

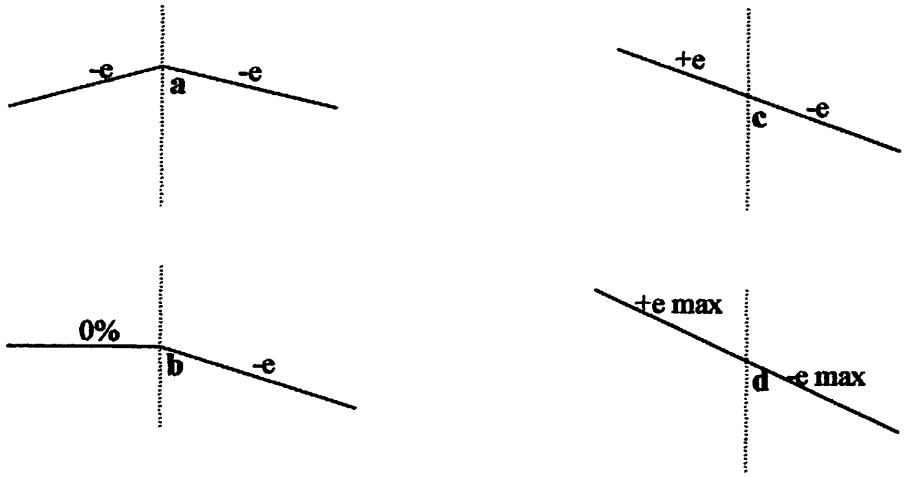
II.5. Diagram Superelevasi

Diagram superelevasi menggambarkan pencapaian superelevasi dari lereng normal ke superelevasi penuh, sehingga dengan mempergunakan diagram superelevasi dapat ditentukan bentuk penampang melintang pada setiap titik disuatu lengkung horisontal yang direncanakan.

Diagram superelevasi digambar berdasarkan tanda positif atau negatif ditinjau dari ketinggian sumbu jalan. Tanda positif untuk elevasi tepi perkerasan yang terletak lebih tinggi dari sumbu jalan dan tanda negatif untuk elevasi tepi perkerasan yang terletak lebih rendah dari sumbu jalan. Pada jalan tanpa median sumbu jalan digunakan sebagai sumbu putar. Metode ini paling umum dipergunakan untuk jalan 2 jalur 2 arah tanpa median (jalan raya tidak terpisah).

Pada jalan lurus, kemiringan tersebut dinamakan kemiringan normal dimana kemiringan melintang diambil minimum (= e_n) yang berguna pula untuk keperluan drainase yang efektif. Sedangkan untuk jalan yang mempunyai kelengkungan, kemiringannya bukan lagi sama dengan kemiringan minimum melainkan akan bertambah menuju kemiringan maksimum (= e_{maks}) pada pertengahan lengkungan.

Perubahan kemiringan tersebut bisa dilihat pada gambar berikut :

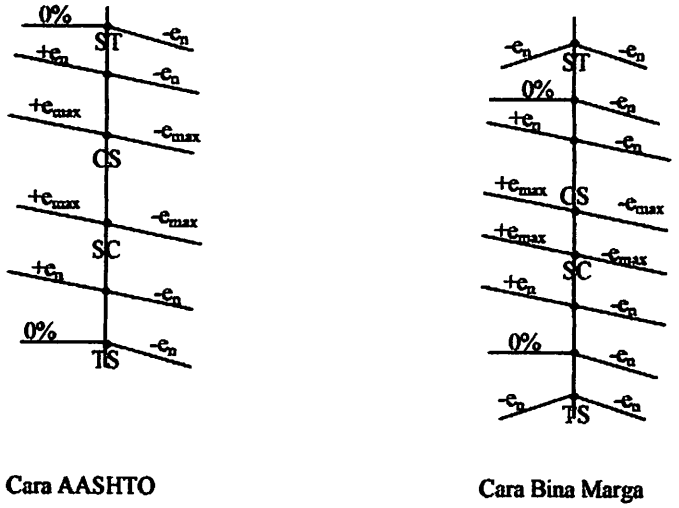


Gambar 2.10.
Kemiringan badan jalan

Posisi dari kemiringan badan jalan pada masing-masing titik pada lengkungan ditentukan dengan bantuan sebuah diagram super elevasi. Super elevasi dicapai secara bertahap dari kemiringan melintang normal pada bagian jalan lurus sampai ke kemiringan penuh (superelevasi) pada bagian lengkung.
Pembuatan diagram superelevasi antara cara AASHTO dan Bina Marga terdapat sedikit perbedaan, yaitu :

a. cara AASHTO, penampang melintang sudah mulai berubah pada titik TS.

b. cara Bina Marga, penampang melintang pada titik TS masih berupa penampang melintang normal.



Cara AASHTO

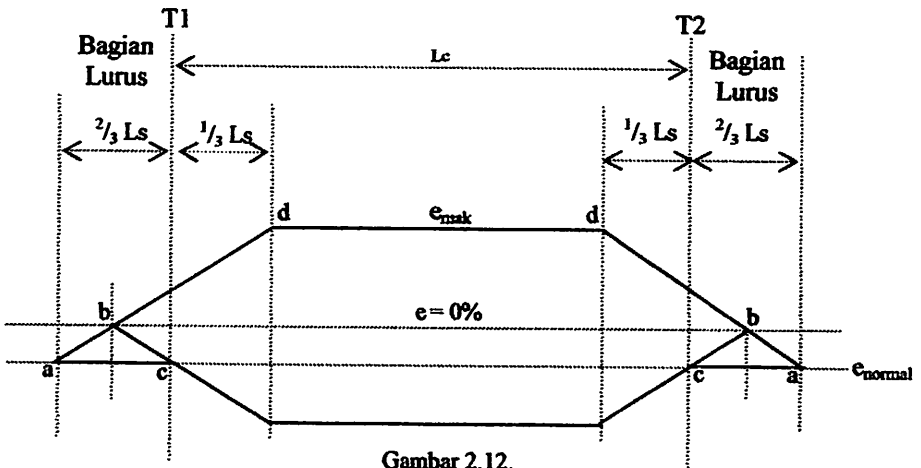
Cara Bina Marga

Gambar 2.11.
Perbedaan pembuatan diagram superelevasi

Metode yang digunakan untuk melakukan superelevasi yaitu merubah lereng potongan melintang, dilakukan dengan bentuk profil dari tepi pekerasan. Ada dua macam metode dalam pencapaian superelevasi yaitu :

a. Metode superelevasi pada tikungan tipe Full Circle (cara Bina Marga)

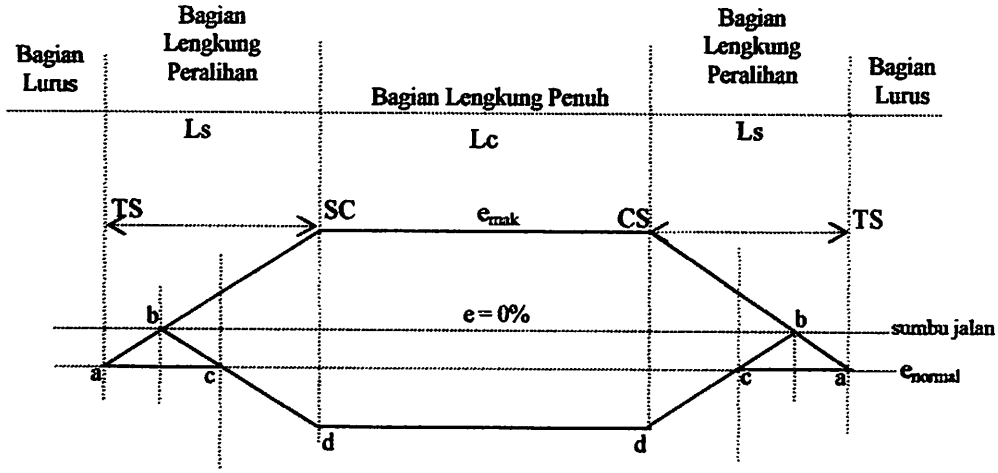
Pada tikungan FullCircle, pencapaian superelevasi dilakukan secara linear, diawali dari bagian lurus sepamjang $^2/_3$ Ls sampai dengan bagian lingkaran penuh sepanjang $^1/_3$ Ls.



Gambar 2.12.
Metode pencapaian superelevasi pada tikungan tipe Full Circle

b. Metode superelevasi pada tikungan tipe Spiral CircleSpiral

Pada tikungan SpiralCircleSpiral, pencapaian superelevasi dilakukan secara linear, diawali dari bentuk normal ( ) sampai awal lengkung peralihan (TS) yang berbentuk ( ) pada bagian lurus jalan, lalu dilanjutkan sampai superelevasi penuh ( ) pada akhir bagian lengkung peralihan (SC).



Gambar 2.13.
Metode pencapaian superelevasi pada tikungan tipe Spiral Circle Spiral

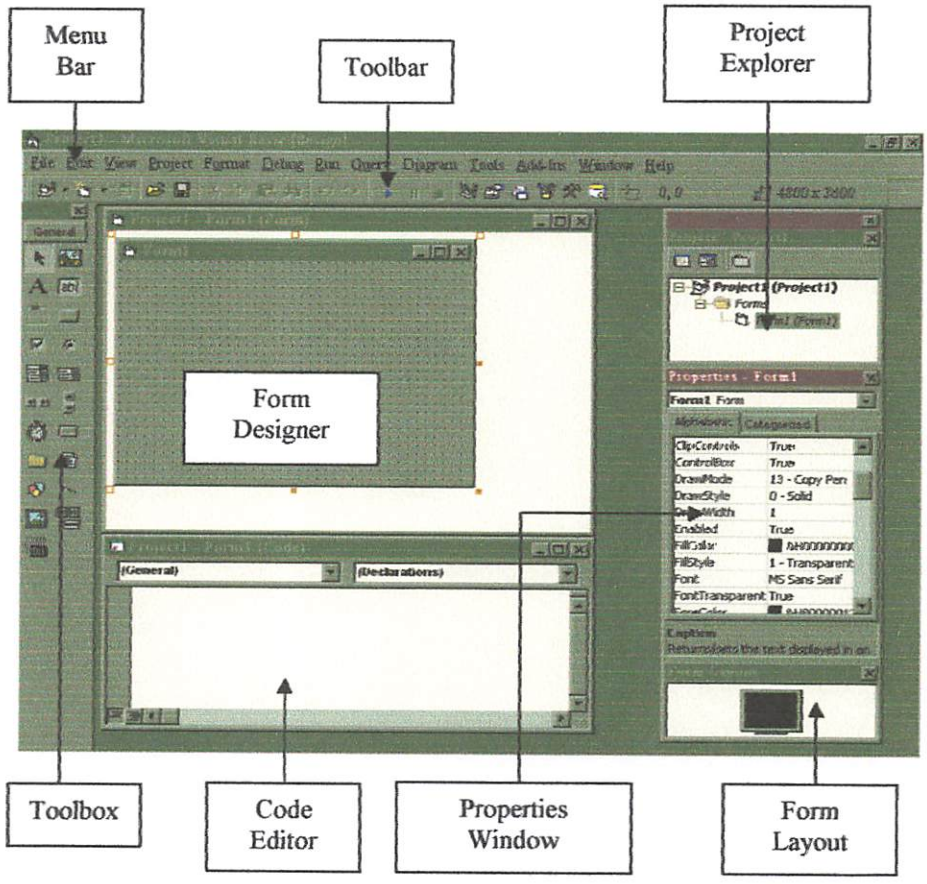
II.6. Visual Basic 6.0 dan Autocad 2004

Sistem pemrograman Visual Basic merupakan suatu bahasa pemrograman yang mengkombinasikan kemampuan bahasa *basic* dan piranti desain visual, bahasa ini menyediakan kesederhanaan dan kemudahan pemakaian tanpa mengorbankan kinerja atau fasilitas grafik yang menyebabkan *windows* menjadi lingkungan kerja yang menyenangkan. Visual basic adalah suatu bahasa computer yang mendukung pemrograman *event – driven* yaitu gaya pemrograman yang cocok untuk pemakai grafis. Pada pemrograman *event – driven* sasarannya adalah menjadikan pemakai sebagai orang yang berkuasa.

Autocad adalah suatu paket program yang mempunyai kemampuan dalam hal pengolahan grafis yang sangat bagus dan sering dipergunakan. Kemampuan yang ada pada Autocad 2004 dapat dimanfaatkan pada aplikasi yang akan dibuat dengan Visual Basic 6.0 dengan menggunakan fasilitas OLE, karena pada

Autocad 2004 telah disediakan *AutoCad Active Objects* yang dapat diakses oleh bahasa pemrograman lain, salah satunya adalah Visual Basic 6.0.

Tampilan layar program Microsoft Visual Basic 6.0 dapat dilihat pada gambar di bawah ini :



Gambar 2.14.
Tampilan Visual basic

Keterangan tampilan layer Visual Basic :

1. Menu Bar

   Menu Bar terdiri dari beberapa pilihan atau menu seperti : File, Edit, View, dll, yang memiliki akses ke submenu.

2. Toolbar

   Toolbar adalah jalan pintas untuk mengakses menu dengan jalan mengklik toolbar yang bersesuaian dengan suatu menu.

3. Toolbox

   Toolbox berfungsi sebagai tool untuk menempatkan berbagai macam tombol control di atas form designer.

4. Project Explorer

   Project Explorer yaitu sebagai tempat untuk memonitoring jumlah form, module, class yang digunakan dalam suatu project.

5. Properties Window

   Properties Window adalah tempat untuk mengedit karakteristik dari sebuah object yang terdapat didalam project misalnya ukuran, warna dan text dari suatu tombol.

6. Form Layout

   Form Layout adalah jendela yang menggambarkan posisi form dalam layar monitor. Dengan tampilan tersebut dapat dilihat posisi aplikasi yang dibuat saat dijalankan dalam layer monitor.

7. Form Designer

   Form Designer adalah tempat untuk menempatkan beberapa object, misalnya : label, teks, gambar, dan lain – lain. Semua object yang ditempatkan pada form akan dijalankan atau ditampilkan pada layer window. Pada Form inilah sebagai dasar dari aplikasi yang akan dibuat.

8. Code Editor

   Code Editor adalah tempat untuk mengetik kode program.

Beberapa contoh dasar – dasar *Object Oriented Programing* yang terdapat dalam Visual Basic antara lain :

1. Property

   Property ialah sifat atau cirri yang dimiliki oleh suatu *object*, contohnya *object* label mempunyai warna tulisan putih merah, warna *background* abu – abu. Warna tulisan dan warna background disebut property dari object label. Salah satu contoh dari property object form dapat diliahat pada tabel

No.	Property	Keterangan
1.	Name	Nama Object
2.	Picture	Gambar background form
3.	Caption	Judul Form
4.	Font	Menentukan jenis dan ukuran huruf pada form
5.	Enabled	Mengaktifkan form atau tidak mengaktifkan
6.	Border Style	Jenis pembatas di sekeliling form
7.	Icon	Menentukan gambar icon yang ditampilkan
8.	dll	

Tabel 2.1
Property Object Form

No.	Property	Keterangan
1.	Name	Nama Object
2.	Enabled	Menentukan aktif tidaknya kontrol textbox
3.	Visible	Tampak atau tidaknya tampilan textbox
4.	Aligment	Perataan teks pada kotak object textbox
5.	Back Color	Warna latar belakang dari textbox
6.	Width	Menentukan lebar textbox
7.	dll	

Tabel 2.2
Property Object Text

2. Method

Method ialah suatu aksi yang dapat diterapkan pada suatu *object*. Contohnya : *txttotal.Text = txt jml.Text*txtharga.Text* artinya menghitung jumlah (barang) dengan harga satuan, kemudian akan ditampilkan pada kontrol (*txttotal.Text*). Proses ini akan bekerja apabila menekan tombol (*command*).

```
Private Sub Command1_Click()
txttotal.Text =Format( txt jml.Text*txtharga.Text, "###,###,###.00)
EndSub
```

3. Event

Event ialah kejadian yang menimpa suatu *object*. Programmer dapat mengontrol apa yang akan dilakukan program pada saat suatu kejadian berlangsung.

No	Property	Keterangan
1.	On Change	Saat dirubah
2.	On Get Focus	Saat mendapat fokus
3.	On Lost Focus	Saat kehilangan fokus
4.	On Click	Saat diklik
5.	On Key Down	Saat tombol ditekan
6.	On Mouse Move	Saat Mouse digerakkan di atas object

Tabel 2.3
event object textbox

II.7. Obyek Lingking and Embedding (OLE)

Kelebihan yang dimiliki oleh Visual Basic adalah fasilitas OLE (Obyek Lingking and Embedding) yang memungkinkan untuk membuat suatu obyek dalam suatu aplikasi yang berisi data dari aplikasi lain, yang ditempatkan di dalam program Visual Basic.

Lingking, obyek dari aplikasi lain yang dihubungkan dengan aplikasi Visual Basic. Sewaktu pemakai keluar dari aplikasi, obyek yang diperbaharui disimpan dalam aplikasi sumbernya. Obyek yang sama dihubungkan dengan beberapa aplikasi lain.

Embedding, obyek dari aplikasi lain yang dihubungkan dengan aplikasi Visual Basic dan ketika pemakai keluar dari aplikasi, obyek otomatis diperbaharui dan disimpan dalam aplikasi Visual Basic. Tidak ada aplikasi lain yang mempunyai akses ke data yang di-*embed*.

Jenis aplikasi yang dapat dibuat dengan Visual Basic antara lain :

1. Aplikasi data base terbaru dengan format yang dapat dibuat dengan memanfaatkan fasilitas *data acces*.
2. ActeveX Document, yaitu aplikasiyang digunakan dalam dunia imternet, dimana pemakaiannya hanya bisa menggunakan browser internet.

3. Aplikasi yang memanfaatkan fasilitas dari aplikasi lain seperti Microsoft Word, Excel, Autocad, dll.
4. Aplikasi yang menggunakan run time dynamic – link library.
5. ActiveX control, yaitu file kontrol yang digunakan dalam dunia pengembangan software.
6. Aplikasi umum seperti game, animasi, dll.

BAB III

PELAKSANAAN PENELITIAN

III.1. Materi Penelitian

Materi penelitian terdiri dari data perencanaan yang meliputi :

1. Data kurva horisontal

   a. Tipe Full Circle (selisih busur, polar, poligon) :

No	Data perencanaan	Keterangan
1.	$X_A, Y_A, X_B, Y_B, X_C, Y_C$	X_A, Y_A : Koordinat awal kurva X_B, Y_B : Koordinat pusat / PI X_C, Y_C : Koordinat akhir kurva
2.	R	Radius atau jari – jari lingkaran
3.	Titik stacking out	Jumlah titik stacking out

Tabel 3.1.
Data input Kurva Horisontal

   b. Tipe Spiral Circle Spiral

No.	Data perencanaan	Keterangan
1.	$X_A, Y_A, X_{PI}, Y_{PI}, X_C, Y_C$	X_A, Y_A : Koordinat awal kurva X_{PI}, Y_{PI} : Koordinat pusat / PI X_C, Y_C : Koordinat akhir kurva
2.	R	Jari – jari lingkaran
3.	Vr	Kecepatan rata – rata
4.	Stacking out I	Jumlah titik stacking out untuk busur spiral
5.	Stacking out II	Jumlah titik stacking out untuk busur lingkaran

Tabel 3.2.
Data input Busur Spiral

Catatan : apabila data koordinat titik tidak ada, maka data koordinat bisa diganti dengan data jarak dan azimuth.

2. Data kurva Vertikal

Cara input I :

No.	Data perencanaan
1.	Elevasi awal
2.	Elevasi tengah
3.	Elevasi akhir
4.	Acuan tinggi
5.	Jumlah titik stacking out
6.	Panjang / jarak busur

Cara input II :

No.	Data perencanaan
1.	Elevasi awal
2.	g1 (gradien I)
3.	g2 (gradien II)
4.	Acuan tinggi
5.	Jumlah titik stacking out
6.	Panjang / jarak busur

Tabel 3.3
Data input Kurva Vertikal

3. Diagram Superelevasi

a. Full Circle

No.	Data perencanaan
1.	e_n (kemiringan normal)
2.	emax (kemiringan maksimal)
3.	Lebar badan jalan
4.	Panjang Ls fiktif / garis lurus
5.	Panjang busur lingkaran (Lc)

b. Spiral Circle Spiral

No.	Data perencanaan
1.	e_n (kemiringan normal)
2.	emax (kemiringan maksimal)
3.	Lebar badan jalan
4.	Panjang busur spiral (Ls)
5.	Panjang busur lingkaran (Lc)

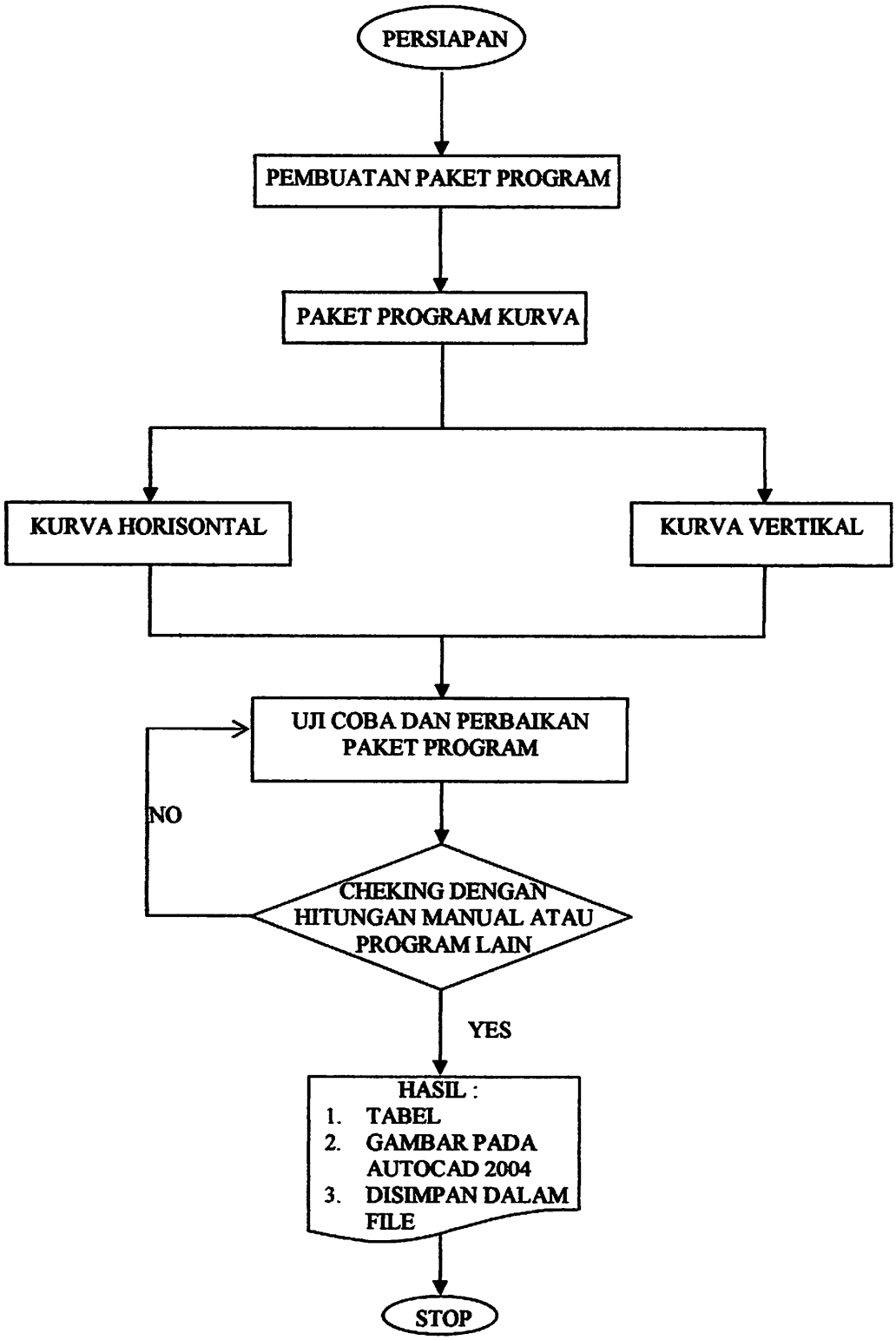
Tabel 3.4.
Data input Diagram Superelevasi

III.2. Alat Penelitian

Alat – alat penelitian yang digunakan untuk memproses dan menyelesaikan penelitian ini adalah :

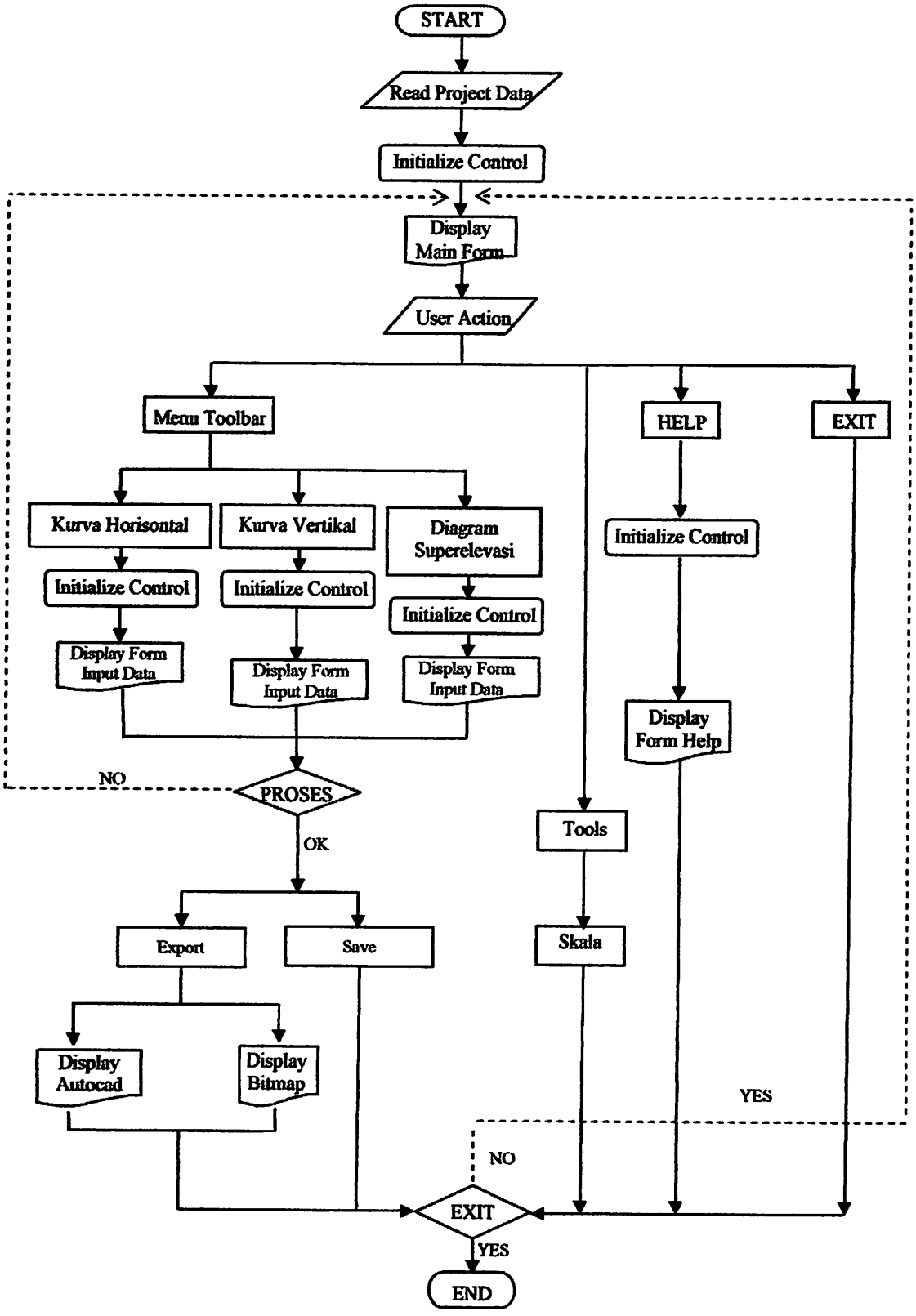
1. Perangkat Keras
   - PC Pentium III
   - Hardisk 100 Mega byte
   - Memori 128 mega byte
   - Monitor
2. Perangkat Lunak
   - Visual Basic 6.0
   - Autocad 2004

DIAGRAM ALIR PENELITIAN



Gambar 3.1.
Diagram Penelitian

Flowchart Program



Gambar 3.2.
Diagram Alir Program

Keterangan Flowchart Program

Untuk pembuatan program, diperlukan beberapa tahapan yang harus dilakukan, yaitu :

1. Start, yaitu memulai program.
2. Read Project data, merupakan pemanggilan data yang telah diproses atau dibentuk sebelumnya.
3. Initialize Control, yaitu mengenali perintah (control) untuk menjalankan program.
4. Display Form, yaitu menampilkan form.
5. User Action, yaitu pengguna (user) dapat menjalankan perintah melalui tombol toolbar.
6. Menu Toolbar , berisi fasilitas – fasilitas yang akan digunakan oleh user.
7. Help, merupakan menu bantuan yang berisikan petunjuk (guide) dalam menjalankan program.
8. Exit, yaitu untuk mengakhiri atau keluar dari program.

III.3. Pembuatan Program

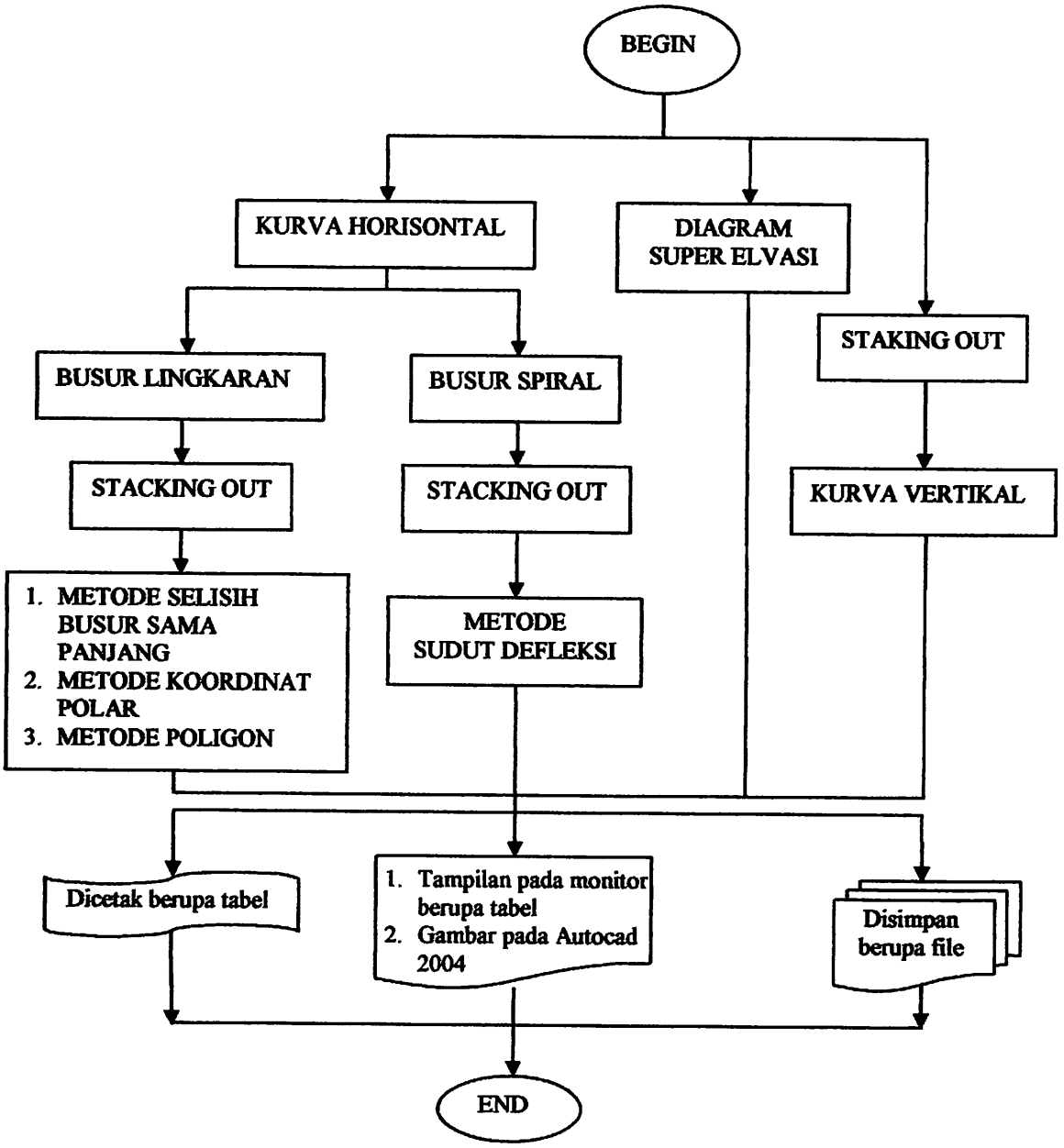
Dalam pembuatan program akan dibahas menurut materi sebagaimana terdapat pada volume pekerjaan. Untuk pembuatan suatu program ada tahapan – tahapan yang harus dilakukan diantaranya adalah :

1. Rancangan Program
2. Pembuatan form awal
3. Pembuatan form menu utama program
4. Pembuatan form untuk Pemrosesan Data

III.3.1. Rancangan program

Rancangan program ini meliputi : Metode selisih busur, polar, poligon, spiral (SCS), kurva vertikal, Diagram Superelevasi (FC dan SCS) yang akan ditampilkan dalam bentuk diagram alir.

RANCANGAN PROGRAM UTAMA



Gambar 3.3.
Rancangan Program Utama

III.3.1.1. Rancangan Program Kurva Horisontal

Tujuan dari pembuatan program ini adalah untuk mendapatkan koordinat titik detail dan data poligon yang merupakan hasil data pengukuran yang akan digunakan dalam proses penggambaran.

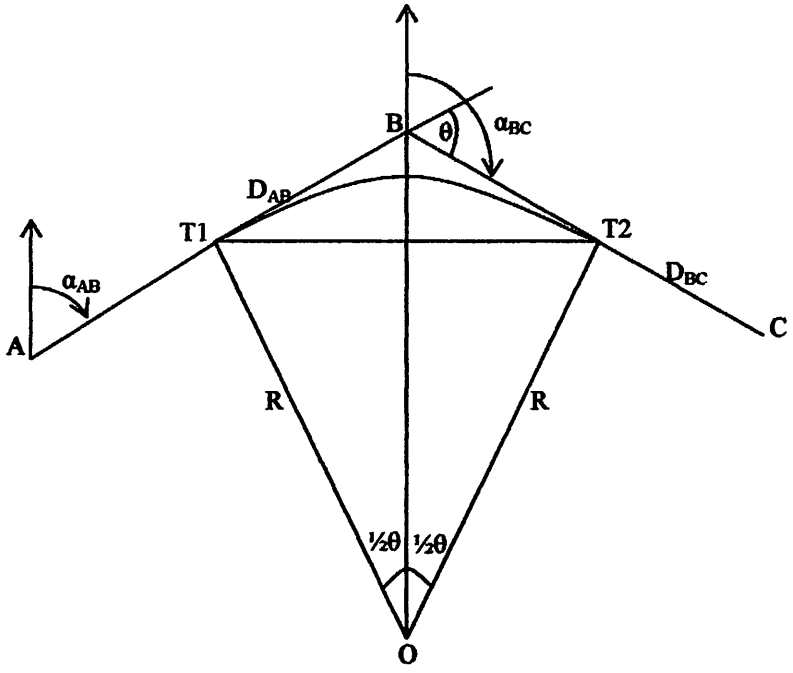
Berdasarkan bentuk busurnya, maka kurva horisontal dibagi menjadi dua antara lain :

A. Busur Lingkaran

Metode yang digunakan antara lain :

1. Pembuatan Program Metode Selisih Busur Sama Panjang
2. Rancangan Program Metode Koordinat Polar
3. Rancangan Program Metode Poligon

Data inputnya terdiri dari 2 cara, yaitu :



Gambar 3.4.
Gambar Kurva Horisontal

Cara I :

a. Jari-jari R dalam satuan meter
b. Koordinat lapangan : (XA, YA), (XB,YB),(XC,YC).
c. Jumlah staking out (n).

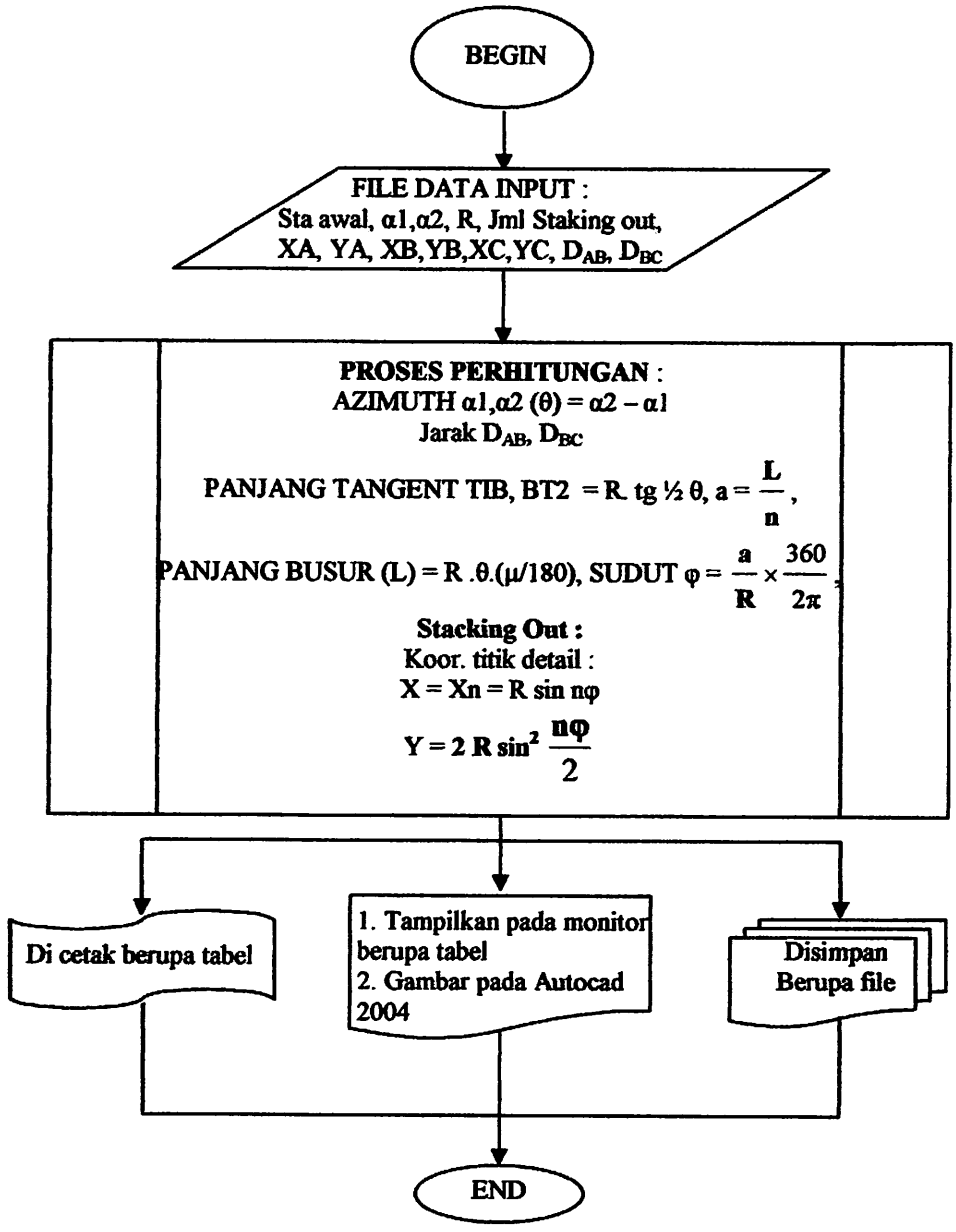
Cara II :

a Jari-jari R dalam satuan meter

b Azimuth AB,BC (α_{AB}, α_{BC}), dan Jarak AB, BC (D_{AB}, D_{BC})

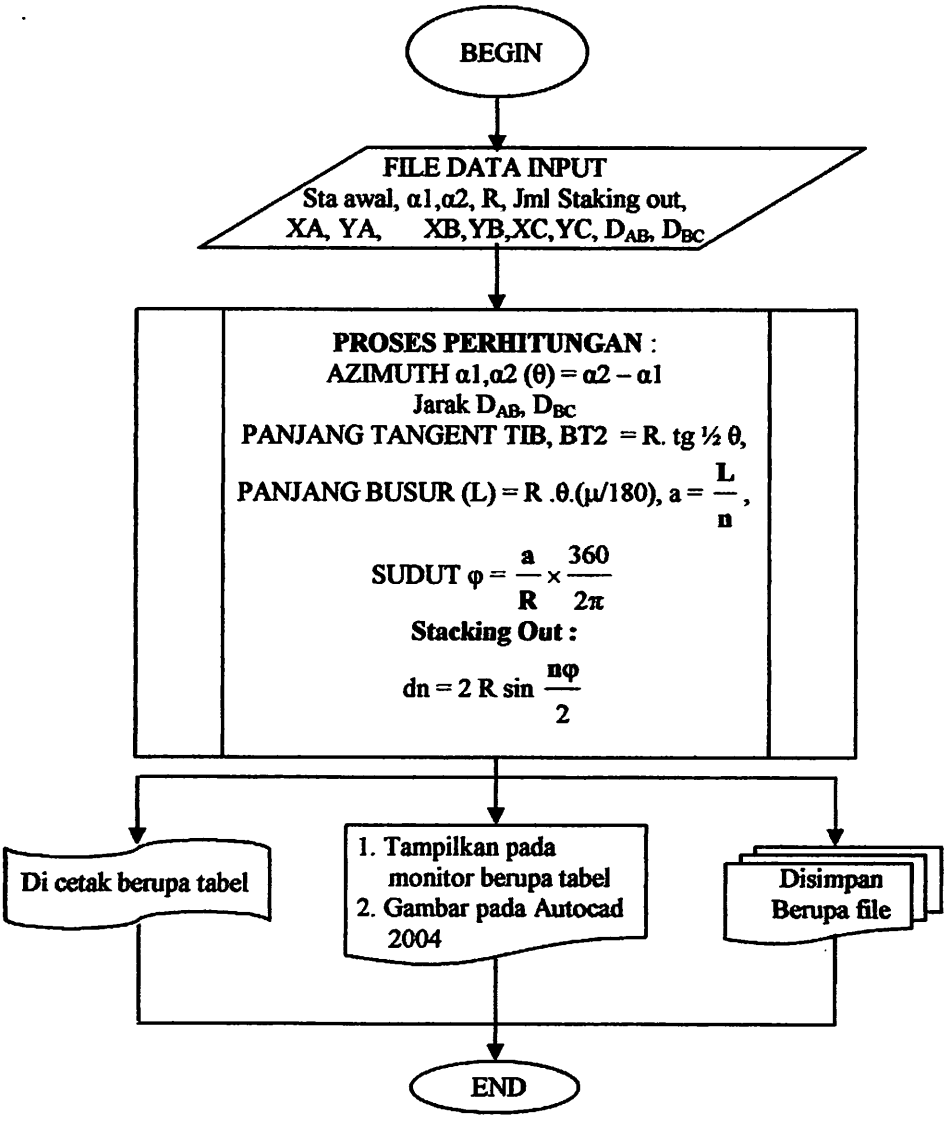
c Jumlah staking out (n).

RANCANGAN PROGRAM KURVA HORISONTAL BUSUR LINGKARAN DENGAN METODE KOORDINAT SELISIH BUSUR SAMA PANJANG



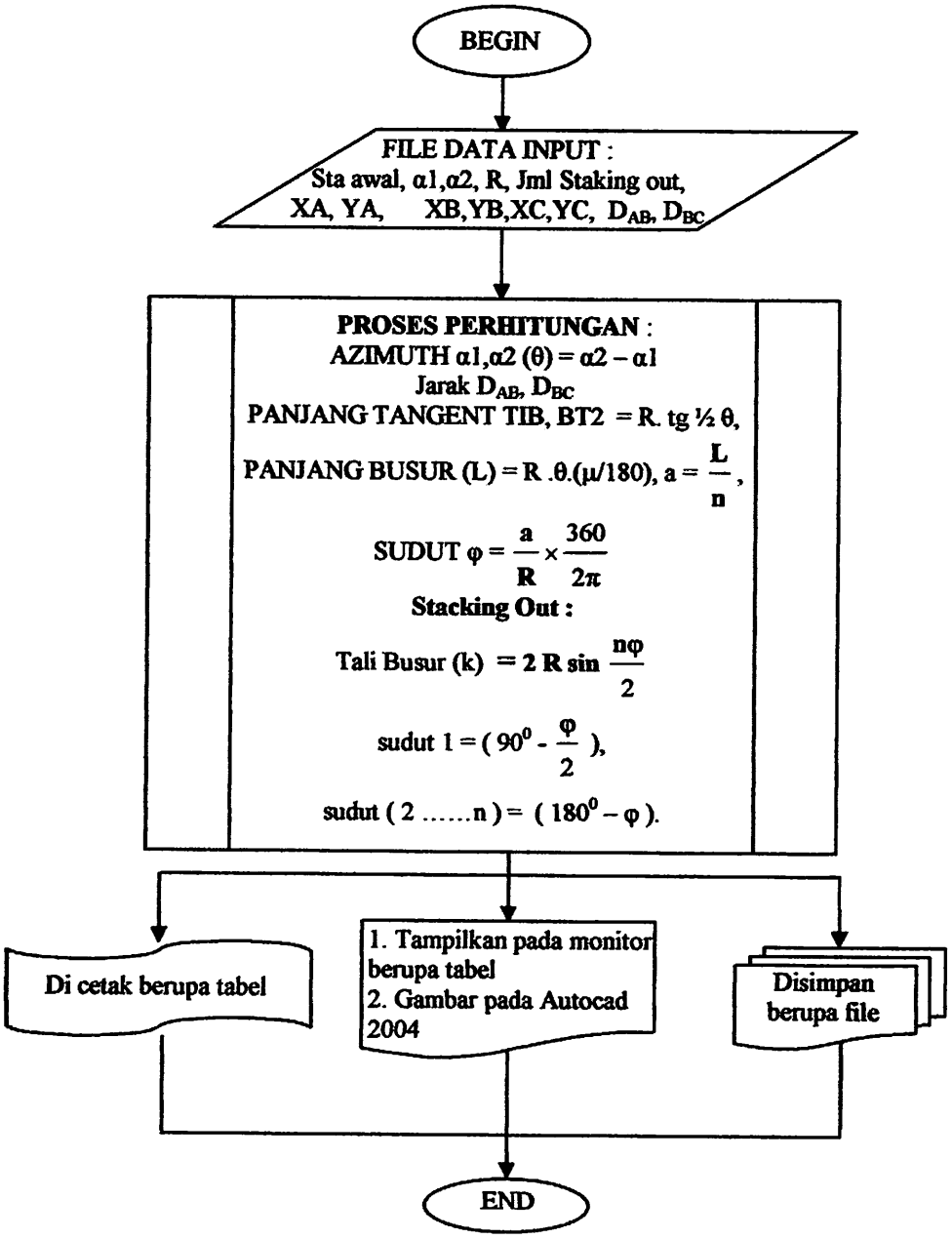
Gambar 3.5.
Rancangan Program Metode Selisih Busur

RANCANGAN PROGRAM KURVA HORISONTAL BUSUR LINGKARAN DENGAN METODE KOORDINAT POLAR



Gambar 3.6.
Rancangan Program Metode Polar

RANCANGAN PROGRAM KURVA HORISONTAL BUSUR LINGKARAN DENGAN METODE POLIGON



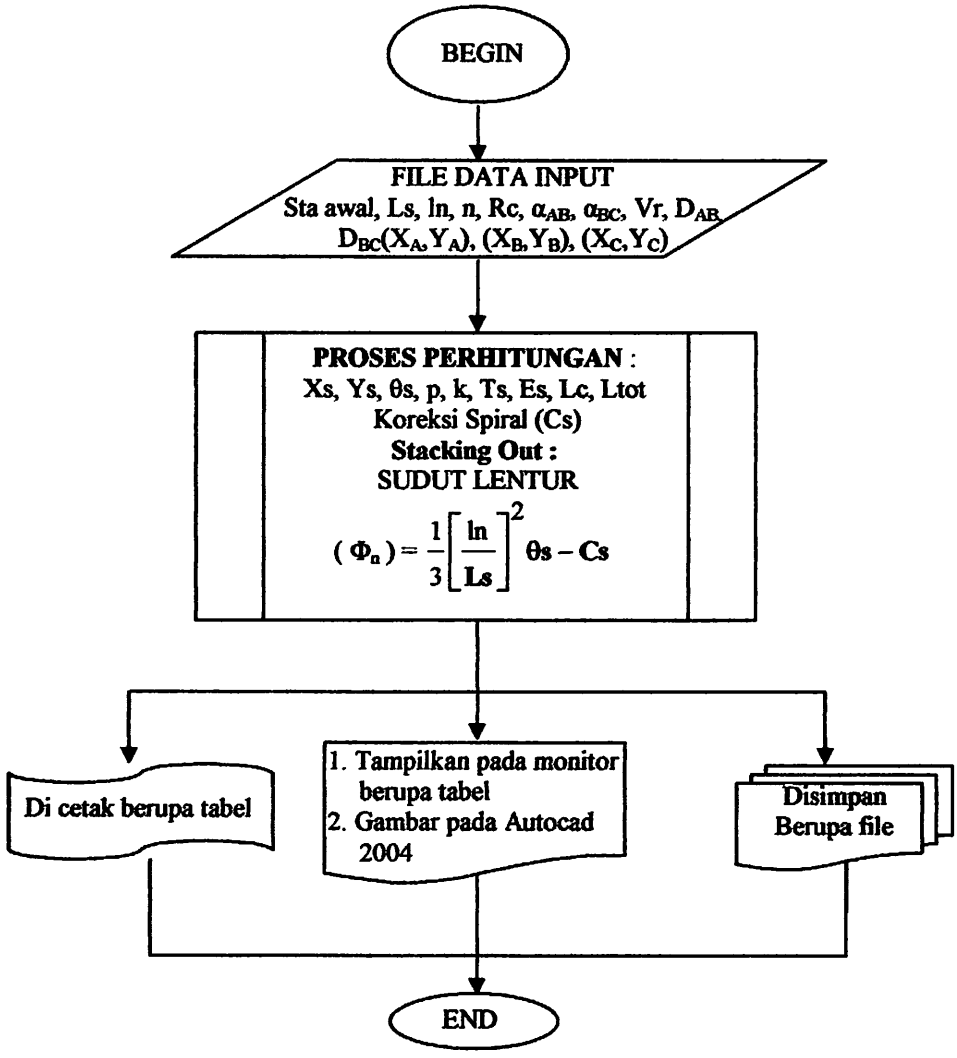
Gambar 3.7.
Rancangan Program Metode Poligon

B. Busur Spiral

Pada busur spiral, metode yang digunakan adalah metode sudut defleksi. Adapun data inputnya meliputi :

a. Panjang busur spiral Ls dalam satuan meter
b. Jari-jari busur lingkaran Rc dalam satuan meter
c. Jarak stake out sta dalam satuan meter (ln)
d. Jumlah staking out (n),
e. Koordinat awal (koordinat lapangan) : XA, YA,XB,YB,XC,YC atau bisa diisi dengan Azimuth
f. Kecepatan rata – rata (Vr)
g. Jarak AB dan BC

RANCANGAN PROGRAM KURVA HORISONTAL BUSUR SPIRAL DENGAN METODE SUDUT DEFLEKSI



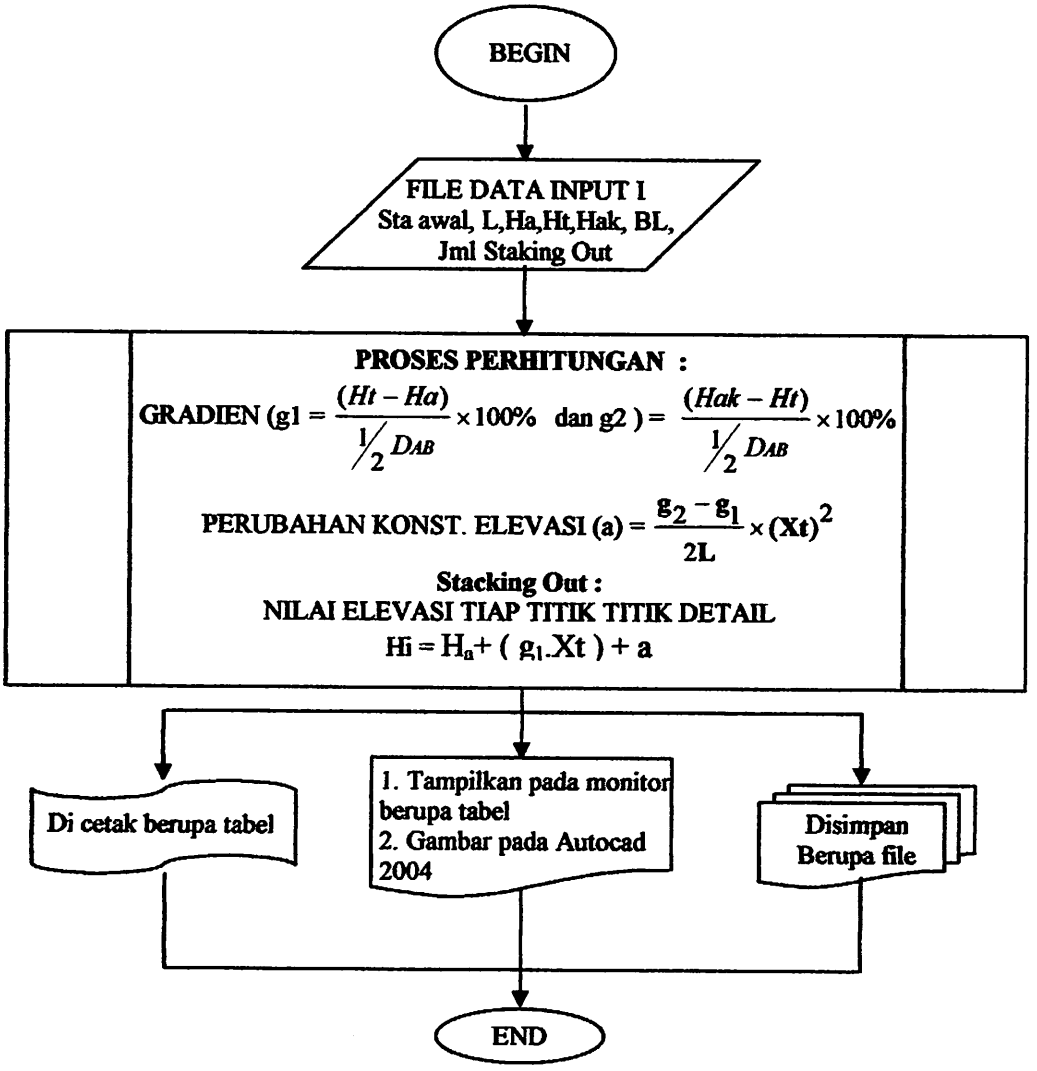
Gambar 3.8.
Rancangan Program Metode Spiral

III.3.1.2. Rancangan Program Kurva Vertikal

Program ini dibuat untuk mendapatkan elevasi (ketinggian) titik detail secara teliti. Rancangan programnya sebagai berikut :

1. Data input perhitungan meliputi :
   a. Panjang kurva vertikal (L) dalam satuan meter
   b. Elevasi awal lengkungan (Ha) dalam satuan meter
   c. Elevasi tengah (Ht) dalam satuan meter
   d. Elevasi akhir lengkungan (Hak) dalam satuan meter
   e. Base level (besar elevasi terkecil dalam gambar) BL = dalam satuan meter
   f. Jumlah titik staking out

RANCANGAN PROGRAM KURVA VERTIKAL



Gambar 3.9.
Rancangan Program Kurva Vertikal

III.3.1.3. Rancangan Program Diagram Superelevasi

Pada Diagram Superelevasi terdapat 2 metode yaitu :

A. Metode SCS (Spiral Circle Spiral)

Data inputnya antara lain :

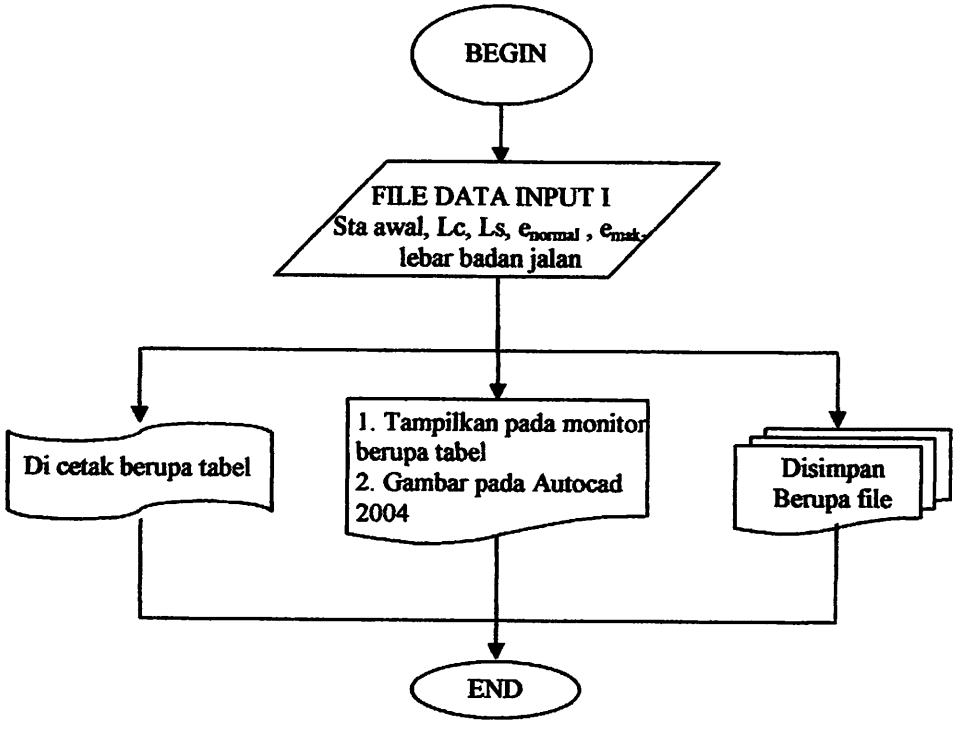
- Nilai kemiringan normal (e_{normal})
- Nilai kemiringan maksimum (e_{mak})
- Panjang Busur Lingkaran (Lc)
- Panjang Lengkung Peralihan (Ls)
- Lebar Badan Jalan

B. Metode FC (Full Cicle)

Data inputnya antara lain :

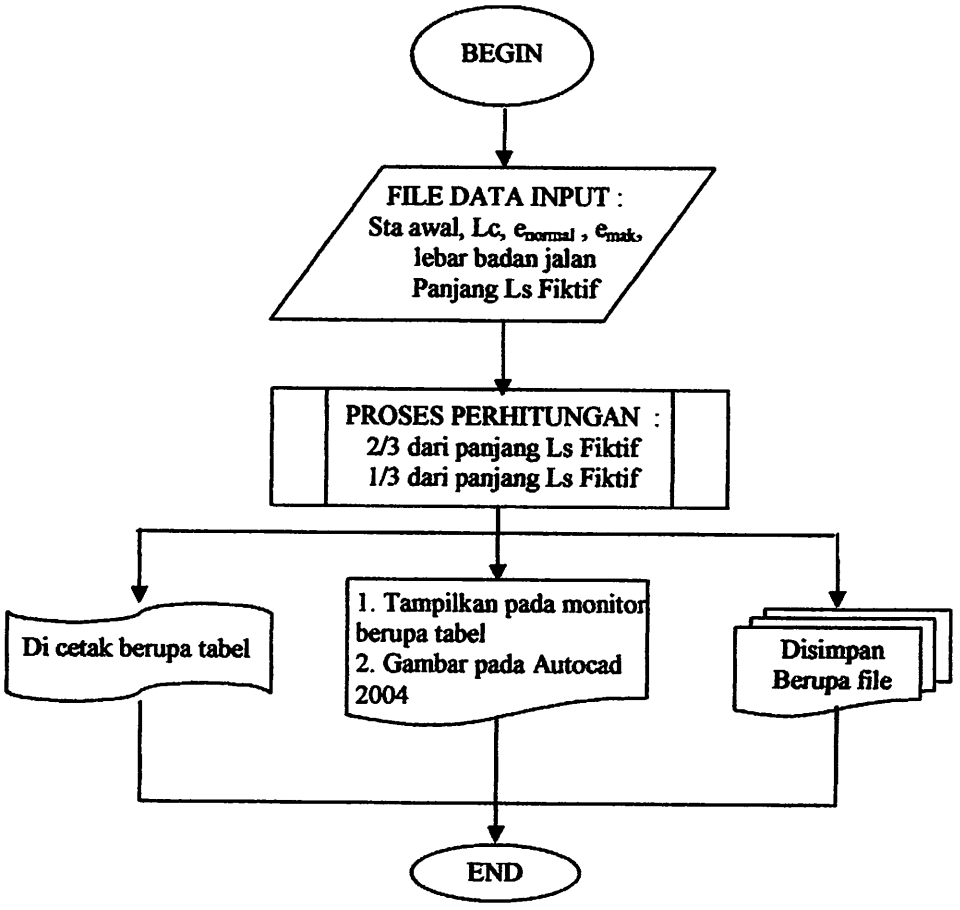
- Nilai kemiringan normal (e_{normal})
- Nilai kemiringan maksimum (e_{mak})
- Panjang Busur Lingkaran (Lc)
- Panjang Ls Fiktif
- Lebar Badan Jalan

METODE SPIRAL CIRCLE SPIRAL



Gambar 3.10.
Rancangan Program Diagram Superelevasi
Metode SCS

RANCANGAN PROGRAM DIAGRAM SUPER ELEVASI METODE FULL CIRCLE



Gambar 3.11.
Rancangan Program Diagram Superelevasi
Metode Full Circle

III.3.2. Pembuatan form awal

Pada tahapan ini akan dibuat suatu form untuk tampilan awal dari suatu program dan tampilan menu utama program yang berisi menu – menu bar, toolbar dan tempat layout untuk gambar.

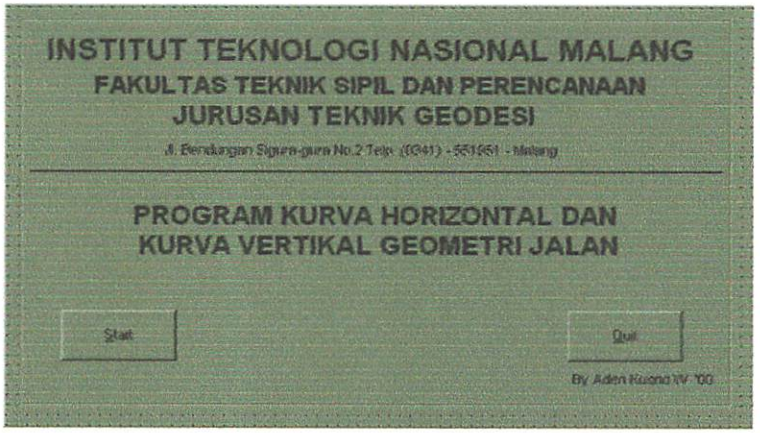
a. Form untuk tampilan awal

Untuk tampilan awal, pilih kontrol *Label*  , *frame*  dan *Command*  / klik ganda kontrol *Label, frame* dan *Command,* tambahkan ke dalam form sebanyak 6 kontrol label, 1 frame dan 2 kontrol command. Tulis kode form pada jendela code form yang dapat dilihat pada lampiran listing program. Atur setiap kontrol seperti tabel di bawah ini :

Komponen	**Property**	
	Type	**Nilai**
Label 1,2,3,4,5,6	Alligment	2 – center
Label 1 Label 2 Label 3 Label 4 Label 5 Label 6 Command 1 Command 2	Caption	INSTITUT TEKNOLOGI NASIONAL MALANG FAKULTAS TEKNIK SIPIL DAN PERENCANAAN JURUSAN TEKNIK GEODESI Jl. Bendungan Sigura-gura No.2 Telp. (0341) - 551951 – Malang PROGRAM KURVA HORIZONTAL DAN KURVA VERTIKAL GEOMETRI JALAN By Aden Kusno W '00 Start Quit

Tabel 3.5.
Pengaturan kontrol untuk form awal

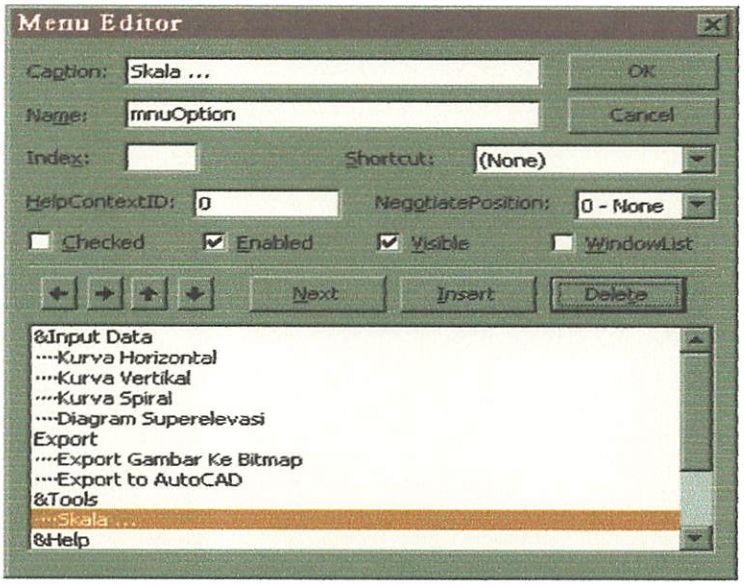
Hasil dari pembuatan tampilan awal adalah seperti dibawah ini :



Gambar 3.12.
Tampilan awal program

b. Tampilan menu utama program

Pada menu utama terdapat menu bar, dimana pembuatannya menggunakan *Menu editor*  , contoh pengisian untuk pembuatan menu toolbar.

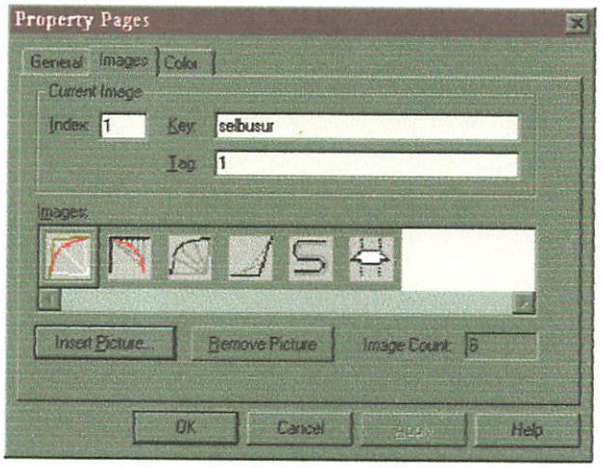


Gambar 3.13.
Menu Editor

Untuk penulisan &Input Data, &Export, &Tools merupakan nama dari menu bar sedangkan Kurva Horisontal, Kurva Vertikal, Skala, Export to Autocad adalah sub menu. Hasil dari pembuatan menu bar dapat dilihat pada gambar di bawah ini :

Gambar 3.14.
Menu Bar dan Sub Menu

Selain menubar, pada menu utama juga ditambahaan kontrol picture box untuk penggambaran dan toolbar. Untuk membuat toolbar, pilih kontrol image list , klik kanan pilih properties.

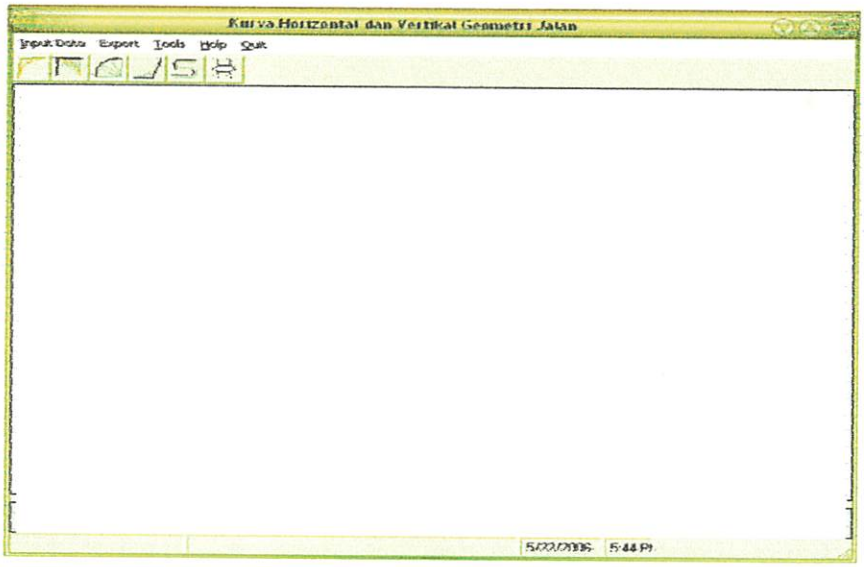


Gambar 3.15.
Tampilan Image List

Hasil dari pembuatan toolbar adalah sebagai berikut :

Gambar 3.16.
Hasil pembuatan toolbar

Dalam pembuatan icon toolbar, pemggambarannya dilakukan pada *Paint* dan disimpan dalam format *BMP. Hasil keseluruhan untuk pembuatan menu utama dapat dilihat pada gambar di bawah ini :



Gambar 3.17.
Tampilan Menu Utama Program

III.3.4. Pembuatan form untuk Pemrosesan Data

Pada tahapan ini dilakukan analisa suatu permasalahan yang akan dibuat paket programnya, sehingga dapat menentukan data apa yang akan diproses (input), bagaimana memprosesnya dan apa hasil dari program (output).

Paket program untuk perhitungan data dan penggambaran terdiri dari :

- Kurva Horisontal (metode selisih busur, polar, dan poligon) dan spiral
- Kurva vertikal
- Diagram Superelevasi

A. Metode selisih busur, polar, poligon

Langkah pembuatan form :

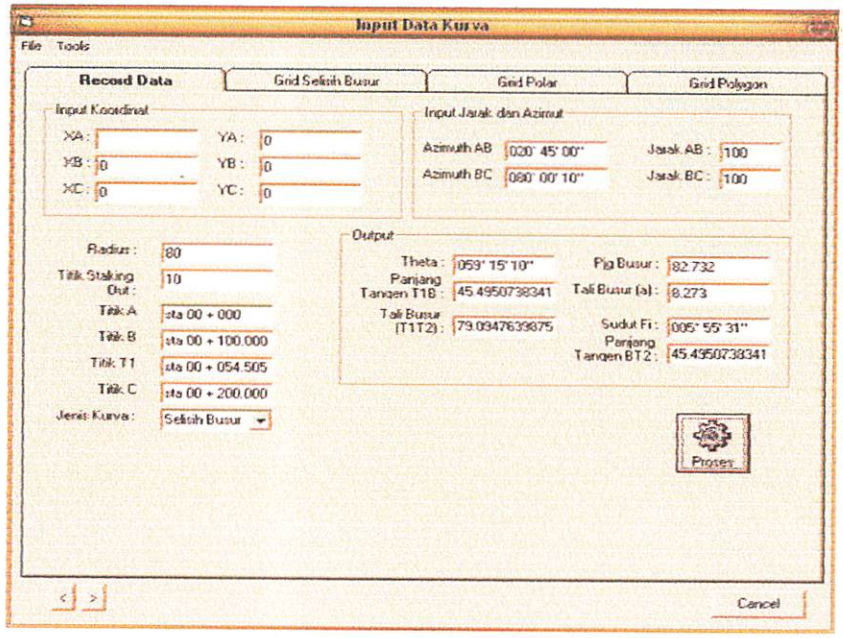
1. Tambahkan 20 kontrol *Text* untuk data input dan data output.
2. Tambahkan 21 kontrol *label*, untuk memberikan keterangan dalam menginput data dan melihat hasil output data.
3. Tambahkan 4 kontrol command sebagai perintah, dan 1 combo box untuk pemilihan jenis metode.

4. Pada form tersebut tambahkan 1 kontrol SS Tab dan 3 kontrol Data Grid untuk hasil stacking out. Pengaturan kontrol SS Tab dapat dilihat pada tabel di bawah ini :

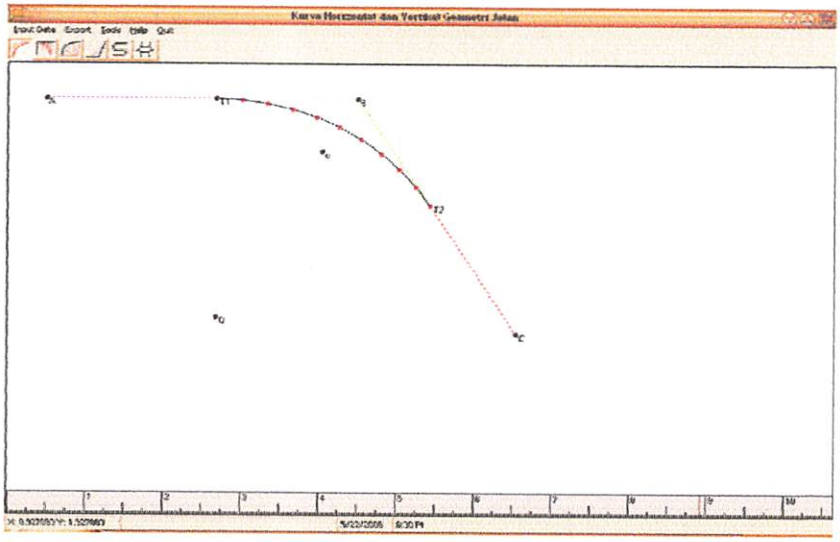
Komponen	Property	
	Type	Nilai
SS Tab	Tab	4
	TabsPerRow	4
	Caption	Record Data Grid Selisih Busur Grid Polar Grid Poligon

Tabel 3.6.
Pengaturan kontrol SS Tab untuk kurva Horisontal

5. Tambahkan Menu Editor untuk membuat menu bar (File dan Tools) dan sub menu (New, Save, Close, dan Gambar).
6. Tulislah kode program pada jendela code yang dapat dilihat pada lampiran listing program.
7. Tambahkan 3 kontrol *Frame* untuk untuk lebih mudah dalam pengelompokan.
8. Hasil Pembuatan Form dapat dilihat pada gambar di bawah ini :



Gambar 3.18.
Tampilan form kurva horisontal

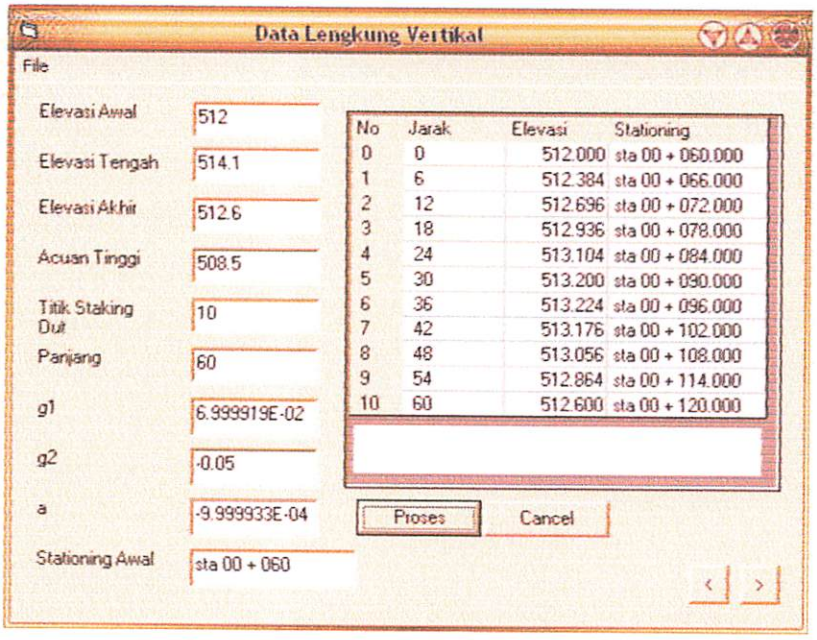


Gambar 3.19.
Contoh Tampilan gambar kurva horisontal

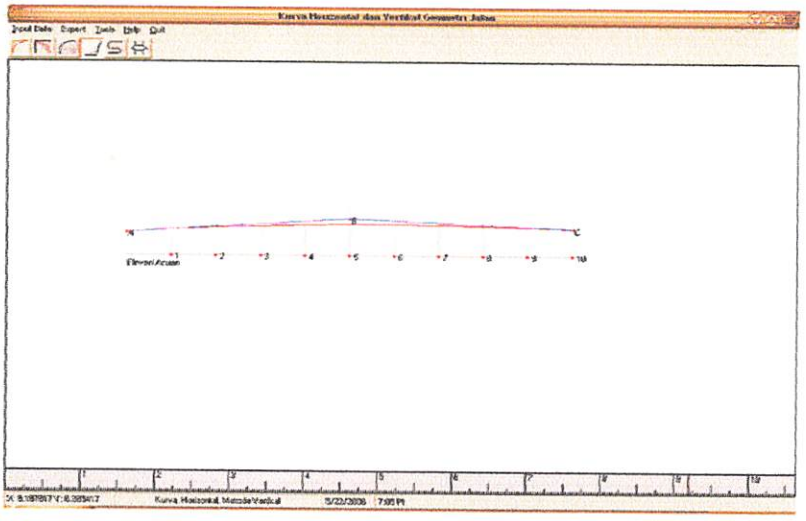
B. Kurva Vertikal

Langkah Pembuatan Form :

1. Tambahkan 1 kontrol MsFlexGrid untuk hasil stacking out.
2. Tambahkan 10 kontrol *label*, untuk memberikan keterangan dalam menginput data dan melihat hasil output data.
3. Tambahkan Menu Editor untuk membuat menu bar (File) dengan sub menunya (New, Save, Close,).
4. Tambahkan 10 kontrol *Text* untuk data input dan data output.
5. Tambahkan 2 kontrol command sebagai perintah.
6. Tulislah kode program pada jendela code yang dapat dilihat pada lampiran listing program.
7. Hasil Pembuatan Form dapat dilihat pada gambar di bawah ini :



Gambar 3.20.
Tampilan form kurva Vertikal



Gambar 3.21.
Contoh Tampilan gambar kurva Vertikal

C. Spiral

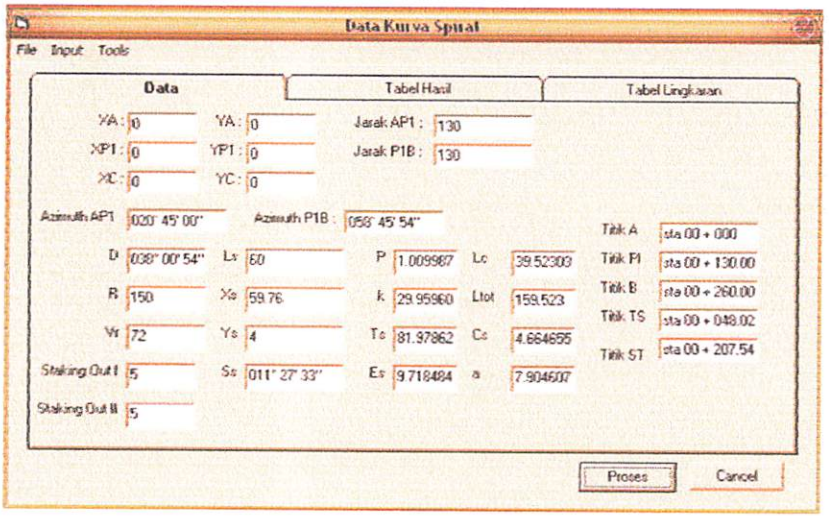
Langkah pembuatan form :

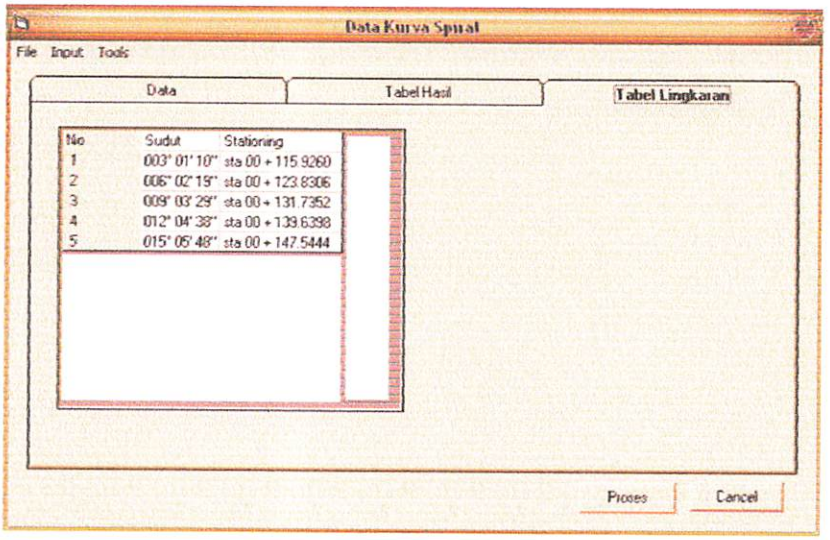
1. Tambahkan 32 kontrol *Text* untuk data input dan data output.
2. Tambahkan 34 kontrol *label*, untuk memberikan keterangan dalam menginput data dan melihat hasil output data.
3. Tambahkan 2 kontrol command sebagai perintah.
4. Pada form tersebut tambahkan 1 kontrol SS Tab dan 3 kontrol MS Flex Grid untuk hasil stacking out. Pengaturan kontrol SS Tab dapat dilihat pada tabel di bawah ini :

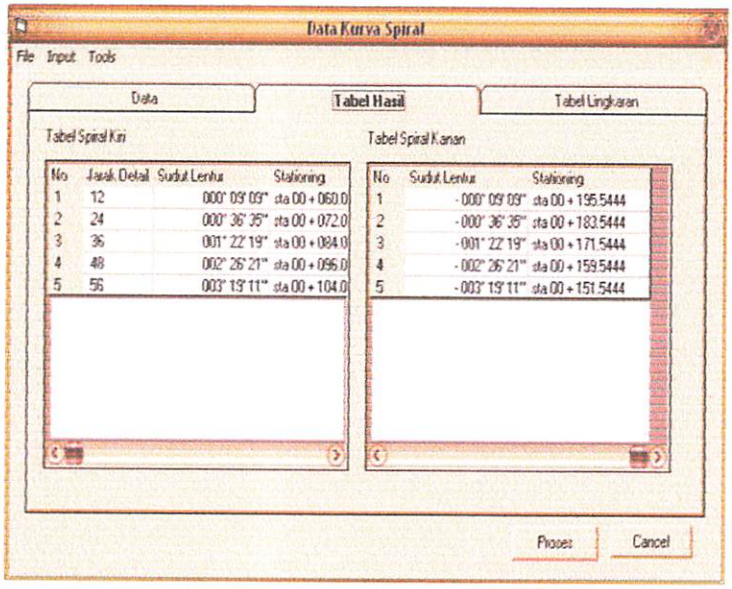
Komponen	**Property**	
	Type	**Nilai**
SS Tab	Tab	3
	TabsPerRow	3
	Caption	Data Tabel Hasil Tabel Lingkaran

Tabel 3.7.
Pengaturan kontrol SS Tab untuk Busur Spiral

5. Tambahkan Menu Editor untuk membuat menu File dengan sub menunya yaitu New, Save, Close, Input dengan submenunya jarak, dan Tools dengan submenunya Prosess, Gambar ke Layar.
6. Tulislah kode program pada jendela code yang dapat dilihat pada lampiran listing program.
7. Hasil Pembuatan Form dapat dilihat pada gambar di bawah ini :



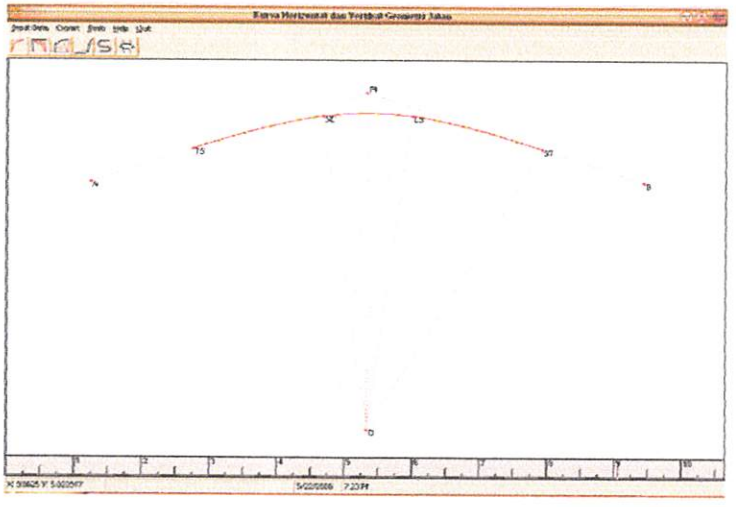




Gambar 3.22.
Tampilan form Busur Spiral

Keterangan :

- Tabel spiral kiri : Untuk busur spiral sebelah kiri.
- Tabel spiral kanan : Untuk busur spiral sebelah kanan.
- Tanda (-) : Perhitungan sudut berlawanan arah jarum jam



Gambar 3.23.
Contoh Tampilan gambar Busur Spiral

D. Diagram Superelevasi

Langkah pembuatan form :

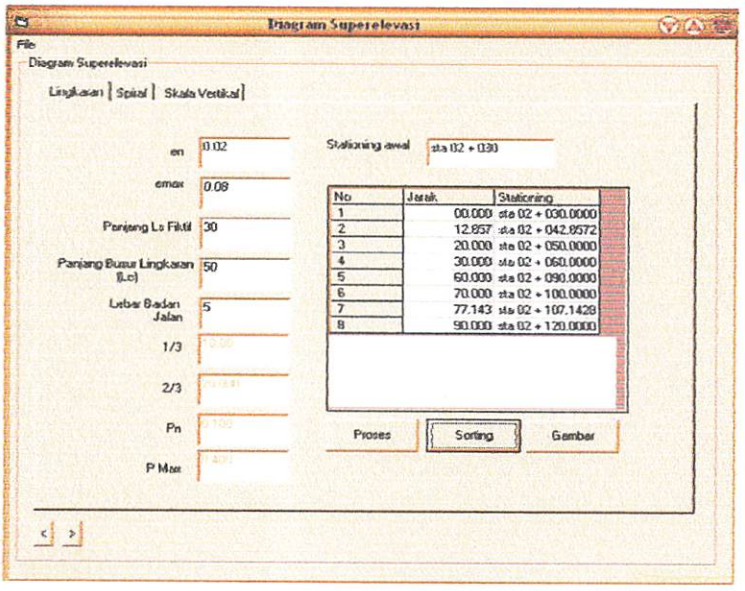
Pada form tersebut tambahkan 1 kontrol SS Tab. Pengaturan kontrol SS Tab dapat dilihat pada tabel di bawah ini :

Komponen	Property	
	Type	Nilai
SS Tab	Tab	3
	TabsPerRow	3
	Caption	Lingkaran Spiral Skala Vertikal

Tabel 3.8
Pengaturan kontrol SS Tab untuk Diagram Superelevasi

Untuk Diagram Superelevasi metode Lingkaran, pembuatan form sebagai berikut:

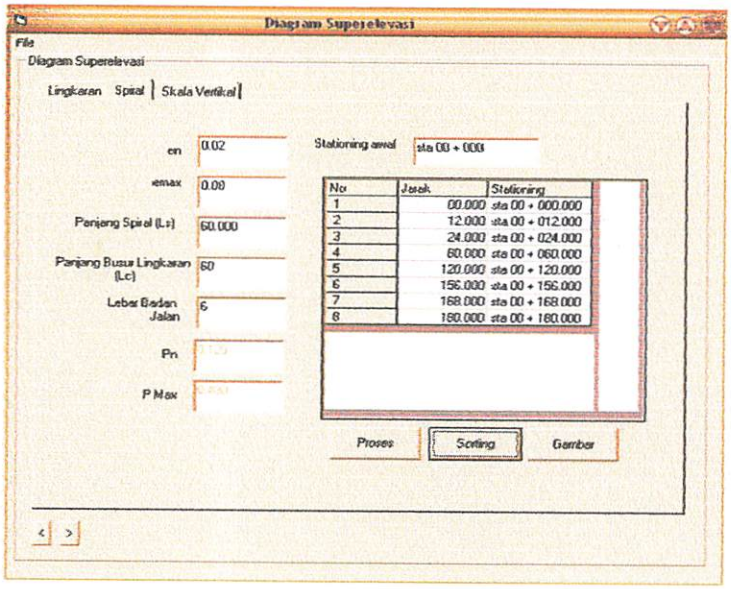
1. Tambahkan 10 kontrol *Text* untuk data input dan data output.
2. Tambahkan 10 kontrol *label*, untuk memberikan keterangan dalam menginput data dan melihat hasil output data.
3. Tambahkan 5 kontrol command sebagai perintah.
4. Tambahkan 1 kontrol MSFlex Grid untuk hasil stacking out.
5. Tambahkan Menu Editor untuk membuat menu bar (File) dengan sub menunya (New, Save, Close, Jump to Autocad (spiral) dan Jump to Autocad (lingkaran).
6. Tulislah kode program pada jendela code yang dapat dilihat pada lampiran listing program.
7. Hasil Pembuatan Form dapat dilihat pada gambar di bawah ini :



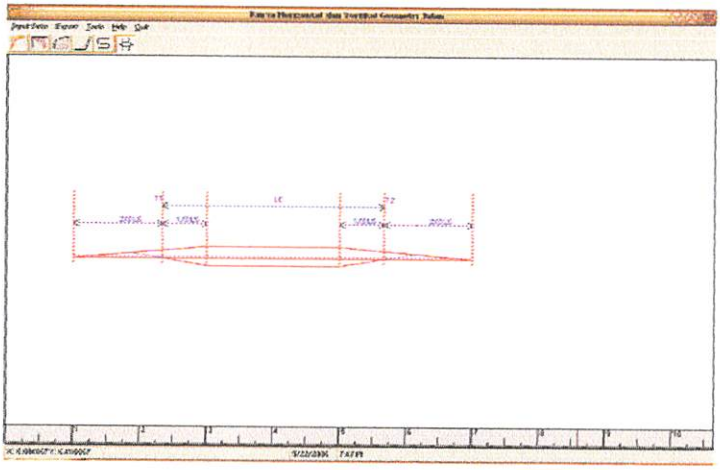
Gambar 3.24.
Tampilan form Diagram Superelevasi
Metode Lingkaran

Untuk Diagram Superelevasi metode Spiral, pembuatan form sebagai berikut:

1. Tambahkan 8 kontrol *Text* untuk data input dan data output
2. Tambahkan 8 kontrol *label*, untuk memberikan keterangan dalam menginput data dan melihat hasil output data
3. Tambahkan 5 kontrol command sebagai perintah.
4. Tambahkan 1 kontrol MSFlex Grid untuk hasil stacking out
5. Tulislah kode program pada jendela code yang dapat dilihat pada lampiran listing program.
6. Hasil Pembuatan Form dapat dilihat pada gambar di bawah ini :



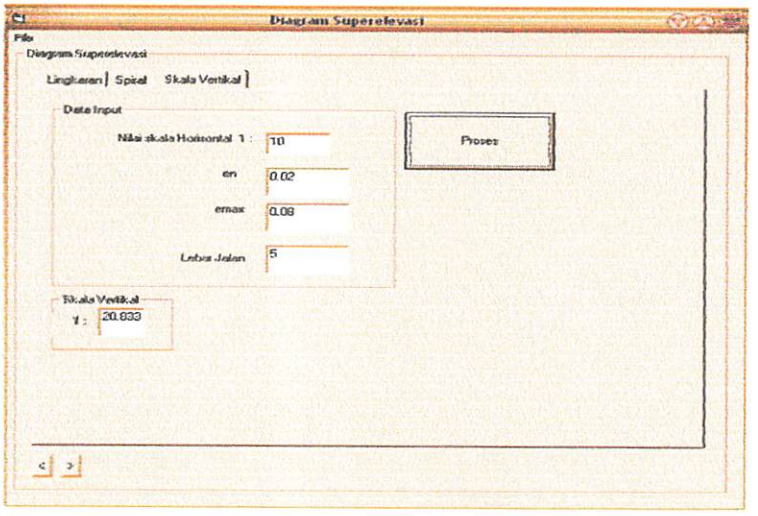
Gambar 3.25.
Tampilan form Diagram Superelevasi
Metode Spiral



Gambar 3.26.
Contoh Tampilan gambar Diagram Superelevasi

Untuk pembuatan skala vertikal, pembuatan form sebagai berikut:

1. Tambahkan 5 kontrol *Text* untuk data input dan data output
2. Tambahkan 5 kontrol *label*, untuk memberikan keterangan dalam menginput data dan melihat hasil output data
3. Tambahkan 1 kontrol command sebagai perintah.
4. Tambahkan 2 frame untuk mengelompokkan data input dan data output (skala vertikal).
5. Tulislah kode program pada jendela code yang dapat dilihat pada lampiran listing program.
6. Hasil Pembuatan Form dapat dilihat pada gambar di bawah ini :



Gambar 3.27.
Tampilan penentuan skala vertikal
pada Diagram Superelevasi

BAB IV
HASIL DAN PEMBAHASAN

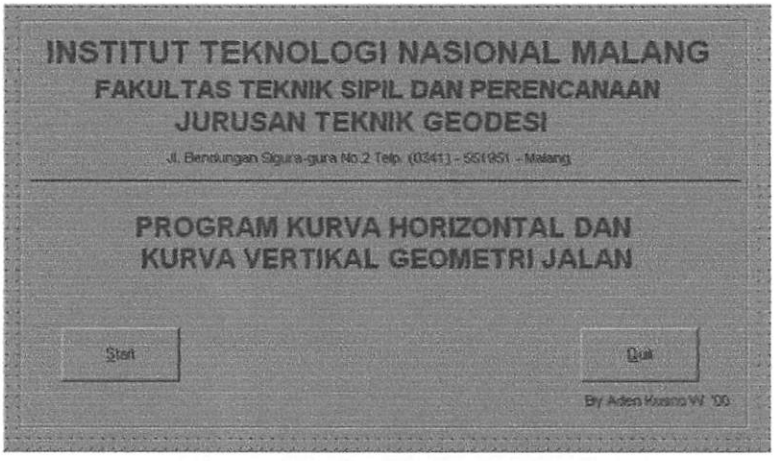
IV.1. Paket Program

Paket program yang akan dihasilkan adalah suatu paket program yang berfungsi untuk mengolah data hasil pengukuran ataupun dari data perencanaan. Paket program ini mempunyai kemampuan untuk penggambaran dari hasil perhitungan, dapat diexport ke Autocad sehingga dapat mudah dirubah atau diedit.

Paket program ini terdiri dari beberapa proses perhitungan yaitu :

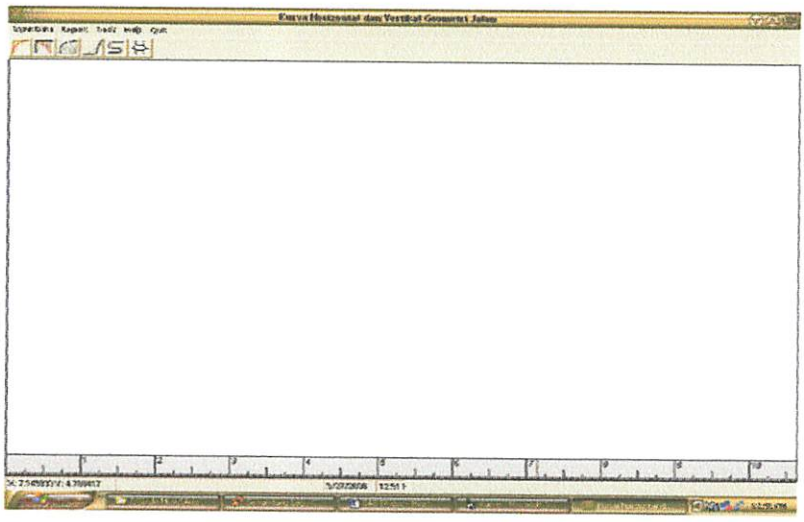
a. Perhitungan Titik Detail Kurva Horisontal (Metode Selisih Busur, Polar dan Poligon)
b. Perhitungan Titik Detail Kurva Vertikal
c. Perhitungan Titik Detail Kurva Spiral (Metode Sudut Defleksi)
d. Perhitungan Diagram Superelevasi

IV.1.1. Tampilan menu utama



Gambar 4.1.
Tampilan Utama Program

Gambar 4.1, merupakan tampilan utama dari program geometri jalan, pilih *Start* untuk memulai program.



Gambar 4.2.
Tampilan Menu Utama Program

Gambar 4.3.
Menu Bar dan Toolbar

Gambar 4.4.
Fasilitas – fasilitas yang terdapat padaMenu Bar

IV.1.2. Pemrosesan data

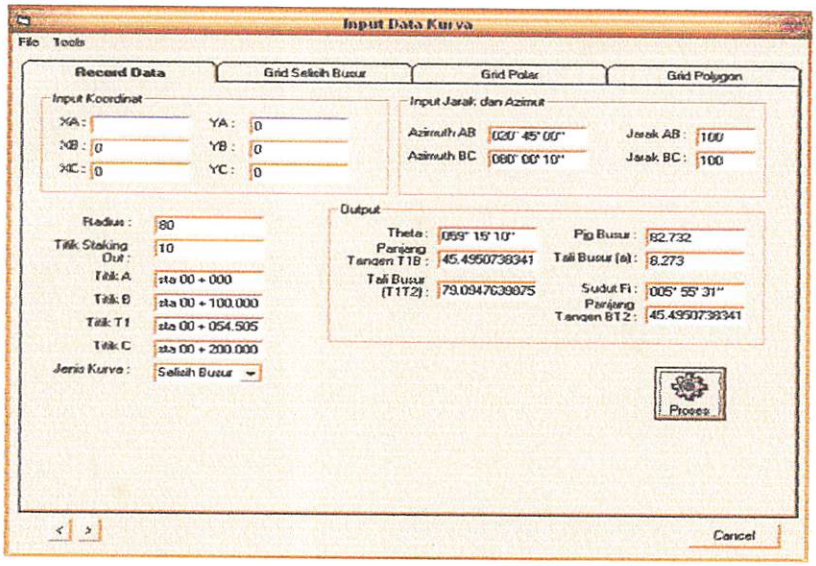
Pemrosesan data dilakukan pada setiap masing – masing form, mulai dari data input sampai data output (dalam bentuk tabel atau gambar).

Dalam penggambaran, apabila hasil gambar terlalu besar, maka dapat diatur dengan skala gambar, pada menu tool pilih skala (skala yang digunakan adalah dalam satuan inchi).

a. Kurva Horisontal

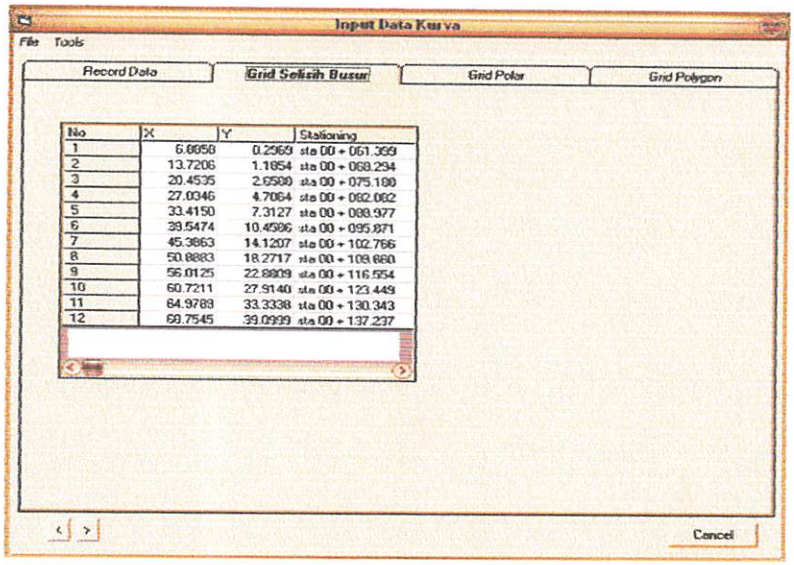
Input data dilakukan melalui menu *Input data* kemudian pilih input yang dikehendaki. Input data pada metode selisih busur, polar, dan poligon terdiri dari 2 cara :

1. Nilai koordinat (XA,YA,XB,YB,XC,YC), Radius dan Jumlah stacking out
2. Radius, Jumlah stacking out, Jarak AB, Jarak BC, Azimut AB, Azimut BC, kemudian tekan *proses* untuk melihat hasilnya :

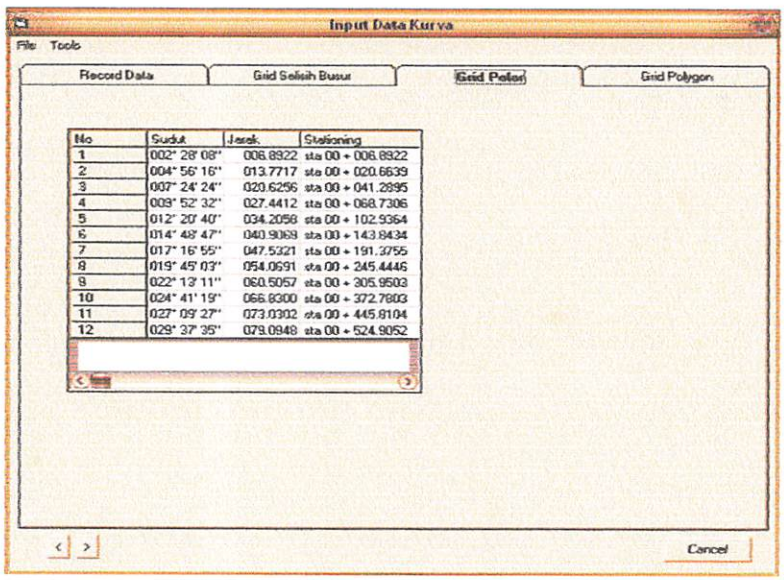


Gambar 4.5.
Hasil perhitungan kurva horisontal

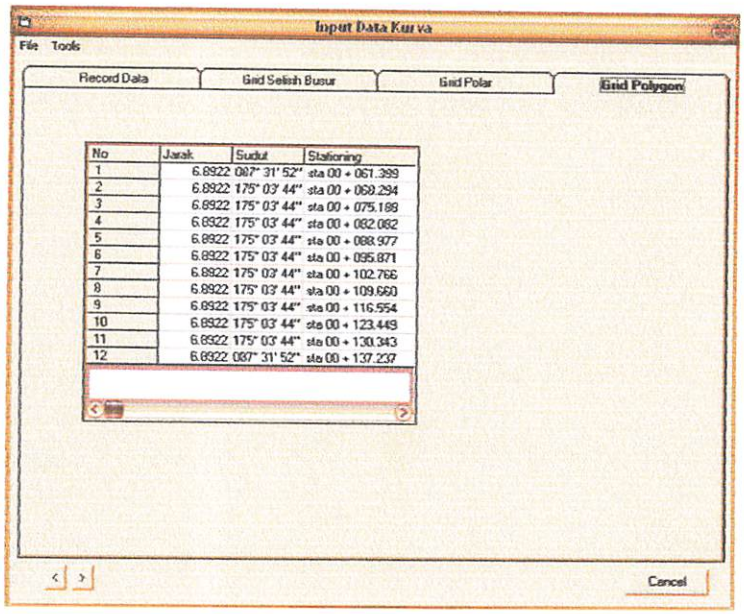
Dalam perhitungan kurva horisontal ini terdapat metode selisih busur, polar, poligon. Untuk memilih salah satu, gunakan pilihan pada jenis kurva, dan hasil stacking out dapat dilihat pada masing – masing tabel.



Gambar 4.6.
Hasil perhitungan staking out metode selisih busur

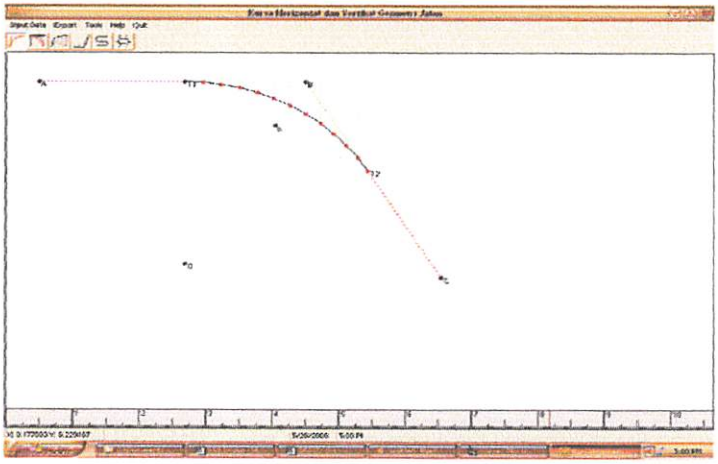


Gambar 4.7.
Hasil perhitungan staking out metode polar

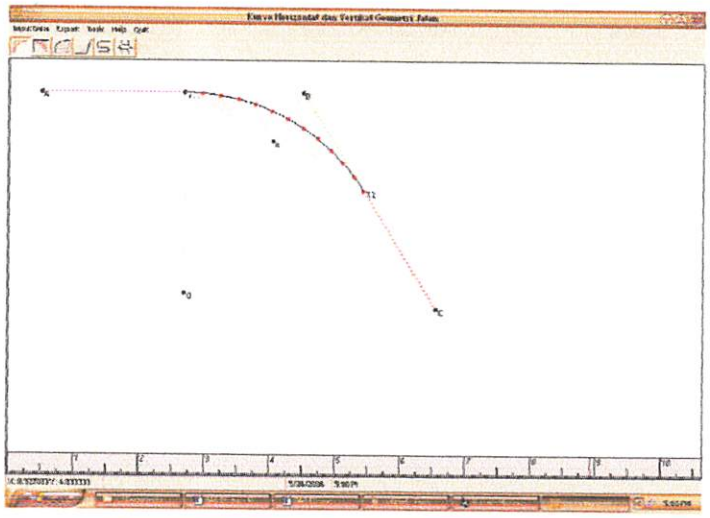


Gambar 4.8.
Hasil perhitungan metode poligon

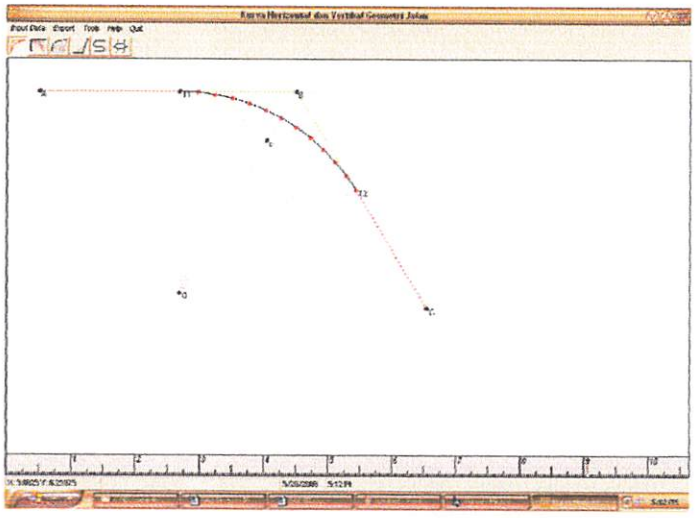
Cara untuk proses penggambaran pada kurva horisontal (metode selisih busur, polar, poligon, busur spiral) yaitu : pilih menu bar *tool* pilih *gambar* .



Gambar 4.9.
Hasil Output Kurva horisontal
Metode Selisih Busur



Gambar 4.10.
Hasil Output Kurva horisontal
Metode Polar



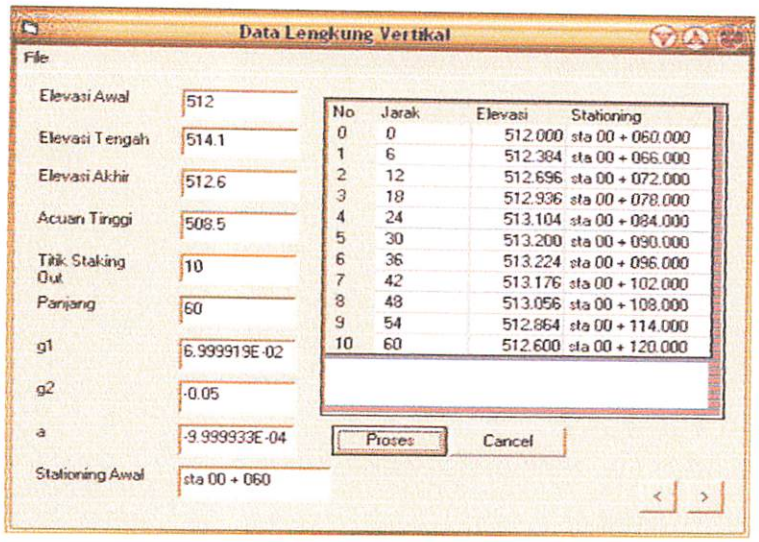
Gambar 4.11.
Hasil Output Kurva horisontal
Metode Poligon

b. Kurva Vertikal

Input data dilakukan dengan 2 cara yaitu :

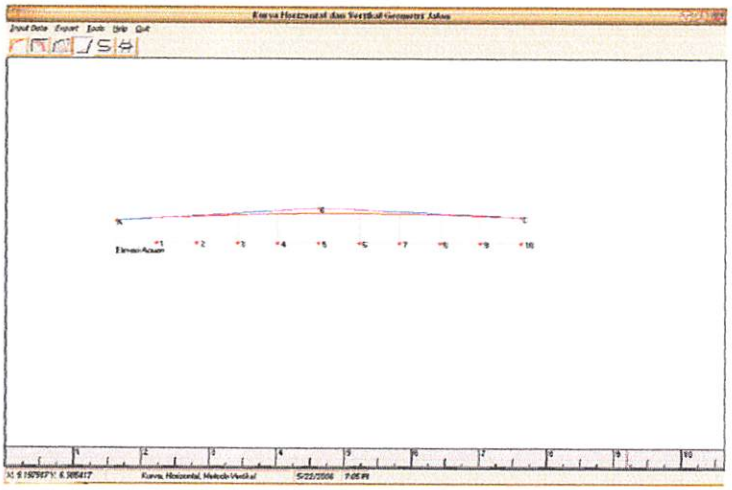
1. Elevasi awal, tengah, akhir, Elevasi Acuan, Panjang, Titik Stacking Out, dan Stationing awal.
2. Elevasi awal, g1, g2, , Elevasi Acuan, Panjang, Titik Stacking Out, dan Stationing awal.

kemudian tekan *proses* untuk melihat hasilnya :



Gambar 4.12.
Hasil output data Kurva Vertikal

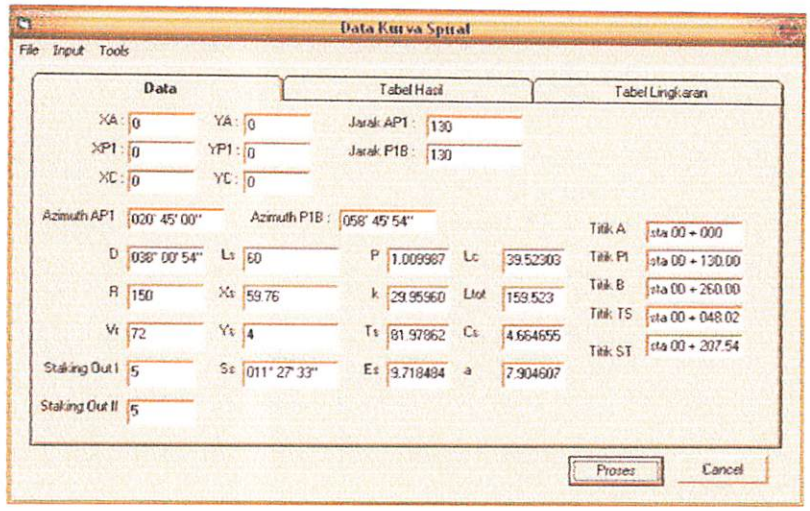
Untuk kurva vertikal klik disembarang tempat pada field gambar, kemudian klik yang kedua untuk melihat hasil gambar.



Gambar 4.13.
Hasil output gambar Kurva Vertikal

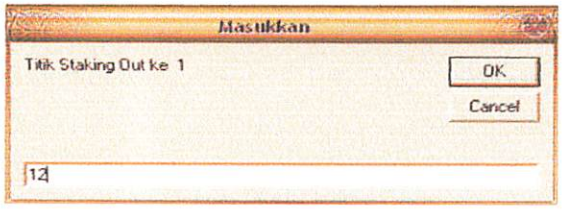
c. Busur Spiral

Data – data yang harus diisikan pada perhitungan ini adalah : Jarak, Azimuth, Jari – jari (R), Kecepatan rata – rata (Vr), stacking out I untuk jumlah titik pada busur spiral, stacking out II untuk jumlah titik pada busur lingkaran. Tekan *proses* untuk melihat hasil perhitungan.

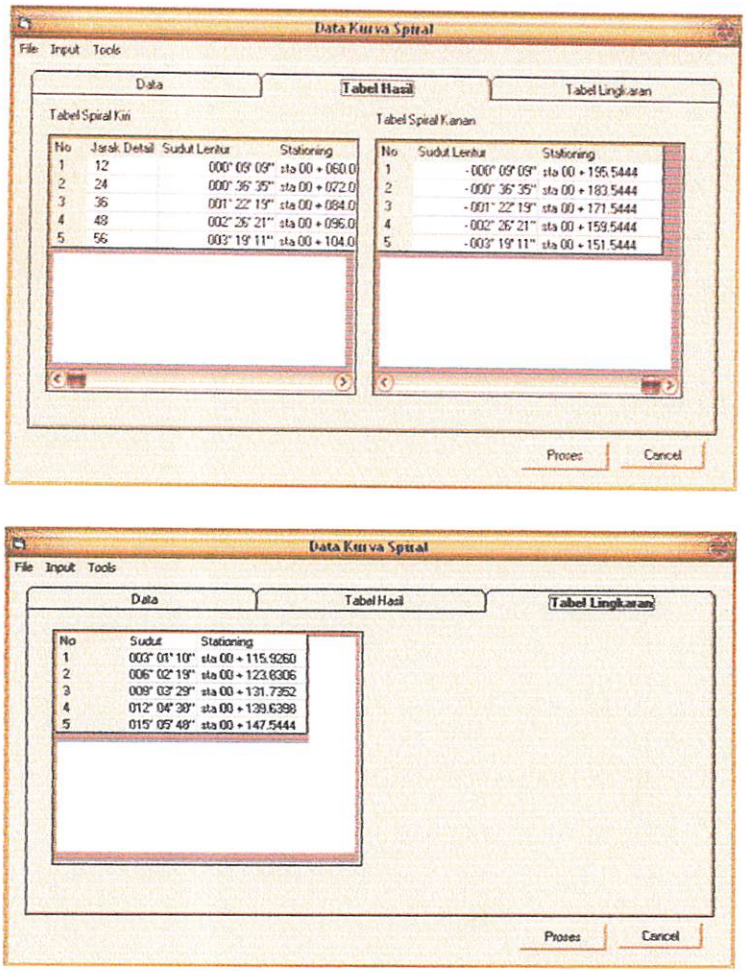


Gambar 4.14.
Hasil output data Busur Spiral

Pilih menu *input*, kemudian pilih *jarak*. Untuk nilai stacking out busur spiral dapat diisikan secara acak.

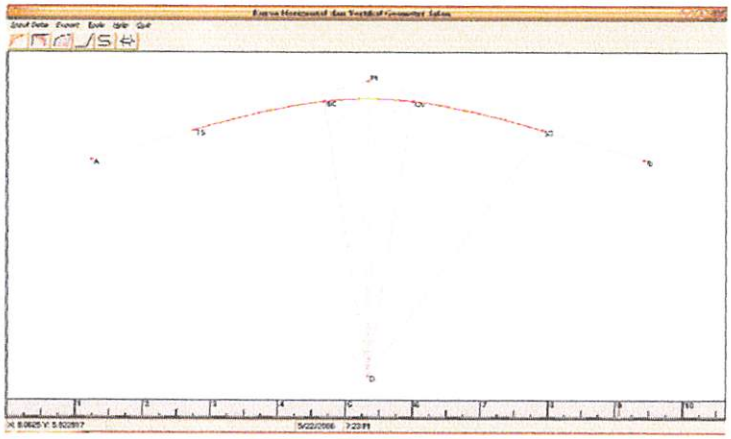


Gambar 4.15.
Pengisian titik stacking out secara acak



Gambar 4.16.
Hasil output stacking out Busur Spiral dan Busur Lingkaran

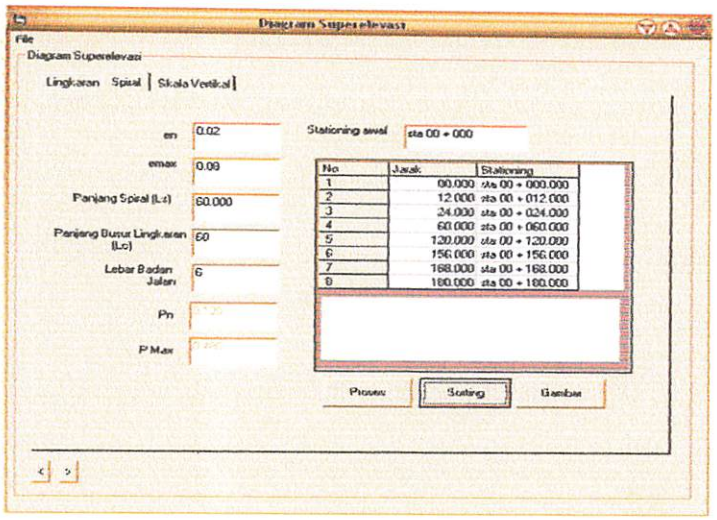
Cara untuk proses penggambaran pada Busur Spiral yaitu : pilih menu bar *tool* pilih *gambar*.



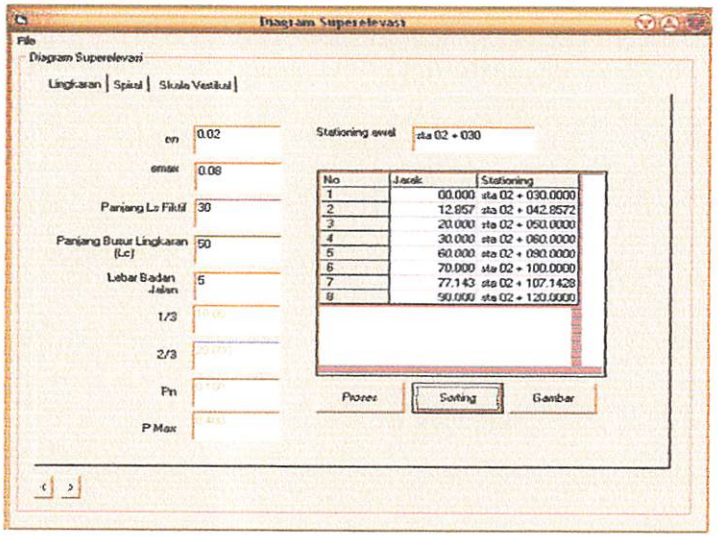
Gambar 4.17.
Hasil output Gambart Busur Spiral

d. Diagram Superelevasi

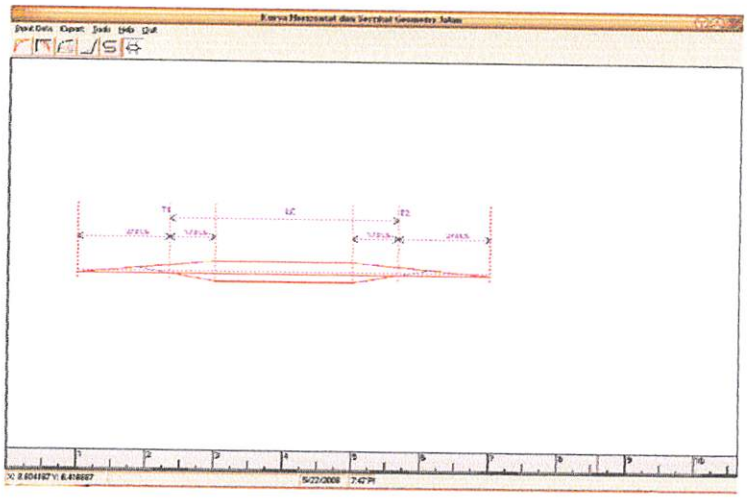
Pada tipe spiral isikan seluruhnya kecuali nilai Pn dan Pmax, sedangkantipe lingkaran isikan seluruhnya kecuali nilai Pn, Pmax, 1/3Ls, dan 2/3 Ls. Tekan proses untuk menghitung jarak dan stationing, kemudian pilih Sorting untuk memperbaiki susunan tabel. Tekan Gambar untuk melihat hasil gambar.



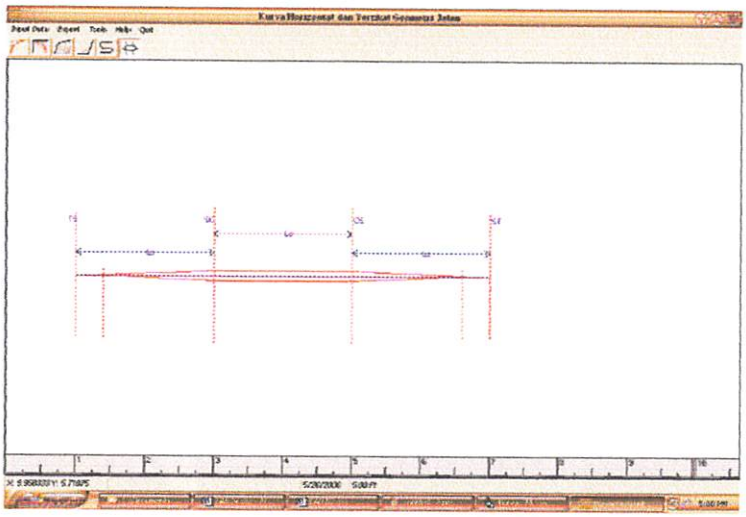
Gambar 4.18.
Contoh hasil output data Diagram Superelevasi
Tipe Spiral Circle Spiral



Gambar 4.19.
Contoh hasil output data Diagram Superelevasi
Tipe Full Circle

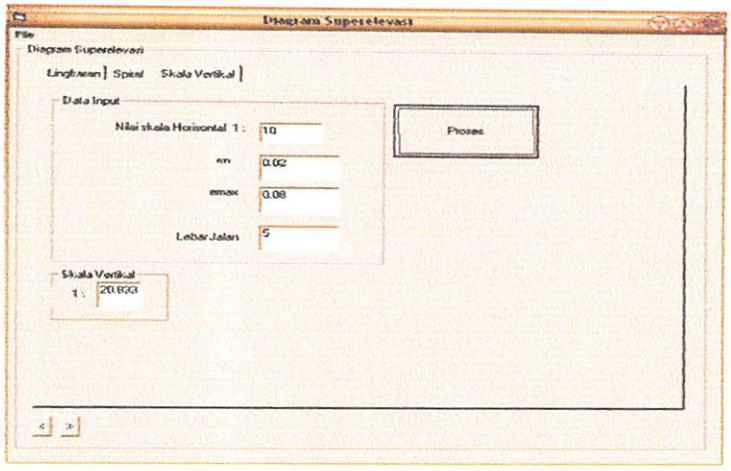


Gambar 4.20.
Contoh hasil output gambar Diagram Superelevasi
Tipe Lingkaran



Gambar 4.21.
Contoh hasil output gambar Diagram Superelevasi
Tipe Spiral

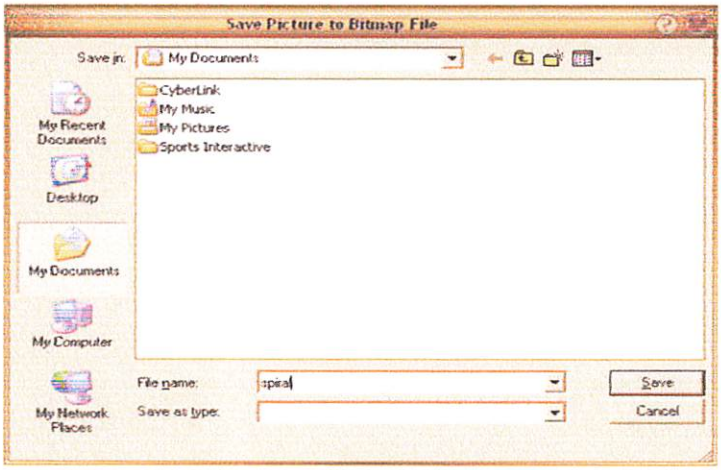
Pada Diagram Superelevasi, memiliki skala vertikal untuk menentukan nilai lebar jalan pada gambar. Data inputnya yaitu : skala horizontal, e_n, e_{max}, dan lebar jalan. Tekan proses untuk hasil perhitungan skala vertikal.



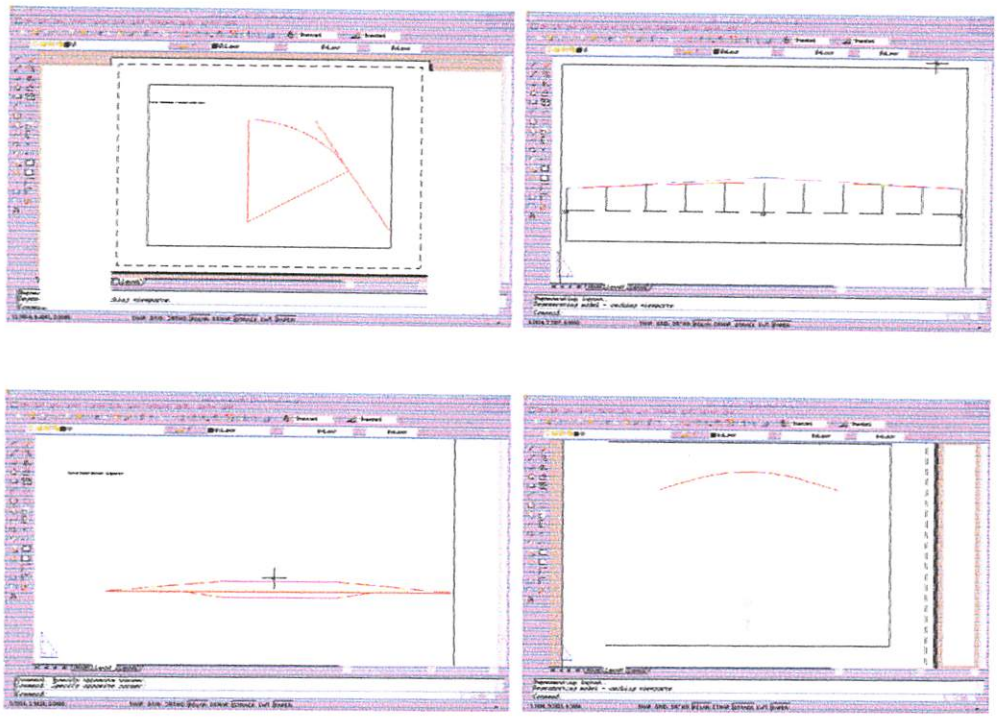
Gambar 4.22.
Perhitungan untuk penentuan skala vertikal

Gambar diatas menunjukkan bahwa skala vertikal dari Digram Superelevasi adalah 1 : 20.833 (1 inchi = 20,833 meter)

Gambar yang telah dihasilkan dapat disimpan dalam format *BMP atau diexport ke Autocad



Gambar 4.23.
Penyimpanan gambar dalam format *BMP



Gambar 4.24.
Hasil gambar yang telah diexport ke Autocad

IV.2. Uji Ketelitian

Uji ketelitian ini digunakan untuk mengetahui kebenaran hasil perhitungan, maka hasil perhitungan perlu dibandingkan dengan perhitungan dengan program lain dalam hal ini yang digunakan adalah Microsoft Excel.

IV.2.1. Uji ketelitian stacking out metode selisih busur sama panjang

Uji ketelitian pada perhitungan selisih busur sama panjang antara perhitungan Excel dengan Visual Basic 6.0 adalah sebagai berikut :

Excel

X	Y
8.25928	0.42749
16.43029	1.70539
24.42571	3.82005
32.16008	6.74886
39.55075	10.46053
46.51873	14.91538
52.98956	20.06581
58.89407	25.85677
64.16916	32.22638
68.75846	39.10655

Visual Basic 6.0

No	X	Y	Stationing
1	8.2581	0.4274	sta 00+062.780
2	16.4280	1.7049	sta 00+071.053
3	24.4224	3.8190	sta 00+079.326
4	32.1558	6.7470	sta 00+087.599
5	39.5457	10.4577	sta 00+095.872
6	46.5131	14.9113	sta 00+104.145
7	52.9835	20.0604	sta 00+112.418
8	58.8878	25.8499	sta 00+120.690
9	64.1629	32.2180	sta 00+128.963
10	68.7525	39.0966	sta 00+137.236

Tabel 4.1.
Hasil perbandingan perhitungan antara Excel dengan
Visual Basic 6.0 pada metode selisih busur sama panjang

IV.2.2. Uji ketelitian stacking out metode polar

Uji ketelitian pada perhitungan metode polar antara perhitungan Excel dengan Visual Basic 6.0 adalah sebagai berikut :

Excel

Jarak	Sudut
8.27033697	2^0 57' 46.5"
16.51856235	5^0 55' 33"
24.72262366	8^0 53' 19.5"
32.8605865	11^0 51' 06"
40.9106932	14^0 48' 52.5"
48.85142097	17^0 46' 39"
56.66153946	20^0 44' 22.5"
64.32016753	23^0 42' 12"
71.80682904	26^0 39' 58.5"
79.10150764	29^0 37' 45"

Visual Basic 6.0

No	Sudut	Jarak	Stationing
1	002° 57' 45"	008.2692	sta 00 + 008.2692
2	005° 55' 30"	016.5162	sta 00 + 024.7854
3	008° 53' 15"	024.7192	sta 00 + 049.5046
4	011° 51' 00"	032.8560	sta 00 + 082.3606
5	014° 48' 45"	040.9051	sta 00 + 123.2657
6	017° 46' 30"	048.8448	sta 00 + 172.1105
7	020° 44' 15"	056.6539	sta 00 + 228.7644
8	023° 42' 00"	064.3116	sta 00 + 293.0760
9	026° 39' 45"	071.7975	sta 00 + 364.8735
10	029° 37' 30"	079.0914	sta 00 + 443.9649

Tabel 4.2.
Hasil perbandingan perhitungan antara Excel dengan
Visual Basic 6.0 pada metode polar

IV.2.3. Uji ketelitian stacking out metode poligon

Uji ketelitian pada perhitungan metode poligon antara perhitungan Excel dengan Visual Basic 6.0 adalah sebagai berikut :

Excel

Sudut	Jarak
87^0 02' 13.5"	8.27034
174^0 04' 26.76"	8.27034
174^0 04' 26.76"	8.27034
174^0 04' 26.76"	8.27034
174^0 04' 26.76"	8.27034
174^0 04' 26.76"	8.27034
174^0 04' 26.76"	8.27034
174^0 04' 26.76"	8.27034
174^0 04' 26.76"	8.27034
87^0 02' 13.5"	8.27034

Visual Basic 6.0

No	Jarak	Sudut	Stationing
1	8.2692	087° 02' 15"	sta 00 + 062.780
2	8.2692	174° 04' 30"	sta 00 + 071.053
3	8.2692	174° 04' 30"	sta 00 + 079.326
4	8.2692	174° 04' 30"	sta 00 + 087.599
5	8.2692	174° 04' 30"	sta 00 + 095.872
6	8.2692	174° 04' 30"	sta 00 + 104.145
7	8.2692	174° 04' 30"	sta 00 + 112.418
8	8.2692	174° 04' 30"	sta 00 + 120.690
9	8.2692	174° 04' 30"	sta 00 + 128.963
10	8.2692	087° 02' 15"	sta 00 + 137.236

Tabel 4.3.
Hasil perbandingan perhitungan antara Excel dengan
Visual Basic 6.0 pada metode poligon

IV.2.4. Uji ketelitian stacking out kurva vertikal

Uji ketelitian pada perhitungan kurva vertikal antara perhitungan Excel dengan Visual Basic 6.0 adalah sebagai berikut :

Excel

Elevasi	Jarak
512.00000	0
512.42222	6.6667
512.75555	13.3333
513.00000	20
513.15556	26.6667
513.22222	33.3333
513.20000	40
513.08889	46.6667
512.88889	53.3334
512.60000	60

Visual Basic 6.0

No	Jarak	Elevasi	Stationing
0	0	512.000	sta 00 + 060.000
1	6.666667	512.422	sta 00 + 066.667
2	13.33333	512.756	sta 00 + 073.333
3	20	513.000	sta 00 + 080.000
4	26.66667	513.156	sta 00 + 086.667
5	33.33333	513.222	sta 00 + 093.333
6	40	513.200	sta 00 + 100.000
7	46.66667	513.089	sta 00 + 106.667
8	53.33334	512.889	sta 00 + 113.333
9	60	512.600	sta 00 + 120.000

Tabel 4.4.
Hasil perbandingan perhitungan antara Excel dengan
Visual Basic 6.0 pada kurva vertikal

IV.2.3. Uji ketelitian stacking out busur spiral

Uji ketelitian pada perhitungan busur spiral antara perhitungan Excel dengan Visual Basic 6.0 adalah sebagai berikut :

Excel

jarak	Sudut lentur
12	0^0 9' 9.42"
24	0^0 36' 37.71"
36	1^0 22' 24.85"
48	2^0 26' 30.84"
60	3^0 48' 55.69"

Visual Basic 6.0

No	Jarak Detail	Sudut Lentur	Stationing
1	12	000° 09' 09"	sta 00 + 060.0
2	24	000° 36' 35"	sta 00 + 072.0
3	36	001° 22' 19"	sta 00 + 084.0
4	48	002° 26' 21"	sta 00 + 096.0
5	60	003° 48' 40"	sta 00 + 108.0

Tabel 4.5.
Hasil perbandingan perhitungan antara Excel dengan
Visual Basic 6.0 pada Busur Spirall

Dari hasil perhitungan tersebut dapat dianalisa bahwa hasil perhitungan program dan hasil hitungan Microsoft Excel adalah sebagai berikut :

No.	Metode Selisih Busur Sama Panjang	
	X	Y
1	0.00118	0.00009
2	0.00229	0.00049
3	0.00331	0.00105
4	0.00428	0.00186
5	0.00505	0.00283
6	0.00563	0.00408
7	0.00606	0.00541
8	0.00627	0.00687
9	0.00626	0.00838
10	0.00596	0.00995

Tabel 4.6
Hasil perbandingan atau selisih dari perhitungan antara Excel dengan Visual Basic 6.0 pada metode selisih busur sama panjang

No.	Metode Polar	
	Jarak (m)	Sudut
1	0.00113	0° 0' 1.5"
2	0.00236	0° 0' 3"
3	0.00342	0° 0' 4.5"
4	0.00458	0° 0' 6"
5	0.00559	0° 0' 7.5"
6	0.00662	0° 0' 9"
7	0.00763	0° 0' 10.5"
8	0.00856	0° 0' 12"
9	0.00932	0° 0' 13.5"
10	0.0096	0° 0' 15"

Tabel 4.7
Hasil perbandingan atau selisih dari perhitungan jarak dan sudut antara Excel dengan Visual Basic 6.0 pada metode polar

No.	Metode Poligon	
	Jarak (m)	Sudut
1	0.00114	0° 0' 1.5"
2	-	0° 0' 3.24 "

Tabel 4.8
Hasil perbandingan atau selisih dari perhitungan jarak dan sudut antara Excel dengan Visual Basic 6.0 pada metode poligon

No.	Kurva Vertikal	
	Jarak	Elevasi
1	0	0.00000
2	6.6667	0.00022
3	13.3333	-0.00045
4	20	0.00000
5	26.6667	-0.00044
6	33.3333	0.00022
7	40	0.00000
8	46.6667	-0.00011
9	53.3334	-0.00011
10	60	0.00000

Tabel 4.9
Hasil perbandingan atau selisih dari perhitungan elevasi antara Excel dengan Visual Basic 6.0 pada kurva vertikal

No.	Busur Spiral	
	Jarak (m)	Sudut Lentur
1	12	0° 0' 0.42"
2	24	0° 0' 2.71"
3	36	0° 0' 5.85"
4	48	0° 0' 9.84"
5	60	0° 0' 15.69"

Tabel 4.10
Hasil perbandingan atau selisih dari perhitungan sudut lentur antara Excel dengan Visual Basic 6.0 pada Busur Spiral

BAB V
PENUTUP

V.1. Kesimpulan

1. Software dari hasil penelitian ini dapat dipergunakan untuk menghitung beberapa perencanaan geometri jalan antara lain :
   a. Perhitungan stacking out kurva horisontal metode selisih busur sama panjang.
   b. Perhitungan stacking out kurva horisontal metode polar.
   c. Perhitungan stacking out kurva horisontal metode poligon.
   d. Perhitungan stacking out kurva vertical.
   e. Perhitungan stacking out Busur Spiral metode Spiral Circle Spiral.
   f. Perhitungan Diagram Superelevasi.
2. Hasil program yang berupa gambar dapat disimpan dalam format *BMP atau diexport ke Autocad, sedangkan yang berupa data dapat langsung di simpan ke Ms Acces atau dari Acces diexport ke Excel untuk dicetak.
3. Hasil perbandingan dari perhitungan Visual basic 6.0 dengan Ms Excel memiliki selisih jarak antara 0.001 m – 0.009 m, dan selisih sudut antara $0^0\ 0'\ 0" - 0^0\ 0'\ 16"$, sehingga program tersebut sudah dianggap benar.

V.2. Saran

1. Sebelum melakukan proses perhitungan, sebaiknya harus sudah dilengkapi dengan Ms Acces dan Autocad 2004.
2. Software ini bisa dikembangkan kembali, misalnya dengan pembuatan kurva lebih dari 1 lengkungan.

DAFTAR PUSTAKA

Alamsyah, Ansyori, Alik, Ir.,2001, *Rekayasa Jalan Raya*, Edisi Pertama, UMM Pres, Malang.

B.S.Wirshing, R.James, and B.I.E. Wirshing, H.Roy, 1995, *Teori dan Soal – Soal Pengantar Pemetaan*, Erlangga, Jakarta.

Brinker, C.Russel, Paul, R.Wolf, Djoko, Walijatun, 1997, *Dasar – dasar Pengukuran Tanah (Surveying)*, Edisi VII jilid 2, Erlangga, Jakarta.

Clarkson, H.Oglesby, and Hicks, R.Gary, 1997, *Teknik Jalan Raya*,Edisi IV jilid 1, Erlangga, Jakarta.

Dewobroto, Wiryanto, Ir., 2003, *Aplikasi Sain dan Teknik dengan Visual Basic 6.0*, P.T. Media Komputindo, Jakarta.

Mangkulo, Alexander, Hengky, 2003, *Membangun Sistem Database dengan Visual Basic 6.0 dan Acces 2000*, P.T. Elex Media Komputindo, Jakarta.

Meyer, F.Carl, and David, W. Gibson, 1984, *Survei dan Perencanaan Lintas Jalur*, Edisi V, Erlangga, Jakarta.

Sinaga, Indra, Ir., 1992, *Pengukuran dan Pemetaan Pekerjaan Konstruksi*, Pustaka Sinar Harapan, Jakarta.

S, Hendriatiningsih, *Geometris Jalan Raya dan Stake Out*, Edisi II, Jurusan Geodesi FTSP, ITB, Bandung.

Sukirman, Silvia, 1999, *Dasar – dasar Perencanaan Geometrik Jalan*, Nova, Bandung.

Wongsotjitro, Soetomo, 1980, *Ilmu Ukur Tanah*, Kanisius, Yogyakarta.

LISTING PROGRAM

LISTING PROGRAM

TAMPILAN AWAL

```
Option Explicit

Private Sub Command1_Click()
 Me.Hide
 MainFrm.Show
End Sub

Private Sub Command2_Click()
Unload Me
End Sub

Private Sub Form_KeyPress(KeyAscii As Integer)
   Unload Me
End Sub
```

TAMPILAN FORM UTAMA

```
Option Explicit

Dim PosX, oldx, x1 As Single
Dim PosY, oldy As Single
Dim ClickIndex As Byte
Dim Drawing As Boolean
Dim DrawComplete As Boolean
Dim lastX As Single, lastY As Single
Dim PanelText(1 To 15) As String

'Create the Ruler
Private Sub DrawRuler(MSize)
   Dim Sincr As Single, RScale As Integer
   Dim i As Integer
   'Scalemode is in TWIPS 1440 per inch
   RScale = 1440
   'Number of segment across form
   Sincr = RScale / MSize
   Do While Sincr < HRuler.ScaleWidth
      'Number of sections
      For i = 1 To MSize
         'Size of Tics
         If i = MSize Then
            HRuler.Line (Sincr, 0)-(Sincr, HRuler.ScaleHeight)
            HRuler.CurrentY = 0
            HRuler.Print Int(Sincr / RScale)
         Else
            If i = Int(MSize * 0.75) Then
```

```
            HRuler.Line (Sincr, HRuler.ScaleHeight - (HRuler.ScaleHeight *
0.25))-(Sincr, HRuler.ScaleHeight)
          Else
            If i = Int(MSize * 0.5) Then
              HRuler.Line (Sincr, HRuler.ScaleHeight - (HRuler.ScaleHeight *
0.5))-(Sincr, HRuler.ScaleHeight)
            Else
              If i = MSize * 0.25 Then
                HRuler.Line (Sincr, HRuler.ScaleHeight - (HRuler.ScaleHeight
* 0.25))-(Sincr, HRuler.ScaleHeight)
              Else
                HRuler.Line (Sincr, HRuler.ScaleHeight - (HRuler.ScaleHeight
* 0.125))-(Sincr, HRuler.ScaleHeight)
              End If
            End If
          End If
        End If
        Sincr = Sincr + (RScale / MSize)
      Next
    Loop
End Sub
Private Sub gambar_titik(X As Integer, Y As Integer, color As Integer)
Pic.PSet (X, Y), QBColor(color)
End Sub
Private Sub gambar_garis(x1 As Integer, Y1 As Integer, x2 As Integer, Y2 As
Integer, color As Integer)
Pic.Line (x1, Y1)-(x2, Y2), QBColor(color)
End Sub
Private Sub hotSpot(ByVal X As Single, ByVal Y As Single, ByVal Text As
String)
Pic.DrawWidth = 4
Pic.PSet (X, Y), QBColor(12)
Pic.DrawWidth = 1
Pic.DrawStyle = vbSolid
Pic.CurrentX = X + 25
Pic.CurrentY = Y - 50
Pic.Print Text
Pic.DrawStyle = vbDot
Pic.Line (X, Y)-(X, Y - 1200), QBColor(7)
Pic.DrawStyle = vbSolid
End Sub
Private Sub hotText(ByVal X As Single, ByVal Y As Single, ByVal Text As
String)
Pic.DrawWidth = 4
Pic.PSet (X, Y), QBColor(12)
Pic.DrawWidth = 1
Pic.DrawStyle = vbSolid
```

```
HRuler.Line (Sincr, HRuler.ScaleHeight - (HRuler.ScaleHeight *
0.25))-(Sincr, HRuler.ScaleHeight)
Else
If i = Int(MSize * 0.5) Then
HRuler.Line (Sincr, HRuler.ScaleHeight - (HRuler.ScaleHeight *
0.5))-(Sincr, HRuler.ScaleHeight)
Else
If i = MSize * 0.25 Then
HRuler.Line (Sincr, HRuler.ScaleHeight - (HRuler.ScaleHeight
* 0.25))-(Sincr, HRuler.ScaleHeight)
Else
HRuler.Line (Sincr, HRuler.ScaleHeight - (HRuler.ScaleHeight
* 0.125))-(Sincr, HRuler.ScaleHeight)
End If
End If
End If
End If
Sincr = Sincr + (RScale / MSize)
Next
Loop
End Sub
Private Sub gambar_titik(X As Integer, Y As Integer, color As Integer)
Pic.PSet (X, Y), QBColor(color)
End Sub
Private Sub gambar_garis(x1 As Integer, Y1 As Integer, x2 As Integer, Y2 As
Integer, color As Integer)
Pic.Line (x1, Y1)-(x2, Y2), QBColor(color)
End Sub
Private Sub hotSpot(ByVal X As Single, ByVal Y As Single, ByVal Text As
String)
Pic.DrawWidth = 4
Pic.PSet (X, Y), QBColor(12)
Pic.DrawWidth = 1
Pic.DrawStyle = vbSolid
Pic.CurrentX = X + 25
Pic.CurrentY = Y - 50
Pic.Print Text
Pic.DrawStyle = vbDot
Pic.Line (X, Y)-(X, Y - 1200), QBColor(7)
Pic.DrawStyle = vbSolid
End Sub
Private Sub hotText(ByVal X As Single, ByVal Y As Single, ByVal Text As
String)
Pic.DrawWidth = 4
Pic.PSet (X, Y), QBColor(12)
Pic.DrawWidth = 1
Pic.DrawStyle = vbSolid
```

```
Pic.CurrentX = X + 25
Pic.CurrentY = Y - 50
Pic.Print Text

End Sub
Private Sub Form_Load()
On Error Resume Next
'   HRuler.Width = MainFrm.ScaleWidth - HRuler.Left
    MainFrm.Show
      'Draw Ruler 16ths of an inch
    DrawRuler 32
    ClickIndex = 0
    PropertiesFrm.Show
    DrawComplete = False
    Drawing = False
    PanelText(1) = "Kurva, Horizontal, Metode Selisih Busur"
    PanelText(2) = "Kurva, Horizontal, Metode Polar"
    PanelText(3) = "Kurva, Horizontal, Metode Polygon"
    PanelText(4) = "Kurva, Horizontal, Metode Vertikal"
    PanelText(5) = "Kurva, Horizontal, Metode Spiral"
    Position(0).X = 0
    Position(0).Y = 0
    TextPoint(1) = "A"
    TextPoint(2) = "B"
    TextPoint(3) = "C"
    MaxPoint = 3
    GScale = 10
    DEKurva.Conn.Open
  App.HelpFile = App.Path & "\kurva.hlp"
End Sub

Private Sub mnuAbout_Click()
frmAbout.Show
End Sub

Private Sub mnuData_Click()
DataFrm.Show
End Sub


Private Sub GambarKeAutoCAd()
    On Error Resume Next
    Dim i, j As Integer
    Dim strCmd As String
    Dim LineObj As AcadLine
    Dim dimObj As AcadDimAligned
    Dim textObj As AcadText
```

```
Pic.CurrentX = X + 25
Pic.CurrentY = Y - 50
Pic.Print Text

End Sub
Private Sub Form_Load()
On Error Resume Next
    HRuler.Width = MainFrm.ScaleWidth - HRuler.Left
    MainFrm.Show
    'Draw Ruler 16ths of an inch
    DrawRuler 32
    ClickIndex = 0
    PropertiesFrm.Show
    DrawComplete = False
    Drawing = False
    PanelText(1) = "Kurva, Horizontal, Metode Selisih Busur"
    PanelText(2) = "Kurva, Horizontal, Metode Polar"
    PanelText(3) = "Kurva, Horizontal, Metode Poligon"
    PanelText(4) = "Kurva, Horizontal, Metode Vertikal"
    PanelText(5) = "Kurva, Horizontal, Metode Spiral"
    Position(0).X = 0
    Position(0).Y = 0
    TextPoint(1) = "A"
    TextPoint(2) = "B"
    TextPoint(3) = "C"
    MaxPoint = 3
    GScale = 16
    DEKurva.Conn.Open
    App.HelpFile = App.Path & "\Kurva.hlp"
End Sub

Private Sub mnuAbout_Click()
frmAbout.Show
End Sub

Private Sub mnuData_Click()
DataFrm.Show
End Sub


Private Sub GambarKeAutoCAD()
    On Error Resume Next
    Dim i, j As Integer
    Dim strCmd As String
    Dim LineObj As AcadLine
    Dim dimObj As AcadDimAligned
    Dim textObj As AcadText
```

```
Dim startPoint(0 To 2) As Double
Dim endPoint(0 To 2) As Double
Dim basePoint(0 To 2) As Double
Dim TAcuan(0 To 2) As Double
Dim location(0 To 2) As Double
Dim color As AcadAcCmColor

Dim entry As AcadLineType
Dim found As Boolean
Dim arText(0 To 4) As String
Dim x1, Y1, x2, Y2 As Double
Dim lai As Integer

arText(1) = "XA"
arText(2) = "XB"
arText(3) = "XC"
found = False

Set AcadApp = GetObject("AutoCAD.Application")
If Err Then
'MsgBox Err.Description
Set AcadApp = CreateObject("AutoCAD.Application")

End If

Set AcadDoc = AcadApp.ActiveDocument

AcadDoc.WindowState = acMax
Set Lays = AcadDoc.Layers.Add("Elevasi")
Set color = AcadApp.GetInterfaceObject("AutoCAD.AcCmColor.16")
Call color.SetRGB(0, 0, 255)
Lays.TrueColor = color

Set Lays = AcadDoc.Layers.Add("Garis")
Set color = AcadApp.GetInterfaceObject("AutoCAD.AcCmColor.16")
Call color.SetRGB(255, 0, 0)
Lays.TrueColor = color

Set Lays = AcadDoc.Layers.Add("Helper")
Set color = AcadApp.GetInterfaceObject("AutoCAD.AcCmColor.16")
Call color.SetRGB(209, 209, 209)
Lays.TrueColor = color
Lays.Linetype = "DASHED"
AcadDoc.Regen (acActiveViewport)

For Each entry In AcadDoc.Linetypes
   If StrComp(entry.Name, "DASHED", 1) = 0 Then
```

```
Dim startPoint(0 To 2) As Double
Dim endPoint(0 To 2) As Double
Dim basePoint(0 To 2) As Double
Dim TAcuan(0 To 2) As Double
Dim location(0 To 2) As Double
Dim color As AcadAcCmColor

Dim entry As AcadLineType
Dim found As Boolean
Dim arText(0 To 4) As String
Dim x1, Y1, x2, Y2 As Double
Dim lai As Integer

arText(1) = "XA"
arText(2) = "XB"
arText(3) = "XC"
found = False

Set AcadApp = GetObject("AutoCAD.Application")
If Err Then
'MsgBox Err.Description
Set AcadApp = CreateObject("AutoCAD.Application")

End If

Set AcadDoc = AcadApp.ActiveDocument

AcadDoc.WindowState = acMax
Set Lays = AcadDoc.Layers.Add("Elevasi")
Set color = AcadApp.GetInterfaceObject("AutoCAD.AcCmColor.16")
Call color.SetRGB(0, 0, 255)
Lays.TrueColor = color

Set Lays = AcadDoc.Layers.Add("Garis")
Set color = AcadApp.GetInterfaceObject("AutoCAD.AcCmColor.16")
Call color.SetRGB(255, 0, 0)
Lays.TrueColor = color

Set Lays = AcadDoc.Layers.Add("Helper")
Set color = AcadApp.GetInterfaceObject("AutoCAD.AcCmColor.16")
Call color.SetRGB(209, 209, 209)
Lays.TrueColor = color
Lays.Linetype = "DASHED"
AcadDoc.Regen (acActiveViewport)

For Each entry In AcadDoc.Linetypes
 If StrComp(entry.Name, "DASHED", 1) = 0 Then
```

```
      found = True
      Exit For
    End If
Next
If Not (found) Then AcadDoc.Linetypes.Load "DASHED", "acad.lin"


Select Case Operation
Case Is <= 3:
Open App.Path & "\horizontal.txt" For Input As #1

location(0) = 0
location(1) = 0
location(2) = 0
If Operation = 1 Then
Set textObj = AcadDoc.ModelSpace. _
        AddText("Kurva  Horizontal - Metode Selisih Busur", location, 1)
textObj.Update
ElseIf Operation = 2 Then
Set textObj = AcadDoc.ModelSpace. _
        AddText("Kurva  Horizontal - Metode Polar", location, 1)
textObj.Update
ElseIf Operation = 3 Then
Set textObj = AcadDoc.ModelSpace. _
        AddText("Kurva  Horizontal - Metode Polygon", location, 1)
textObj.Update
End If

While Not EOF(1)
Input #1, x1, Y1, x2, Y2, lai
startPoint(0) = x1 * GScale / 1440
startPoint(1) = -Y1 * GScale / 1440
startPoint(2) = 0
endPoint(0) = x2 * GScale / 1440
endPoint(1) = -Y2 * GScale / 1440
endPoint(2) = 0
Set LineObj = AcadDoc.ModelSpace.AddLine(startPoint, endPoint)
If lai = 1 Then
LineObj.Layer = "Garis"
Else
LineObj.Layer = "Helper"
LineObj.Linetype = "DASHED"
End If
Wend
Close #1

Case Is = 4
```

```
                found = True
                Exit For
            End If
        Next
    If Not (found) Then AcadDoc.Linetypes.Load "DASHED", "acad.lin"

    Select Case Operation
    Case Is <= 3:
    Open App.Path & "\horizontal.txt" For Input As #1

    location(0) = 0
    location(1) = 0
    location(2) = 0
    If Operation = 1 Then
    Set textObj = AcadDoc.ModelSpace. _
        AddText("Kurva Horizontal - Metode Selisih Busur", location, 1)
    textObj.Update
    ElseIf Operation = 2 Then
    Set textObj = AcadDoc.ModelSpace. _
        AddText("Kurva Horizontal - Metode Polar", location, 1)
    textObj.Update
    ElseIf Operation = 3 Then
    Set textObj = AcadDoc.ModelSpace. _
        AddText("Kurva Horizontal - Metode Polygon", location, 1)
    textObj.Update
    End If

    While Not EOF(1)
    Input #1, x1, Y1, x2, Y2, lai
    startPoint(0) = x1 * GScale / 1440
    startPoint(1) = -Y1 * GScale / 1440
    startPoint(2) = 0
    endPoint(0) = x2 * GScale / 1440
    endPoint(1) = -Y2 * GScale / 1440
    endPoint(2) = 0
    Set LineObj = AcadDoc.ModelSpace.AddLine(startPoint, endPoint)
    If lai = 1 Then
    LineObj.Layer = "Garis"
    Else
    LineObj.Layer = "Helper"
    LineObj.Linetype = "DASHED"
    End If
    Wend
    Close #1

    Case Is = 4
```

```
    With VertCurve
    For i = 0 To .Iterasi
    startPoint(0) = ArrLine(i - 1).X
    startPoint(1) = ArrLine(i - 1).Y
    startPoint(2) = 0
    endPoint(0) = ArrLine(i).X
    endPoint(1) = ArrLine(i).Y
    endPoint(2) = 0
    TAcuan(0) = ArrLine(i).X
    TAcuan(1) = .Acuan
    TAcuan(2) = 0
    If i > 0 Then
    Set LineObj = AcadDoc.ModelSpace.AddLine(startPoint, endPoint)
    LineObj.Layer = "Garis"
    End If
    'strCmd = strCmd & Str(PX) & "," & Str(PY) & vbCrLf
    Set LineObj = AcadDoc.ModelSpace.AddLine(endPoint, TAcuan)
    Next i

    startPoint(0) = ArrLine(0).X
    startPoint(1) = .Acuan
    startPoint(2) = 0
    endPoint(0) = ArrLine(.Iterasi).X
    endPoint(1) = .Acuan
    endPoint(2) = 0
    location(0) = .Panjang / 2
    location(1) = .Acuan - 5
    location(2) = 0
    Set LineObj = AcadDoc.ModelSpace.AddLine(startPoint, endPoint)
    LineObj.Linetype = "DASHED"

    Set dimObj = AcadDoc.ModelSpace.AddDimAligned(startPoint, endPoint,
location)

    For i = 0 To 2
    startPoint(0) = .Panjang * i / 2
    startPoint(1) = .Elev(i)
    startPoint(2) = 0
    endPoint(0) = .Panjang * (i + 1) / 2
    endPoint(1) = .Elev(i + 1)
    endPoint(2) = 0
    location(0) = (.Panjang * (i) / 2) - 0.5
    location(1) = .Acuan - 0.5
    location(2) = 0
    If i < 2 Then
    Set LineObj = AcadDoc.ModelSpace.AddLine(startPoint, endPoint)
    LineObj.Layer = "Elevasi"
```

```
With VertCurve
For i = 0 To .Iterasi
startPoint(0) = .ArrLine(i - 1).X
startPoint(1) = .ArrLine(i - 1).Y
startPoint(2) = 0
endPoint(0) = .ArrLine(i).X
endPoint(1) = .ArrLine(i).Y
endPoint(2) = 0
TAcuan(0) = .ArrLine(i).X
TAcuan(1) = .Acuan
TAcuan(2) = 0
If i > 0 Then
Set LineObj = AcadDoc.ModelSpace.AddLine(startPoint, endPoint)
LineObj.Layer = "Garis"
End If
'strCmd = strCmd & Str(PX) & "," & Str(PY) & vbCrLf
Set LineObj = AcadDoc.ModelSpace.AddLine(endPoint, TAcuan)
Next i

startPoint(0) = .ArrLine(0).X
startPoint(1) = .Acuan
startPoint(2) = 0
endPoint(0) = .ArrLine(.Iterasi).X
endPoint(1) = .Acuan
endPoint(2) = 0
location(0) = .Panjang / 2
location(1) = .Acuan - 5
location(2) = 0
Set LineObj = AcadDoc.ModelSpace.AddLine(startPoint, endPoint)
LineObj.Linetype = "DASHED"

Set dimObj = AcadDoc.ModelSpace.AddDimAligned(startPoint, endPoint,
location)

For i = 0 To 2
startPoint(0) = .Panjang * i / 2
startPoint(1) = .Elev(i)
startPoint(2) = 0
endPoint(0) = .Panjang * (i + 1) / 2
endPoint(1) = .Elev(i + 1)
endPoint(2) = 0
location(0) = (.Panjang * (i / 2) - 0.5
location(1) = .Acuan - 0.5
location(2) = 0
If i < 2 Then
Set LineObj = AcadDoc.ModelSpace.AddLine(startPoint, endPoint)
LineObj.Layer = "Elevasi"
```

```
LineObj.Linetype = "DASHED"
End If

Set textObj = AcadDoc.ModelSpace. _
        AddText(arText(i + 1), location, 0.5)
textObj.Update
Next
End With
Case 5: ' Menggambar kurva spiral
Open App.Path & "\spiral.txt" For Input As #1

location(0) = 0
location(1) = 0
location(2) = 0
Set textObj = AcadDoc.ModelSpace. _
        AddText("Kurva  Spiral", location, 0.5)
textObj.Update
While Not EOF(1)
Input #1, x1, Y1, x2, Y2, lai
startPoint(0) = x1
startPoint(1) = -Y1
startPoint(2) = 0
endPoint(0) = x2
endPoint(1) = -Y2
endPoint(2) = 0
Set LineObj = AcadDoc.ModelSpace.AddLine(startPoint, endPoint)
If lai = 1 Then
LineObj.Layer = "Garis"
Else
LineObj.Layer = "Helper"
LineObj.Linetype = "DASHED"
End If
Wend
Close #1
Case 6: ' Menggambar kurva super elevasi
'Dim x1, Y1, x2, Y2 As Double
'Dim lai As Integer
If JenisElev = 0 Then
Open App.Path & "\selingkaran.txt" For Input As #2
location(0) = 0
location(1) = 0
location(2) = 0
Set textObj = AcadDoc.ModelSpace. _
        AddText("Kurva Super Elevasi : Lingkaran", location, 0.5)
textObj.Update
Else
Open App.Path & "\sespiral.txt" For Input As #2
```

```
            LineObj.Linetype = "DASHED"
            End If

            Set textObj = AcadDoc.ModelSpace. _
                AddText(arText(i + 1), location, 0.5)
            textObj.Update
            Next
            End With
            Case 5: ' Menggambar kurva spiral
            Open App.Path & "\spiral.txt" For Input As #1

            location(0) = 0
            location(1) = 0
            location(2) = 0
            Set textObj = AcadDoc.ModelSpace. _
                AddText("Kurva  Spiral", location, 0.5)
            textObj.Update
            While Not EOF(1)
            Input #1, x1, Y1, x2, Y2, lai
            startPoint(0) = x1
            startPoint(1) = -Y1
            startPoint(2) = 0
            endPoint(0) = x2
            endPoint(1) = -Y2
            endPoint(2) = 0
            Set LineObj = AcadDoc.ModelSpace.AddLine(startPoint, endPoint)
            If lai = 1 Then
            LineObj.Layer = "Garis"
            Else
            LineObj.Layer = "Helper"
            LineObj.Linetype = "DASHED"
            End If
            Wend
            Close #1
            Case 6: ' Menggambar kurva super elevasi
            'Dim x1, Y1, x2, Y2 As Double
            'Dim lai As Integer
            If JenisElev = 0 Then
            Open App.Path & "\selingkaran.txt" For Input As #2
            location(0) = 0
            location(1) = 0
            location(2) = 0
            Set textObj = AcadDoc.ModelSpace. _
                AddText("Kurva Super Elevasi : Lingkaran", location, 0.5)
            textObj.Update
            Else
            Open App.Path & "\sespiral.txt" For Input As #2
```

```
    location(0) = 0
    location(1) = 0
    location(2) = 0
    Set textObj = AcadDoc.ModelSpace. _
            AddText("Kurva Super Elevasi : Spiral", location, 0.5)
    textObj.Update
    End If
    While Not EOF(2)
    Input #2, x1, Y1, x2, Y2, lai
    startPoint(0) = x1 * GScale / 1440
    startPoint(1) = -Y1 * GScale / 1440
    startPoint(2) = 0
    endPoint(0) = x2 * GScale / 1440
    endPoint(1) = -Y2 * GScale / 1440
    endPoint(2) = 0
    Set LineObj = AcadDoc.ModelSpace.AddLine(startPoint, endPoint)
    If lai = 1 Then
    LineObj.Layer = "Garis"
    Else
    LineObj.Layer = "Helper"
    End If
    Wend
    Close #2
   End Select
   AcadDoc.Regen (acActiveViewport)
   With AcadApp
   .Visible = True
   .ZoomExtents
   End With
End Sub

Private Sub mnucontent_Click()
'Me.HelpContextID = 1
With CommonDialog1
.HelpFile = App.Path & "\kurva.hlp"
.HelpCommand = 1
.HelpContext = 1
.ShowHelp
End With
End Sub

Private Sub mnuExpAUTOCAD_Click()
 GambarKeAutoCAd
End Sub

Private Sub mnuKurvaVertikal_Click()
Operation = 4
```

```
location(0) = 0
location(1) = 0
location(2) = 0
Set textObj = AcadDoc.ModelSpace. _
AddText("Kurva Super Elevasi : Spiral", location, 0.5)
textObj.Update
End If
While Not EOF(2)
Input #2, x1, Y1, x2, Y2, lat
startPoint(0) = x1 * GScale / 1440
startPoint(1) = -Y1 * GScale / 1440
startPoint(2) = 0
endPoint(0) = x2 * GScale / 1440
endPoint(1) = -Y2 * GScale / 1440
endPoint(2) = 0
Set LineObj = AcadDoc.ModelSpace.AddLine(startPoint, endPoint)
If lat = 1 Then
LineObj.Layer = "Garis"
Else
LineObj.Layer = "Helper"
End If
Wend
Close #2
End Select
AcadDoc.Regen (acActiveViewport)
With AcadApp
.Visible = True
.ZoomExtents
End With
End Sub

Private Sub mnucontent_Click()
'Me.HelpContextID = 1
With CommonDialog1
.HelpFile = App.Path & "\kurva.hlp"
.HelpCommand = 1
.HelpContext = 1
.ShowHelp
End With
End Sub

Private Sub mnuExpAUTOCAD_Click()
GambarKeAutoCAd
End Sub

Private Sub mnuKurvaVertikal_Click()
Operation = 4
```

```
StatusBar1.Panels(2).Text = PanelText(Operation)
Toolbar1.Buttons(4).Value = tbrPressed
MaxPoint = 1
DataVertikal.Show
End Sub

Private Sub mnuKurvaVt_Click()

End Sub

Private Sub mnuKurvHorizonal_Click()
InputFrm1.Show
Toolbar1.Buttons(1).Value = tbrPressed
End Sub

Private Sub mnuProperties_Click()
PropertiesFrm.Show
End Sub

Private Sub mnuSave_Click()
ChSaveDlg.Show 0, MainFrm
End Sub

Private Sub mnuSaveToBitmap_Click()
Dim NamaFile As String
With CommonDialog1
.DialogTitle = "Save Picture to Bitmap File"
.ShowSave
NamaFile = .FileName + ".bmp"
SavePicture Me.Pic.Image, NamaFile
End With
End Sub


Private Sub mnuSBType1_Click()
InputFrm1.Show
End Sub


Private Sub mnuSpirals_Click()
Operation = 5
SpiralFrm.Show
End Sub

Private Sub mnuSuperelevasi_Click()
 Operation = 6
 SuperlevasiFrm.Show
```

```
StatusBar1.Panels(2).Text = PanelText(Operation)
Toolbar1.Buttons(4).Value = tbrPressed
MaxPoint = 1
DataVertikal.Show
End Sub

Private Sub mnuKurvaVt_Click()

End Sub

Private Sub mnuKurvHorizonal_Click()
InputFrm1.Show
Toolbar1.Buttons(1).Value = tbrPressed
End Sub

Private Sub mnuProperties_Click()
PropertiesFrm.Show
End Sub

Private Sub mnuSave_Click()
CbSaveDlg.Show 0, MainFrm
End Sub

Private Sub mnuSaveToBitmap_Click()
Dim NamaFile As String
With CommonDialog1
.DialogTitle = "Save Picture to Bitmap File"
.ShowSave
NamaFile = .FileName + ".bmp"
SavePicture MoPic.Image, NamaFile
End With
End Sub

Private Sub mnuSBType1_Click()
InputFrm1.Show
End Sub

Private Sub mnuSpirals_Click()
Operation = 5
SpiralFrm.Show
End Sub

Private Sub mnuSuperelevasi_Click()
Operation = 6
SuperlevasiFrm.Show
```

```
End Sub

Private Sub pic_MouseDown(Button As Integer, Shift As Integer, X As Single, Y
As Single)

If Operation > 0 Then
   If Button = 1 Then
      Drawing = True
   Else
      Drawing = False
      ClickIndex = 0

   End If
   If Drawing Then
      ClickIndex = ClickIndex + 1

      If ClickIndex > MaxPoint Then
       If Drawing Then DrawComplete = True
       ClickIndex = 0
       Drawing = False
      Else
      DrawComplete = False
      Position(ClickIndex).X = X
      Position(ClickIndex).Y = Y
      AcPos(ClickIndex).X = X / 1440
      AcPos(ClickIndex).Y = (Me.Pic.ScaleHeight - Y) / 1440
      If ClickIndex <= MaxPoint Then
      PropertiesFrm.KoorX(ClickIndex - 1).Caption = "X" & Chr(64 +
ClickIndex) & " : " & Str(AcPos(ClickIndex).X)
      PropertiesFrm.KoorY(ClickIndex - 1).Caption = "Y" & Chr(64 +
ClickIndex) & " : " & Str(AcPos(ClickIndex).Y)
      End If
      'Pic.DrawStyle = vbDot
      If ClickIndex <= 1 Then
      Pic.PSet (X, Y), QBColor(4)
      Else
      Pic.Line (lastX, lastY)-(Position(ClickIndex).X, Position(ClickIndex).Y),
RGB(10, 200, 255)
      End If
      '
      hotSpot X, Y, TextPoint(ClickIndex)

      lastX = X
      lastY = Y

      End If
   End If
```

```
End Sub

Private Sub pic_MouseDown(Button As Integer, Shift As Integer, X As Single, Y
As Single)

If Operation > 0 Then
  If Button = 1 Then
   Drawing = True
  Else
   Drawing = False
   ClickIndex = 0

  End If
  If Drawing Then
   ClickIndex = ClickIndex + 1

   If ClickIndex > MaxPoint Then
    If Drawing Then DrawComplete = True
    ClickIndex = 0
    Drawing = False
   Else
   DrawComplete = False
   Position(ClickIndex).X = X
   Position(ClickIndex).Y = Y
   AcPos(ClickIndex).X = X / 1440
   AcPos(ClickIndex).Y = (Me.Pic.ScaleHeight - Y) / 1440
   If ClickIndex <= MaxPoint Then
   PropertiesFrm.KoorX(ClickIndex - 1).Caption = "X" & Chr(64 +
ClickIndex) & " : " & Str(AcPos(ClickIndex).X)
   PropertiesFrm.KoorY(ClickIndex - 1).Caption = "Y" & Chr(64 +
ClickIndex) & " : " & Str(AcPos(ClickIndex).Y)
   End If
   Pic.DrawStyle = vbDot
   If ClickIndex <= 1 Then
   Pic.PSet (X, Y), QBColor(4)
   Else
   Pic.Line (lastX, lastY)-(Position(ClickIndex).X, Position(ClickIndex).Y),
RGB(10, 200, 255)
   End If

   hotSpot X, Y, TextPoint(ClickIndex)

   lastX = X
   lastY = Y

   End If
  End If
```

```
    If DrawComplete And (Button = 1) Then
     CurveOperation Operation

    End If
 End If

End Sub

Private Sub Pic_MouseMove(Button As Integer, Shift As Integer, X As Single, Y
As Single)
   Dim virtY As Single
   HRuler.DrawMode = 6
   HRuler.Line (X, 0)-(X, HRuler.ScaleHeight)
   If x1 > 0 Then
      HRuler.Line (x1, 0)-(x1, HRuler.ScaleHeight)
   End If
   HRuler.DrawMode = 13
   x1 = X
   PosX = X / 1440
   PosY = (Y - HRuler.Height) / 1440
   virtY = (Pic.Height - Y) / 1440
   If Y < Pic.Height Then StatusBar1.Panels(1).Text = "X:" & Str(PosX) & " Y:"
& Str(virtY)

End Sub

Private Sub Form_Resize()
   'HRuler.Width = Me.ScaleWidth - HRuler.Left
   Pic.Height = Me.ScaleHeight - HRuler.Height
   HRuler.Cls
     'Draw Ruler 16ths of an inch
   DrawRuler 32
End Sub

Private Sub Form_Unload(Cancel As Integer)
End
End Sub

Private Sub mnuClear_Click()
Pic.Cls
If Pic.Image <> Null Then
Pic.Image = Null
End If
End Sub

Private Sub mnuPolar_Click()
Operation = 2
```

```
If DrawComplete And (Button = 1) Then
CurveOperation Operation

End If
End If

End Sub

Private Sub Pic_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
Dim virtY As Single
HRuler.DrawMode = 6
HRuler.Line (X, 0)-(X, HRuler.ScaleHeight)
If x1 > 0 Then
HRuler.Line (x1, 0)-(x1, HRuler.ScaleHeight)
End If
HRuler.DrawMode = 13
x1 = X
PosX = X / 1440
PosY = (Y - HRuler.Height) / 1440
virtY = (Pic.Height - Y) / 1440
If Y < Pic.Height Then StatusBar1.Panels(1).Text = "X:" & Str(PosX) & " Y:" & Str(virtY)

End Sub

Private Sub Form_Resize()
'HRuler.Width = Me.ScaleWidth - HRuler.Left
Pic.Height = Me.ScaleHeight - HRuler.Height
HRuler.Cls
'Draw Ruler 16ths of an inch
DrawRuler 32
End Sub

Private Sub Form_Unload(Cancel As Integer)
End
End Sub

Private Sub mnuClear_Click()
Pic.Cls
If Pic.Image <> Null Then
Pic.Image = Null
End If
End Sub

Private Sub mnuPolar_Click()
Operation = 2
```

```
StatusBar1.Panels(2).Text = PanelText(Operation)
Toolbar1.Buttons(2).Value = tbrPressed
MaxPoint = 3
End Sub

Private Sub mnuPolygon_Click()
Operation = 3
StatusBar1.Panels(2).Text = PanelText(Operation)
Toolbar1.Buttons(3).Value = tbrPressed
MaxPoint = 3
End Sub

Private Sub mnuQuit_Click()
If MsgBox("Ingin Keluar dari program", vbYesNo + vbQuestion, "Sudah Yakin",
"", 0) = vbYes Then End
End Sub

Private Sub mnuSelisih_Click()
 Operation = 1
 StatusBar1.Panels(2).Text = PanelText(Operation)
 Toolbar1.Buttons(1).Value = tbrPressed
 MaxPoint = 3
End Sub

Private Sub mnuSpiral_Click()
 Operation = 5
 StatusBar1.Panels(2).Text = PanelText(Operation)
 Toolbar1.Buttons(5).Value = tbrPressed
 MaxPoint = 3
End Sub

Private Function GetCenterX(ByRef Titik1 As Point, ByRef Titik2 As Point,
ByVal Radius As Single, ByVal Sudut As Single) As Single
Dim T1C As Single
Dim rad1 As Single
Dim t2 As Single
Dim jarak1 As Single
jarak1 = JarakTitik(Titik1.X, Titik1.Y, Titik2.X, Titik2.Y)
rad1 = Sudut * PI / 180
T1C = jarak1 * Sin(rad1)
GetCenterX = Titik1.X + T1C
End Function
Private Function GetCenterY(ByRef Titik1 As Point, ByRef Titik2 As Point,
ByVal Radius As Single, ByVal Sudut As Single) As Single
Dim T1C As Single
Dim rad1 As Single
Dim t2 As Single
```

```
StatusBar1.Panels(2).Text = PanelText(Operation)
Toolbar1.Buttons(2).Value = tbrPressed
MaxPoint = 3
End Sub

Private Sub mnuPolygon_Click()
Operation = 3
StatusBar1.Panels(2).Text = PanelText(Operation)
Toolbar1.Buttons(3).Value = tbrPressed
MaxPoint = 3
End Sub

Private Sub mnuQuit_Click()
If MsgBox("Ingin Keluar dari program", vbYesNo + vbQuestion, "Sudah Yakin",
"") = vbYes Then End
End Sub

Private Sub mnuSelisih_Click()
Operation = 1
StatusBar1.Panels(2).Text = PanelText(Operation)
Toolbar1.Buttons(1).Value = tbrPressed
MaxPoint = 3
End Sub

Private Sub mnuSpiral_Click()
Operation = 5
StatusBar1.Panels(2).Text = PanelText(Operation)
Toolbar1.Buttons(5).Value = tbrPressed
MaxPoint = 3
End Sub

Private Function GetCenterX(ByRef Titik1 As Point, ByRef Titik2 As Point,
ByVal Radius As Single, ByVal Sudut As Single) As Single
Dim T1C As Single
Dim rad1 As Single
Dim t2 As Single
Dim jarak1 As Single
jarak1 = JarakTitik(Titik1.X, Titik1.Y, Titik2.X, Titik2.Y)
rad1 = Sudut * Pi / 180
T1C = jarak1 * Sin(rad1)
GetCenterX = Titik1.X + T1C
End Function
Private Function GetCenterY(ByRef Titik1 As Point, ByRef Titik2 As Point,
ByVal Radius As Single, ByVal Sudut As Single) As Single
Dim T1C As Single
Dim rad1 As Single
Dim t2 As Single
```

```
Dim jarak1 As Single
jarak1 = JarakTitik(Titik1.X, Titik1.Y, Titik2.X, Titik2.Y)
rad1 = Sudut * PI / 180
T1C = jarak1 * Sin(rad1)
t2 = jarak1 * Radius / T1C
GetCenterY = t2

End Function
Private Sub Bulatan(ByVal X As Single, ByVal Y As Single, ByVal Warna As
Long)
 Pic.DrawWidth = 3
 Pic.PSet (X, Y), Warna
 Pic.DrawWidth = 1
End Sub

Private Sub GambarKurvaHorizontal(ByRef Posix() As Point, ByVal Index As
Integer)
Dim i As Integer
Dim R As Single
Dim Px0, Px1 As Single
Dim PBusur, N As Single
Dim Dt, t As Single
Dim PT, J1, J2 As Single
Dim STegak, SAwal, STitik As Single
Dim sudutAsal, sudutAkhir As Single

If Curve.Jarak_AB > 0 Or Curve.Jarak_BC > 0 Then
J1 = Curve.Jarak_AB
J2 = Curve.Jarak_BC
Sudut(1) = 0
Sudut(2) = Curve.theta * PI / 180
Sudut(3) = Curve.theta * PI / 180
Else
Sudut(1) = TangenSudut(Posix(1).X, Posix(1).Y, Posix(2).X, Posix(2).Y)
Sudut(2) = TangenSudut(Posix(2).X, Posix(2).Y, Posix(3).X, Posix(3).Y)
Sudut(3) = Sudut(2)
Radsudut(1) = PI / 2 - Sudut(1)
Radsudut(2) = 2.5 * PI - Sudut(2)
Radsudut(3) = Radsudut(2) - Radsudut(1)

If Radsudut(1) < 0 Then Radsudut(1) = Radsudut(1) + 2 * PI
If Radsudut(2) > 2 * PI Then
Radsudut(2) = Radsudut(2) - 2 * PI
End If

If Radsudut(3) > 2 * PI Then Radsudut(3) = Radsudut(3) - 2 * PI
```

```
Dim jarak1 As Single
jarak1 = JarakTitik(Titik1.X, Titik1.Y, Titik2.X, Titik2.Y)
rad1 = Sudut * PI / 180
TIC = jarak1 * Sin(rad1)
t2 = jarak1 * Radius / TIC
GetCenterY = t2

End Function
Private Sub Bulatan(ByVal X As Single, ByVal Y As Single, ByVal Warna As
Long)
 Pic.DrawWidth = 3
 Pic.PSet (X, Y), Warna
 Pic.DrawWidth = 1
End Sub

Private Sub GambarKurvaHorizontal(ByRef Posisi() As Point, ByVal Index As
Integer)
Dim i As Integer
Dim R As Single
Dim Px0, Px1 As Single
Dim PBusur, N As Single
Dim Dt, t As Single
Dim PI, J1, J2 As Single
Dim STegak, SAwal, STitik As Single
Dim sudutAsal, sudutAkhir As Single

If Curve.Jarak_AB > 0 Or Curve.Jarak_BC > 0 Then
J1 = Curve.Jarak_AB
J2 = Curve.Jarak_BC
Sudut(1) = 0
Sudut(2) = Curve.theta * PI / 180
Sudut(3) = Curve.theta * PI / 180
Else
Sudut(1) = TangenSudut(Posisi(1).X, Posisi(1).Y, Posisi(2).X, Posisi(2).Y)
Sudut(2) = TangenSudut(Posisi(2).X, Posisi(2).Y, Posisi(3).X, Posisi(3).Y)
Sudut(3) = Sudut(2)
Radsudut(1) = PI / 2 - Sudut(1)
Radsudut(2) = 2.5 * PI - Sudut(2)
Radsudut(3) = Radsudut(2) - Radsudut(1)

If Radsudut(1) < 0 Then Radsudut(1) = Radsudut(1) + 2 * PI
If Radsudut(2) > 2 * PI Then
Radsudut(2) = Radsudut(2) - 2 * PI
End If

If Radsudut(3) > 2 * PI Then Radsudut(3) = Radsudut(3) - 2 * PI
```

```
PropertiesFrm.SudutAB.Text = Str(Radsudut(1) * 180 / PI)
PropertiesFrm.SudutBC.Text = Str(Radsudut(2) * 180 / PI)
PropertiesFrm.SudutTT.Text = Str(Radsudut(3) * 180 / PI)

Pic.DrawStyle = vbSolid
J1 = JarakTitik(Posix(1).X, Posix(1).Y, Posix(2).X, Posix(2).Y)
J2 = JarakTitik(Posix(2).X, Posix(2).Y, Posix(3).X, Posix(3).Y)
End If
If PropertiesFrm.Radius.Text <> "" Then
R = Val(PropertiesFrm.Radius.Text) * 1440
Else
If Curve.Radius = 0 Then
R = Val(InputBox("Nilai Radius", "Masukkan")) * 1440
PropertiesFrm.Radius.Text = Str(R / 1440)
Else
R = Curve.Radius
End If
End If
If R > J1 Or R > J2 Then
 MsgBox "Radius tidak boleh melebihi jarak antar dua titik", vbOKOnly, "Error",
"", 0
End If
PT = PanjangTangen(R, Radsudut(3))
If PT > J1 Or PT > J2 Then
MsgBox "Panjang Tangen tidak boleh melebihi jarak A-B atau B-C"
   Exit Sub
End If
Px0 = J1 - PT
Px1 = PT
Pic.Circle (Posix(1).X, Posix(1).Y), 700, QBColor(12), Sudut(1), 0.5 * PI
Pic.Circle (Posix(2).X, Posix(2).Y), 700, QBColor(12), Sudut(2), 0.5 * PI
' Curve.Jarak_AB > 0 Or Curve.Jarak_BC > 0 Then

'Else
ArcPoint(1).X = Posix(1).X + Px0 * Cos(Sudut(1))
ArcPoint(1).Y = Posix(1).Y - Px0 * Sin(Sudut(1))
ArcPoint(2).X = Posix(2).X + Px1 * Cos(Sudut(2))
ArcPoint(2).Y = Posix(2).Y - Px1 * Sin(Sudut(2))
ArcPoint(3).X = ArcPoint(1).X + R * Cos(Radsudut(1))
ArcPoint(3).Y = ArcPoint(1).Y + R * Sin(Radsudut(1))
'End If
hotSpot ArcPoint(1).X, ArcPoint(1).Y, "T1"
hotSpot ArcPoint(2).X, ArcPoint(2).Y, "T2"
Bulatan ArcPoint(3).X, ArcPoint(3).Y, QBColor(12)
Pic.Print "O"
Pic.DrawStyle = vbDot
```

```
PropertiesFrm.SudutAB.Text = Str(Radsudut(1) * 180 / PI)
PropertiesFrm.SudutBC.Text = Str(Radsudut(2) * 180 / PI)
PropertiesFrm.SudutTT.Text = Str(Radsudut(3) * 180 / PI)

Pic.DrawStyle = vbSolid
J1 = JarakTitik(Posix(1).X, Posix(1).Y, Posix(2).X, Posix(2).Y)
J2 = JarakTitik(Posix(2).X, Posix(2).Y, Posix(3).X, Posix(3).Y)
End If
If PropertiesFrm.Radius.Text <> "" Then
R = Val(PropertiesFrm.Radius.Text) * 1440
Else
If Curve.Radius = 0 Then
R = Val(InputBox("Nilai Radius", "Masukkan")) * 1440
PropertiesFrm.Radius.Text = Str(R / 1440)
Else
R = Curve.Radius
End If
End If
If R > J1 Or R > J2 Then
MsgBox "Radius tidak boleh melebihi jarak antar dua titik", vbOKOnly, "Error",
"", 0
End If
PT = PanjangTangen(R, Radsudut(3))
If PT > J1 Or PT > J2 Then
MsgBox "Panjang Tangen tidak boleh melebihi jarak A-B atau B-C"
  Exit Sub
End If
Px0 = J1 - PT
Px1 = PT
Pic.Circle (Posix(1).X, Posix(1).Y), 700, QBColor(12), Sudut(1), 0.5 * PI
Pic.Circle (Posix(2).X, Posix(2).Y), 700, QBColor(12), Sudut(2), 0.5 * PI
' Curve.Jarak_AB > 0 Or Curve.Jarak_BC > 0 Then

'Else
ArcPoint(1).X = Posix(1).X + Px0 * Cos(Sudut(1))
ArcPoint(1).Y = Posix(1).Y - Px0 * Sin(Sudut(1))
ArcPoint(2).X = Posix(2).X - Px1 * Cos(Sudut(2))
ArcPoint(2).Y = Posix(2).Y - Px1 * Sin(Sudut(2))
ArcPoint(3).X = ArcPoint(1).X + R * Cos(Radsudut(1))
ArcPoint(3).Y = ArcPoint(1).Y + R * Sin(Radsudut(1))
'End If
hotSpot ArcPoint(1).X, ArcPoint(1).Y, "T1"
hotSpot ArcPoint(2).X, ArcPoint(2).Y, "T2"
Bulatan ArcPoint(3).X, ArcPoint(3).Y, QBColor(12)
Pic.Print "O"
Pic.DrawStyle = vbDot
```

```
Pic.Line (ArcPoint(3).X, ArcPoint(3).Y)-(Posix(2).X, Posix(2).Y), QBColor(7)
Pic.Line (ArcPoint(1).X, ArcPoint(1).Y)-(ArcPoint(2).X, ArcPoint(2).Y),
QBColor(7)
Pic.Line (ArcPoint(1).X, ArcPoint(1).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(13)
Pic.Line (ArcPoint(2).X, ArcPoint(2).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(13)

Pic.DrawStyle = vbSolid
sudutAsal = TangenSudut(ArcPoint(3).X, ArcPoint(3).Y, ArcPoint(2).X,
ArcPoint(2).Y)
If Radsudut(3) > 2 * PI Then
sudutAkhir = sudutAsal + Radsudut(3) - 2 * PI
PBusur = Radsudut(3) - 2 * PI
Else
sudutAkhir = sudutAsal + Radsudut(3)
PBusur = Radsudut(3)
End If


If PropertiesFrm.Iterasi.Text <> "" Then
N = Val(PropertiesFrm.Iterasi.Text)
Else
If Curve.N = 0 Then
N = Val(InputBox("Jumlah Iterasi (n)", "Masukkan"))
PropertiesFrm.Iterasi.Text = Str(N)
Else
N = Curve.N
End If
End If
Dt = (PBusur) / N
i = 0
t = sudutAsal
If Not DataFromEntry Then
With Curve
.Radius = R
.N = N
.Jarak_AB = J1
.Jarak_BC = J2
.Sudut_AB = Sudut(1) * 180 / PI
.Sudut_BC = Sudut(2) * 180 / PI
.theta = Radsudut(3) * 180 / PI
End With
End If
SAwal = TangenSudut(ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(2).X,
ArcPoint(2).Y)
Do While t <= sudutAkhir
```

```
Pic.Line (ArcPoint(3).X, ArcPoint(3).Y)-(Posix(2).X, Posix(2).Y), QBColor(7)
Pic.Line (ArcPoint(1).X, ArcPoint(1).Y)-(ArcPoint(2).X, ArcPoint(2).Y),
QBColor(7)
Pic.Line (ArcPoint(1).X, ArcPoint(1).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(15)
Pic.Line (ArcPoint(2).X, ArcPoint(2).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(15)

Pic.DrawStyle = vbSolid
sudutAsal = TangenSudut(ArcPoint(3).X, ArcPoint(3).Y, ArcPoint(2).X,
ArcPoint(2).Y)
If Radsudut(3) > 2 * PI Then
sudutAkhir = sudutAsal + Radsudut(3) - 2 * PI
PBusur = Radsudut(3) - 2 * PI
Else
sudutAkhir = sudutAsal + Radsudut(3)
PBusur = Radsudut(3)
End If


If PropertiesFrm.Iterasi.Text <> "" Then
N = Val(PropertiesFrm.Iterasi.Text)
Else
If Curve.N = 0 Then
N = Val(InputBox("Jumlah iterasi (n)", "Masukkan"))
PropertiesFrm.Iterasi.Text = Str(N)
Else
N = Curve.N
End If
End If
Dt = (PBusur) / N
i = 0
t = sudutAsal
If Not DataFromEntry Then
With Curve
.Radius = R
.N = N
.Jarak_AB = J1
.Jarak_BC = J2
.Sudut_AB = Sudut(1) * 180 / PI
.Sudut_BC = Sudut(2) * 180 / PI
.theta = Radsudut(3) * 180 / PI
End With
End If
SAwal = TangenSudut(ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(2).X,
ArcPoint(2).Y)
Do While t <= sudutAkhir
```

```
 i = i + 1
 ArrLine(i).X = ArcPoint(3).X + R * Cos(t)
 ArrLine(i).Y = ArcPoint(3).Y - R * Sin(t)
 If i > 1 Then
 Pic.Line (ArrLine(i - 1).X, ArrLine(i - 1).Y)-(ArrLine(i).X, ArrLine(i).Y),
QBColor(0)
 Select Case Index
 Case Is = 1
 arrVi(i) = t * 180 / PI
 arrJarak(i) = R * Dt / 1440
 Pic.Line (ArrLine(i).X, ArrLine(i).Y)-(ArrLine(i).X, ArcPoint(2).Y), QBColor(7)
 Case Is = 2
 Pic.DrawStyle = vbDot
 Pic.Line (ArcPoint(1).X, ArcPoint(1).Y)-(ArrLine(i).X, ArrLine(i).Y),
QBColor(7)
 Pic.Line (ArcPoint(3).X, ArcPoint(3).Y)-(ArrLine(i).X, ArrLine(i).Y),
QBColor(7)
 Pic.DrawStyle = vbSolid
 arrJarak(1) = JarakTitik(ArcPoint(1).X, ArcPoint(2).X, ArcPoint(1).Y,
ArcPoint(2).Y)
 'arrVi(i) = TangenSudut(ArcPoint(2).X, ArrLine(i).X, ArcPoint(2).Y,
ArrLine(i).Y)
 arrVi(i) = (t - sudutAsal) * 180 / PI
 arrJarak(i) = JarakTitik(ArcPoint(1).X, ArrLine(i).X, Me.Pic.ScaleHeight -
ArcPoint(1).Y, Me.Pic.ScaleHeight - ArrLine(i).Y)
 Case Is = 3
 Pic.DrawStyle = vbDot
  Pic.Line (ArcPoint(3).X, ArcPoint(3).Y)-(ArrLine(i).X, ArrLine(i).Y),
QBColor(7)
 Pic.DrawStyle = vbSolid
 arrVi(i) = (t - sudutAsal) * 180 / PI
 arrJarak(i) = R * Dt / 1440
 End Select
 Else
 arrVi(i) = (t - sudutAsal) * 180 / PI
 arrJarak(i) = R * Dt / 1440
 End If
 Bulatan ArrLine(i).X, ArrLine(i).Y, QBColor(12)
 t = t + Dt
Loop
MaxLine = i
Pic.Line (ArrLine(i).X, ArrLine(i).Y)-(ArcPoint(1).X, ArcPoint(1).Y),
QBColor(0)


End Sub
Private Sub CurveOperation(ByVal Index As Integer)
```

```
i = i + 1
ArrLine(i).X = ArcPoint(3).X + R * Cos(t)
ArrLine(i).Y = ArcPoint(3).Y - R * Sin(t)
If i > 1 Then
Pic.Line (ArrLine(i - 1).X, ArrLine(i - 1).Y)-(ArrLine(i).X, ArrLine(i).Y),
QBColor(0)
Select Case Index
Case Is = 1
arrVi(i) = t * 180 / PI
arrJarak(i) = R * Dt / 1440
Pic.Line (ArrLine(i).X, ArrLine(i).Y)-(ArrLine(i).X, ArcPoint(2).Y), QBColor(7)
Case Is = 2
Pic.DrawStyle = vbDot
Pic.Line (ArcPoint(1).X, ArcPoint(1).Y)-(ArrLine(i).X, ArrLine(i).Y),
QBColor(7)
Pic.Line (ArcPoint(3).X, ArcPoint(3).Y)-(ArrLine(i).X, ArrLine(i).Y),
QBColor(7)
Pic.DrawStyle = vbSolid
arrJarak(i) = JarakTitik(ArcPoint(1).X, ArcPoint(2).X, ArcPoint(1).Y,
ArcPoint(2).Y)
'arrVi(i) = TangenSudut(ArcPoint(2).X, ArrLine(i).X, ArcPoint(2).Y,
ArrLine(i).Y)
arrVi(i) = (t - sudutAsal) * 180 / PI
arrJarak(i) = JarakTitik(ArcPoint(1).X, ArrLine(i).X, Me.Pic.ScaleHeight -
ArcPoint(1).Y, Me.Pic.ScaleHeight - ArrLine(i).Y)
Case Is = 3
Pic.DrawStyle = vbDot
Pic.Line (ArcPoint(3).X, ArcPoint(3).Y)-(ArrLine(i).X, ArrLine(i).Y),
QBColor(7)
Pic.DrawStyle = vbSolid
arrVi(i) = (t - sudutAsal) * 180 / PI
arrJarak(i) = R * Dt / 1440
End Select
Else
arrVi(i) = (t - sudutAsal) * 180 / PI
arrJarak(i) = R * Dt / 1440
End If
Bulatan ArrLine(i).X, ArrLine(i).Y, QBColor(12)
t = t + Dt
Loop
MaxLine = i
Pic.Line (ArrLine(i).X, ArrLine(i).Y)-(ArcPoint(1).X, ArcPoint(1).Y),
QBColor(0)

End Sub
Private Sub CurveOperation(ByVal Index As Integer)
```

```
  Select Case Index
   Case Is = 1
     GambarKurvaHorizontal Position, 1
   Case Is = 2
     GambarKurvaHorizontal Position, 2
   Case Is = 3
     GambarKurvaHorizontal Position, 3
   Case Is = 4
     GambarKurvaVertikal Position(1)


  End Select

End Sub
Private Sub GambarKurvaVertikal(ByRef Posix As Point)
Dim i As Integer
Dim PoX, Poy As Single
With VertCurve
ArcPoint(1).X = Posix.X
ArcPoint(1).Y = Posix.Y + (.Elev(0) - .Acuan) * 1440 / GScale

ArcPoint(2).X = Posix.X + (.Panjang / GScale / 2) * 1440
ArcPoint(2).Y = ArcPoint(1).Y - ((.Elev(1) - .Acuan) * 1440 / GScale)
hotSpot ArcPoint(2).X, ArcPoint(2).Y, "B"

ArcPoint(3).X = Posix.X + (.Panjang / GScale) * 1440
ArcPoint(3).Y = ArcPoint(1).Y - ((.Elev(2) - .Acuan) * 1440 / GScale)
hotSpot ArcPoint(3).X, ArcPoint(3).Y, "C"
'garis bantuan
Me.Pic.Line (Posix.X, Posix.Y)-(Posix.X, ArcPoint(1).Y), QBColor(7)
Me.Pic.Line (ArcPoint(3).X, ArcPoint(3).Y)-(ArcPoint(3).X, ArcPoint(1).Y),
QBColor(7)
Me.Pic.Line (Posix.X, ArcPoint(1).Y)-(ArcPoint(3).X, ArcPoint(1).Y),
QBColor(7)
oldx = Posix.X
oldy = Posix.Y
Pic.CurrentX = Posix.X
Pic.CurrentY = ArcPoint(1).Y + 102
Pic.Print "Elevasi Acuan"
For i = 1 To .Iterasi
 PoX = Posix.X + ArrLine(i).X * 1440 / GScale
 Poy = ArcPoint(1).Y - (ArrLine(i).Y - .Acuan) * 1440 / GScale
 Me.Pic.Line (PoX, Poy)-(PoX, ArcPoint(1).Y), QBColor(7)
 Me.Pic.Line (oldx, oldy)-(PoX, Poy), QBColor(12)
 hotText PoX, ArcPoint(1).Y, Str(i)
 oldx = PoX
 oldy = Poy
```

```
Select Case Index
    Case Is = 1
        GambarKurvaHorizontal Position, 1
    Case Is = 2
        GambarKurvaHorizontal Position, 2
    Case Is = 3
        GambarKurvaHorizontal Position, 3
    Case Is = 4
        GambarKurvaVertikal Position(1)


End Select

End Sub
Private Sub GambarKurvaVertikal(ByRef Posix As Point)
Dim i As Integer
Dim PoX, Poy As Single
With VertCurve
ArcPoint(1).X = Posix.X
ArcPoint(1).Y = Posix.Y + (.Elev(0) - .Acuan) * 1440 / GScale

ArcPoint(2).X = Posix.X + (.Panjang / GScale / 2) * 1440
ArcPoint(2).Y = ArcPoint(1).Y - ((.Elev(1) - .Acuan) * 1440 / GScale)
hotSpot ArcPoint(2).X, ArcPoint(2).Y, "B"

ArcPoint(3).X = Posix.X + (.Panjang / GScale) * 1440
ArcPoint(3).Y = ArcPoint(1).Y - ((.Elev(2) - .Acuan) * 1440 / GScale)
hotSpot ArcPoint(3).X, ArcPoint(3).Y, "C"
'garis bantuan
Me.Pic.Line (Posix.X, Posix.Y)-(Posix.X, ArcPoint(1).Y), QBColor(7)
Me.Pic.Line (ArcPoint(3).X, ArcPoint(3).Y)-(ArcPoint(3).X, ArcPoint(1).Y),
QBColor(7)
Me.Pic.Line (Posix.X, ArcPoint(1).Y)-(ArcPoint(3).X, ArcPoint(1).Y),
QBColor(7)
oldx = Posix.X
oldy = Posix.Y
Pic.CurrentX = Posix.X
Pic.CurrentY = ArcPoint(1).Y + 102
Pic.Print "Elevasi Acuan"
For i = 1 To .Iterasi
 PoX = Posix.X + ArrLine(i).X * 1440 / GScale
 Poy = ArcPoint(1).Y - (ArrLine(i).Y - .Acuan) * 1440 / GScale
 Me.Pic.Line (PoX, Poy)-(PoX, ArcPoint(1).Y), QBColor(7)
 Me.Pic.Line (oldx, oldy)-(PoX, Poy), QBColor(12)
 hotText PoX, ArcPoint(1).Y, Str(i)
 oldx = PoX
 oldy = Poy
```

```
Next
'gambar garis busur
Me.Pic.Line (Posix.X, Posix.Y)-(ArcPoint(2).X, ArcPoint(2).Y), QBColor(9)
Me.Pic.Line (ArcPoint(2).X, ArcPoint(2).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(9)
End With
End Sub
Private Sub KurvaVertikal()

End Sub
Private Sub Toolbar1_ButtonClick(ByVal Button As MSComctlLib.Button)
If MsgBox("Ingin masukkan data secara manual", vbYesNo, "Konfirmasi", "", 0)
= vbYes Then
 DataFromEntry = True
 Else
 DataFromEntry = False
 End If
Select Case Button.Tag
   Case Is = 1
   If DataFromEntry Then
   InputFrm1.Show
   InputFrm1.Combo1.ListIndex = 0
   Else
   mnuSelisih_Click
   End If
   Case Is = 2
   If DataFromEntry Then
   InputFrm1.Show
   InputFrm1.Combo1.ListIndex = 1
   Else
   mnuPolar_Click
   End If
   Case Is = 3
   If DataFromEntry Then
   InputFrm1.Show
   InputFrm1.Combo1.ListIndex = 2
   Else
   mnuPolygon_Click
   End If
   Case Is = 4

   Operation = 4
   MaxPoint = 1

   Case Is = 5
   Operation = 6
   If DataFromEntry Then
```

```
Next
'gambar garis busur
Me.Pic.Line (Posix.X, Posix.Y)-(ArcPoint(2).X, ArcPoint(2).Y), QBColor(9)
Me.Pic.Line (ArcPoint(2).X, ArcPoint(2).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(9)
End With
End Sub
Private Sub KurvaVertikal()

End Sub
Private Sub Toolbar1_ButtonClick(ByVal Button As MSComctlLib.Button)
If MsgBox("Ingin masukkan data secara manual", vbYesNo, "Konfirmasi", "", 0)
= vbYes Then
 DataFromEntry = True
 Else
 DataFromEntry = False
 End If
Select Case Button.Tag
  Case Is = 1
  If DataFromEntry Then
  InputFrm1.Show
  InputFrm1.Combo1.ListIndex = 0
  Else
  mnuSelisih_Click
  End If
  Case Is = 2
  If DataFromEntry Then
  InputFrm1.Show
  InputFrm1.Combo1.ListIndex = 1
  Else
  mnuPolar_Click
  End If
  Case Is = 3
  If DataFromEntry Then
  InputFrm1.Show
  InputFrm1.Combo1.ListIndex = 2
  Else
  mnuPolygon_Click
  End If
  Case Is = 4

  Operation = 4
  MaxPoint = 1

  Case Is = 5
  Operation = 6
  If DataFromEntry Then
```

```
    SpiralFrm.Show
    End If
    Case Is = 6
    If DataFromEntry Then
    SuperlevasiFrm.Show
    End If
    Operation = 6
    StatusBar1.Panels(2).Text = PanelText(Operation)
    Toolbar1.Buttons(6).Value = tbrPressed
    MaxPoint = 2
    End Select
End Sub

Private Sub Toolbar1_ButtonMenuClick(ByVal ButtonMenu As
MSComctlLib.ButtonMenu)
If Operation > 0 Then
Select Case ButtonMenu.Tag
 Case Is = 7
   mnuSaveToBitmap_Click
 End Select
Else
MsgBox "Not Operation Can be done, You Must choose one of methode to draw
the curve"
End If
End Sub
```

```
SpiralFrm.Show
End If
Case Is = 6
If DataFromEntry Then
SuperlevasiFrm.Show
End If
Operation = 6
StatusBar1.Panels(2).Text = PanelText(Operation)
Toolbar1.Buttons(6).Value = tbrPressed
MaxPoint = 2
End Select
End Sub

Private Sub Toolbar1_ButtonMenuClick(ByVal ButtonMenu As
MSComctlLib.ButtonMenu)
If Operation > 0 Then
Select Case ButtonMenu.Tag
Case Is = 7
mnuSaveToBitmap_Click
End Select
Else
MsgBox "Not Operation Can be done. You Must choose one of methode to draw
the curve"
End If
End Sub
```

PROPERTIES

```
Option Explicit
Dim DockHandler As New clsDockingHandler
Private Sub Form_Load()
 Set DockHandler.ParentForm = Me
 DockHandler.AlwaysOnTop = False
End Sub

Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single,
Y As Single)
If Button = vbLeftButton Then
      ' OK - Here we go. We pass the cursor coordinates
      ' at the start of the drag
      DockHandler.StartDockDrag X, Y
   End If

End Sub

Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single,
Y As Single)
If Button = vbLeftButton Then
      ' Continuing with the drag
      DockHandler.UpdateDockDrag X, Y
   End If
End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
Cancel = 0
End Sub

Private Sub Form_Unload(Cancel As Integer)
Cancel = 0
End Sub
```

PROPERTIES

```
Option Explicit
Dim DockHandler As New clsDockingHandler
Private Sub Form_Load()
 Set DockHandler.ParentForm = Me
 DockHandler.AlwaysOnTop = False
End Sub

Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single,
Y As Single)
If Button = vbLeftButton Then
    ' OK - Here we go. We pass the cursor coordinates
    ' at the start of the drag
    DockHandler.StartDockDrag X, Y
  End If

End Sub

Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single,
Y As Single)
If Button = vbLeftButton Then
    ' Continuing with the drag
    DockHandler.UpdateDockDrag X, Y
  End If
End Sub

Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
Cancel = 0
End Sub

Private Sub Form_Unload(Cancel As Integer)
Cancel = 0
End Sub
```

KURVA HORISONTAL

```
Option Explicit
Dim Kurva As DataKurva
Dim Poss(1 To 3) As Point
Dim tt, l, Fi, tgT1B, tgBT2, a, R, x1, Y1 As Single
Dim zab, zbc, dab, dbc, d, k As Single
Dim APoint(3) As Point
Dim X, Y, oldx, oldy As Single
Dim jrkT1T2 As Double
Dim ORec, OEx As ADODB.Recordset
Private Sub Bulatan(ByVal X As Single, ByVal Y As Single, ByVal Warna As
Long)
 MainFrm.Pic.DrawWidth = 5
 MainFrm.Pic.PSet (X, Y), Warna
 MainFrm.Pic.DrawWidth = 1
End Sub
Private Sub gambar_titik(X As Integer, Y As Integer, color As Integer)
MainFrm.Pic.PSet (X, Y), QBColor(color)
End Sub
Private Sub gambar_garis(x1 As Integer, Y1 As Integer, x2 As Integer, Y2 As
Integer, color As Integer)
MainFrm.Pic.Line (x1, Y1)-(x2, Y2), QBColor(color)
End Sub
Private Sub hotSpot(ByVal X As Single, ByVal Y As Single, ByVal Text As
String)
Bulatan X, Y, 12
With MainFrm
.Pic.DrawStyle = vbSolid
.Pic.CurrentX = X + 50
.Pic.CurrentY = Y - 20
.Pic.Print Text

End With
End Sub
Private Sub GambarKurva()
Dim i As Integer
Dim Px0, Px1 As Single
Dim PBusur, N As Single
Dim Dt, t As Single
Dim PT, J1, J2 As Single
Dim STegak, SAwal, STitik As Single
Dim sudutAsal, sudutAkhir As Single
Dim SudutC, JarT1T2 As Single
Open App.Path & "\horizontal.txt" For Output As #1
With Curve
.Radius = R
```

KURVA HORISONTAL

```
Option Explicit
Dim Kurva As DataKurva
Dim Poss(1 To 3) As Point
Dim tt, I, Fi, tgT1B, tgBT2, n, R, x1, Y1 As Single
Dim zab, zbc, dab, dbc, d, k As Single
Dim APoint(3) As Point
Dim X, Y, oldx, oldy As Single
Dim jrkT1T2 As Double
Dim ORec, OEx As ADODB.Recordset
Private Sub Bulatan(ByVal X As Single, ByVal Y As Single, ByVal Warna As Long)
MainFrm.Pic.DrawWidth = 5
MainFrm.Pic.PSet (X, Y), Warna
MainFrm.Pic.DrawWidth = 1
End Sub
Private Sub gambar_titik(X As Integer, Y As Integer, color As Integer)
MainFrm.Pic.PSet (X, Y), QBColor(color)
End Sub
Private Sub gambar_garis(x1 As Integer, Y1 As Integer, x2 As Integer, Y2 As Integer, color As Integer)
MainFrm.Pic.Line (x1, Y1)-(x2, Y2), QBColor(color)
End Sub
Private Sub hotSpot(ByVal X As Single, ByVal Y As Single, ByVal Text As String)
Bulatan X, Y, 12
With MainFrm
.Pic.DrawStyle = vbSolid
.Pic.CurrentX = X + 50
.Pic.CurrentY = Y - 20
.Pic.Print Text

End With
End Sub
Private Sub GambarKurva()
Dim i As Integer
Dim Px0, Px1 As Single
Dim PBusur, N As Single
Dim Dt, t As Single
Dim PT, J1, J2 As Single
Dim STegak, SAwal, STitik As Single
Dim sudutAsal, sudutAkhir As Single
Dim SudutC, JarT1T2 As Single
Open App.Path & "\horizontal.txt" For Output As #1
With Curve
.Radius = R
```

```
.N = it
.Jarak_AB = dab
.Jarak_BC = dbc
.Sudut_AB = zab
.Sudut_BC = zbc
.theta = tt
End With
With MainFrm
Position(1).X = 720
Position(1).Y = 720
Position(2).X = Position(1).X + dab * 1440 / GScale
Position(2).Y = Position(1).Y
Position(3).X = Position(2).X + dbc * Cos((tt) * PI / 180) * 1440 / GScale
Position(3).Y = Position(2).Y + dbc * Sin((tt) * PI / 180) * 1440 / GScale

APoint(1).X = 720
APoint(1).Y = 720
APoint(2).X = APoint(1).X + dab * 1440 / GScale
APoint(2).Y = APoint(1).Y
APoint(3).X = APoint(2).X + dbc * Cos((tt) * PI / 180) * 1440 / GScale
APoint(3).Y = APoint(2).Y + dbc * Sin((tt) * PI / 180) * 1440 / GScale

Px0 = dab - tgT1B
Px1 = tgBT2
ArcPoint(1).X = APoint(1).X + Px0 * 1440 / GScale
ArcPoint(1).Y = APoint(1).Y
ArcPoint(2).X = APoint(2).X + Px1 * Cos((tt) * PI / 180) * 1440 / GScale
ArcPoint(2).Y = APoint(2).Y + Px1 * Sin((tt) * PI / 180) * 1440 / GScale
ArcPoint(3).X = ArcPoint(1).X
ArcPoint(3).Y = ArcPoint(1).Y + R * 1440 / GScale
Radsudut(3) = tt * PI / 180
.Pic.Line (APoint(1).X, APoint(1).Y)-(ArcPoint(1).X, ArcPoint(1).Y),
QBColor(12)
Write #1, APoint(1).X, APoint(1).Y, ArcPoint(1).X, ArcPoint(1).Y, 1
.Pic.Line (ArcPoint(2).X, ArcPoint(2).Y)-(APoint(3).X, APoint(3).Y),
QBColor(12)
Write #1, ArcPoint(2).X, ArcPoint(2).Y, APoint(3).X, APoint(3).Y, 1
.Pic.Line (ArcPoint(1).X, ArcPoint(1).Y)-(APoint(2).X, APoint(2).Y),
QBColor(10)
Write #1, ArcPoint(1).X, ArcPoint(1).Y, APoint(2).X, APoint(2).Y, 1
.Pic.Line (ArcPoint(2).X, ArcPoint(2).Y)-(APoint(2).X, APoint(2).Y),
QBColor(10)

Write #1, ArcPoint(2).X, ArcPoint(2).Y, APoint(2).X, APoint(2).Y, 1

.Pic.Line (ArcPoint(1).X, ArcPoint(1).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(7)
```

```
.N = n
.Jarak_AB = dab
.Jarak_BC = dbc
.Sudut_AB = zab
.Sudut_BC = zbc
.theta = tt
End With
With MainFrm
Position(1).X = 720
Position(1).Y = 720
Position(2).X = Position(1).X + dab * 1440 / GScale
Position(2).Y = Position(1).Y
Position(3).X = Position(2).X + dbc * Cos((tt) * PI / 180) * 1440 / GScale
Position(3).Y = Position(2).Y + dbc * Sin((tt) * PI / 180) * 1440 / GScale

APoint(1).X = 720
APoint(1).Y = 720
APoint(2).X = APoint(1).X + dab * 1440 / GScale
APoint(2).Y = APoint(1).Y
APoint(3).X = APoint(2).X + dbc * Cos((tt) * PI / 180) * 1440 / GScale
APoint(3).Y = APoint(2).Y + dbc * Sin((tt) * PI / 180) * 1440 / GScale

Px0 = dab - tgBT1
Px1 = tgBT2
ArcPoint(1).X = APoint(1).X + Px0 * 1440 / GScale
ArcPoint(1).Y = APoint(1).Y
ArcPoint(2).X = APoint(2).X + Px1 * Cos((tt) * PI / 180) * 1440 / GScale
ArcPoint(2).Y = APoint(2).Y + Px1 * Sin((tt) * PI / 180) * 1440 / GScale
ArcPoint(3).X = ArcPoint(1).X
ArcPoint(3).Y = ArcPoint(1).Y + R * 1440 / GScale
Radsudut(3) = tt * PI / 180
.PicLine (APoint(1).X, APoint(1).Y)-(ArcPoint(1).X, ArcPoint(1).Y), QBColor(12)
Write #1, APoint(1).X, APoint(1).Y, ArcPoint(1).X, ArcPoint(1).Y, 1
.PicLine (ArcPoint(2).X, ArcPoint(2).Y)-(APoint(3).X, APoint(3).Y), QBColor(12)
Write #1, ArcPoint(2).X, ArcPoint(2).Y, APoint(3).X, APoint(3).Y, 1
.PicLine (ArcPoint(1).X, ArcPoint(1).Y)-(APoint(2).X, APoint(2).Y), QBColor(10)
Write #1, ArcPoint(1).X, ArcPoint(1).Y, APoint(2).X, APoint(2).Y, 1
.PicLine (ArcPoint(2).X, ArcPoint(2).Y)-(APoint(2).X, APoint(2).Y), QBColor(10)

Write #1, ArcPoint(2).X, ArcPoint(2).Y, APoint(2).X, APoint(2).Y, 1

.PicLine (ArcPoint(1).X, ArcPoint(1).Y)-(ArcPoint(3).X, ArcPoint(3).Y), QBColor(7)
```

```
Write #1, ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(3).X, ArcPoint(3).Y, 1
.Pic.Line (ArcPoint(2).X, ArcPoint(2).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(7)
Write #1, ArcPoint(2).X, ArcPoint(2).Y, ArcPoint(3).X, ArcPoint(3).Y, 1
.Pic.DrawStyle = vbDot
.Pic.Line (ArcPoint(1).X, ArcPoint(1).Y)-(ArcPoint(2).X, ArcPoint(2).Y),
QBColor(7)
Write #1, ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(2).X, ArcPoint(2).Y, 2
.Pic.Line (APoint(2).X, APoint(2).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(7)
Write #1, APoint(2).X, APoint(2).Y, ArcPoint(3).X, ArcPoint(3).Y, 2
SudutC = TangenSudut(ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(2).X,
ArcPoint(2).Y)
JarT1T2 = JarakTitik(ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(1).X,
ArcPoint(2).Y)
ArcPoint(4).X = ArcPoint(1).X + JarT1T2 * Cos(SudutC)
ArcPoint(4).Y = ArcPoint(1).Y - JarT1T2 * Sin(SudutC)

hotSpot APoint(1).X, APoint(1).Y, "A"
hotSpot APoint(2).X, APoint(2).Y, "B"
hotSpot APoint(3).X, APoint(3).Y, "C"
hotSpot ArcPoint(1).X, ArcPoint(1).Y, "T1"
hotSpot ArcPoint(2).X, ArcPoint(2).Y, "T2"
hotSpot ArcPoint(3).X, ArcPoint(3).Y, "O"
hotSpot ArcPoint(4).X, ArcPoint(4).Y, "c"

SAwal = TangenSudut(ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(2).X,
ArcPoint(2).Y)
.Pic.DrawStyle = vbSolid
sudutAsal = TangenSudut(ArcPoint(3).X, ArcPoint(3).Y, ArcPoint(2).X,
ArcPoint(2).Y)
If Radsudut(3) > 2 * PI Then
sudutAkhir = sudutAsal + Radsudut(3) - 2 * PI
PBusur = Radsudut(3) - 2 * PI
Else
sudutAkhir = sudutAsal + Radsudut(3)
PBusur = Radsudut(3)
End If
Dt = (PBusur) / it
i = 0
t = sudutAsal
Do While t <= sudutAkhir
 i = i + 1
 ArrLine(i).X = ArcPoint(3).X + R * Cos(t) * 1440 / GScale
 ArrLine(i).Y = ArcPoint(3).Y - R * Sin(t) * 1440 / GScale
 If i > 1 Then
```

```
Write #1, ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(3).X, ArcPoint(3).Y, 1
.PicLine (ArcPoint(2).X, ArcPoint(2).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(7)
Write #1, ArcPoint(2).X, ArcPoint(2).Y, ArcPoint(3).X, ArcPoint(3).Y, 1
.PicDrawStyle = vbDot
.PicLine (ArcPoint(1).X, ArcPoint(1).Y)-(ArcPoint(2).X, ArcPoint(2).Y),
QBColor(7)
Write #1, ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(2).X, ArcPoint(2).Y, 2
.PicLine (APoint(2).X, APoint(2).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(7)
Write #1, APoint(2).X, APoint(2).Y, ArcPoint(3).X, ArcPoint(3).Y, 2
SudutC = TangenSudut(ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(2).X,
ArcPoint(2).Y)
JarT1T2 = JarakTitik(ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(1).X,
ArcPoint(2).Y)
ArcPoint(4).X = ArcPoint(1).X + JarT1T2 * Cos(SudutC)
ArcPoint(4).Y = ArcPoint(1).Y - JarT1T2 * Sin(SudutC)

hotSpot APoint(1).X, APoint(1).Y, "A"
hotSpot APoint(2).X, APoint(2).Y, "B"
hotSpot APoint(3).X, APoint(3).Y, "C"
hotSpot ArcPoint(1).X, ArcPoint(1).Y, "T1"
hotSpot ArcPoint(2).X, ArcPoint(2).Y, "T2"
hotSpot ArcPoint(3).X, ArcPoint(3).Y, "O"
hotSpot ArcPoint(4).X, ArcPoint(4).Y, "c"

SAwal = TangenSudut(ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(2).X,
ArcPoint(2).Y)
.PicDrawStyle = vbSolid
sudutAsal = TangenSudut(ArcPoint(3).X, ArcPoint(3).Y, ArcPoint(2).X,
ArcPoint(2).Y)
If Radsudut(3) > 2 * PI Then
sudutAkhir = sudutAsal + Radsudut(3) - 2 * PI
PBusur = Radsudut(3) - 2 * PI
Else
sudutAkhir = sudutAsal + Radsudut(3)
PBusur = Radsudut(3)
End If
Dt = (PBusur) / n
i = 0
t = sudutAsal
Do While t <= sudutAkhir
i = i + 1
ArrLine(i).X = ArcPoint(3).X + R * Cos(t) * 1440 / GScale
ArrLine(i).Y = ArcPoint(3).Y - R * Sin(t) * 1440 / GScale
If i > 1 Then
```

```
 .Pic.Line (ArrLine(i - 1).X, ArrLine(i - 1).Y)-(ArrLine(i).X, ArrLine(i).Y),
QBColor(0)
 Write #1, ArrLine(i - 1).X, ArrLine(i - 1).Y, ArrLine(i).X, ArrLine(i).Y, 1
 Select Case Combo1.ListIndex
 Case Is = 0
 arrVi(i) = t * 180 / PI
 arrJarak(i) = R * Dt
 .Pic.DrawStyle = vbDot
 .Pic.Line (ArrLine(i).X, Position(1).Y)-(ArrLine(i).X, ArrLine(i).Y), QBColor(7)
 Write #1, ArrLine(i).X, Position(1).Y, ArrLine(i).X, ArrLine(i).Y, 2
 .Pic.DrawStyle = vbSolid
 Case Is = 1
 .Pic.DrawStyle = vbDot
 .Pic.Line (ArcPoint(1).X, ArcPoint(1).Y)-(ArrLine(i).X, ArrLine(i).Y),
QBColor(7)
 Write #1, ArcPoint(1).X, ArcPoint(1).Y, ArrLine(i).X, ArrLine(i).Y, 2
 .Pic.DrawStyle = vbSolid
 arrJarak(1) = JarakTitik(ArcPoint(1).X, ArcPoint(2).X, ArcPoint(1).Y,
ArcPoint(2).Y)
 arrVi(i) = TangenSudut(ArcPoint(2).X, ArrLine(i).X, ArcPoint(2).Y,
ArrLine(i).Y)
 arrVi(i) = (t - sudutAsal) * 180 / PI
 arrJarak(i) = JarakTitik(ArcPoint(1).X, ArrLine(i).X, .Pic.ScaleHeight -
ArcPoint(1).Y, .Pic.ScaleHeight - ArrLine(i).Y)
 Case Is = 2
 .Pic.DrawStyle = vbDot
 .Pic.Line (ArcPoint(3).X, ArcPoint(3).Y)-(ArrLine(i).X, ArrLine(i).Y),
QBColor(7)
 Write #1, ArcPoint(3).X, ArcPoint(3).Y, ArrLine(i).X, ArrLine(i).Y, 2
 .Pic.DrawStyle = vbSolid
 arrVi(i) = (t - sudutAsal) * 180 / PI
 arrJarak(i) = R * Dt / 1440
 End Select
 Else
 arrVi(i) = (t - sudutAsal) * 180 / PI
 arrJarak(i) = R * Dt / 1440
 End If
 Bulatan ArrLine(i).X, ArrLine(i).Y, QBColor(12)
 t = t + Dt
Loop
MaxLine = i

.Pic.Line (ArrLine(i).X, ArrLine(i).Y)-(ArcPoint(1).X, ArcPoint(1).Y),
QBColor(0)
Write #1, ArrLine(i).X, ArrLine(i).Y, ArcPoint(1).X, ArcPoint(1).Y, 1
.Pic.DrawStyle = vbDot
.Pic.Line (APoint(2).X, APoint(2).Y)-(ArcPoint(2).X, Position(1).Y), QBColor(7)
```

```
.Pic.Line (ArrLine(i - 1).X, ArrLine(i - 1).Y)-(ArrLine(i).X, ArrLine(i).Y), QBColor(0)
Write #1, ArrLine(i - 1).X, ArrLine(i - 1).Y, ArrLine(i).X, ArrLine(i).Y, 1
Select Case Combo1.ListIndex
Case Is = 0
arrVi(i) = t * 180 / PI
arrJarak(i) = R * Dt
.Pic.DrawStyle = vbDot
.Pic.Line (ArrLine(i).X, Position(1).Y)-(ArrLine(i).X, ArrLine(i).Y), QBColor(7)
Write #1, ArrLine(i).X, Position(1).Y, ArrLine(i).X, ArrLine(i).Y, 2
.Pic.DrawStyle = vbSolid
Case Is = 1
.Pic.DrawStyle = vbDot
.Pic.Line (ArcPoint(1).X, ArcPoint(1).Y)-(ArrLine(i).X, ArrLine(i).Y), QBColor(7)
Write #1, ArcPoint(1).X, ArcPoint(1).Y, ArrLine(i).X, ArrLine(i).Y, 2
.Pic.DrawStyle = vbSolid
arrJarak(1) = JarakTitik(ArcPoint(1).X, ArcPoint(2).X, ArcPoint(1).Y, ArcPoint(2).Y)
arrVi(i) = TangenSudut(ArcPoint(2).X, ArrLine(i).X, ArcPoint(2).Y, ArrLine(i).Y)
arrVi(i) = (t - sudutAsal) * 180 / PI
arrJarak(i) = JarakTitik(ArcPoint(1).X, ArrLine(i).X, .Pic.ScaleHeigt - ArcPoint(1).Y, .Pic.ScaleHeight - ArrLine(i).Y)
Case Is = 2
.Pic.DrawStyle = vbDot
.Pic.Line (ArcPoint(3).X, ArcPoint(3).Y)-(ArrLine(i).X, ArrLine(i).Y), QBColor(7)
Write #1, ArcPoint(3).X, ArcPoint(3).Y, ArrLine(i).X, ArrLine(i).Y, 2
.Pic.DrawStyle = vbSolid
arrVi(i) = (t - sudutAsal) * 180 / PI
arrJarak(i) = R * Dt / 1440
End Select
Else
arrVi(i) = (t - sudutAsal) * 180 / PI
arrJarak(i) = R * Dt / 1440
End If
Bulatan ArrLine(i).X, ArrLine(i).Y, QBColor(12)
t = t + Dt
Loop
MaxLine = i

.Pic.Line (ArrLine(i).X, ArrLine(i).Y)-(ArcPoint(1).X, ArcPoint(1).Y), QBColor(0)
Write #1, ArrLine(i).X, ArrLine(i).Y, ArcPoint(1).X, ArcPoint(1).Y, 1
.Pic.DrawStyle = vbDot
.Pic.Line (APoint(2).X, APoint(2).Y)-(ArcPoint(2).X, Position(1).Y), QBColor(7)
```

```
Write #1, APoint(2).X, APoint(2).Y, ArcPoint(2).X, Position(1).Y, 2
.Pic.Line (ArcPoint(2).X, Position(1).Y)-(ArcPoint(2).X, ArcPoint(2).Y),
QBColor(7)
Write #1, ArcPoint(2).X, Position(1).Y, ArcPoint(2).X, ArcPoint(2).Y, 2
End With
Close #1
End Sub

Private Sub cmdCancel_Click()
Unload Me
End Sub
Private Sub GambarKurva2()
 Dim i As Integer


 With MainFrm
  For i = 1 To it
   ' X = Val(MGrid.TextMatrix(i, 1)) * 1440 / GScale
    'y = .Pic.ScaleHeight - (Val(MGrid.TextMatrix(i, 2)) * 1440 / GScale) -
.HRuler.ScaleHeight

    If i = 1 Then
    Bulatan X, Y, QBColor(10)

    Else
    .Pic.Line (oldx, oldy)-(X, Y), QBColor(12)
    End If
    oldx = X
    oldy = Y

  Next i
 End With
End Sub

Private Sub cmdOK_Click()
If Val(JAB.Text) < Val(txtT1B.Text) Or Val(JBC.Text) < Val(txtT1B.Text) _
 Then
MsgBox "Panjang T1B tidak boleh melebihi jarak AB atau BC", vbCritical +
vbOKOnly, "Error"
Else
GambarKurva
End If
DataFromEntry = True
MainFrm.Toolbar1.Buttons(Combo1.ListIndex + 1).Value = tbrPressed
Unload Me
End Sub
```

```
Write #1, APoint(2).X, APoint(2).Y, ArcPoint(2).X, Position(1).Y, 2
.PicLine (ArcPoint(2).X, Position(1).Y)-(ArcPoint(2).X, ArcPoint(2).Y),
QBColor(7)
Write #1, ArcPoint(2).X, Position(1).Y, ArcPoint(2).X, ArcPoint(2).Y, 3
End With
Close #1
End Sub

Private Sub cmdCancel_Click()
Unload Me
End Sub
Private Sub GambarKurva2()
Dim i As Integer


With MainFrm
For i = 1 To n
' X = Val(MGrid.TextMatrix(i, 1)) * 1440 / GScale
Y = .Pic.ScaleHeight - (Val(MGrid.TextMatrix(i, 2)) * 1440 / GScale) -
.HRuler.ScaleHeight

If i = 1 Then
Bulatan X, Y, QBColor(10)

Else
.PicLine (oldx, oldy)-(X, Y), QBColor(12)
End If
oldx = X
oldy = Y

Next i
End With
End Sub

Private Sub cmdOK_Click()
If Val(JAB.Text) < Val(txtT1B.Text) Or Val(JBC.Text) < Val(txtT1E.Text)
Then
MsgBox "Panjang T1B tidak boleh melebihi jarak AB atau BC", vbCritical +
vbOKOnly, "Error"
Else
GambarKurva
End If
DataFromEntry = True
MainFrm.Toolbar1.Buttons(Combo1.ListIndex + 1).Value = tbrPressed
Unload Me
End Sub
```

```
Private Sub ProsesPerhitungan(ByVal Jenis As String)
Dim jrab, jrbc As Single
Dim Sudut(3), Radsudut(3) As Single


If Val(KX(1).Text) <> 0 _
And Val(KY(1).Text) <> 0 _
And Val(KX(0).Text) <> 0 _
And Val(KY(0).Text) <> 0 _
Then
Sudut(1) = Atn((Val(KX(1).Text) - Val(KX(0).Text)) / (Val(KY(1).Text) -
Val(KY(0).Text)))
Sudut(2) = Atn((Val(KX(2).Text) - Val(KX(1).Text)) / (Val(KY(2).Text) -
Val(KY(1).Text)))
Sudut(3) = Sudut(2) - Sudut(1)
jrab = Sqr((Val(KX(1).Text) - Val(KX(0).Text)) ^ 2 + (Val(KY(1).Text) -
Val(KY(0).Text)) ^ 2)
jrbc = Sqr((Val(KX(2).Text) - Val(KX(1).Text)) ^ 2 + (Val(KY(2).Text) -
Val(KY(1).Text)) ^ 2)
JAB.Text = Format(jrab, "##0.000")
JBC.Text = Format(jrbc, "##0.000")
zab = Sudut(1) * 180 / PI
zbc = Sudut(2) * 180 / PI
If zbc < 0 Then zbc = zbc + 180
SudutAB.Text = SudutString(zab)
SudutBC.Text = SudutString(zbc)

Else
zab = DerajatToReal(SudutAB.Text) 'Val(SudutAB.Text)
zbc = DerajatToReal(SudutBC.Text) 'Val(SudutBC.Text)
dab = Val(JAB.Text)
dbc = Val(JBC.Text)

End If
R = Val(Radius.Text)
it = Val(Iterasi.Text)
      If zab > zbc Then
         tt = Abs((zab - zbc))
      Else
        tt = zbc - zab
      End If
      If zbc < 0 Then
         zbc = ((zbc + 0.18))
         tt = (zbc - zab)
      End If
l = (R * tt * (PI / 180))
a = (l / it)
```

```
Private Sub ProsesPerhitungan(ByVal Jenis As String)
Dim jrab, jrbc As Single
Dim Sudut(3), Radsudut(3) As Single


If Val(KX(1).Text) <> 0 _
And Val(KY(1).Text) <> 0 _
And Val(KX(0).Text) <> 0 _
And Val(KY(0).Text) <> 0 _
Then
Sudut(1) = Atn((Val(KX(1).Text) - Val(KX(0).Text)) / (Val(KY(1).Text) -
Val(KY(0).Text)))
Sudut(2) = Atn((Val(KX(2).Text) - Val(KX(1).Text)) / (Val(KY(2).Text) -
Val(KY(1).Text)))
Sudut(3) = Sudut(2) - Sudut(1)
jrab = Sqr((Val(KX(1).Text) - Val(KX(0).Text)) ^ 2 + (Val(KY(1).Text) -
Val(KY(0).Text)) ^ 2)
jrbc = Sqr((Val(KX(2).Text) - Val(KX(1).Text)) ^ 2 + (Val(KY(2).Text) -
Val(KY(1).Text)) ^ 2)
JAB.Text = Format(jrab, "##0.000")
JBC.Text = Format(jrbc, "##0.000")
zab = Sudut(1) * 180 / PI
zbc = Sudut(2) * 180 / PI
If zbc < 0 Then zbc = zbc + 180
SudutAB.Text = SudutString(zab)
SudutBC.Text = SudutString(zbc)

Else
zab = DerajatToReal(SudutAB.Text) 'Val(SudutAB.Text)
zbc = DerajatToReal(SudutBC.Text) 'Val(SudutBC.Text)
dab = Val(JAB.Text)
dbc = Val(JBC.Text)

End If
R = Val(Radius.Text)
it = Val(Iterasi.Text)
    If zab > zbc Then
        tt = Abs((zab - zbc))
    Else
        tt = zbc - zab
    End If
    If zbc < 0 Then
        zbc = ((zbc + 0.18))
        tt = (zbc - zab)
    End If
l = (R * tt * (PI / 180))
a = (l / it)
```

```
Select Case Jenis
Case "Selisih Busur":
Fi = ((a / R) * (360 / (2 * PI)))
Case "Polar":
Fi = tt / it
d = (2 * R * Sin(0.5 * Fi * PI / 180))
Case "Polygon":
Fi = tt / it
k = (2 * R * Sin(0.5 * Fi * PI / 180))
End Select
jrkT1T2 = Abs((2 * R) * Sin(tt * PI / 2 / 180))
tgT1B = (R * (Tan((tt / 2) * PI / 180)))
tgBT2 = (R * (Tan((tt / 2) * PI / 180)))
'JarakT1T2.Text = Format(jrkT1T2, "##0.000")

Text3.Text = "sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(dab,
"000.000")
Text2.Text = "sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(dab -
tgT1B, "000.000")
Text4.Text = "sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(dab + dbc,
"000.000")

MaxLine = it
End Sub
Private Sub SelisihBusur()
On Error GoTo hadn
Dim i As Integer
Dim sta As Single
Dim strsta As String
Dim datexist As Boolean

Dim strConn As String
Dim OConn As ADODB.Connection
strConn = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & App.Path &
"/db/data.mdb" & _
  ";User ID=admin"
Set OConn = New ADODB.Connection
Set ORec = New ADODB.Recordset
OConn.Open strConn
ORec.Open "select * from DetilSelisihBusur where IdHOrizontal=" & _
Str(DEKurva.rsHorizontal("ID")), strConn, adOpenDynamic,
adLockBatchOptimistic
sta = dab - tgT1B

If ORec.RecordCount > 1 Then
Set OEx = OConn.Execute("Delete from DetilSelisih Busur where IdHorizontal="
& _
```

```
Select Case jenis
Case "Selisih Busur":
Fi = ((a / R) * (360 / (2 * PI)))
Case "Polar":
Fi = tt / n
d = (2 * R * Sin(0.5 * Fi * PI / 180))
Case "Polygon":
Fi = tt / n
k = (2 * R * Sin(0.5 * Fi * PI / 180))
End Select
jrkTTT2 = Abs((2 * R) * Sin(tt * PI / 2 / 180))
tgT1B = (R * (Tan((tt / 2) * PI / 180)))
tgBT2 = (R * (Tan((tt / 2) * PI / 180)))
'JarakT1T2.Text = Format(jrkTTT2, "##0.000")

Text3.Text = "sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(dab,
"000.000")
Text2.Text = "sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(dab -
tgT1B, "000.000")
Text4.Text = "sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(dab + dbc,
"000.000")

MaxLine = n
End Sub
Private Sub SelisihBusur()
On Error GoTo hadn
Dim i As Integer
Dim sta As Single
Dim strsta As String
Dim dataexist As Boolean

Dim strConn As String
Dim OConn As ADODB.Connection
strConn = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & App.Path &
"\db\data.mdb" & _
";User ID=admin"
Set OConn = New ADODB.Connection
Set ORec = New ADODB.Recordset
OConn.Open strConn
ORec.Open "select * from DetilSelisihBusur where IdHOrizontal=" & _
Str(DBKurva.rsHorizontal!("ID")), strConn, adOpenDynamic,
adLockBatchOptimistic
sta = dab - tgT1B

If ORec.RecordCount > 1 Then
Set OEx = OConn.Execute("Delete from DetilSelisih Busur where IdHorizontal="
& _
```

```
Str(DEKurva.rsHorizontal("ID")))
datexist = True
Else
datexist = False
End If
With MSFlexGrid1
.ColWidth(3) = 1500
.Rows = 1
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "X"
.TextMatrix(0, 2) = "Y"
.TextMatrix(0, 3) = "Stationing"

For i = 1 To it
X = (R * Sin(i * Fi * (PI / 180)))
Y = R - (R * (Cos(i * Fi * PI / 180)))
sta = sta + a
'sta = sta + X
strsta = "sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(sta, "000.000")
.AddItem Str(i) & vbTab & Format(X, "###0.0000") & vbTab & Format(Y,
"###0.0000") & vbTab & strsta

Next
End With
Set ORec = Nothing
Exit Sub
hadn:
 MsgBox "Error" & Err.Description
End Sub
Private Sub Polar()
Dim i As Integer
Dim Sudut As Single
Dim jarak, akum As Single


akum = dab - tgT1B
With MSFlexGrid2
.ColWidth(3) = 1500
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "Sudut"
.TextMatrix(0, 2) = "Jarak"
.TextMatrix(0, 3) = "Stationing"
akum = 0
For i = 1 To it
Sudut = Sudut + 0.5 * Fi
jarak = (2 * R * Sin(Sudut * PI / 180))
X = (R * Sin(i * Fi * (PI / 180)))
```

```
Str(DEKurva.rsHorizontal("ID"))
datexist = True
Else
datexist = False
End If
With MSFlexGrid1
.ColWidth(3) = 1500
.Rows = i
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "X"
.TextMatrix(0, 2) = "Y"
.TextMatrix(0, 3) = "Stationing"

For i = 1 To n
X = (R * Sin(i * Fi * (PI / 180)))
Y = R - (R * (Cos(i * Fi * PI / 180)))
sta = sta + a
'sta = sta + X
strsta = "sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(sta, "000.000")
.AddItem Str(i) & vbTab & Format(X, "##0.0000") & vbTab & Format(Y,
"##0.0000") & vbTab & strsta

Next
End With
Set ORec = Nothing
Exit Sub
hcdn:
MsgBox "Error" & Err.Description
End Sub
Private Sub Polar()
Dim i As Integer
Dim Sudut As Single
Dim jarak, akum As Single


akum = dab - tgT1B
With MSFlexGrid2
.ColWidth(3) = 1500
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "Sudut"
.TextMatrix(0, 2) = "Jarak"
.TextMatrix(0, 3) = "Stationing"
akum = 0
For i = 1 To n
Sudut = Sudut + 0.5 * Fi
jarak = (2 * R * Sin(Sudut * PI / 180))
X = (R * Sin(i * Fi * (PI / 180)))
```

```
Y = R - (R * (Cos(i * Fi * PI / 180)))
ArrGaris(i).X = X
ArrGaris(i).Y = Y
akum = akum + jarak
.AddItem Str(i) & vbTab & SudutString(Sudut) & vbTab & Format(jarak,
"000.0000") & _
vbTab & "sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(akum,
"000.0000")

Next
End With

End Sub
Private Sub Polygon()
Dim i As Integer
Dim Sudut As Single
Dim jarak, sta As Single
With MSFlexGrid3
.ColWidth(3) = 1500
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "Jarak"
.TextMatrix(0, 2) = "Sudut"
.TextMatrix(0, 3) = "Stationing"
sta = dab - tgT1B
For i = 1 To it
If i = 1 Or i = it Then
Sudut = 90 - Fi / 2
Else
Sudut = 180 - Fi
End If
jarak = (2 * R * Sin(0.5 * Fi * PI / 180))
X = (R * Sin(i * Fi * (PI / 180)))
Y = R - (R * (Cos(i * Fi * PI / 180)))
ArrGaris(i).X = X
ArrGaris(i).Y = Y
sta = sta + a
.AddItem Str(i) & vbTab & Format(jarak, "###0.0000") & vbTab &
SudutString(Sudut) & vbTab & _
  "sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(sta, "000.000")

Next
End With

End Sub
```

```
Y = R - (R * (Cos(i * Fi * PI / 180)))
ArrGaris(i).X = X
ArrGaris(i).Y = Y
akum = akum + jarak
.AddItem Str(i) & vbTab & SudutString(Sudut) & vbTab & Format(jarak, _
"000.0000") & _
vbTab & "sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(akum, _
"000.0000")

Next
End With

End Sub
Private Sub Polygon()
Dim i As Integer
Dim Sudut As Single
Dim jarak, sta As Single
With MSFlexGrid3
.ColWidth(3) = 1500
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "Jarak"
.TextMatrix(0, 2) = "Sudut"
.TextMatrix(0, 3) = "Stationing"
sta = dab - tgT1B
For i = 1 To it
If i = 1 Or i = it Then
Sudut = 90 - Fi / 2
Else
Sudut = 180 - Fi
End If
jarak = (2 * R * Sin(0.5 * Fi * PI / 180))
X = (R * Sin(i * Fi * (PI / 180)))
Y = R - (R * (Cos(i * Fi * PI / 180)))
.ArrGaris(i).X = X
ArrGaris(i).Y = Y
sta = sta + a
.AddItem Str(i) & vbTab & Format(jarak, "##0.0000") & vbTab & _
SudutString(Sudut) & vbTab & _
"sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(sta, "000.000")

Next
End With

End Sub
```

```
Private Sub cmdNav_Click(Index As Integer)
Select Case Index
 Case 0:
  If Not DEKurva.rsHorizontal.BOF Then DEKurva.rsHorizontal.MovePrevious
 Case 1:
  If Not DEKurva.rsHorizontal.EOF Then DEKurva.rsHorizontal.MoveNext

End Select
End Sub

Private Sub cmdProses_Click()

ProsesPerhitungan DEKurva.rsHorizontal("Jenis")

With DEKurva
.rsHorizontal("Theta") = SudutString(tt)
.rsHorizontal("LBusur") = l
'txtA.Text = Format(a, "###.0000")
.rsHorizontal("Fi") = SudutString(Fi)
.rsHorizontal("TanIB") = tgT1B
.rsHorizontal("TanBT") = tgBT2
.rsHorizontal("a") = a
.rsHorizontal("TaliBusur") = jrkT1T2

mnuSave_Click
If .rsHorizontal("jenis") <> "" Then
Select Case DEKurva.rsHorizontal("Jenis")
Case "Selisih Busur":
 SelisihBusur
Case "Polar":
 Polar
Case "Polygon":
 Polygon
End Select


End If
End With
End Sub

Private Sub Form_Load()
Dim vScale As String


Open App.Path & "\option.ini" For Input As #1
Input #1, vScale
GScale = Val(vScale)
```

```
Private Sub cmdNav_Click(Index As Integer)
Select Case Index
 Case 0:
 If Not DEKurva.rsHorizontal.BOF Then DEKurva.rsHorizontal.MovePrevious
 Case 1:
 If Not DEKurva.rsHorizontal.EOF Then DEKurva.rsHorizontal.MoveNext

End Select
End Sub

Private Sub cmdProses_Click()

ProsesPerhitungan DEKurva.rsHorizontal("Jenis")

With DEKurva
 .rsHorizontal("Theta") = SudutString(t)
 .rsHorizontal("LBusur") = l
 txtA.Text = Format(a, "##.0000")
 .rsHorizontal("Fi") = SudutString(Fi)
 .rsHorizontal("TanB") = tgT1B
 .rsHorizontal("TanBT") = tgBT2
 .rsHorizontal("a") = a
 .rsHorizontal("TaliBusur") = jdkT1T2

 mnuSave_Click
 If .rsHorizontal("Jenis") <> "" Then
 Select Case DEKurva.rsHorizontal("Jenis")
 Case "Selisih Busur":
 SelisihBusur
 Case "Polar":
 Polar
 Case "Polygon":
 Polygon
 End Select


 End If
 End With
 End Sub

 Private Sub Form_Load()
 Dim vScale As String


 Open App.Path & "\option.ini" For Input As #1
 Input #1, vScale
 GScale = Val(vScale)
```

```
Close #1
With Combo1
.AddItem "Selisih Busur"
.AddItem "Polar"
.AddItem "Polygon"
End With
'DEKurva.rsDetilSelisihBusur.Open

End Sub

Private Sub Form_Unload(Cancel As Integer)
Operation = Combo1.ListIndex + 1
End Sub

Private Sub KX_LostFocus(Index As Integer)
Dim dummy As String
If Index = 2 Then
 If Val(KX(2).Text) < Val(KX(0).Text) Then
   dummy = KX(0).Text
   KX(0).Text = KX(2).Text
   KX(2).Text = dummy
 End If
End If
End Sub

Private Sub mnuGambar_Click()
GambarKurva
End Sub

Private Sub mnuNew_Click()
DEKurva.rsHorizontal.AddNew
KX(0).Text = "0"
KY(0).Text = "0"
KX(1).Text = "0"
KY(1).Text = "0"
KX(2).Text = "0"
KY(2).Text = "0"

End Sub

Private Sub mnuOpen_Click()
With DEKurva
.rsHorizontal.MoveLast

End With
End Sub
```

```
Close #1
With Combo1
.AddItem "Selisih Besar"
.AddItem "Polar"
.AddItem "Polygon"
End With
DEKurva.rsDenSelisihBesar.Open

End Sub

Private Sub Form_Unload(Cancel As Integer)
Operation = Combo1.ListIndex + 1
End Sub

Private Sub KX_LostFocus(Index As Integer)
Dim dummy As String
If Index = 2 Then
 If Val(KX(2).Text) < Val(KX(0).Text) Then
  dummy = KX(0).Text
  KX(0).Text = KX(2).Text
  KX(2).Text = dummy
 End If
End If
End Sub

Private Sub mnuGambar_Click()
GambarKurva
End Sub

Private Sub mnuNew_Click()
DEKurva.rsHorizontal.AddNew
KX(0).Text = "0"
KY(0).Text = "0"
KX(1).Text = "0"
KY(1).Text = "0"
KX(2).Text = "0"
KY(2).Text = "0"

End Sub

Private Sub mnuOpen_Click()
With DEKurva
.rsHorizontal.MoveLast

End With
End Sub
```

```
Case "Polar":
 ReportPolar.Refresh
 ReportPolar.Show

Case "Polygon":
 ReportPolygon.Refresh
 ReportPolygon.Show

 Case Else:
 ReportSelisihBusur.Refresh
 ReportSelisihBusur.Show
End Select
End Sub
Sub save_detil_selisih_busur()
Dim i As Integer
Dim Recs As ADODB.Recordset
Dim constr As String
constr = get_connection
With DEKurva
DeleteTable "DetilSelisihbusur", "Idhorizontal=" & .rsHorizontal("Id")
Set Recs = New ADODB.Recordset

Recs.Open "select * from DetilSelisihBusur", constr, adOpenDynamic,
adLockBatchOptimistic
For i = 1 To MSFlexGrid1.Rows - 1
Recs.AddNew
Recs("IdDetil") = MSFlexGrid1.TextMatrix(i, 0)
Recs("Idhorizontal") = .rsHorizontal("Id")
Recs("X") = Val(Replace(MSFlexGrid1.TextMatrix(i, 1), ",", "."))
Recs("Y") = Val(Replace(MSFlexGrid1.TextMatrix(i, 2), ",", "."))
Recs("Stationing") = MSFlexGrid1.TextMatrix(i, 3)
Recs.UpdateBatch adAffectAll
Next
End With
Set Recs = Nothing

End Sub
Sub save_detil_polar()
Dim i As Integer
```

```
Case "Polar":
 ReportPolar.Refresh
 ReportPolar.Show

Case "Polygon":
 ReportPolygon.Refresh
 ReportPolygon.Show

Case Else:
 ReportSelisihBusur.Refresh
 ReportSelisihBusur.Show
End Select
End Sub
Sub save_detil_selisih_busur()
Dim i As Integer
Dim Recs As ADODB.Recordset
Dim constr As String
constr = get_connection
With DEKurva
DeleteTable "DetilSelisihBusur", "Idhorizontal=" & .rsHorizontal("Id")
Set Recs = New ADODB.Recordset

Recs.Open "select * from DetilSelisihBusur", constr, adOpenDynamic,
adLockBatchOptimistic
For i = 1 To MSFlexGrid1.Rows - 1
Recs.AddNew
Recs("IdDetil") = MSFlexGrid1.TextMatrix(i, 0)
Recs("Idhorizontal") = .rsHorizontal("Id")
Recs("X") = Val(Replace(MSFlexGrid1.TextMatrix(i, 1), ",", "."))
Recs("Y") = Val(Replace(MSFlexGrid1.TextMatrix(i, 2), ",", "."))
Recs("Stationing") = MSFlexGrid1.TextMatrix(i, 3)
Recs.UpdateBatch adAffectAll
Next
End With
Set Recs = Nothing

End Sub
Sub save_detil_polar()
Dim i As Integer
```

```
Dim Recs As ADODB.Recordset
Dim constr As String
constr = get_connection
With DEKurva
DeleteTable "DetilPolar", "Idhorizontal=" & .rsHorizontal("Id")
Set Recs = New ADODB.Recordset

Recs.Open "select * from DetilPolar", constr, adOpenDynamic,
adLockBatchOptimistic
For i = 1 To MSFlexGrid2.Rows - 1
Recs.AddNew
Recs("IdPolar") = MSFlexGrid2.TextMatrix(i, 0)
Recs("Idhorizontal") = .rsHorizontal("Id")
Recs("Sudut") = MSFlexGrid2.TextMatrix(i, 1)
Recs("Jarak") = Val(Replace(MSFlexGrid2.TextMatrix(i, 2), ",", "."))
Recs("Stationing") = MSFlexGrid2.TextMatrix(i, 3)
Recs.UpdateBatch adAffectAll
Next
End With
Set Recs = Nothing

End Sub
Sub save_detil_polygon()
Dim i As Integer
Dim Recs As ADODB.Recordset
Dim constr As String
constr = get_connection
With DEKurva
DeleteTable "DetilPolygon", "Idhorizontal=" & .rsHorizontal("Id")
Set Recs = New ADODB.Recordset

Recs.Open "select * from DetilPolygon", constr, adOpenDynamic,
adLockBatchOptimistic
For i = 1 To MSFlexGrid3.Rows - 1
Recs.AddNew
Recs("IdPolygon") = MSFlexGrid3.TextMatrix(i, 0)
Recs("Idhorizontal") = .rsHorizontal("Id")
Recs("Jarak") = Val(Replace(MSFlexGrid3.TextMatrix(i, 1), ",", "."))
Recs("Sudut") = MSFlexGrid3.TextMatrix(i, 2)

Recs("Stationing") = MSFlexGrid3.TextMatrix(i, 3)
Recs.UpdateBatch adAffectAll
Next
End With
Set Recs = Nothing

End Sub
```

```
Dim Recs As ADODB.Recordset
Dim constr As String
constr = get_connection
With DEKurva
DeleteTable "DetilPolar", "Idhorizontal=" & .rsHorizontal("Id")
Set Recs = New ADODB.Recordset

Recs.Open "select * from DetilPolar", constr, adOpenDynamic,
adLockBatchOptimistic
For i = 1 To MSFlexGrid2.Rows - 1
Recs.AddNew
Recs("IdPolar") = MSFlexGrid2.TextMatrix(i, 0)
Recs("Idhorizontal") = .rsHorizontal("Id")
Recs("Sudut") = MSFlexGrid2.TextMatrix(i, 1)
Recs("Jarak") = Val(Replace(MSFlexGrid2.TextMatrix(i, 2), ",", "."))
Recs("Stationing") = MSFlexGrid2.TextMatrix(i, 3)
Recs.UpdateBatch adAffectAll
Next
End With
Set Recs = Nothing

End Sub
Sub save_detil_polygon()
Dim i As Integer
Dim Recs As ADODB.Recordset
Dim constr As String
constr = get_connection
With DEKurva
DeleteTable "DetilPolygon", "Idhorizontal=" & .rsHorizontal("Id")
Set Recs = New ADODB.Recordset

Recs.Open "select * from DetilPolygon", constr, adOpenDynamic,
adLockBatchOptimistic
For i = 1 To MSFlexGrid3.Rows - 1
Recs.AddNew
Recs("IdPolygon") = MSFlexGrid3.TextMatrix(i, 0)
Recs("Idhorizontal") = .rsHorizontal("Id")
Recs("Jarak") = Val(Replace(MSFlexGrid3.TextMatrix(i, 1), ",", "."))
Recs("Sudut") = MSFlexGrid3.TextMatrix(i, 2)

Recs("Stationing") = MSFlexGrid3.TextMatrix(i, 3)
Recs.UpdateBatch adAffectAll
Next
End With
Set Recs = Nothing

End Sub
```

```
Private Sub mnuRefresh_Click()

DEKurva.rsHorizontal.Resync


End Sub

Private Sub mnuSave_Click()
With DEKurva
If .rsHorizontal.EditMode = adEditAdd Then
.rsHorizontal.UpdateBatch adAffectAll
End If

Select Case .rsHorizontal("Jenis")
Case "Selisih Busur":
 Call save_detil_selisih_busur
Case "Polar":
 Call save_detil_polar
Case "Polygon":
 Call save_detil_polygon
End Select
End With
End Sub
```

```
Private Sub mnuRefresh_Click()

DEKurva.rsHorizontal.Resync

End Sub

Private Sub mnuSave_Click()
With DEKurva
If .rsHorizontal.EditMode = adEditAdd Then
.rsHorizontal.UpdateBatch adAffectAll
End If

Select Case .rsHorizontal("Jenis")
Case "Selisih Busur":
 Call save_detil_selisih_busur
Case "Polar":
 Call save_detil_polar
Case "Polygon":
 Call save_detil_polygon
End Select
End With
End Sub
```

KURVA VERTIKAL

```
Option Explicit
Dim g1, g2, a As Single
Dim AddNewFlag As Boolean
Private Sub ProsesPenghitungan()
'DEKurva.rsVertical.UpdateBatch adAffectAll
With VertCurve
   .Acuan = DEKurva.rsVertical("Acuan")
   .Panjang = DEKurva.rsVertical("Panjang")
   .Iterasi = DEKurva.rsVertical("sto")
   If DEKurva.rsVertical("g1") <> 0# Or DEKurva.rsVertical("g2") <> 0# Then
   .Elev(0) = DEKurva.rsVertical("EAwal")
   .Elev(1) = .Elev(0) + (DEKurva.rsVertical("g1") * (.Panjang / 2))
   .Elev(2) = .Elev(1) + (DEKurva.rsVertical("g2") * (.Panjang / 2))

   .g1 = DEKurva.rsVertical("g1")
   .g2 = DEKurva.rsVertical("g2")
   'DEKurva.rsVertical("EAwal") = .Elev(0)
   'DEKurva.rsVertical("ETengah") = .Elev(1)
   'DEKurva.rsVertical("EAkhir") = .Elev(2)
   If .Acuan > DEKurva.rsVertical("EAwal") Or _
   .Acuan > DEKurva.rsVertical("Etengah") Or _
    .Acuan > DEKurva.rsVertical("EAkhir") Then
    MsgBox "Nilai Acuan tidak boleh melebihi tinggi elevasi", vbCritical
    Acuan.SetFocus
    End If
   Else
   .Elev(0) = DEKurva.rsVertical("EAwal")
   .Elev(1) = DEKurva.rsVertical("ETengah")
   .Elev(2) = DEKurva.rsVertical("EAkhir")
      If .Elev(0) = .Elev(1) Then
      .g1 = 0
      Else
      .g1 = (.Elev(1) - .Elev(0)) / (.Panjang / 2)
      End If
      If .Elev(2) = .Elev(1) Then
      .g2 = 0
      Else
      .g2 = (.Elev(2) - .Elev(1)) / (.Panjang / 2)
      End If
   End If

.a = (.g2 - .g1) / 2 / .Panjang
DEKurva.rsVertical("a") = .a
DEKurva.rsVertical("g1") = .g1
DEKurva.rsVertical("g2") = .g2
```

KURVA VERTIKAL

```
Option Explicit
Dim g1, g2, a As Single
Dim AddNewFlag As Boolean
Private Sub ProsesPenghitungan()
'DEKurva.rsVertical.UpdateBatch adAffectAll
With VertCurve
    .Acuan = DEKurva.rsVertical("Acuan")
    .Panjang = DEKurva.rsVertical("Panjang")
    .Iterasi = DEKurva.rsVertical("sto")
    If DEKurva.rsVertical("g1") <> 0# Or DEKurva.rsVertical("g2") <> 0# Then
    .Elev(0) = DEKurva.rsVertical("EAwal")
    .Elev(1) = .Elev(0) + (DEKurva.rsVertical("g1") * (.Panjang / 2))
    .Elev(2) = .Elev(1) + (DEKurva.rsVertical("g2") * (.Panjang / 2))

    .g1 = DEKurva.rsVertical("g1")
    .g2 = DEKurva.rsVertical("g2")
    'DEKurva.rsVertical("EAwal") = .Elev(0)
    'DEKurva.rsVertical("ETengah") = .Elev(1)
    'DEKurva.rsVertical("EAkhir") = .Elev(2)
    If .Acuan > DEKurva.rsVertical("EAwal") Or _
    .Acuan > DEKurva.rsVertical("Etengah") Or _
    .Acuan > DEKurva.rsVertical("EAkhir") Then
    MsgBox "Nilai Acuan tidak boleh melebihi tinggi elevasi", vbCritical
    .Acuan.SetFocus
    End If
    Else
    .Elev(0) = DEKurva.rsVertical("EAwal")
    .Elev(1) = DEKurva.rsVertical("ETengah")
    .Elev(2) = DEKurva.rsVertical("EAkhir")
        If .Elev(0) = .Elev(1) Then
        .g1 = 0
        Else
        .g1 = (.Elev(1) - .Elev(0)) / (.Panjang / 2)
        End If
        If .Elev(2) = .Elev(1) Then
        .g2 = 0
        Else
        .g2 = (.Elev(2) - .Elev(1)) / (.Panjang / 2)
        End If
    End If

.a = (.g2 - .g1) / 2 / .Panjang
DEKurva.rsVertical("a") = .a
DEKurva.rsVertical("g1") = .g1
DEKurva.rsVertical("g2") = .g2
```

```
'DEKurva.rsVertical.UpdateBatch adAffectAll
End With

End Sub
Private Sub Command1_Click()
Dim i As Integer
Dim Y, X As Single
Dim delta, sta As Single
With VertCurve
Call ProsesPenghitungan
delta = .Panjang / .Iterasi
With MGrid
If .Rows > 1 Then
 .Rows = 1

 End If
.Refresh
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "Jarak"
.TextMatrix(0, 2) = "Elevasi"
.TextMatrix(0, 3) = "Stationing"
.ColWidth(0) = 500
.ColWidth(3) = 1500
sta = Val(Right(Text1.Text, 3))
.AddItem Str(0) & vbTab & Str(0) & vbTab &
FormatNumber(VertCurve.Elev(0), 3) & vbTab & _
   "sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(sta, "000.000")
ArrLine(0).X = 0#
ArrLine(0).Y = VertCurve.Elev(0)

 For i = 1 To VertCurve.Iterasi
   X = X + delta
   Y = VertCurve.a * (X ^ 2) + VertCurve.g1 * X + VertCurve.Elev(0)
   ArrLine(i).X = X
   ArrLine(i).Y = Y
    .AddItem Str(i) & vbTab & Str(X) & vbTab & FormatNumber(Y, 3) & vbTab
& _
   "sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(sta + X, "000.000")
 Next i
End With
 MaxLine = .Iterasi
MGrid.FixedRows = 1
End With
End Sub

Private Sub Command2_Click()
Operation = 4
```

```
'DEKurva.rsVertical.UpdateBatch adAffectAll
End With

End Sub
Private Sub Command1_Click()
Dim i As Integer
Dim Y, X As Single
Dim delta, sta As Single
With VertCurve
Call ProsesPenghitungan
delta = .Panjang / .Iterasi
With MGrid
If .Rows < 1 Then
.Rows = 1

End If
.Refresh
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "Jarak"
.TextMatrix(0, 2) = "Elevasi"
.TextMatrix(0, 3) = "Stationing"
.ColWidth(0) = 500
.ColWidth(3) = 1500
sta = Val(Right(Text1.Text, 3))
.AddItem Str(0) & vbTab & Str(0) & vbTab & _
FormatNumber(VertCurve.Elev(0), 3) & vbTab & _
"sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(sta, "000.000")
ArrLine(0).X = 0#
ArrLine(0).Y = VertCurve.Elev(0)

For i = 1 To VertCurve.Iterasi
X = X + delta
Y = VertCurve.a * (X ^ 2) + VertCurve.gi * X + VertCurve.Elev(0)
ArrLine(i).X = X
ArrLine(i).Y = Y
.AddItem Str(i) & vbTab & Str(X) & vbTab & FormatNumber(Y, 3) & vbTab
& _
"sta " & Right(Left(Text1.Text, 6), 2) & " + " & Format(sta + X, "000.000")
Next i
End With
.MaxLine = .Iterasi
.MGrid.FixedRows = 1
End With
End Sub

Private Sub Command2_Click()
Operation = 4
```

```
MaxPoint = 1
MainFrm.Toolbar1.Buttons(4).Value = tbrPressed
Unload Me
End Sub

Private Sub mnuGambar_Click()
Command1_Click

End Sub

Private Sub mnuACAD_Click()

End Sub

Private Sub Command3_Click()
If Not DEKurva.rsVertical.BOF Then DEKurva.rsVertical.MovePrevious
End Sub

Private Sub Command4_Click()
If Not DEKurva.rsVertical.EOF Then DEKurva.rsVertical.MoveNext
End Sub

Private Sub Form_Load()
If DEKurva.rsVertical.RecordCount < 1 Then
 DEKurva.rsVertical.AddNew
 End If
AddNewFlag = False
End Sub

Private Sub Form_Unload(Cancel As Integer)
Cancel = False
End Sub

Private Sub mnuClose_Click()
Unload Me
End Sub

Private Sub mnuNew_Click()
 AddNewFlag = True
 DEKurva.rsVertical.AddNew
End Sub

Private Sub mnuSave_Click()
On Error Resume Next
With DEKurva.rsVertical
.UpdateBatch adAffectAll
End With
```

```
Dim objCon As ADODB.Connection
Dim objRec As ADODB.Recordset
Dim connstr As String
Dim strSQL As String
Dim i As Integer
connstr = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & App.Path & _
"\db\data.mdb" & ";User ID=Admin;Password=;"
Set objCon = New ADODB.Connection
objCon.Open connstr
Set objRec = objCon.Execute("Delete from detilvertikal where IDVertikal=" &
DEKurva.rsVertical("IdVertikal"))

Set objRec = New ADODB.Recordset
strSQL = "select * from detilvertikal where IDVertikal=" &
DEKurva.rsVertical("IdVertikal")
objRec.Open strSQL, connstr, adOpenDynamic, adLockOptimistic

For i = 1 To MGrid.Rows - 1
 objRec.AddNew
 objRec("IdDetil") = i
 objRec("IdVertikal") = DEKurva.rsVertical("IdVertikal")
 objRec("Jarak") = Val(MGrid.TextMatrix(i, 1))
 objRec("Elevasi") = MGrid.TextMatrix(i, 2)
 objRec("Stationing") = MGrid.TextMatrix(i, 3)
 objRec.UpdateBatch adAffectAll
Next

End Sub
```

```
Dim objCon As ADODB.Connection
Dim objRec As ADODB.Recordset
Dim connstr As String
Dim strSQL As String
Dim i As Integer
connstr = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & App.Path &
"/db/data.mdb" & ";User ID=Admin;Password=;"
Set objCon = New ADODB.Connection
objCon.Open connstr
Set objRec = objCon.Execute("Delete from detilvertikal where IDVertikal=" &
DEKurva.rsVertical("IdVertikal"))

Set objRec = New ADODB.Recordset
strSQL = "select * from detilvertikal where IDVertikal=" &
DEKurva.rsVertical("IdVertikal")
objRec.Open strSQL, connstr, adOpenDynamic, adLockOptimistic

For i = 1 To MGrid.Rows - 1
objRec.AddNew
objRec("IdDetil") = i
objRec("IdVertikal") = DEKurva.rsVertical("IdVertikal")
objRec("Jarak") = Val(MGrid.TextMatrix(i, 1))
objRec("Elevasi") = MGrid.TextMatrix(i, 2)
objRec("Station") = MGrid.TextMatrix(i, 3)
objRec.UpdateBatch adAffectAll
Next

End Sub
```

BUSUR SPIRAL

```
Const phi = 3.14159265358979
Dim d, Xs, Ys, Ss, p, k, Es, Lc, Ltot, Ts As Double
Dim rc, Vr, Ls, a As Integer
Dim MaxLine As Integer
Dim delta, dab, dbc As Double
Dim Cs As Double
Private Sub Bulatan(ByVal X As Single, ByVal Y As Single, ByVal Warna As
Long)
 MainFrm.Pic.DrawWidth = 4
 MainFrm.Pic.PSet (X, Y), Warna
 MainFrm.Pic.DrawWidth = 1
End Sub
Private Sub gambar_titik(X As Integer, Y As Integer, color As Integer)
MainFrm.Pic.PSet (X, Y), QBColor(color)
End Sub
Private Sub gambar_garis(x1 As Integer, Y1 As Integer, x2 As Integer, Y2 As
Integer, color As Integer)
MainFrm.Pic.Line (x1, Y1)-(x2, Y2), QBColor(color)
End Sub
Private Sub hotSpot(ByVal X As Single, ByVal Y As Single, ByVal Text As
String, ByVal WithDot As Boolean)
If WithDot Then Bulatan X, Y, QBColor(12)
With MainFrm
.Pic.DrawStyle = vbSolid
.Pic.CurrentX = X + 50
.Pic.CurrentY = Y - 20
.Pic.Print Text

End With
End Sub
Private Sub ProsesPenghitungan()
Dim sud As Single
With DEKurva
If (.rsSpiral("YP1").Value - .rsSpiral("YA").Value) <> 0& _
And (.rsSpiral("YC").Value - .rsSpiral("YP1").Value) <> 0& _
Then
'Sudut(1) = Atn((Val(KX(1).Text) - Val(KX(0).Text)) / (Val(KY(1).Text) -
Val(KY(0).Text)))
Sudut(1) = Atn((.rsSpiral("XP1") - .rsSpiral("XA")) / (.rsSpiral("YP1") -
.rsSpiral("YA")))

'Sudut(2) = Atn((Val(KX(2).Text) - Val(KX(1).Text)) / (Val(KY(2).Text) -
Val(KY(1).Text)))
Sudut(2) = Atn((.rsSpiral("XC") - .rsSpiral("XP1")) / (.rsSpiral("YC") -
.rsSpiral("YP1")))
```

BUSUR SPIRAL

```
Const phi = 3.14159265358979
Dim d, Xs, Ys, Ss, p, k, Es, Lc, Ltot, Ts As Double
Dim rc, Vr, Ls, a As Integer
Dim MaxLine As Integer
Dim delta, dab, dbc As Double
Dim Cs As Double
Private Sub Bulatan(ByVal X As Single, ByVal Y As Single, ByVal Warna As Long)
 MainFrm.Pic.DrawWidth = 4
 MainFrm.Pic.PSet (X, Y), Warna
 MainFrm.Pic.DrawWidth = 1
End Sub
Private Sub gambar_titik(X As Integer, Y As Integer, color As Integer)
MainFrm.Pic.PSet (X, Y), QBColor(color)
End Sub
Private Sub gambar_garis(x1 As Integer, Y1 As Integer, x2 As Integer, Y2 As Integer, color As Integer)
MainFrm.Pic.Line (x1, Y1)-(x2, Y2), QBColor(color)
End Sub
Private Sub hotSpot(ByVal X As Single, ByVal Y As Single, ByVal Text As String, ByVal WithDot As Boolean)
If WithDot Then Bulatan X, Y, QBColor(12)
With MainFrm
 .Pic.DrawStyle = vbSolid
 .Pic.CurrentX = X + 50
 .Pic.CurrentY = Y - 20
 .Pic.Print Text

End With
End Sub
Private Sub ProsesPenghitungan()
Dim sud As Single
With DEKurva
If (.rsSpiral("YP1").Value - .rsSpiral("YA").Value) <> 0& _
And (.rsSpiral("YC").Value - .rsSpiral("YP1").Value) <> 0& _
Then
'Sudut(1) = Atn((Val(KX(1).Text) - Val(KX(0).Text)) / (Val(KY(1).Text) - Val(KY(0).Text)))
Sudut(1) = Atn((.rsSpiral("XP1") - .rsSpiral("XA")) / (.rsSpiral("YP1") - .rsSpiral("YA")))

'Sudut(2) = Atn((Val(KX(2).Text) - Val(KX(1).Text)) / (Val(KY(2).Text) - Val(KY(1).Text)))
Sudut(2) = Atn((.rsSpiral("XC") - .rsSpiral("XP1")) / (.rsSpiral("YC") - .rsSpiral("YP1")))
```

```
Sudut(3) = Sudut(2) - Sudut(1)
.rsSpiral("JarakAP1") = Sqr((.rsSpiral("XP1") - .rsSpiral("XA")) ^ 2 +
(.rsSpiral("YP1") - .rsSpiral("YA")) ^ 2)
.rsSpiral("JarakP1B") = Sqr((.rsSpiral("XC") - .rsSpiral("XP1")) ^ 2 +
(.rsSpiral("YC") - .rsSpiral("YP1")) ^ 2)
zab = Sudut(1) * 180 / PI
zbc = Sudut(2) * 180 / PI
If zbc < 0 Then zbc = zbc + 180
d = zbc - zab
.rsSpiral("ZAP1") = SudutString(zab)
.rsSpiral("ZP1B") = SudutString(zbc)
.rsSpiral("d") = SudutString(d)
Else
If SudutAB.Text <> "" And SudutBC.Text <> "" Then
zab = DerajatToReal(SudutAB.Text) 'Val(SudutAB.Text)
zbc = DerajatToReal(SudutBC.Text) 'Val(SudutBC.Text)

If zbc < 0 Then zbc = zbc + 180
d = zbc - zab
Else
d = DerajatToReal(Txt0.Text)
End If
.rsSpiral("d") = SudutString(d)
End If

If JAB.Text <> "" And JBC.Text <> "" Then
dab = .rsSpiral("JarakAP1")
dbc = .rsSpiral("JarakP1B")
End If
'd = DerajatToReal(Txt0.Text)
rc = .rsSpiral("R")
Vr = .rsSpiral("Vr")
.rsSpiral("Ls") = (Vr / 3.6) * 3
.rsSpiral("Xs") = .rsSpiral("Ls") * (1 - (.rsSpiral("Ls") ^ 2 / (40 * .rsSpiral("R") ^
2)))
.rsSpiral("Ys") = .rsSpiral("Ls") ^ 2 / (6 * .rsSpiral("R"))
Ss = (90 / phi) * (.rsSpiral("Ls") / rc)
.rsSpiral("Ss") = SudutString(Ss)

.rsSpiral("p") = (.rsSpiral("Ls") ^ 2 / (6 * rc)) - (rc * (1 - Cos(Ss * phi / 180)))
.rsSpiral("k") = (.rsSpiral("Ls") - (.rsSpiral("Ls") ^ 3 / (40 * rc ^ 2)) - (rc * Sin(Ss
* phi / 180)))
.rsSpiral("Ts") = (((rc + .rsSpiral("p")) * Tan(0.5 * d * phi / 180)) + .rsSpiral("k"))
.rsSpiral("Es") = ((rc + .rsSpiral("p")) * 1 / Cos(0.5 * d * phi / 180) - rc)

sud = (d - (2 * Ss)) / 180
```

```
Sudut(3) = Sudut(2) - Sudut(1)
.rsSpiral("JarakAP1") = Sqr((.rsSpiral("XP1") - .rsSpiral("XA")) ^ 2 +
(.rsSpiral("YP1") - .rsSpiral("YA")) ^ 2)
.rsSpiral("JarakP1B") = Sqr((.rsSpiral("XC") - .rsSpiral("XP1")) ^ 2 +
(.rsSpiral("YC") - .rsSpiral("YP1")) ^ 2)
zab = Sudut(1) * 180 / PI
zbc = Sudut(2) * 180 / PI
If zbc < 0 Then zbc = zbc + 180
d = zbc - zab
.rsSpiral("ZAP1") = SudutString(zab)
.rsSpiral("ZP1B") = SudutString(zbc)
.rsSpiral("d") = SudutString(d)
Else
If SudutAB.Text <> "" And SudutBC.Text <> "" Then
zab = DerajatToReal(SudutAB.Text) 'Val(SudutAB.Text)
zbc = DerajatToReal(SudutBC.Text) 'Val(SudutBC.Text)

If zbc < 0 Then zbc = zbc + 180
d = zbc - zab
Else
d = DerajatToReal(Txt0.Text)
End If
.rsSpiral("d") = SudutString(d)
End If

If JAB.Text <> "" And JBC.Text <> "" Then
dab = .rsSpiral("JarakAP1")
dbc = .rsSpiral("JarakP1B")
End If
'd = DerajatToReal(Txt0.Text)
rc = .rsSpiral("R")
Vr = .rsSpiral("Vr")
.rsSpiral("Ls") = (Vr / 3.6) * 3
.rsSpiral("Xs") = .rsSpiral("Ls") * (1 - (.rsSpiral("Ls") ^ 2 / (40 * .rsSpiral("R") ^
2)))
.rsSpiral("Ys") = .rsSpiral("Ls") ^ 2 / (6 * .rsSpiral("R"))
Ss = (90 / phi) * (.rsSpiral("Ls") / rc)
.rsSpiral("Ss") = SudutString(Ss)

.rsSpiral("p") = (.rsSpiral("Ls") ^ 2 / (6 * rc)) - (rc * (1 - Cos(Ss * phi / 180)))
.rsSpiral("k") = (.rsSpiral("Ls") - (.rsSpiral("Ls") ^ 3 / (40 * rc ^ 2)) - (rc * Sin(Ss
* phi / 180)))
.rsSpiral("Ts") = ((rc + .rsSpiral("p")) * Tan(0.5 * d * phi / 180)) + .rsSpiral("k")
.rsSpiral("Es") = (rc + .rsSpiral("p")) * 1 / Cos(0.5 * d * phi / 180) - rc

sud = (d - (2 * Ss)) / 180
```

```
Lc = sud * phi * rc
.rsSpiral("Lc") = Lc
.rsSpiral("a") = .rsSpiral("Lc") / .rsSpiral("sto2")
.rsSpiral("Ltot") = .rsSpiral("Lc") + (2 * .rsSpiral("Ls"))
delta = .rsSpiral("Ls") / .rsSpiral("sto1")
.rsSpiral("Cs") = 0.0031 * (Ss ^ 3)
.rsSpiral.UpdateBatch adAffectAll
TextP1.Text = "sta " & Right(Left(TextA.Text, 6), 2) & _
" + " & Format(dab, "000.000")
TextB.Text = "sta " & Right(Left(TextA.Text, 6), 2) & _
" + " & Format(dab + dbc, "000.000")
TextTS.Text = "sta " & Right(Left(TextA.Text, 6), 2) & _
" + " & Format(dab - .rsSpiral("Ts"), "000.000")
ST = dab - .rsSpiral("Ts") + .rsSpiral("Ltot")
TextST.Text = "sta " & Right(Left(TextA.Text, 6), 2) & _
" + " & Format(ST, "000.000")
End With

End Sub
Private Sub Command1_Click()
Dim i As Integer
Dim X As Double
Dim jrab, jrbc, ST As Single
Dim sdte, delta, station, sd, started As Single
Dim Sudut(3), Radsudut(3) As Single

ProsesPenghitungan

With FGHasil
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "Jarak Detail (Ls)"
.TextMatrix(0, 2) = "Sudut Lentur"
.TextMatrix(0, 3) = "Stationing"
MaxLine = Val(txtI.Text)
sto1 = MaxLine
.Rows = MaxLine + 1
.ColWidth(0) = 500
.ColWidth(2) = 1600
End With
'properti grid untuk lingkaran
sdte = d - (2 * Ss)
delta = sdte / Val(Text9.Text)
FGLing.Rows = Val(Text9.Text) + 1
sd = 0
station = dab - Ts + Ls
With FGLing
.ColWidth(2) = 1500
```

```
Lc = sud * phi * rc
.rsSpiral("Lc") = Lc
.rsSpiral("a") = .rsSpiral("Lc") / .rsSpiral("sto2")
.rsSpiral("Ltot") = .rsSpiral("Lc") + (2 * .rsSpiral("Ls"))
delta = .rsSpiral("Ls") / .rsSpiral("sto1")
.rsSpiral("Cs") = 0.0031 * (Ss ^ 3)
.rsSpiral.UpdateBatch adAffectAll
TextP1.Text = "sta " & Right(Left(TextA.Text, 6), 2) & _
" + " & Format(dab, "000.000")
TextE.Text = "sta " & Right(Left(TextA.Text, 6), 2) & _
" + " & Format(dab + dbe, "000.000")
TextTS.Text = "sta " & Right(Left(TextA.Text, 6), 2) & _
" + " & Format(dab - .rsSpiral("Ts"), "000.000")
ST = dab - .rsSpiral("Ts") + .rsSpiral("Ltot")
TextST.Text = "sta " & Right(Left(TextA.Text, 6), 2) & _
" + " & Format(ST, "000.000")
End With

End Sub
Private Sub Command1_Click()
Dim i As Integer
Dim X As Double
Dim jrab, jrbe, ST As Single
Dim sdte, delta, station, sd, started As Single
Dim Sudut(3), Radsudut(3) As Single

ProsesPenghitungan

With FGHasil
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "Jarak Detail (Ls)"
.TextMatrix(0, 2) = "Sudut Lentur"
.TextMatrix(0, 3) = "Stationing"
MaxLine = Val(txl.Text)
sto1 = MaxLine
.Rows = MaxLine + 1
.ColWidth(0) = 500
.ColWidth(2) = 1600
End With
'properti grid untuk lingkaran
sdte = d - (2 * Ss)
delta = sdte / Val(Text9.Text)
FGLing.Rows = Val(Text9.Text) + 1
sd = 0
station = dab - Ts + Ls
With FGLing
.ColWidth(2) = 1500
```

```
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "Sudut"
.TextMatrix(0, 2) = "Stationing"
For i = 1 To Val(Text9.Text)
 sd = sd + delta
 station = station + DEKurva.rsSpiral("a")
 .TextMatrix(i, 0) = Str(i)
 .TextMatrix(i, 1) = SudutString(sd)
 .TextMatrix(i, 2) = "sta " & Right(Left(TextA.Text, 6), 2) & " + " & _
 Format(station, "000.0000")

Next
End With
sto2 = Val(Text9.Text)
End Sub
Private Sub Command2_Click()
Dim phi, tsud As Double
Dim Csreal, sta, sdte, started As Double
Dim strX, Cstring As String
'Cs = Val(txtCS.Text)
'MsgBox Str(konstan)
Dim ORec As ADODB.Recordset
Dim strConn As String
Dim NewRecord As Boolean
strConn = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & App.Path &
"/db/data.mdb" & _
  ";User ID=admin"
Set ORec = New ADODB.Recordset
DeleteTable "DetilSpiral", "IdSpiral=" & Str(DEKurva.rsSpiral("IdSpiral"))
DeleteTable "DetilSpiralCircle", "IdSpiral=" & Str(DEKurva.rsSpiral("IdSpiral"))
ORec.Open "select * from DetilSpiral where IdSpiral=" & _
Str(DEKurva.rsSpiral("IdSpiral")), strConn, adOpenDynamic,
adLockBatchOptimistic

If ORec.RecordCount >= 1 Then
NewRecord = False
Else
NewRecord = True
End If
Ls = DEKurva.rsSpiral("Ls")
Ss = DerajatToReal(DEKurva.rsSpiral("Ss"))
Cs = 0.0031 * (Ss ^ 3)
'txtCS.Text = Format(Csreal, "###0.0000")
Cstring = SudutString(Cs)
Cs = Val(Left(Right(Cstring, 3), 2)) / 3600
sdte = d - (2 * Ss)
started = (90 - (sdte / 2)) * PI / 180
```

```
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "Sudut"
.TextMatrix(0, 2) = "Stationing"
For i = 1 To Val(Text9.Text)
 sd = sd + delta
station = station + DEKurva.rsSpiral("a")
.TextMatrix(i, 0) = Str(i)
.TextMatrix(i, 1) = SudutString(sd)
.TextMatrix(i, 2) = "sta " & Right(Left(TextA.Text, 6), 2) & " + " & _
Format(station, "000.0000")

Next
End With
sto2 = Val(Text9.Text)
End Sub
Private Sub Command2_Click()
Dim phi, tsud As Double
Dim Csreal, sta, sdte, started As Double
Dim strX, Cstring As String
'Cs = Val(txtCS.Text)
'MsgBox Str(konstan)
Dim ORec As ADODB.Recordset
Dim strConn As String
Dim NewRecord As Boolean
strConn = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & App.Path & _
"\db\data.mdb" & _
  ";User ID=admin"
Set ORec = New ADODB.Recordset
DeleteTable "DetilSpiral", "IdSpiral=" & Str(DEKurva.rsSpiral("IdSpiral"))
DeleteTable "DetilSpiralCircle", "IdSpiral=" & Str(DEKurva.rsSpiral("IdSpiral"))
ORec.Open "select * from DetilSpiral where IdSpiral=" & _
Str(DEKurva.rsSpiral("IdSpiral")), strConn, adOpenDynamic,
adLockBatchOptimistic

If ORec.RecordCount >= 1 Then
NewRecord = False
Else
NewRecord = True
End If
Ls = DEKurva.rsSpiral("Ls")
Ss = DerajatToReal(DEKurva.rsSpiral("Ss"))
Cs = 0.0031 * (Ss ^ 3)
'txtCS.Text = Format(Csreal, "##0.0000")
Cstring = SudutString(Cs)
Cs = Val(Left(Right(Cstring, 3), 2)) / 3600
sdte = d - (2 * Ss)
started = (90 - (sdte / 2)) * PI / 180
```

```
With FGHasil
.ColWidth(0) = 500
.ColWidth(3) = 1500
For i = 1 To MaxLine
strX = InputBox("Titik Staking Out ke " & Str(i), "Masukkan")
X = Val(strX)
arrJarak(i) = X
If X <= Ls Then
tsud = 0.333 * ((X / Ls) ^ 2) * (Ss - Cs)
Else
  MsgBox "Anda hanya boleh memasukkan Maksimal Angka " & Str(Ls)
  strX = InputBox("Titik Staking Out ke " & Str(i), "Masukkan")
  X = Val(strX)
  arrJarak(i) = X
 End If
'phi = Atn(tsud) * 180 / PI
sta = dab - DEKurva.rsSpiral("Ts") + X
arrVi(i) = tsud
 .TextMatrix(i, 0) = Str(i)
 .TextMatrix(i, 1) = Str(X)
 .TextMatrix(i, 2) = SudutString(tsud)
 .TextMatrix(i, 3) = "sta " & Right(Left(TextA.Text, 6), 2) & _
 " + " & Format(sta, "000.0000")
ORec.AddNew
ORec("Id") = i
ORec("IdSpiral") = DEKurva.rsSpiral("IdSpiral")
ORec("Jarak") = X
ORec("SudutLentur") = .TextMatrix(i, 2)
ORec("Stationing") = .TextMatrix(i, 3)
ORec.UpdateBatch adAffectAll

Next i
.FixedRows = 1
End With
With FGHasil2
 .ColWidth(0) = 500
 .ColWidth(2) = 1700
.ColWidth(3) = 1700
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "Jarak Detail (Ls)"
.TextMatrix(0, 2) = "Sudut Lentur"
.TextMatrix(0, 3) = "Stationing"
Dim j As Integer
j = i - 1
For i = 1 To FGHasil.Rows - 1
j = j + 1
tsud = -arrVi(i)
```

```
With FGHasil
.ColWidth(0) = 500
.ColWidth(3) = 1500
For i = 1 To MaxLine
strX = InputBox("Titik Staking Out ke " & Str(i), "Masukkan")
X = Val(strX)
arrJarak(i) = X
If X <= Ls Then
tsud = 0.333 * ((X / Ls) ^ 2) * (Ss - Cs)
Else
MsgBox "Anda hanya boleh memasukkan Maksimal Angka " & Str(Ls)
strX = InputBox("Titik Staking Out ke " & Str(i), "Masukkan")
X = Val(strX)
arrJarak(i) = X
End If
phi = Atn(tsud) * 180 / PI
sta = dab - DEKurvaTsSpiral("Ts") + X
arrVi(i) = tsud
.TextMatrix(i, 0) = Str(i)
.TextMatrix(i, 1) = Str(X)
.TextMatrix(i, 2) = SudutString(tsud)
.TextMatrix(i, 3) = "sta " & Right(Left(TextA.Text, 6), 2) & _
" + " & Format(sta, "000.0000")
ORec.AddNew
ORec("Id") = i
ORec("IdSpiral") = DEKurvaTsSpiral("IdSpiral")
ORec("Jarak") = X
ORec("SudutLentur") = .TextMatrix(i, 2)
ORec("Stationing") = .TextMatrix(i, 3)
ORec.UpdateBatch adAffectAll

Next i
.FixedRows = 1
End With
With FGHasil2
.ColWidth(0) = 500
.ColWidth(2) = 1700
.ColWidth(3) = 1700
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "Jarak Detail (Ls)"
.TextMatrix(0, 2) = "Sudut Lentur"
.TextMatrix(0, 3) = "Stationing"
Dim j As Integer
j = i - 1
For i = 1 To FGHasil.Rows - 1
i = j + 1
tsud = -arrVi(i)
```

```
 X = arrJarak(i)
 sta = dab - DEKurva.rsSpiral("Ts") + DEKurva.rsSpiral("Ltot") - X
  .AddItem Str(i) & vbTab & Format(X, "000.000") & vbTab & SudutString(tsud)
& _
  vbTab & "sta " & Right(Left(TextA.Text, 6), 2) & _
  " + " & Format(sta, "000.0000")
 ORec.AddNew
 ORec("Id") = j
 ORec("IdSpiral") = DEKurva.rsSpiral("IdSpiral")
 ORec("Jarak") = X
 ORec("SudutLentur") = .TextMatrix(i, 2)
 ORec("Stationing") = .TextMatrix(i, 3)
 ORec.UpdateBatch adAffectAll
 Next
  .FixedRows = 1

 End With
End Sub

Private Sub Command3_Click()
MainFrm.Toolbar1.Buttons(5).Value = tbrPressed
Unload Me
End Sub

Private Sub GambarKurva()
Dim theta As Single
Dim started, ended As Single
Dim jari, sd, ak, it, sdte, pj, der As Single
Dim X, Y, delta, oldx, oldy, jari2 As Single
Dim XStart, YStart, XEnd, YEnd As Single
Dim i, k As Integer
Open App.Path & "\spiral.txt" For Output As #1

 Ss = DerajatToReal(DEKurva.rsSpiral("Ss"))
d = DerajatToReal(DEKurva.rsSpiral("d"))
sdte = d - (2 * Ss)
started = (90 - (sdte / 2)) * PI / 180
ended = started + (sdte * PI / 180)
theta = 360 - (d / 2)
dab = DEKurva.rsSpiral("JarakAP1")
dbc = DEKurva.rsSpiral("JarakP1B")
Ts = DEKurva.rsSpiral("Ts")
rc = DEKurva.rsSpiral("R")
Es = DEKurva.rsSpiral("Es")
With MainFrm
ArcPoint(2).X = .Pic.Width \ 2
ArcPoint(2).Y = 720
```

```
X = arrJarak(i)
sta = dab - DEKurva.rsSpiral("Ts") - DEKurva.rsSpiral("Ltot") - X
.AddItem Str(i) & vbTab & Format(X, "000.000") & vbTab & SudutString(tsud)
&
vbTab & "sta " & Right(Left(TextA.Text, 0), 2) & _
" + " & Format(sta, "000.0000")
ORec.AddNew
ORec("Id") = i
ORec("IdSpiral") = DEKurva.rsSpiral("IdSpiral")
ORec("Jarak") = X
ORec("SudutLentur") = .TextMatrix(i, 2)
ORec("Stationing") = .TextMatrix(i, 3)
ORec.UpdateBatch adAffectAll
Next
.FixedRows = 1

End With
End Sub

Private Sub Command3_Click()
MainFrm.Toolbar1.Buttons(5).Value = tbrPressed
Unload Me
End Sub

Private Sub GambarKurva()
Dim theta As Single
Dim started, ended As Single
Dim jari, sd, ak, tt, sdte, pj, der As Single
Dim X, Y, delta, oldx, oldy, jari2 As Single
Dim XStart, YStart, XEnd, YEnd As Single
Dim i, k As Integer
Open App.Path & "\spiral.txt" For Output As #1

Ss = DerajatToReal(DEKurva.rsSpiral("Ss"))
d = DerajatToReal(DEKurva.rsSpiral("d"))
sdte = d - (2 * Ss)
started = (90 - (sdte / 2)) * PI / 180
ended = started - (sdte * PI / 180)
theta = 360 - (d / 2)
dab = DEKurva.rsSpiral("JarakAPI")
dbc = DEKurva.rsSpiral("JarakPIB")
Ts = DEKurva.rsSpiral("Ts")
rc = DEKurva.rsSpiral("R")
Es = DEKurva.rsSpiral("Es")
With MainFrm
ArcPoint(2).X = Pic.Width / 2
ArcPoint(2).Y = 720
```

```
ArcPoint(3).X = ArcPoint(2).X + Ts * Cos(theta * PI / 180) * 1440 / GScale
ArcPoint(3).Y = ArcPoint(2).Y - Ts * Sin(theta * PI / 180) * 1440 / GScale
ArcPoint(1).X = ArcPoint(2).X - Ts * Cos(theta * PI / 180) * 1440 / GScale
ArcPoint(1).Y = ArcPoint(2).Y - Ts * Sin(theta * PI / 180) * 1440 / GScale
ArcPoint(4).X = ArcPoint(2).X
ArcPoint(4).Y = ArcPoint(2).Y + ((Es + rc) * 1440 / GScale)
ArrLine(0).X = ArcPoint(1).X - (dab - Ts) * Cos(theta * PI / 180) * 1440 / GScale
ArrLine(0).Y = ArcPoint(1).Y - (dab - Ts) * Sin(theta * PI / 180) * 1440 / GScale
ArrLine(1).X = ArcPoint(3).X + (dbc - Ts) * Cos(theta * PI / 180) * 1440 /
GScale
ArrLine(1).Y = ArcPoint(3).Y - (dbc - Ts) * Sin(theta * PI / 180) * 1440 / GScale
XStart = ArcPoint(4).X + rc * Cos(started) * 1440 / GScale
YStart = ArcPoint(4).Y - rc * Sin(started) * 1440 / GScale
XEnd = ArcPoint(4).X + rc * Cos(ended) * 1440 / GScale
YEnd = ArcPoint(4).Y - rc * Sin(ended) * 1440 / GScale

hotSpot ArrLine(0).X, ArrLine(0).Y, "A", True
hotSpot ArrLine(1).X, ArrLine(1).Y, "B", True
hotSpot ArcPoint(1).X, ArcPoint(1).Y, "TS", True
hotSpot ArcPoint(2).X, ArcPoint(2).Y - 144, "PI", False
Bulatan ArcPoint(2).X, ArcPoint(2).Y, QBColor(12)
hotSpot ArcPoint(3).X, ArcPoint(3).Y, "ST", True
hotSpot ArcPoint(4).X, ArcPoint(4).Y, "O", True
.Pic.Line (ArcPoint(1).X, ArcPoint(1).Y)-(ArcPoint(2).X, ArcPoint(2).Y),
QBColor(7)
Write #1, ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(2).X, ArcPoint(2).Y, 2
.Pic.Line (ArcPoint(2).X, ArcPoint(2).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(7)
Write #1, ArcPoint(2).X, ArcPoint(2).Y, ArcPoint(3).X, ArcPoint(3).Y, 2
.Pic.Line (ArcPoint(2).X, ArcPoint(2).Y)-(ArcPoint(4).X, ArcPoint(4).Y),
QBColor(7)
Write #1, ArcPoint(2).X, ArcPoint(2).Y, ArcPoint(4).X, ArcPoint(4).Y, 2
.Pic.Line (ArcPoint(1).X, ArcPoint(1).Y)-(ArrLine(0).X, ArrLine(0).Y),
QBColor(7)
Write #1, ArcPoint(1).X, ArcPoint(1).Y, ArrLine(0).X, ArrLine(0).Y, 2
.Pic.Line (ArcPoint(3).X, ArcPoint(3).Y)-(ArrLine(1).X, ArrLine(1).Y),
QBColor(7)
Write #1, ArcPoint(3).X, ArcPoint(3).Y, ArrLine(1).X, ArrLine(1).Y, 2
.Pic.Line (ArcPoint(4).X, ArcPoint(4).Y)-(XStart, YStart), QBColor(7)
Write #1, ArcPoint(4).X, ArcPoint(4).Y, XStart, YStart, 2
.Pic.Line (ArcPoint(4).X, ArcPoint(4).Y)-(XEnd, YEnd), QBColor(7)
Write #1, ArcPoint(4).X, ArcPoint(4).Y, XEnd, YEnd, 2
.Pic.Line (ArcPoint(4).X, ArcPoint(4).Y)-(ArcPoint(1).X, ArcPoint(1).Y),
QBColor(7)
Write #1, ArcPoint(4).X, ArcPoint(4).Y, ArcPoint(1).X, ArcPoint(1).Y, 2
.Pic.Line (ArcPoint(4).X, ArcPoint(4).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(7)
```

```
ArcPoint(3).X = ArcPoint(2).X + Ts * Cos(theta * PI / 180) * 1440 / GScale
ArcPoint(3).Y = ArcPoint(2).Y - Ts * Sin(theta * PI / 180) * 1440 / GScale
ArcPoint(1).X = ArcPoint(2).X - Ts * Cos(theta * PI / 180) * 1440 / GScale
ArcPoint(1).Y = ArcPoint(2).Y - Ts * Sin(theta * PI / 180) * 1440 / GScale
ArcPoint(4).X = ArcPoint(2).X
ArcPoint(4).Y = ArcPoint(2).Y + ((Es + rc) * 1440 / GScale)
ArrLine(0).X = ArcPoint(1).X - (dab - Ts) * Cos(theta * PI / 180) * 1440 / GScale
ArrLine(0).Y = ArcPoint(1).Y - (dab - Ts) * Sin(theta * PI / 180) * 1440 / GScale
ArrLine(1).X = ArcPoint(3).X + (dbc - Ts) * Cos(theta * PI / 180) * 1440 /
GScale
ArrLine(1).Y = ArcPoint(3).Y - (dbc - Ts) * Sin(theta * PI / 180) * 1440 / GScale
XStart = ArcPoint(4).X + rc * Cos(started) * 1440 / GScale
YStart = ArcPoint(4).Y - rc * Sin(started) * 1440 / GScale
XEnd = ArcPoint(4).X - rc * Cos(ended) * 1440 / GScale
YEnd = ArcPoint(4).Y - rc * Sin(ended) * 1440 / GScale

hotSpot ArrLine(0).X, ArrLine(0).Y, "A", True
hotSpot ArrLine(1).X, ArrLine(1).Y, "B", True
hotSpot ArcPoint(1).X, ArcPoint(1).Y, "TS", True
hotSpot ArcPoint(2).X, ArcPoint(2).Y - 144, "PI", False
Bulatan ArcPoint(2).X, ArcPoint(2).Y, QBColor(12)
hotSpot ArcPoint(3).X, ArcPoint(3).Y, "ST", True
hotSpot ArcPoint(4).X, ArcPoint(4).Y, "O", True
.PicLine (ArcPoint(1).X, ArcPoint(1).Y)-(ArcPoint(2).X, ArcPoint(2).Y),
QBColor(7)
Write #1, ArcPoint(1).X, ArcPoint(1).Y, ArcPoint(2).X, ArcPoint(2).Y, 2
.PicLine (ArcPoint(2).X, ArcPoint(2).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(7)
Write #1, ArcPoint(2).X, ArcPoint(2).Y, ArcPoint(3).X, ArcPoint(3).Y, 2
.PicLine (ArcPoint(2).X, ArcPoint(2).Y)-(ArcPoint(4).X, ArcPoint(4).Y),
QBColor(7)
Write #1, ArcPoint(2).X, ArcPoint(2).Y, ArcPoint(4).X, ArcPoint(4).Y, 2
.PicLine (ArcPoint(1).X, ArcPoint(1).Y)-(ArrLine(0).X, ArrLine(0).Y),
QBColor(7)
Write #1, ArcPoint(1).X, ArcPoint(1).Y, ArrLine(0).X, ArrLine(0).Y, 2
.PicLine (ArcPoint(3).X, ArcPoint(3).Y)-(ArrLine(1).X, ArrLine(1).Y),
QBColor(7)
Write #1, ArcPoint(3).X, ArcPoint(3).Y, ArrLine(1).X, ArrLine(1).Y, 2
.PicLine (ArcPoint(4).X, ArcPoint(4).Y)-(XStart, YStart), QBColor(7)
Write #1, ArcPoint(4).X, ArcPoint(4).Y, XStart, YStart, 2
.PicLine (ArcPoint(4).X, ArcPoint(4).Y)-(XEnd, YEnd), QBColor(7)
Write #1, ArcPoint(4).X, ArcPoint(4).Y, XEnd, YEnd, 2
.PicLine (ArcPoint(4).X, ArcPoint(4).Y)-(ArcPoint(1).X, ArcPoint(1).Y),
QBColor(7)
Write #1, ArcPoint(4).X, ArcPoint(4).Y, ArcPoint(1).X, ArcPoint(1).Y, 2
.PicLine (ArcPoint(4).X, ArcPoint(4).Y)-(ArcPoint(3).X, ArcPoint(3).Y),
QBColor(7)
```

```
Write #1, ArcPoint(4).X, ArcPoint(4).Y, ArcPoint(3).X, ArcPoint(3).Y, 2
.Pic.DrawStyle = vbSolid
oldx = ArcPoint(1).X
oldy = ArcPoint(1).Y
sd = d / 2
k = 0
For i = 1 To FGHasil.Rows - 1
 'der = Val(DerajatToReal(FGHasil.TextMatrix(i, 2)))

 sd = (d / 2) - arrVi(i)
 pj = (Val(FGHasil.TextMatrix(i, 1))) * 1440 / GScale
 X = ArcPoint(1).X + pj * Cos(sd * PI / 180)
 Y = ArcPoint(1).Y - pj * Sin(sd * PI / 180)
 .Pic.Line (oldx, oldy)-(X, Y), QBColor(12)
 Write #1, oldx, oldy, X, Y, 1
 Bulatan X, Y, QBColor(10)
 oldx = X
 oldy = Y
Next
jari2 = rc * 1440 / GScale
.Pic.Line (oldx, oldy)-(XEnd, YEnd), QBColor(12)
Write #1, oldx, oldy, XEnd, YEnd, 1
.Pic.Circle (ArcPoint(4).X, ArcPoint(4).Y), (jari2), QBColor(12), started, ended
.Pic.DrawStyle = vbDot
 oldx = XStart
 oldy = YStart
 Write #1, oldx, oldy, XStart, YStart, 1
 For i = 1 To FGLing.Rows - 1
 der = Val(DerajatToReal(FGLing.TextMatrix(i, 1)))
 sd = started + (der * PI / 180)
 X = ArcPoint(4).X + jari2 * Cos(sd)
 Y = ArcPoint(4).Y - jari2 * Sin(sd)
 Write #1, oldx, oldy, X, Y, 1
 Bulatan X, Y, QBColor(10)
 .Pic.Line (ArcPoint(4).X, ArcPoint(4).Y)-(X, Y), QBColor(7)
 Write #1, ArcPoint(4).X, ArcPoint(4).Y, X, Y, 2
 oldx = X
 oldy = Y
 Next
.Pic.DrawStyle = vbSolid
.Pic.Line (oldx, oldy)-(XEnd, YEnd), QBColor(12)
Write #1, oldx, oldy, XEnd, YEnd, 1
hotSpot XEnd, YEnd, "SC", True
hotSpot XStart, YStart, "CS", True
oldx = ArcPoint(3).X
oldy = ArcPoint(3).Y
sd = d / 2
```

```
Write #1, ArcPoint(4).X, ArcPoint(4).Y, ArcPoint(3).X, ArcPoint(3).Y, 2
.Pic.DrawStyle = vbSolid
oldx = ArcPoint(1).X
oldy = ArcPoint(1).Y
sd = d / 2
k = 0
For i = 1 To FGHasil.Rows - 1
 'der = Val(DerajatToReal(FGHasil.TextMatrix(i, 2)))

sd = (d / 2) - arrVi(i)
pj = (Val(FGHasil.TextMatrix(i, 1)) * 1440 / GScale
X = ArcPoint(1).X + pj * Cos(sd * PI / 180)
Y = ArcPoint(1).Y - pj * Sin(sd * PI / 180)
.Pic.Line (oldx, oldy)-(X, Y), QBColor(12)
Write #1, oldx, oldy, X, Y, 1
Bulatan X, Y, QBColor(10)
oldx = X
oldy = Y
Next
jari2 = rc * 1440 / GScale
.Pic.Line (oldx, oldy)-(XEnd, YEnd), QBColor(12)
Write #1, oldx, oldy, XEnd, YEnd, 1
.Pic.Circle (ArcPoint(4).X, ArcPoint(4).Y), (jari2), QBColor(12), started, ended
.Pic.DrawStyle = vbDot
oldx = XStart
oldy = YStart
Write #1, oldx, oldy, XStart, YStart, 1
For i = 1 To FGLing.Rows - 1
der = Val(DerajatToReal(FGLing.TextMatrix(i, 1)))
sd = started + (der * PI / 180)
X = ArcPoint(4).X + jari2 * Cos(sd)
Y = ArcPoint(4).Y - jari2 * Sin(sd)
Write #1, oldx, oldy, X, Y, 1
Bulatan X, Y, QBColor(10)
.Pic.Line (ArcPoint(4).X, ArcPoint(4).Y)-(X, Y), QBColor(7)
Write #1, ArcPoint(4).X, ArcPoint(4).Y, X, Y, 2
oldx = X
oldy = Y
Next
.Pic.DrawStyle = vbSolid
.Pic.Line (oldx, oldy)-(XEnd, YEnd), QBColor(12)
Write #1, oldx, oldy, XEnd, YEnd, 1
hotSpot XEnd, YEnd, "SC", True
hotSpot XStart, YStart, "CS", True
oldx = ArcPoint(3).X
oldy = ArcPoint(3).Y
sd = d / 2
```

```
For i = 1 To FGHasil.Rows - 1
 'der = Val(DerajatToReal(FGHasil.TextMatrix(i, 2)))
 sd = (d / 2) - arrVi(i)
 pj = (Val(FGHasil.TextMatrix(i, 1))) * 1440 / GScale
 X = ArcPoint(3).X - pj * Cos(sd * PI / 180)
 Y = ArcPoint(3).Y - pj * Sin(sd * PI / 180)
Write #1, oldx, oldy, X, Y, 1
 .Pic.Line (oldx, oldy)-(X, Y), QBColor(12)
 Bulatan X, Y, QBColor(10)
 oldx = X
 oldy = Y
Next
.Pic.Line (oldx, oldy)-(XStart, YStart), QBColor(12)
Write #1, oldx, oldy, XStart, YStart, 1
End With
Close #1
End Sub

Private Sub cmdc_Click()
Unload Me
End Sub

Private Sub cmdNav_Click(Index As Integer)
With DEKurva.rsSpiral
Select Case Index
Case 0:
If Not .BOF Then .MoveFirst
Case 1:
If Not .BOF Then .MovePrevious
Case 2:
If Not .EOF Then .MoveNext
Case 3:
If Not .EOF Then .MoveLast
End Select
End With
End Sub

Private Sub cmdp_Click()
Command1_Click
End Sub


Private Sub FGHasil_MouseUp(Button As Integer, Shift As Integer, X As Single,
Y As Single)
If Button = 2 Then
 Me.PopupMenu mnuExport, 0, X, Y, mnuSaveAccess
End If
```

```
For i = 1 To FGHasil.Rows - 1
'dor = Val(DerajatToReal(FGHasil.TextMatrix(i, 2)))
sd = (d / 2) - arrV(i)
pj = (Val(FGHasil.TextMatrix(i, 1)) * 1440 / (tScale
X = ArcPoint(3).X - pj * Cos(sd * PI / 180)
Y = ArcPoint(3).Y - pj * Sin(sd * PI / 180)
Write #1, oldx, oldy, X, Y, i
.PicLine (oldx, oldy)-(X, Y), QBColor(12)
Bulatan X, Y, QBColor(10)
oldx = X
oldy = Y
Next
.PicLine (oldx, oldy)-(XStart, YStart), QBColor(12)
Write #1, oldx, oldy, XStart, YStart, i
End With
Close #1
End Sub

Private Sub cmdc_Click()
Unload Me
End Sub

Private Sub cmdNav_Click(Index As Integer)
With DEKurva.rsSpiral
Select Case Index
Case 0:
If Not .BOF Then .MoveFirst
Case 1:
If Not .BOF Then .MovePrevious
Case 2:
If Not .BOF Then .MoveNext
Case 3:
If Not .EOF Then .MoveLast
End Select
End With
End Sub

Private Sub cmdb_Click()
Command1_Click
End Sub


Private Sub FGHasil_MouseUp(Button As Integer, Shift As Integer, X As Single,
Y As Single)
If Button = 2 Then
Me.PopupMenu mnuExport, 0, X, Y, mnuSaveAccess
End If
```

```
End Sub

Private Sub Form_Load()
Dim vScale As String
Open App.Path & "\option.ini" For Input As #1
Input #1, vScale
GScale = Val(vScale)
Close #1
KeyPreview = True
'DEKurva.rsSpiral.Open
End Sub

Private Sub Form_Unload(Cancel As Integer)
DEKurva.rsSpiral.Close
End Sub

Private Sub KX_LostFocus(Index As Integer)
Dim dummy As String
If Index = 2 Then
 If Val(KX(2).Text) < Val(KX(0).Text) Then
   dummy = KX(0).Text
   KX(0).Text = KX(2).Text
   KX(2).Text = dummy
 End If
End If
End Sub

Private Sub mnuClose_Click()
Unload Me
End Sub

Private Sub mnuGambar_Click()
If Val(JAB.Text) < Val(Txt9.Text) Or Val(JBC.Text) < Val(Txt9.Text) _
 Then
MsgBox "Panjang TS tidak boleh melebihi jarak AP1 atau P1B", vbCritical +
vbOKOnly, "Error"
Else
If FGHasil.TextMatrix(1, 2) <> "" Then
GambarKurva
Else
MsgBox "Anda harus memasukkan data jarak staking out terlebih dahulu"

Private Sub mnuJarak_Click()
Command2_Click

End Sub
```

```
Private Sub mnuNew_Click()
DEKurva.rsSpiral.AddNew
End Sub

Private Sub mnuOpen_Click()
DEKurva.rsSpiral.MoveFirst

End Sub

Private Sub mnuProsess_Click()
Command1_Click
End Sub

Private Sub mnuSave_Click()
On Error Resume Next
With DEKurva.rsSpiral
.UpdateBatch adAffectAll
End With
End Sub

Private Sub SudutAB_KeyDown(KeyCode As Integer, Shift As Integer)
If KeyCode = vbKeyF3 Then SudutAB.Text = Left(SudutAB.Text, 3) & "° " &
Mid(SudutAB.Text, 6, 7)

End Sub

Private Sub SudutBC_KeyDown(KeyCode As Integer, Shift As Integer)
If KeyCode = vbKeyF3 Then SudutBC.Text = Left(SudutBC.Text, 3) & "° " &
Mid(SudutBC.Text, 6, 7)
End Sub

Private Sub Txt0_KeyDown(KeyCode As Integer, Shift As Integer)
If KeyCode = vbKeyF3 Then Txt0.Text = Left(Txt0.Text, 3) & "° " &
Mid(Txt0.Text, 6, 7)
End Sub
```

```
Private Sub mnuNew_Click()
DEKurva.rsSpiral.AddNew
End Sub

Private Sub mnuOpen_Click()
DEKurva.rsSpiral.MoveFirst

End Sub

Private Sub mnuProsess_Click()
Command1_Click
End Sub

Private Sub mnuSave_Click()
On Error Resume Next
With DEKurva.rsSpiral
.UpdateBatch adAffectAll
End With
End Sub

Private Sub SudutAB_KeyDown(KeyCode As Integer, Shift As Integer)
If KeyCode = vbKeyF3 Then SudutAB.Text = Left(SudutAB.Text, 3) & "º " &
Mid(SudutAB.Text, 6, 7)

End Sub

Private Sub SudutBC_KeyDown(KeyCode As Integer, Shift As Integer)
If KeyCode = vbKeyF3 Then SudutBC.Text = Left(SudutBC.Text, 3) & "º " &
Mid(SudutBC.Text, 6, 7)
End Sub

Private Sub Txt0_KeyDown(KeyCode As Integer, Shift As Integer)
If KeyCode = vbKeyF3 Then Txt0.Text = Left(Txt0.Text, 3) & "º " &
Mid(Txt0.Text, 6, 7)
End Sub
```

DIAGRAM SUPERELEVASI

```
Option Explicit
Dim pjg, Bsr, Lbr As Double
Dim Koord(100) As Point
Dim Pot As Point
Dim i As Integer
Dim piclengt As Single
Dim emax, ens, sc As Single
Dim m1, m2, deltax As Single
Dim at7, Y As Single
Dim oldx, oldy As Single
Dim starsort(1) As Boolean
Private Function GetCoord(ByVal grad1 As Single, ByVal grad2 As Single, _
ByVal x1 As Single, ByVal x2 As Single) As Point
On Error Resume Next
If grad1 <> grad2 Then
GetCoord.X = (grad2 * (x2 - x1)) / (grad1 - grad2)
Else
GetCoord.X = x1
End If
GetCoord.Y = grad1 * GetCoord.X
End Function
Private Sub hotArrow(ByVal X As Single, ByVal Y As Single, ByVal Direction
As Integer)
With MainFrm
.Pic.DrawWidth = 1
.Pic.DrawStyle = vbSolid
Select Case Direction
Case 0:
.Pic.Line (X - 100, Y - 75)-(X, Y), QBColor(0)
.Pic.Line (X - 100, Y + 75)-(X, Y), QBColor(0)
Case 1:
.Pic.Line (X + 100, Y - 75)-(X, Y), QBColor(0)
.Pic.Line (X + 100, Y + 75)-(X, Y), QBColor(0)

End Select
.Pic.DrawStyle = vbSolid
.Pic.DrawWidth = 1
End With
End Sub
Private Sub GambarKurvaLingkaran()
Dim sc As Single
Dim YEn As Single
Open App.Path & "\selingkaran.txt" For Output As #1
With MainFrm
sc = 30
```

DIAGRAM SUPERELEVASI

```
Option Explicit
Dim pjg, lbr, Lbr As Double
Dim Koord(100) As Point
Dim Pot As Point
Dim i As Integer
Dim pielengt As Single
Dim emax, ens, sc As Single
Dim m1, m2, deltax As Single
Dim at7, Y As Single
Dim oldx, oldy As Single
Dim starsort(1) As Boolean
Private Function GetCoord(ByVal grad1 As Single, ByVal grad2 As Single, _
ByVal x1 As Single, ByVal x2 As Single) As Point
On Error Resume Next
If grad1 <> grad2 Then
GetCoord.X = (grad2 * (x2 - x1)) / (grad1 - grad2)
Else
GetCoord.X = x1
End If
GetCoord.Y = grad1 * GetCoord.X
End Function
Private Sub hotArrow(ByVal X As Single, ByVal Y As Single, ByVal Direction As Integer)
With MainFrm
.PicDrawWidth = 1
.PicDrawStyle = vbSolid
Select Case Direction
Case 0:
.PicLine (X - 100, Y - 75)-(X, Y), QBColor(0)
.PicLine (X - 100, Y + 75)-(X, Y), QBColor(0)
Case 1:
.PicLine (X + 100, Y - 75)-(X, Y), QBColor(0)
.PicLine (X + 100, Y + 75)-(X, Y), QBColor(0)

End Select
.PicDrawStyle = vbSolid
.PicDrawWidth = 1
End With
End Sub
Private Sub GambarKurvaLingkaran()
Dim sc As Single
Dim YEn As Single
Open App.Path & "\celingkaran.txt" For Output As #1
With MainFrm
sc = 30
```

```
.Pic.DrawStyle = vbSolid
.Pic.Line (Koord(0).X, Koord(0).Y)-(Koord(5).X, Koord(5).Y), QBColor(12)
Write #1, Koord(0).X, Koord(0).Y, Koord(5).X, Koord(5).Y, 1
For i = 0 To 8
.Pic.Line (Koord(i).X, Koord(i).Y)-(Koord(i + 1).X, Koord(i + 1).Y),
QBColor(12)
Write #1, Koord(i).X, Koord(i).Y, Koord(i + 1).X, Koord(i + 1).Y, 1
Next
.Pic.DrawStyle = vbDot
YEn = DEKurva.rsSELingkaran("en") * 1440 * sc / GScale
.Pic.Line (Koord(0).X, Koord(0).Y - YEn)-(Koord(5).X, Koord(5).Y - YEn),
QBColor(1)
Write #1, Koord(0).X, Koord(0).Y - YEn, Koord(5).X, Koord(5).Y - YEn, 2
For i = 0 To 9
Select Case i
Case 1, 4:

Case Else:
.Pic.Line (Koord(i).X, Koord(1).Y - 1500)-(Koord(i).X, Koord(5).Y + 150),
QBColor(12)
End Select
Next
.Pic.Line (Koord(0).X, Koord(3).Y - 560)-(Koord(9).X, Koord(3).Y - 560),
QBColor(1)
.Pic.Line (Koord(9).X, Koord(3).Y - 560)-(Koord(2).X, Koord(3).Y - 560),
QBColor(1)
.Pic.Line (Koord(3).X, Koord(3).Y - 560)-(Koord(4).X, Koord(3).Y - 560),
QBColor(1)
.Pic.Line (Koord(4).X, Koord(3).Y - 560)-(Koord(5).X, Koord(3).Y - 560),
QBColor(1)
.Pic.Line (Koord(1).X, Koord(1).Y)-(Koord(9).X, Koord(9).Y), QBColor(1)
Write #1, Koord(1).X, Koord(1).Y, Koord(9).X, Koord(9).Y, 2
.Pic.Line (Koord(4).X, Koord(4).Y)-(Koord(6).X, Koord(6).Y), QBColor(1)
Write #1, Koord(4).X, Koord(4).Y, Koord(6).X, Koord(6).Y, 2
.Pic.Line (Koord(9).X, Koord(3).Y - 1000)-(Koord(6).X, Koord(3).Y - 1000),
QBColor(1)
hotArrow Koord(0).X, Koord(3).Y - 560, 1
hotArrow Koord(9).X, Koord(3).Y - 560, 0
hotArrow Koord(9).X, Koord(3).Y - 560, 1
hotArrow Koord(2).X, Koord(3).Y - 560, 0
hotArrow Koord(9).X, Koord(3).Y - 1000, 1
hotArrow Koord(6).X, Koord(3).Y - 1000, 0
hotArrow Koord(3).X, Koord(3).Y - 560, 1
hotArrow Koord(6).X, Koord(3).Y - 560, 0
hotArrow Koord(6).X, Koord(3).Y - 560, 1
hotArrow Koord(5).X, Koord(3).Y - 560, 0
```

```
Pic.DrawStyle = vbSolid
Pic.Line (Koord(0).X, Koord(0).Y)-(Koord(5).X, Koord(5).Y), QBColor(12)
Write #1, Koord(0).X, Koord(0).Y, Koord(5).X, Koord(5).Y, 1
For i = 0 To 8
Pic.Line (Koord(i).X, Koord(i).Y)-(Koord(i + 1).X, Koord(i + 1).Y),
QBColor(12)
Write #1, Koord(i).X, Koord(i).Y, Koord(i + 1).X, Koord(i + 1).Y, 1
Next
Pic.DrawStyle = vbDot
YEn = DEKurva.rsSELingkaran("en") * 1440 * sc / GScale
Pic.Line (Koord(0).X, Koord(0).Y - YEn)-(Koord(5).X, Koord(5).Y - YEn),
QBColor(1)
Write #1, Koord(0).X, Koord(0).Y - YEn, Koord(5).X, Koord(5).Y - YEn, 2
For i = 0 To 9
Select Case i
Case 1, 4:

Case Else:
Pic.Line (Koord(i).X, Koord(1).Y - 1500)-(Koord(i).X, Koord(5).Y + 150),
QBColor(12)
End Select
Next
Pic.Line (Koord(0).X, Koord(3).Y - 560)-(Koord(9).X, Koord(3).Y - 560),
QBColor(1)
Pic.Line (Koord(9).X, Koord(3).Y - 560)-(Koord(2).X, Koord(3).Y - 560),
QBColor(1)
Pic.Line (Koord(3).X, Koord(3).Y - 560)-(Koord(4).X, Koord(3).Y - 560),
QBColor(1)
Pic.Line (Koord(4).X, Koord(3).Y - 560)-(Koord(5).X, Koord(3).Y - 560),
QBColor(1)
Pic.Line (Koord(1).X, Koord(1).Y)-(Koord(9).X, Koord(9).Y), QBColor(1)
Write #1, Koord(1).X, Koord(1).Y, Koord(9).X, Koord(9).Y, 2
Pic.Line (Koord(4).X, Koord(4).Y)-(Koord(6).X, Koord(6).Y), QBColor(1)
Write #1, Koord(4).X, Koord(4).Y, Koord(6).X, Koord(6).Y, 2
Pic.Line (Koord(9).X, Koord(3).Y - 1000)-(Koord(6).X, Koord(3).Y - 1000),
QBColor(1)
hotArrow Koord(0).X, Koord(3).Y - 560, 1
hotArrow Koord(9).X, Koord(3).Y - 560, 0
hotArrow Koord(9).X, Koord(3).Y - 560, 1
hotArrow Koord(2).X, Koord(3).Y - 560, 0
hotArrow Koord(9).X, Koord(3).Y - 1000, 1
hotArrow Koord(6).X, Koord(3).Y - 1000, 0
hotArrow Koord(3).X, Koord(3).Y - 560, 1
hotArrow Koord(6).X, Koord(3).Y - 560, 0
hotArrow Koord(6).X, Koord(3).Y - 560, 1
hotArrow Koord(5).X, Koord(3).Y - 560, 0
```

```
TextOut Koord(9).X - 200, Koord(2).Y - 1200, "T1", 1
TextOut Koord(6).X, Koord(2).Y - 1200, "T2", 1
TextOut Koord(9).X + (Koord(6).X - Koord(9).X) \ 2, Koord(2).Y - 1200, "LC",
1
TextOut Koord(0).X + (Koord(9).X - Koord(0).X) \ 2, Koord(2).Y - 700, "2/3
LS", 1
TextOut Koord(9).X + (Koord(2).X - Koord(9).X) \ 2 - 200, Koord(2).Y - 700,
"1/3 LS", 1
TextOut Koord(3).X + (Koord(6).X - Koord(3).X) \ 2 - 200, Koord(2).Y - 700,
"1/3 LS", 1
TextOut Koord(6).X + (Koord(5).X - Koord(6).X) \ 2, Koord(2).Y - 700, "2/3
LS", 1


End With
Close #1
End Sub
Private Sub GambarKurvaSpiral()
Dim i As Integer
Open App.Path & "\sespiral.txt" For Output As #1
With MainFrm

.Pic.DrawStyle = vbSolid
For i = 0 To 2
.Pic.Line (Koord(i).X, Koord(i).Y)-(Koord(i + 1).X, Koord(i + 1).Y),
QBColor(12)
Write #1, Koord(i).X, Koord(i).Y, Koord(i + 1).X, Koord(i + 1).Y, 1
Next i
.Pic.Line (Koord(0).X, Koord(0).Y)-(Koord(3).X, Koord(3).Y), QBColor(12)
Write #1, Koord(0).X, Koord(0).Y, Koord(3).X, Koord(3).Y, 1
.Pic.Line (Koord(4).X, Koord(4).Y)-(Koord(5).X, Koord(5).Y), QBColor(12)
Write #1, Koord(4).X, Koord(4).Y, Koord(5).X, Koord(5).Y, 1
.Pic.Line (Koord(8).X, Koord(8).Y)-(Koord(4).X, Koord(4).Y), QBColor(12)
Write #1, Koord(8).X, Koord(8).Y, Koord(4).X, Koord(4).Y, 1
.Pic.Line (Koord(9).X, Koord(9).Y)-(Koord(5).X, Koord(5).Y), QBColor(12)
Write #1, Koord(9).X, Koord(9).Y, Koord(5).X, Koord(5).Y, 1
.Pic.DrawStyle = vbDot
.Pic.Line (Koord(6).X, Koord(6).Y)-(Koord(7).X, Koord(7).Y), QBColor(0)
Write #1, Koord(6).X, Koord(6).Y, Koord(7).X, Koord(7).Y, 2
.Pic.Line (Koord(10).X, Koord(10).Y)-(Koord(11).X, Koord(11).Y),
QBColor(12)
Write #1, Koord(10).X, Koord(10).Y, Koord(11).X, Koord(11).Y, 2
.Pic.Line (Koord(12).X, Koord(12).Y)-(Koord(13).X, Koord(13).Y),
QBColor(12)
Write #1, Koord(12).X, Koord(12).Y, Koord(13).X, Koord(13).Y, 2
```

```
TextOut Koord(9).X - 200, Koord(2).Y - 1200, "T1", 1
TextOut Koord(6).X, Koord(2).Y - 1200, "T2", 1
TextOut Koord(9).X + (Koord(6).X - Koord(9).X) / 2, Koord(2).Y - 1200, "LC",
1
TextOut Koord(0).X + (Koord(9).X - Koord(0).X) / 2, Koord(2).Y - 700, "2/3
LS", 1
TextOut Koord(9).X + (Koord(2).X - Koord(9).X) / 2 - 200, Koord(2).Y - 700,
"1/3 LS", 1
TextOut Koord(3).X + (Koord(6).X - Koord(3).X) / 2 - 200, Koord(2).Y - 700,
"1/3 LS", 1
TextOut Koord(6).X + (Koord(5).X - Koord(6).X) / 2, Koord(2).Y - 700, "2/3
LS", 1


End With
Close #1
End Sub
Private Sub GambarKurvaSpiral()
Dim i As Integer
Open App.Path & "\sespiral.txt" For Output As #1
With MainFrm

.Pic.DrawStyle = vbSolid
For i = 0 To 2
.Pic.Line (Koord(i).X, Koord(i).Y)-(Koord(i + 1).X, Koord(i + 1).Y),
QBColor(12)
Write #1, Koord(i).X, Koord(i).Y, Koord(i + 1).X, Koord(i + 1).Y, 1
Next i
.Pic.Line (Koord(0).X, Koord(0).Y)-(Koord(3).X, Koord(3).Y), QBColor(12)
Write #1, Koord(0).X, Koord(0).Y, Koord(3).X, Koord(3).Y, 1
.Pic.Line (Koord(4).X, Koord(4).Y)-(Koord(5).X, Koord(5).Y), QBColor(12)
Write #1, Koord(4).X, Koord(4).Y, Koord(5).X, Koord(5).Y, 1
.Pic.Line (Koord(8).X, Koord(8).Y)-(Koord(4).X, Koord(4).Y), QBColor(12)
Write #1, Koord(8).X, Koord(8).Y, Koord(4).X, Koord(4).Y, 1
.Pic.Line (Koord(9).X, Koord(9).Y)-(Koord(5).X, Koord(5).Y), QBColor(12)
Write #1, Koord(9).X, Koord(9).Y, Koord(5).X, Koord(5).Y, 1
.Pic.DrawStyle = vbDot
.Pic.Line (Koord(6).X, Koord(6).Y)-(Koord(7).X, Koord(7).Y), QBColor(0)
Write #1, Koord(6).X, Koord(6).Y, Koord(7).X, Koord(7).Y, 2
.Pic.Line (Koord(10).X, Koord(10).Y)-(Koord(11).X, Koord(11).Y),
QBColor(12)
Write #1, Koord(10).X, Koord(10).Y, Koord(11).X, Koord(11).Y, 2
.Pic.Line (Koord(12).X, Koord(12).Y)-(Koord(13).X, Koord(13).Y),
QBColor(12)
Write #1, Koord(12).X, Koord(12).Y, Koord(13).X, Koord(13).Y, 2
```

```
.Pic.Line (Koord(0).X, Koord(0).Y - 1440)-(Koord(0).X, Koord(0).Y + 1440),
QBColor(12)
Write #1, Koord(0).X, Koord(0).Y - 1440, Koord(0).X, Koord(0).Y + 1440, 2
.Pic.Line (Koord(3).X, Koord(3).Y - 1440)-(Koord(3).X, Koord(3).Y + 1440),
QBColor(12)
Write #1, Koord(3).X, Koord(3).Y - 1440, Koord(3).X, Koord(3).Y + 1440, 1
.Pic.Line (Koord(8).X, Koord(8).Y - 140)-(Koord(8).X, Koord(8).Y + 1440),
QBColor(12)
Write #1, Koord(8).X, Koord(8).Y - 140, Koord(8).X, Koord(8).Y + 1440, 1
.Pic.Line (Koord(14).X, Koord(14).Y - 140)-(Koord(14).X, Koord(14).Y + 1440),
QBColor(12)

.Pic.Line (Koord(8).X + deltax, Koord(8).Y - 140)-(Koord(8).X + deltax,
Koord(8).Y + 1440), QBColor(12)
.Pic.Line (Koord(3).X - (2 * deltax), Koord(14).Y - 140)-(Koord(3).X - (2 *
deltax), Koord(14).Y + 1440), QBColor(12)

.Pic.Line (Koord(0).X, Koord(3).Y - 560)-(Koord(4).X, Koord(3).Y - 560),
QBColor(1)
.Pic.Line (Koord(4).X, Koord(3).Y - 1000)-(Koord(2).X, Koord(3).Y - 1000),
QBColor(1)
.Pic.Line (Koord(2).X, Koord(3).Y - 560)-(Koord(3).X, Koord(3).Y - 560),
QBColor(1)
'.Pic.Line (Koord(4).X, Koord(3).Y - 560)-(Koord(5).X, Koord(3).Y - 560),
QBColor(1)
'.Pic.Line (Koord(1).X, Koord(1).Y)-(Koord(9).X, Koord(9).Y), QBColor(1)
'.Pic.Line (Koord(4).X, Koord(4).Y)-(Koord(6).X, Koord(6).Y), QBColor(1)
'.Pic.Line (Koord(2).X, Koord(3).Y - 1000)-(Koord(3).X, Koord(3).Y - 1000),
QBColor(1)

hotArrow Koord(0).X, Koord(3).Y - 560, 1
hotArrow Koord(4).X, Koord(3).Y - 560, 0
hotArrow Koord(4).X, Koord(3).Y - 1000, 1
hotArrow Koord(2).X, Koord(3).Y - 1000, 0
hotArrow Koord(2).X, Koord(3).Y - 560, 1
hotArrow Koord(3).X, Koord(3).Y - 560, 0

TextOut Koord(0).X - 200, Koord(2).Y - 1200, "TS", 1
TextOut Koord(4).X - 240, Koord(2).Y - 1200, "SC", 1
TextOut Koord(2).X, Koord(2).Y - 1200, "CS", 1
TextOut Koord(3).X, Koord(2).Y - 1200, "ST", 1
TextOut Koord(0).X + (Koord(4).X - Koord(0).X) \ 2, Koord(2).Y - 450, "Ls", 1
TextOut Koord(2).X + (Koord(3).X - Koord(2).X) \ 2, Koord(2).Y - 450, "Ls", 1
TextOut Koord(4).X + (Koord(2).X - Koord(4).X) \ 2, Koord(2).Y - 900, "Lc", 1
End With
Close #1
End Sub
```

```
.PicLine (Koord(0).X, Koord(0).Y - 1440)-(Koord(0).X, Koord(0).Y + 1440), QBColor(12)
Write #1, Koord(0).X, Koord(0).Y - 1440, Koord(0).X, Koord(0).Y + 1440, 2
.PicLine (Koord(3).X, Koord(3).Y - 1440)-(Koord(3).X, Koord(3).Y + 1440), QBColor(12)
Write #1, Koord(3).X, Koord(3).Y - 1440, Koord(3).X, Koord(3).Y + 1440,
.PicLine (Koord(8).X, Koord(8).Y - 1440)-(Koord(8).X, Koord(8).Y + 1440), QBColor(12)
Write #1, Koord(8).X, Koord(8).Y - 1440, Koord(8).X, Koord(8).Y + 1440, 1
.PicLine (Koord(14).X, Koord(14).Y - 1440)-(Koord(14).X, Koord(14).Y + 1440), QBColor(12)

.PicLine (Koord(8).X + deltax, Koord(8).Y - 1440)-(Koord(8).X + deltax, Koord(8).Y + 1440), QBColor(12)
.PicLine (Koord(3).X - (2 * deltax), Koord(14).Y - 1440)-(Koord(3).X - (2 * deltax), Koord(14).Y + 1440), QBColor(12)

.PicLine (Koord(0).X, Koord(3).Y - 560)-(Koord(4).X, Koord(3).Y - 560), QBColor(1)
.PicLine (Koord(4).X, Koord(3).Y - 1000)-(Koord(2).X, Koord(3).Y - 1000), QBColor(1)
.PicLine (Koord(2).X, Koord(3).Y - 560)-(Koord(3).X, Koord(3).Y - 560), QBColor(1)
'.PicLine (Koord(4).X, Koord(3).Y - 560)-(Koord(5).X, Koord(3).Y - 560), QBColor(1)
'.PicLine (Koord(1).X, Koord(1).Y)-(Koord(9).X, Koord(9).Y), QBColor(1)
'.PicLine (Koord(4).X, Koord(4).Y)-(Koord(6).X, Koord(6).Y), QBColor(1)
'.PicLine (Koord(2).X, Koord(3).Y - 1000)-(Koord(3).X, Koord(3).Y - 1000), QBColor(1)

hotArrow Koord(0).X, Koord(3).Y - 560, 1
hotArrow Koord(4).X, Koord(3).Y - 560, 0
hotArrow Koord(4).X, Koord(3).Y - 1000, 1
hotArrow Koord(2).X, Koord(3).Y - 1000, 0
hotArrow Koord(2).X, Koord(3).Y - 560, 1
hotArrow Koord(3).X, Koord(3).Y - 560, 0

TextOut Koord(0).X - 200, Koord(2).Y - 1200, "TS", 1
TextOut Koord(4).X - 240, Koord(2).Y - 1200, "SC", 1
TextOut Koord(2).X, Koord(2).Y - 1200, "CS", 1
TextOut Koord(3).X, Koord(2).Y - 1200, "ST", 1
TextOut Koord(0).X + (Koord(4).X - Koord(0).X) / 2, Koord(2).Y - 450, "Ls", 1
TextOut Koord(2).X + (Koord(3).X - Koord(2).X) / 2, Koord(2).Y - 450, "Ls", 1
TextOut Koord(4).X + (Koord(2).X - Koord(4).X) / 2, Koord(2).Y - 900, "Lc", 1
End With
Close #1
End Sub
```

```
Private Sub Perhitungan_Lingkaran()
On Error Resume Next
Dim jaraks, jawal, sc, kawal As Single
Dim i, No As Integer
sc = 30
piclengt = ((4 * DEKurva.rsSELingkaran("Ls") / 3) +
DEKurva.rsSELingkaran("Lc")) * 1440 / GScale
With MainFrm
Koord(0).X = 1440
Koord(0).Y = .Pic.Height \ 2
Koord(2).X = Koord(0).X + DEKurva.rsSELingkaran("Ls") * 1440 / GScale
Koord(2).Y = Koord(0).Y - ((DEKurva.rsSELingkaran("emax") +
DEKurva.rsSELingkaran("en")) * 1440 * sc / GScale)
Koord(3).X = Koord(0).X + piclengt - DEKurva.rsSELingkaran("Ls") * 1440 /
GScale
Koord(3).Y = Koord(0).Y - ((DEKurva.rsSELingkaran("emax") +
DEKurva.rsSELingkaran("en")) * 1440 * sc / GScale)
Koord(5).X = Koord(0).X + piclengt
Koord(5).Y = Koord(0).Y
m1 = ((Koord(2).Y - Koord(0).Y) / (Koord(2).X - Koord(0).X))
deltax = Abs(m1) * (DEKurva.rsSELingkaran("en") * 1440 * sc / GScale)
Koord(6).X = Koord(5).X - (2 * DEKurva.rsSELingkaran("Ls") / 3) * 1440 /
GScale
Koord(6).Y = Koord(0).Y
Koord(7).X = Koord(3).X
Koord(7).Y = Koord(0).Y + ((DEKurva.rsSELingkaran("emax") -
DEKurva.rsSELingkaran("en")) * 1440 * sc / GScale)
Koord(8).X = Koord(2).X
Koord(8).Y = Koord(7).Y
Koord(9).X = Koord(0).X + (2 * DEKurva.rsSELingkaran("Ls") / 3) * 1440 /
GScale
Koord(9).Y = Koord(0).Y
m2 = ((Koord(9).Y - Koord(8).Y) / (Koord(9).X - Koord(8).X))
Pot = GetCoord(m1, m2, Koord(0).X, Koord(9).X)
'MsgBox " X :" & Str(Pot.X) & " Y :" & Str(Pot.Y)
Koord(1).X = Koord(0).X - Pot.X
Koord(1).Y = Koord(9).Y - Pot.Y
Koord(4).X = Koord(5).X + Pot.X
Koord(4).Y = Koord(6).Y - Pot.Y
End With

 With FG1
 .TextMatrix(0, 0) = "No"
 .TextMatrix(0, 1) = "Jarak"
 .TextMatrix(0, 2) = "Stationing"
 End With
 jawal = Val(Mid(DEKurva.rsSELingkaran("Statawal").Value, 10, 6))
```

```
Private Sub Perhitungan_Lingkaran()
On Error Resume Next
Dim jaraks, jawal, sc, Lawal As Single
Dim i, No As Integer
sc = 50
piclengt = ((4 * DEKurva.rsSElingkaran("Ls") / 3) +
DEKurva.rsSElingkaran("Lc")) * 1440 / GScale
With MainFrm
Koord(0).X = 1440
Koord(0).Y = .Pic.Height / 2
Koord(2).X = Koord(0).X + DEKurva.rsSElingkaran("Ls") * 1440 / GScale
Koord(2).Y = Koord(0).Y - ((DEKurva.rsSElingkaran("emax") +
DEKurva.rsSElingkaran("en")) * 1440 * sc / GScale)
Koord(3).X = Koord(0).X + piclengt - DEKurva.rsSElingkaran("Ls") * 1440 /
GScale
Koord(3).Y = Koord(0).Y - ((DEKurva.rsSElingkaran("emax") +
DEKurva.rsSElingkaran("en")) * 1440 * sc / GScale)
Koord(5).X = Koord(0).X + piclengt
Koord(5).Y = Koord(0).Y
m1 = ((Koord(2).Y - Koord(0).Y) / (Koord(2).X - Koord(0).X))
deltax = Abs(m1) * (DEKurva.rsSElingkaran("en") * 1440 * sc / GScale)
Koord(6).X = Koord(5).X - (2 * DEKurva.rsSElingkaran("Ls") / 3) * 1440 /
GScale
Koord(6).Y = Koord(0).Y
Koord(7).X = Koord(3).X
Koord(7).Y = Koord(0).Y - ((DEKurva.rsSElingkaran("emax") -
DEKurva.rsSElingkaran("en")) * 1440 * sc / GScale)
Koord(8).X = Koord(2).X
Koord(8).Y = Koord(7).Y
Koord(9).X = Koord(0).X + (2 * DEKurva.rsSElingkaran("Ls") / 3) * 1440 /
GScale
Koord(9).Y = Koord(0).Y
m2 = ((Koord(9).Y - Koord(8).Y) / (Koord(9).X - Koord(8).X))
Pot = GetCoord(m1, m2, Koord(0).X, Koord(9).X)
'MsgBox "X :" & Str(Pot.X) & " Y :" & Str(Pot.Y)
Koord(1).X = Koord(6).X - Pot.X
Koord(1).Y = Koord(9).Y - Pot.Y
Koord(4).X = Koord(5).X + Pot.X
Koord(4).Y = Koord(6).Y - Pot.Y
End With

With FG1
.TextMatrix(0, 0) = "No"
.TextMatrix(0, 1) = "Jarak"
.TextMatrix(0, 2) = "Stationing"
End With
jawal = Val(Mid(DEKurva.rsSElingkaran("Stawal").Value, 10, 6))
```

```
kawal = Koord(0).X
For i = 0 To 9
arrJarak(i) = (Koord(i).X - kawal) * GScale / 1440
ArrStationing(i) = "sta " & _
Right(Left(DEKurva.rsSELingkaran("Statawal").Value, 6), 2) & _
" + " & Format(arrJarak(i) + jawal, "000.0000")
FG1.TextMatrix(i + 1, 0) = Str(i)
FG1.TextMatrix(i + 1, 1) = Format(arrJarak(i), "##00.000")
FG1.TextMatrix(i + 1, 2) = ArrStationing(i)
Next


End Sub

Private Sub cmdAdd_Click()

End Sub

Private Sub cmdAdd2_Click()

End Sub

Private Sub cmdNav_Click()
On Error Resume Next
DEKurva.rsSELingkaran.Delete adAffectCurrent
DEKurva.rsSELingkaran.MoveLast
End Sub

Private Sub cmdProses_Click(Index As Integer)
Select Case Index
 Case 2:
  GambarKurvaLingkaran
 Case 3:
  GambarKurvaSpiral

End Select

End Sub

Private Sub cmdProsesLingkaran_Click()
Dim pn, pmax As Single

With DEKurva
Perhitungan_Lingkaran
pjg = .rsSELingkaran("Ls").Value
Bsr = .rsSELingkaran("Lc").Value
Lbr = .rsSELingkaran("Lebar").Value
```

```
    kawal = Koord(0).X
    For i = 0 To 9
    arrJarak(i) = (Koord(i).X - kawal) * GScale / 1440
    ArrStationing(i) = "sta " & _
    Right(Left(DEKurva.rsSELingkaran("Statawal").Value, 6), 2) & _
    " + " & Format(arrJarak(i) + jawal, "000.0000")
    FG1.TextMatrix(i + 1, 0) = Str(i)
    FG1.TextMatrix(i + 1, 1) = Format(arrJarak(i), "##00.000")
    FG1.TextMatrix(i + 1, 2) = ArrStationing(i)
    Next

End Sub

Private Sub cmdAdd_Click()

End Sub

Private Sub cmdAdd2_Click()

End Sub

Private Sub cmdNav_Click()
On Error Resume Next
DEKurva.rsSELingkaran.Delete adAffectCurrent
DEKurva.rsSELingkaran.MoveLast
End Sub

Private Sub cmdProses_Click(Index As Integer)
Select Case Index
  Case 2:
    GambarKurvaLingkaran
  Case 3:
    GambarKurvaSpiral

End Select

End Sub

Private Sub cmdProsesLingkaran_Click()
Dim pn, pmax As Single

With DEKurva
Perhitungan_Lingkaran
pjg = .rsSELingkaran("Ls").Value
Bsr = .rsSELingkaran("Lc").Value
Lbr = .rsSELingkaran("Lebar").Value
```

```
pn = .rsSELingkaran("en").Value * .rsSELingkaran("Lebar").Value
pmax = .rsSELingkaran("emax").Value * .rsSELingkaran("Lebar").Value
.rsSELingkaran("Sepertiga").Value = pjg / 3
.rsSELingkaran("Duapertiga").Value = pjg * 2 / 3
 Pnl.Text = Format(pn, "##0.000")
 PMaxl.Text = Format(pmax, "##0.000")
.rsSELingkaran.UpdateBatch adAffectAll

End With
FG1.ColWidth(1) = 1200
FG1.ColWidth(2) = 1400
End Sub

Private Sub cmdSave1_Click()

End Sub

Private Sub cmdSave2_Click()

End Sub

Private Sub Command1_Click()
Dim dummy, Max As Single
Dim i, j, k, dui, jk As Integer
Dim dumm2 As String
If Not starsort(0) Then
dummy = arrJarak(0)
dumm2 = ArrStationing(0)
Max = 9
jk = 0
For i = 1 To Max - 1
For j = i + 1 To Max
If arrJarak(i) > arrJarak(j) Then
 dummy = arrJarak(i)
 dumm2 = ArrStationing(i)
 arrJarak(i) = arrJarak(j)
 ArrStationing(i) = ArrStationing(j)
 arrJarak(j) = dummy
 ArrStationing(j) = dumm2

End If

Next j
Next i
With FG1
For i = 0 To .Rows - 2
 .TextMatrix(i + 1, 0) = Str(i)
```

```
pn = .rsSELingkaran("en").Value * .rsSELingkaran("Lebar").Value
pmax = .rsSELingkaran("emax").Value * .rsSELingkaran("Lebar").Value
.rsSELingkaran("Sepertiga").Value = pig / 3
.rsSELingkaran("Duapertiga").Value = pig * 2 / 3
PnL.Text = Format(pn, "##0.000")
PMaxL.Text = Format(pmax, "##0.000")
.rsSELingkaran.UpdateBatch adAffectAll

End With
FG1.ColWidth(1) = 1200
FG1.ColWidth(2) = 1400
End Sub

Private Sub cmdSave1_Click()

End Sub

Private Sub cmdSave2_Click()

End Sub

Private Sub Command1_Click()
Dim dummy, Max As Single
Dim i, j, k, dat, jk As Integer
Dim dumm2 As String
If Not atarsort() Then
dummy = arrJarak(0)
dumm2 = ArrStationing(0)
Max = 9
jk = 0
For i = 1 To Max - 1
For j = i + 1 To Max
If arrJarak(i) > arrJarak(j) Then
 dummy = arrJarak(i)
 dumm2 = ArrStationing(i)
 arrJarak(i) = arrJarak(j)
 ArrStationing(i) = ArrStationing(j)
 arrJarak(j) = dummy
 ArrStationing(j) = dumm2

End If

Next j
Next i
With FG1
For i = 0 To .Rows - 2
.TextMatrix(i + 1, 0) = Sn(i)
```

```
.TextMatrix(i + 1, 1) = Format(arrJarak(i), "##00.000")
.TextMatrix(i + 1, 2) = ArrStationing(i)
Next
For i = .Rows - 3 To 0 Step -1
If .TextMatrix(i + 1, 1) = .TextMatrix(i + 2, 1) Then
.TextMatrix(i + 1, 1) = .TextMatrix(i + 2, 1)
.TextMatrix(i + 1, 2) = .TextMatrix(i + 2, 2)
.RemoveItem (i + 2)
End If
Next
For i = 1 To .Rows - 1
.TextMatrix(i, 0) = Str(i)
Next
End With
starsort(0) = True
End If
End Sub

Private Sub Command2_Click()
DEKurva.rsSESpiral.Delete adAffectCurrent
End Sub

Private Sub Command3_Click()
FG1.Clear
End Sub

Private Sub Command4_Click()
Select Case Super.Tab
Case 0:
If Not DEKurva.rsSELingkaran.BOF Then
DEKurva.rsSELingkaran.MovePrevious
End If
Case 1:
If Not DEKurva.rsSESpiral.BOF Then
DEKurva.rsSESpiral.MovePrevious
End If
End Select
End Sub

Private Sub Command5_Click()
On Error Resume Next
Select Case Super.Tab
Case 0:
If Not DEKurva.rsSELingkaran.EOF Then
DEKurva.rsSELingkaran.MoveNext
End If
Case 1:
```

```
.TextMatrix(i + 1, 1) = Format(arrJarak(i), "#0.000")
.TextMatrix(i + 1, 2) = ArrStationing(i)
Next
For i = .Rows - 3 To 0 Step -1
If .TextMatrix(i + 1, 1) - .TextMatrix(i + 2, 1) Then
.TextMatrix(i + 1, 1) = .TextMatrix(i + 2, 1)
.TextMatrix(i + 1, 2) = .TextMatrix(i + 2, 2)
.RemoveItem (i + 2)
End If
Next
For i = 1 To .Rows - 1
.TextMatrix(i, 0) = Str(i)
Next
End With
starsort(0) = True
End If
End Sub

Private Sub Command1_Click()
DEKurva.rsSESpiral.Delete adAffectCurrent
End Sub

Private Sub Command3_Click()
PG1.Clear
End Sub

Private Sub Command4_Click()
Select Case Super.Tab
Case 0:
If Not DEKurva.rsSELingkaran.BOF Then
DEKurva.rsSELingkaran.MovePrevious
End If
Case 1:
If Not DEKurva.rsSESpiral.BOF Then
DEKurva.rsSESpiral.MovePrevious
End If
End Select
End Sub

Private Sub Command5_Click()
On Error Resume Next
Select Case Super.Tab
Case 0:
If Not DEKurva.rsSELingkaran.EOF Then
DEKurva.rsSELingkaran.MoveNext
End If
Case 1:
```

```
If Not DEKurva.rsSESpiral.EOF Then
DEKurva.rsSESpiral.MoveNext
End If
End Select
End Sub


Private Sub Command6_Click()
Dim dummy, Max As Single
Dim i, j, k, dui, jk As Integer
Dim dumm2 As String
If Not starsort(0) Then
dummy = arrJarak(0)
dumm2 = ArrStationing(0)
Max = 11
jk = 0
For i = 1 To Max - 1
For j = i + 1 To Max
If arrJarak(i) > arrJarak(j) Then
 dummy = arrJarak(i)
 dumm2 = ArrStationing(i)
 arrJarak(i) = arrJarak(j)
 ArrStationing(i) = ArrStationing(j)
 arrJarak(j) = dummy
 ArrStationing(j) = dumm2

End If

Next j
Next i
With FG2
For i = 0 To .Rows - 2
 .TextMatrix(i + 1, 0) = Str(i)
 .TextMatrix(i + 1, 1) = Format(arrJarak(i), "##00.000")
 .TextMatrix(i + 1, 2) = ArrStationing(i)
Next
For i = .Rows - 3 To 0 Step -1
If .TextMatrix(i + 1, 1) = .TextMatrix(i + 2, 1) Then
 .TextMatrix(i + 1, 1) = .TextMatrix(i + 2, 1)
 .TextMatrix(i + 1, 2) = .TextMatrix(i + 2, 2)
 .RemoveItem (i + 2)
End If
Next
For i = 1 To .Rows - 1
.TextMatrix(i, 0) = Str(i)
Next
End With
```

```
If Not DEKurva.rssESpiral.EOF Then
DEKurva.rssESpiral.MoveNext
End If
End Select
End Sub

Private Sub Command6_Click()
Dim dummy, Max As Single
Dim i, j, k, dui, jk As Integer
Dim dumm2 As String
If Not starson(0) Then
dummy = arrjarak(0)
dumm2 = ArrStationing(0)
Max = 11
jk = 0
For i = 1 To Max - 1
For j = i + 1 To Max
If arrjarak(i) > arrjarak(j) Then
 dummy = arrjarak(i)
 dumm2 = ArrStationing(i)
 arrjarak(i) = arrjarak(j)
 ArrStationing(i) = ArrStationing(j)
 arrjarak(j) = dummy
 ArrStationing(j) = dumm2

End If

Next j
Next i
With FG2
For i = 0 To .Rows - 2
 .TextMatrix(i + 1, 0) = Str(i)
 .TextMatrix(i + 1, 1) = Format(arrjarak(i), "###00.000")
 .TextMatrix(i + 1, 2) = ArrStationing(i)
Next
For i = .Rows - 3 To 0 Step -1
If .TextMatrix(i + 1, 1) = .TextMatrix(i + 2, 1) Then
 .TextMatrix(i + 1, 1) = .TextMatrix(i + 2, 1)
 .TextMatrix(i + 1, 2) = .TextMatrix(i + 2, 2)
 .RemoveItem (i + 2)
End If
Next
For i = 1 To .Rows - 1
 .TextMatrix(i, 0) = Str(i)
Next
End With
```

```
starsort(0) = True
End If
End Sub

Private Sub Command7_Click()
Dim pn, pmax, jawal, kawal As Single
Dim i As Integer
Dim piclengt As Single
Dim emax, ens, sc As Single
Dim atans, deltax As Single

With DEKurva
   pn = .rsSESpiral("en") * .rsSESpiral("Lebar")
   pmax = .rsSESpiral("emax") * .rsSESpiral("Lebar")
   Pns.Text = Format(pn, "##0.000")
   PMaxs.Text = Format(pmax, "##0.000")
   .rsSESpiral.UpdateBatch adAffectAll
   End With
 piclengt = (2 * DEKurva.rsSESpiral("Ls") + DEKurva.rsSESpiral("Lc")) * 1440
/ GScale
sc = 30
'MsgBox "emax " & Format(emax, "0.000000")
Koord(0).X = 1440
Koord(0).Y = MainFrm.Pic.Height \ 2
Koord(1).X = Koord(0).X + DEKurva.rsSESpiral("Ls") * 1440 / GScale
Koord(1).Y = Koord(0).Y - ((DEKurva.rsSESpiral("emax") +
DEKurva.rsSESpiral("en")) * 1440 * sc / GScale)

Koord(2).X = Koord(1).X + DEKurva.rsSESpiral("Lc") * 1440 / GScale
Koord(2).Y = Koord(1).Y
Koord(3).X = Koord(2).X + DEKurva.rsSESpiral("Ls") * 1440 / GScale
Koord(3).Y = Koord(0).Y
Koord(4).X = Koord(1).X
Koord(4).Y = Koord(0).Y + ((DEKurva.rsSESpiral("emax") -
DEKurva.rsSESpiral("en")) * 1440 * sc / GScale)
Koord(5).X = Koord(2).X
Koord(5).Y = Koord(0).Y + ((DEKurva.rsSESpiral("emax") -
DEKurva.rsSESpiral("en")) * 1440 * sc / GScale)
Koord(6).X = Koord(0).X
Koord(6).Y = Koord(0).Y - (DEKurva.rsSESpiral("en") * 1440 * sc / GScale)
Koord(7).X = Koord(3).X
Koord(7).Y = Koord(0).Y - (DEKurva.rsSESpiral("en") * 1440 * sc / GScale)
atans = ((Koord(1).X - Koord(0).X) / (Koord(1).Y - Koord(0).Y))
deltax = Abs(atans) * (DEKurva.rsSESpiral("en") * 1440 * sc / GScale)
Koord(8).X = Koord(0).X + deltax
Koord(8).Y = Koord(6).Y
Koord(9).X = Koord(3).X - deltax
```

```
starsort(0) = True
End If
End Sub

Private Sub Command7_Click()
Dim pn, pmax, jawal, kawal As Single
Dim i As Integer
Dim pielengt As Single
Dim emax, ens, sc As Single
Dim atans, deltax As Single

With DEKurva
  pn = .rsSESpiral("en") * .rsSESpiral("Lebar")
  pmax = .rsSESpiral("emax") * .rsSESpiral("Lebar")
  Pns.Text = Format(pn, "##0.000")
  PMaxs.Text = Format(pmax, "##0.000")
  .rsSESpiral.UpdateBatch adAffectAll
End With
pielengt = (2 * DEKurva.rsSESpiral("Ls") + DEKurva.rsSESpiral("Lc")) * 1440
/ GScale
sc = 50
'MsgBox "emax " & Format(emax, "0.000000")
Koord(0).X = 1440
Koord(0).Y = MainFrm.Pic.Height / 2
Koord(1).X = Koord(0).X + DEKurva.rsSESpiral("Ls") * 1440 / GScale
Koord(1).Y = Koord(0).Y - ((DEKurva.rsSESpiral("emax") -
DEKurva.rsSESpiral("en")) * 1440 * sc / GScale)

Koord(2).X = Koord(1).X + DEKurva.rsSESpiral("Lc") * 1440 / GScale
Koord(2).Y = Koord(1).Y
Koord(3).X = Koord(2).X + DEKurva.rsSESpiral("Ls") * 1440 / GScale
Koord(3).Y = Koord(0).Y
Koord(4).X = Koord(1).X
Koord(4).Y = Koord(0).Y + ((DEKurva.rsSESpiral("emax") -
DEKurva.rsSESpiral("en")) * 1440 * sc / GScale)
Koord(5).X = Koord(2).X
Koord(5).Y = Koord(0).Y + ((DEKurva.rsSESpiral("emax") -
DEKurva.rsSESpiral("en")) * 1440 * sc / GScale)
Koord(6).X = Koord(0).X
Koord(6).Y = Koord(0).Y - (DEKurva.rsSESpiral("en") * 1440 * sc / GScale)
Koord(7).X = Koord(3).X
Koord(7).Y = Koord(0).Y - (DEKurva.rsSESpiral("en") * 1440 * sc / GScale)
atans = ((Koord(1).X - Koord(0).X) / (Koord(1).Y - Koord(0).Y))
deltax = Abs(atans) * (DEKurva.rsSESpiral("en") * 1440 * sc / GScale)
Koord(8).X = Koord(0).X + deltax
Koord(8).Y = Koord(6).Y
Koord(9).X = Koord(3).X - deltax
```

```
Koord(9).Y = Koord(6).Y
Koord(10).X = Koord(1).X
Koord(10).Y = Koord(1).Y - 1440
Koord(11).X = Koord(4).X
Koord(11).Y = Koord(4).Y + 1440
Koord(12).X = Koord(2).X
Koord(12).Y = Koord(2).Y - 1440
Koord(13).X = Koord(5).X
Koord(13).Y = Koord(5).Y + 1440
Koord(14).X = Koord(3).X - deltax
Koord(14).Y = Koord(6).Y
  With FG2
 .TextMatrix(0, 0) = "No"
 .TextMatrix(0, 1) = "Jarak"
 .TextMatrix(0, 2) = "Stationing"
 .ColWidth(1) = 1200
.ColWidth(2) = 1400
.Rows = 1
 jawal = Val(Mid(DEKurva.rsSESpiral("Statawal").Value, 10, 6))
 kawal = Koord(0).X
  For i = 0 To 9
  arrJarak(i) = (Koord(i).X - kawal) * GScale / 1440
  ArrStationing(i) = "sta " & _
 Right(Left(DEKurva.rsSESpiral("Statawal").Value, 6), 2) & _
 " + " & Format(arrJarak(i) + jawal, "000.000")
  .AddItem Str(i) & vbTab & Format(arrJarak(i), "##0.000") & vbTab &
ArrStationing(i)
  Next
   arrJarak(i) = (arrJarak(8) - arrJarak(0)) * 2
  ArrStationing(i) = "sta " & _
 Right(Left(DEKurva.rsSESpiral("Statawal").Value, 6), 2) & _
 " + " & Format((arrJarak(i)) + jawal, "000.000")
  .AddItem Str(i) & vbTab & Format(arrJarak(i), "##0.000") & vbTab &
ArrStationing(i)
  i = i + 1
  arrJarak(i) = (arrJarak(9) - arrJarak(8))
  ArrStationing(i) = "sta " & _
 Right(Left(DEKurva.rsSESpiral("Statawal").Value, 6), 2) & _
 " + " & Format((arrJarak(i)) + jawal, "000.000")
  .AddItem Str(i) & vbTab & Format(arrJarak(i), "##0.000") & vbTab &
ArrStationing(i)

  End With
End Sub
Sub load_detil_lingkaran()
Dim i As Integer
Dim Recs As ADODB.Recordset
```

```
Koord(9).Y = Koord(6).Y
Koord(10).X = Koord(1).X
Koord(10).Y = Koord(1).Y - 1440
Koord(11).X = Koord(4).X
Koord(11).Y = Koord(4).Y + 1440
Koord(12).X = Koord(2).X
Koord(12).Y = Koord(2).Y - 1440
Koord(13).X = Koord(5).X
Koord(13).Y = Koord(5).Y + 1440
Koord(14).X = Koord(3).X - deltax
Koord(14).Y = Koord(6).Y
 With FG2
 .TextMatrix(0, 0) = "No"
 .TextMatrix(0, 1) = "Jarak"
 .TextMatrix(0, 2) = "Stationing"
 .ColWidth(1) = 1200
 .ColWidth(2) = 1400
 .Rows = 1
jawal = Val(Mid(DEKurva.rsSESpiral("Statawal").Value, 10, 6))
kawal = Koord(0).X
 For i = 0 To 9
 arrJarak(i) = (Koord(i).X - kawal) * GScale / 1440
 ArrStationing(i) = "sta " &
Right(Left(DEKurva.rsSESpiral("Statawal").Value, 6), 2) & _
 " + " & Format(arrJarak(i) + jawal, "000.000")
.AddItem Str(i) & vbTab & Format(arrJarak(i), "##0.000") & vbTab &
ArrStationing(i)
 Next
 arrJarak(i) = (arrJarak(8) - arrJarak(0)) * 2
 ArrStationing(i) = "sta " &
Right(Left(DEKurva.rsSESpiral("Statawal").Value, 6), 2) & _
 " + " & Format(arrJarak(i) + jawal, "000.000")
.AddItem Str(i) & vbTab & Format(arrJarak(i), "##0.000") & vbTab &
ArrStationing(i)
 i = i + 1
 arrJarak(i) = (arrJarak(9) - arrJarak(8))
 ArrStationing(i) = "sta " &
Right(Left(DEKurva.rsSESpiral("Statawal").Value, 6), 2) & _
 " + " & Format(arrJarak(i) + jawal, "000.000")
.AddItem Str(i) & vbTab & Format(arrJarak(i), "##0.000") & vbTab &
ArrStationing(i)

 End With
End Sub
Sub load_detil_lingkaran()
Dim i As Integer
Dim Recs As ADODB.Recordset
```

```
Set Recs = New ADODB.Recordset
DEKurva.rsSELingkaran.MoveFirst
Recs.Open "select * from detilselingkaran where IdSuper=" & _
DEKurva.rsSELingkaran("IdSuper"), get_connection, adOpenDynamic,
adLockBatchOptimistic
FG1.ColWidth(1) = 1500
FG1.ColWidth(2) = 2500

'FG1.Rows = Recs.RecordCount
While Not Recs.EOF
i = i + 1
FG1.TextMatrix(i, 0) = Recs("No")
FG1.TextMatrix(i, 1) = Recs("Jarak")
FG1.TextMatrix(i, 2) = Recs("Stationing")
Recs.MoveNext
Wend

Set Recs = Nothing
End Sub
Sub load_detil_spiral()
Dim i As Integer
Dim Recs As ADODB.Recordset
Set Recs = New ADODB.Recordset
DEKurva.rsSELingkaran.MoveFirst
Recs.Open "select * from detilsespiral where IdSuper=" & _
DEKurva.rsSESpiral("IdSuper"), get_connection, adOpenDynamic,
adLockBatchOptimistic
FG2.ColWidth(1) = 1500
FG2.ColWidth(2) = 2500

'FG1.Rows = Recs.RecordCount
While Not Recs.EOF
i = i + 1
FG2.TextMatrix(i, 0) = Recs("No")
FG2.TextMatrix(i, 1) = Recs("Jarak")
FG2.TextMatrix(i, 2) = Recs("Stationing")
Recs.MoveNext
Wend

Set Recs = Nothing
End Sub
Private Sub Form_Load()
Dim vScale As String


Open App.Path & "\option.ini" For Input As #1
Input #1, vScale
```

```
Set Recs = New ADODB.Recordset
DEKurva.rsSELingkaran.MoveFirst
Recs.Open "select * from detilselingkaran where IdSuper=" & _
DEKurva.rsSELingkaran("IdSuper"), get_connection, adOpenDynamic,
adLockBatchOptimistic
FG1.ColWidth(1) = 1500
FG1.ColWidth(2) = 2500

'FG1.Rows = Recs.RecordCount
While Not Recs.EOF
i = i + 1
FG1.TextMatrix(i, 0) = Recs("No")
FG1.TextMatrix(i, 1) = Recs("Jarak")
FG1.TextMatrix(i, 2) = Recs("Stationing")
Recs.MoveNext
Wend

Set Recs = Nothing
End Sub
Sub load_detil_spiral()
Dim i As Integer
Dim Recs As ADODB.Recordset
Set Recs = New ADODB.Recordset
DEKurva.rsSELingkaran.MoveFirst
Recs.Open "select * from detilsespiral where IdSuper=" & _
DEKurva.rsSESpiral("IdSuper"), get_connection, adOpenDynamic,
adLockBatchOptimistic
FG2.ColWidth(1) = 1500
FG2.ColWidth(2) = 2500

'FG1.Rows = Recs.RecordCount
While Not Recs.EOF
i = i + 1
FG2.TextMatrix(i, 0) = Recs("No")
FG2.TextMatrix(i, 1) = Recs("Jarak")
FG2.TextMatrix(i, 2) = Recs("Stationing")
Recs.MoveNext
Wend

Set Recs = Nothing
End Sub
Private Sub Form_Load()
Dim vScale As String

Open App.Path & "\option.ini" For Input As #1
Input #1, vScale
```

```
GScale = Val(vScale)
Close #1
Call load_detil_lingkaran
Call load_detil_spiral

starsort(0) = False
starsort(1) = False
JenisElev = 0
End Sub

Private Sub Form_Unload(Cancel As Integer)

MainFrm.Toolbar1.Buttons(6).Value = tbrPressed

End Sub
Private Sub GambarLingkaran_Click()
GambarKurvaLingkaran
End Sub

Private Sub mnuNew_Click()
Select Case Super.Tab
Case 0:
DEKurva.rsSELingkaran.AddNew
Case 1:
DEKurva.rsSESpiral.AddNew
End Select
End Sub

Private Sub mnuPrint_Click()
On Error Resume Next
Select Case Super.Tab
Case 0
 ReportSELingkaran.Refresh
 ReportSELingkaran.Show
Case 1
 ReportSESpiral.Refresh
  ReportSESpiral.Show

End Select
End Sub
Sub save_lingkaran()
Dim i As Integer
Dim Recs As ADODB.Recordset
Dim constr As String
constr = get_connection
With DEKurva
DeleteTable "DetilSELingkaran", "IdSuper=" & .rsSELingkaran("IdSuper")
```

```
GScale = Val(vScale)
Close #1
Call load_detil_lingkaran
Call load_detil_spiral

starsort(0) = False
starsort(1) = False
JenisElev = 0
End Sub

Private Sub Form_Unload(Cancel As Integer)

MainFrm.Toolbar1.Buttons(6).Value = tbrPressed

End Sub
Private Sub GambarLingkaran_Click()
GambarKurvaLingkaran
End Sub

Private Sub mnuNew_Click()
Select Case Super.Tab
Case 0:
DEKurva.rsSELingkaran.AddNew
Case 1:
DEKurva.rsSESpiral.AddNew
End Select
End Sub

Private Sub mnuPrint_Click()
On Error Resume Next
Select Case Super.Tab
Case 0
ReportSELingkaran.Refresh
ReportSELingkaran.Show
Case 1
ReportSESpiral.Refresh
ReportSESpiral.Show

End Select
End Sub
Sub save_lingkaran()
Dim i As Integer
Dim Recs As ADODB.Recordset
Dim constr As String
constr = get_connection
With DEKurva
DeleteTable "DetilSELingkaran", "idSuper=" & .rsSELingkaran("idsuper")
```

```
Set Recs = New ADODB.Recordset

Recs.Open "select * from DetilSELingkaran", constr, adOpenDynamic,
adLockBatchOptimistic
For i = 1 To FG1.Rows - 1
Recs.AddNew
Recs("No") = FG1.TextMatrix(i, 0)
Recs("IdSuper") = .rsSELingkaran("IdSuper")
Recs("Jarak") = Val(FG1.TextMatrix(i, 1)) + Val(Right(FG1.TextMatrix(i, 1), 3))
/ 1000
Recs("Stationing") = FG1.TextMatrix(i, 2)
Recs.UpdateBatch adAffectAll
Next
End With
Set Recs = Nothing
End Sub
Sub save_spiral()
Dim i As Integer
Dim Recs As ADODB.Recordset
Dim constr As String
constr = get_connection

With DEKurva
DeleteTable "DetilSESpiral", "IdSuper=" & .rsSESpiral("IdSuper")
Set Recs = New ADODB.Recordset

Recs.Open "select * from DetilSESpiral", constr, adOpenDynamic,
adLockBatchOptimistic

For i = 1 To FG2.Rows - 1
Recs.AddNew
Recs("No") = FG2.TextMatrix(i, 0)
Recs("IdSuper") = .rsSESpiral("IdSuper")
Recs("Jarak") = Val(FG2.TextMatrix(i, 1)) + Val(Right(FG2.TextMatrix(i, 1), 3))
/ 1000
Recs("Stationing") = FG2.TextMatrix(i, 2)
Recs.UpdateBatch adAffectAllChapters
Next
End With
Set Recs = Nothing
End Sub
Private Sub mnuSave_Click()
Select Case Super.Tab
Case 0:
Call save_lingkaran

'dEKurva.rsSELingkaran.Requery
```

```
Set Recs = New ADODB.Recordset

Recs.Open "select * from DetilSELingkaran", constr, adOpenDynamic,
adLockBatchOptimistic
For i = 1 To FG1.Rows - 1
Recs.AddNew
Recs("No") = FG1.TextMatrix(i, 0)
Recs("IdSuper") = rsSELingkaran("IdSuper")
Recs("Jarak") = Val(FG1.TextMatrix(i, 1)) + Val(Right(FG1.TextMatrix(i, 1), 3))
/ 1000
Recs("Stationing") = FG1.TextMatrix(i, 2)
Recs.UpdateBatch adAffectAll
Next
End With
Set Recs = Nothing
End Sub
Sub save_spiral()
Dim i As Integer
Dim Recs As ADODB.Recordset
Dim constr As String
constr = get_connection

With DEKurva
DeleteTable "DetilSESpiral", "IdSuper=" & rsSESpiral("IdSuper")
Set Recs = New ADODB.Recordset

Recs.Open "select * from DetilSESpiral", constr, adOpenDynamic,
adLockBatchOptimistic

For i = 1 To FG2.Rows - 1
Recs.AddNew
Recs("No") = FG2.TextMatrix(i, 0)
Recs("IdSuper") = rsSESpiral("IdSuper")
Recs("Jarak") = Val(FG2.TextMatrix(i, 1)) + Val(Right(FG2.TextMatrix(i, 1), 3))
/ 1000
Recs("Stationing") = FG2.TextMatrix(i, 2)
Recs.UpdateBatch adAffectAllChapters
Next
End With
Set Recs = Nothing
End Sub
Private Sub mnuSave_Click()
Select Case Super.Tab
Case 0:
Call save_lingkaran

'dEKurva.rsSELingkaran.Requery
```

```
Case 1:
Call save_spiral

'DEKurva.rsSESpiral.Requery
End Select

End Sub

Private Sub Super_Click(PreviousTab As Integer)
JenisElev = Super.Tab
End Sub
```

SKALA

```
Option Explicit
Dim skala As String
Private Sub CancelButton_Click()
GScale = 1
Unload Me
End Sub

Private Sub Form_Load()
Open App.Path & "\option.ini" For Input As #1
Input #1, skala
txtSkala.Text = skala
Close #1
End Sub

Private Sub OKButton_Click()
GScale = Val(txtSkala.Text)
Open App.Path & "\option.ini" For Output As #1
Write #1, GScale
Close #1
Unload Me
End Sub
```

```
Case 1:
Call save_spiral
'DEKurva.rsSESpiral.Requery
End Select
End Sub
Private Sub Super_Click(PreviousTab As Integer)
JenisElev = Super.Tab
End Sub
```

SKALA

```
Option Explicit
Dim skala As String
Private Sub CancelButton_Click()
GScale = 1
Unload Me
End Sub
Private Sub Form_Load()
Open App.Path & "\option.ini" For Input As #1
Input #1, skala
txtSkala.Text = skala
Close #1
End Sub
Private Sub OKButton_Click()
GScale = Val(txtSkala.Text)
Open App.Path & "\option.ini" For Output As #1
Write #1, GScale
Close #1
Unload Me
End Sub
```