

PEMBUATAN GAME "DANCE DANCE REVOLUTION" DENGAN VISUAL BASIC

Skripsi

untuk memenuhi salah satu syarat kelulusan

Program Pasca Sarjana (S-1)

Teknik Informatika



Oleh :

Nama : Nicolas Priyono

NIM : 005314009



**JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS SANATA DHARMA
YOGYAKARTA
2004**

MAKING THE 'DANCE DANCE REVOLUTION' GAME USING VISUAL BASIC



by :

Name : Nicolas Priyono

NIM : 005314009



**INFORMATION TECHNOLOGY
SANATA DHARMA UNIVERSITY
YOGYAKARTA
2004**

HALAMAN PERSETUJUAN

SKRIPSI

**PEMBUATAN GAME "DANCE-DANCE REVOLUTION"
DENGAN VISUAL BASIC**

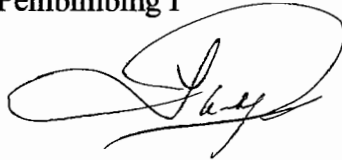
disusun oleh:

Nama : Nicolas Priyono

NIM : 005314009

disetujui oleh:

Dosen Pembimbing I



(Ir. F. Soesianto, B. Sc. E, Ph. D.)

Tanggal : 28 September 2004

Dosen Pembimbing II



(Dr. Riyad Mubarak, B. Sc., M. Sc.)

Tanggal :

HALAMAN PENGESAHAN

SKRIPSI

MAKING THE 'DANCE DANCE REVOLUTION' GAME USING VISUAL BASIC

dipersiapkan dan disusun oleh:

Nicolas Priyono

NIM : 005314009

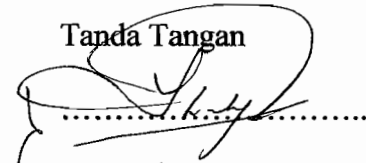
Telah dipertahankan di depan Panitia Penguji
pada tanggal 16 Agustus 2004 dan dinyatakan memenuhi syarat.

Susunan Panitia Penguji

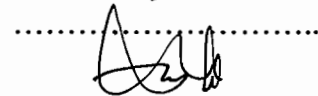
Nama Lengkap

Tanda Tangan

Penguji I : Ir. F. Soesianto, B. Sc. E, Ph. D.



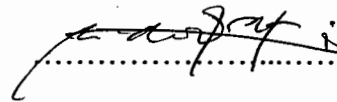
Penguji II : Dr. Riyad Mubarak, B. Sc., M. Sc.



Penguji III : A.M. Polina, S. Kom., M. Sc.



Penguji IV : Ridowati Gunawan, S. Kom., M. T.



Yogyakarta, 30 September 2004

Fakultas Teknik



Ir. Greg. Heliarko, S.J., S.S., B.ST., MA., M.SC.

Skripsi ini dipersembahkan untuk:

Mama, Papa, Thomas, dan Susan.

Terima kasih atas segala cinta, dukungan dan waktu yang telah kalian berikan.

Juga kepada para gamers Indonesia.

Jangan hanya main saja, kita bisa buat sendiri game kita sendiri.

Di mana ada kemauan dan usaha,

Di sana pasti ada jalan.

Orang gagal adalah orang yang tidak mau mencoba.

Hargai orang lain seperti kamu menginginkan kamu dihargai orang lain.

KATA PENGANTAR

Permainan komputer tentunya sudah tidak asing lagi. Malah banyak orang yang mengenal komputer dan mulai tertarik untuk mendalaminya karena mengenal program permainan pada komputer terlebih dahulu. Bahkan saya sendiri merupakan orang yang mengenal permainan komputer terlebih dahulu sebelum mendalami pemrograman komputer. Saya memperhatikan bahwa hanya sedikit sekali program permainan komputer yang dibuat oleh bangsa sendiri. Karena itulah saya mengumpulkan informasi untuk dapat membuat program permainan komputer sendiri dan mengambil langkah awal untuk membuatnya sebagai karya tulis.

Pada kesempatan ini juga, saya ingin mengucapkan terima kasih secara tertulis kepada pihak-pihak berikut:

1. Ir.Greg.Heliarko,SJ.,SS.,B.ST.,MA.,M.SC. selaku Dekan Teknik Universitas Sanata Dharma.
2. Ir. Gregorius Harjanto selaku Dekan Teknik Universitas Sanata Dharma periode sebelumnya.
3. Agnes Maria Polina, S.Kom., M.Sc. selaku Kaprodi Teknik Informatika Universitas Sanata Dharma yang baru dan J.B. Budi Darmawan, M. Sc. selaku Kaprodi Teknik Informatika Universitas Sanata Dharma pada periode sebelumnya.
4. Mama dan Papa, atas dukungan materi dan moril kalian hingga kini.

5. Pak Soesianto, yang telah bersedia menjadi dosen pembimbing I dalam proses penulisan karya tulis ini.
6. Pak Riyad, selaku dosen pembimbing yang masih memberikan kesempatan kepada saya dan teman-teman untuk meminta bantuan Bapak. Selamat jalan. Kami berharap Bapak dapat kembali lagi ke Indonesia.
7. Pak Puspa, Pak Antok, Bu Rido, Pak Donny, Pak Antok, dan segenap dosen Teknik Informatika Universitas Sanata Dharma terima kasih atas segala ilmu yang telah diberikan pada saya.
8. Pak Belle, Mas Danang dan Mas Catur yang telah membantu saya dalam kelancaran proses studi saya.
9. Susan, atas segala dukungan dan dorongannya di saat saya sedang membutuhkannya. v(^u^)^v
10. Jerry dan Donny atas peminjaman cartridge printernya.
11. dan kepada pihak-pihak yang disebutkan namun turut membantu dalam proses pengembangan karya tulis ini.

Terakhir, saya berharap agar karya tulis ini juga berguna bagi para pembaca. Saya sangat berterima kasih bila pembaca bersedia mengirimkan saran, kritik, atau pertanyaan seputar pembuatan karya tulis ini ke NickoAtStage@Yahoo.com.

Terima kasih.

Yogyakarta, 9 September 2004

Nicolas Priyono

DAFTAR ISI



HALAMAN JUDUL.....	i
HALAMAN PERSETUJUAN.....	iii
HALAMAN PENGESAHAN.....	iv
HALAMAN PERSEMBAHAN.....	v
HALAMAN MOTTO.....	vi
KATA PENGANTAR.....	vii
DAFTAR ISI.....	ix
DAFTAR GAMBAR.....	xii
DAFTAR TABEL.....	xiv
INTISARI.....	xv
ABSTRACT.....	xvi
PERNYATAAN KEASLIAN KARYA.....	xvii
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang Masalah.....	1
1.2 Rumusan Masalah.....	2
1.3 Batasan Masalah.....	2
1.4 Tujuan Penulisan.....	3
1.5 Metodologi Penelitian.....	3
1.6 Sistematika Penulisan.....	6
BAB II LANDASAN TEORI.....	7
2.1 <i>Dance Dance Revolution</i> (DDR).....	7
2.2 <i>Microsoft Visual Basic 6.0</i> (VB6).....	8
2.2.1 Pengenalan VB.....	8
2.2.2 Dasar Pemrograman VB.....	9
2.3 <i>Corel Draw</i>	19
2.4 <i>Adobe Photoshop 7.0</i>	20
2.5 <i>File Inialisasi (INI File)</i>	22
2.6 <i>Windows Application Programming Interface (API)</i>	22

2.7 API pada VB	25
2.7.1 <i>Bit Block Transfer</i> (BitBlt).....	27
2.7.2 Menulis Nilai <i>String</i> ke dalam <i>File</i> INI (WritePrivateProfileString).....	33
2.7.3 Mengambil Nilai <i>String</i> dari <i>File</i> INI (GetPrivateProfileString).	35
2.8 Grafik pada VB	36
2.9 Dasar Animasi	38
BAB III ANALISIS dan RANCANGAN SISTEM	40
3.1 Analisis Sistem	40
3.2 Kebutuhan Perangkat-keras.....	41
3.3 Kebutuhan Perangkat-lunak	41
3.4 Rancangan Sistem	42
3.4.1 Layar <i>Splash Screen</i>	43
3.4.2 Layar Utama.....	44
3.4.3 Layar Mulai Permainan.....	45
3.4.4 Layar <i>Loading</i>	46
3.4.5 Layar Permainan	47
3.4.6 Layar Penilaian.....	48
3.4.7 Layar Nilai Tertinggi.....	49
3.4.8 Layar Editor.....	51
3.4.9 Layar Pengaturan.....	52
3.4.10 Layar Trims	53
3.4.11 Jendela Pendukung.....	54
3.4.12 <i>File</i> Inisialisasi	56
BAB IV IMPLEMENTASI dan HASIL PROGRAM	59
4.1 Layar <i>Splash Screen</i>	60
4.2 Layar Utama	60
4.3 Layar Mulai Permainan	63
4.4 Layar <i>Loading</i>	64
4.5 Layar Permainan.....	66
4.6 Layar Penilaian.....	76

4.7 Layar Nilai Tertinggi.....	80
4.8 Layar Editor.....	83
4.9 Layar Pengaturan.....	90
4.10 Layar Trims	92
4.11 Jendela Pendukung.....	93
4.11.1 Konfirmasi.....	93
4.11.2 <i>Browse</i>	94
4.12 Menampilkan, Menyimpan, dan Membaca Data	95
4.13 Modul dan Kelas	95
4.13.1 Modul DirectX	96
4.13.2 Modul mdlOnTop.....	96
4.13.3 Modul PublicSub.....	96
4.13.4 Modul Skinning.....	96
4.13.5 Modul Variabel.....	96
4.13.6 Modul WindowsAPI	97
4.13.7 Kelas MP3.....	97
BAB V KESIMPULAN dan SARAN.....	98
5.1 Kesimpulan.....	98
5.2 Saran.....	99
DAFTAR PUSTAKA.....	101
LAMPIRAN	
Lampiran A. Daftar Istilah	L-1
Lampiran B. Ketentuan dalam Penulisan	L-2

DAFTAR GAMBAR

Gambar 2.1 Contoh layar permainan DDR dan cara memainkannya (<i>snapshot</i> dari DDR IV <i>PlayStation</i>).	7
Gambar 2.2 Tampilan VB.	8
Gambar 2.3 Diagram alir perintah <code>if</code> .	15
Gambar 2.4 Perbandingan diagram alir perintah <code>if</code> dan <code>Select Case</code> .	16
Gambar 2.5 Diagram alir perintah <code>For Next</code> .	17
Gambar 2.6 Perbandingan perintah <code>Do While Loop</code> dan <code>Do Loop While</code> .	18
Gambar 2.7 Tampilan <i>Corel Draw</i> .	19
Gambar 2.8 Tampilan <i>Adobe Photoshop 7.0</i> .	21
Gambar 2.9 Operasi Bitblt.	29
Gambar 2.10 Contoh <i>sprite</i> .	30
Gambar 2.11 Contoh <i>mask</i> .	30
Gambar 2.12 Contoh gambar latar yang telah dioperasikan dengan <i>mask</i> .	32
Gambar 2.13 Contoh gambar latar yang telah dioperasikan dengan <i>sprite</i> .	33
Gambar 2.14 Sistem koordinat VB.	37
Gambar 2.15 Dasar animasi.	38
Gambar 3.1 Struktur program (diagram alir).	42
Gambar 3.2 Rancangan tampilan <i>splash screen</i> .	43
Gambar 3.3 Rancangan tampilan layar utama.	44
Gambar 3.4 Rancangan tampilan layar mulai permainan.	45
Gambar 3.5 Rancangan tampilan layar <i>loading</i> .	46
Gambar 3.6 Rancangan tampilan layar permainan.	47
Gambar 3.7 Rancangan tampilan layar penilaian.	48
Gambar 3.8 Rancangan tampilan layar nilai tertinggi.	50
Gambar 3.9 Rancangan tampilan layar editor.	51
Gambar 3.10 Rancangan tampilan layar pengaturan.	53
Gambar 3.11 Rancangan tampilan trims.	54
Gambar 3.12 Rancangan tampilan konfirmasi.	55

Gambar 3.13 Rancangan tampilan <i>browse</i>	55
Gambar 3.14 Rancangan tampilan saat menyimpan data.....	55
Gambar 3.15 Rancangan tampilan saat memanggil data.	56
Gambar 3.16 Rancangan tampilan saat menampilkan data.....	56
Gambar 4.1 Peta layar dengan rancangan tampilan sistem.	59
Gambar 4.2 Contoh tampilan implementasi layar <i>splash screen</i>	60
Gambar 4.3 Contoh tampilan implementasi layar utama.	61
Gambar 4.4 Contoh tampilan implementasi layar mulai permainan.....	63
Gambar 4.5 Contoh tampilan implementasi layar <i>loading</i>	65
Gambar 4.6 Diagram alir algoritma yang digunakan (Majalah Level, 2003).	66
Gambar 4.7 Contoh tampilan implementasi layar permainan.	67
Gambar 4.8 Diagram alir algoritma tahap <i>update state</i>	67
Gambar 4.9 Diagram alir algoritma tahap <i>render</i>	69
Gambar 4.10 Ilustrasi peletakan gambar layar permainan.	70
Gambar 4.11 Diagram alir algoritma tahap <i>check state</i>	70
Gambar 4.12 Ilustrasi area.....	75
Gambar 4.11 Contoh tampilan implementasi layar penilaian.	76
Gambar 4.12 Contoh tampilan impementasi layar nilai tertinggi.	81
Gambar 4.13 Contoh tampilan implementasi layar editor.	83
Gambar 4.14 Contoh tampilan implementasi layar pengaturan.	91
Gambar 4.15 Contoh tampilan impementasi layar trims.....	93
Gambar 4.16 Contoh tampilan implementasi jendela konfirmasi.....	93
Gambar 4.17 Contoh tampilan implementasi <i>browse</i>	95
Gambar 4.18 Contoh tampilan implementasi saat membaca file INI.	95

DAFTAR TABEL

Tabel 2.1 Bagian-bagian VB.....	9
Tabel 2.2 Tipe-tipe variabel dan konstanta pada VB.....	11
Tabel 2.3 Operator perbandingan pada VB.....	14
Tabel 2.4 Operator logika pada VB.....	14
Tabel 2.5 Bagian-bagian <i>Corel Draw</i>	20
Tabel 2.6 Bagian-bagian <i>Adobe Photoshop 7.0</i>	21
Tabel 2.7 Daftar <i>file-file</i> dll yang sering digunakan.....	24

INTISARI

Tema penulisan ini adalah mengenai cara pembuatan program permainan kompute dengan menggunakan *Microsoft Visual Basic 6*. Jenis permainan komputer yang dibuat tergolong pada ‘musikal’ yang mengutamakan kecepatan dan ketepatan. Permainan ini dikembangkan pertama kali oleh suatu perusahaan pembuat permainan komputer dari Jepang (Konami) dan kemudian mendapatkan sambutan yang sangat baik di Indonesia. Permainan yang dimaksud adalah “Dance Dance Revolution”. Permainan ini muncul pertama kali pada mesin *arcade* yang dimainkan dengan pijakan kaki (pemain mengikuti irama dan petunjuk pada layar monitor dan menari di atas tempat yang disediakan). Kemudian, konsep permainan yang sama berkembang pada mesin konsol seperti *Sony PlayStation* (dengan perangkat tambahan agar pemain dapat bermain dengan pijakan kaki).

Program menjalankan suatu musik (MP3) dan dengan suatu *script* tertentu untuk menampilkan petunjuk panah (yang harus ditekan oleh pemain) pada layar monitor. Sehingga permainan ini tidak memerlukan teknologi AI, karena tingkat kesulitan tergantung pada *script* tersebut.

Penekanan pengembangan adalah pada bagaimana membuat interaksi yang menarik antara manusia dengan komputer dalam hubungannya dengan komputer sebagai media hiburan.

Kelebihan dari program ini adalah dengan disertakannya fasilitas untuk menambah lagu dan membuat *script* yang telah disebutkan tadi, sehingga pemain dapat menambahkan lagu kesukaannya dalam program permainan ini. Sehingga, program menjadi tidak membosankan bagi pemain.

Desain grafis dikembangkan dengan menggunakan Adobe Photoshop 7.0 dan Corel Draw 11, namun penekanan karya tulis tetap pada sisi pemrogramannya.

ABSTRACT

This project was about making a computer game using Microsoft Visual Basic 6.0. The game is a kind of musical that counting on speed and accuracy. This game was developed for the first time by a computer game company from Japan (Konami) and then got a nice welcome in Indonesia. The game named "Dance Dance Revolution". This game introduced in arcade machine that can be played using foot steps (players try to follow a song rhythm and the signs on the monitor screen and dance on the stepping ground). Then, the same game concept was developed on the console machine game ,e.g. Sony Playstation (with an extended device so the players can play using foot steps).

The software plays a music (MP3) and a certain script is made to display arrows sign (that should be pressed by player) on the monitor screen. So this game doesn't need artificial intelligence technology because the difficulty level depends on the scripts.

The development focused on how to build an attractive interactive between human and machine especially for entertainment purpose.

The speciality of this software is a facility to add songs and to build the scripts, so the players can add his/her favourite songs into the game by themself. So, the players won't get bored easily.

The graphic was designed using Adobe Photoshop 7.0 and Corel Draw 11, but still focus on the programming.

PERNYATAAN KEASLIAN KARYA

Saya menyatakan dengan sesungguhnya bahwa yang saya tulis tidak memuat karya atau bagian karya orang lain, kecuali yang telah saya sebutkan dalam kutipan dan daftar pustaka sebagaimana layaknya karya ilmiah.

Yogyakarta, Juni 2004

Nicolas Priyono

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Terdapat banyak macam permainan dengan menggunakan komputer sebagai medianya, seperti *action game*, RPG (*Role Playing Game*), RTS (*Real Time Strategy*), FPS (*First Person Shooter*), dan lain sebagainya. Bahkan banyak permainan komputer yang tipenya merupakan gabungan dari tipe-tipe yang telah disebutkan di atas, hingga permainannya menjadi semakin rumit. Salah satu yang menarik perhatian adalah jenis permainan dengan dukungan musik seperti DDR (*Dance-Dance Revolution*), 'Para-Para', VOS (*Virtual Orchestra Studio*), dan lain-lain. Jenis permainan ini 'menantang' pemain untuk mengikuti alunan musik dengan tarian kaki, tangan, atau dengan alat musik seperti gitar, drum, piano, dan lain-lain.

Pada dasarnya, jenis permainan ini mengutamakan keserasian gerakan pemainnya pada irama dengan mengikuti petunjuk pada layar monitor. Maka dipilih salah satu yang cukup populer, yaitu DDR. Perangkat-lunak dirancang mengikuti permainan DDR disertai tambahan fasilitas bagi penggunaanya untuk mengubah dan membuat 'gerakan'.

Selain DDR adalah yang paling populer diantara permainan sejenisnya, DDR juga merupakan salah satu permainan yang mudah dipahami, sehingga lebih mudah untuk diikuti dan dimainkan oleh berbagai kalangan. Bahkan jenis permainan ini dapat berfungsi sebagai media untuk olah raga.

1.2 Rumusan Masalah

Pertanyaan yang timbul dalam rencana pembuatan program permainan ini diantaranya:

1. Bagaimana menampilkan program dalam mode *fullscreen*?
2. Bagaimana mengimplementasikan animasi yang baik padahal banyak objek yang harus 'digerakkan'?
3. Bagaimana menyelaraskan suara musik dengan piranti masukan?
4. Bagaimana program permainan dapat menyimpan daftar urutan 'panah' beserta daftar nilai untuk setiap lagu dan tingkat kesulitan?
5. Bagaimana program dapat mendeteksi piranti masukan dan memberikan nilai?
6. Bagaimana membuat objek dari VB (*Visual Basic*) *ActiveX* tidak 'berbingkai' dan tidak memberikan kesan timbul?
7. Bagaimana menampilkan gambar/teks yang separuh transparan pada layar permainan?

1.3 Batasan Masalah

DDR adalah permainan yang dikembangkan untuk mesin VGC dan mesin *Arcade*, sehingga beberapa hal tidak diikuti sertakan dalam pembuatan DDR ini, seperti fasilitas untuk digunakan oleh dua orang pemain, objek 3-D (3 Dimensi) yang bergerak menari pada bagian latar, bonus yang didapatkan pemain ketika menyelesaikan permainan dengan baik, dan lain-lain. Perangkat masukan yang digunakan hanya papan ketik (*keyboard*) sebagai ganti dari perangkat masukan

berupa pijakan kaki pada VGC dan mesin *Arcade*. Tidak semua tombol pada papan ketik digunakan, pengguna cukup memilih empat tombol diantara sekian banyak tombol papan ketik yang digunakannya untuk dapat memainkan permainan ini, yaitu tombol panah atas, tombol panah bawah, tombol panah kiri, dan tombol panah kanan.

Lagu adalah faktor penting dalam program permainan ini, namun pengerjaannya tidak diutamakan. Sehingga semua lagu dalam program diambil dari *format file* MP3 yang sudah ada. Pengerjaan suara hanya dilakukan untuk keperluan *sound effect*.

1.4 Tujuan Penulisan

Manfaat yang diharapkan oleh penulis dari tugas akhir ini adalah:

1. Membuat program permainan yang menarik bagi semua kalangan.
2. Menerapkan VB untuk pembuatan program permainan.
3. Menerapkan teori animasi menggunakan beberapa perangkat-lunak :
 - VB
 - Corel Draw
 - Adobe Photoshop

1.5 Metodologi Penelitian

- **Pengamatan**

Untuk membantu penulisan, pengamatan dilakukan pada program permainan DDR dan sejenisnya seperti VOS. DDR yang diamati adalah yang

beredar pada *PlayStation* versi *4th Mix* dan *5th Mix* dan pada mesin *Arcade*. Selain itu pengamatan juga dilakukan pada contoh *source code* program-program permainan yang didapatkan dari berbagai sumber seperti *internet*, buku pemrograman yang disertai dengan *source code*, dan sumber lain yang tidak dapat disebutkan.

- **Studi Pustaka**

Setelah melalui proses pengamatan bentuk permainan komputer sesungguhnya, kemudian dipikirkan mengenai apa saja yang diperlukan dalam pembuatan program permainan. Untuk itu juga diambil dari berbagai sumber seperti *internet*, buku pemrograman program permainan dan dosen (dalam hal ini adalah dosen pembimbing). Maka didapatkan beberapa hal yang dapat membantu dalam pemrograman permainan tersebut, yaitu:

1. Pemrograman dengan VB,
2. Pengolahan grafis dengan *Corel Draw* dan *Adobe Photoshop*,
3. *Microsoft DirectX* dengan VB,
4. *Windows API* dengan VB,
5. Dasar pemrograman grafik dengan VB, dan
6. Dasar animasi

- **Simulasi**

Perangkat-lunak ini diujicobakan pada sistem komputer dengan spesifikasi:

a) Perangkat-keras

- Prosesor : Athlon Thunderbird 1 GHz
- Memori : DDR-SDRAM 256 MB
- *Virtual* Memori : 894.624 KB
- *Motherboard* : ABIT KG7-Lite
- Kartu VGA : Pixelview Nvidia GeForce4 MX420 SDRAM 64MB
- Kartu Suara : Trident Microsystems 4Dwave NX PCI (WDM)
- *Harddisk* : Seagate ST320423A 5400rpm 20GB
- *Monitor* : AOC Spectrum 4Vn 14"

b) Perangkat-lunak

- BIOS : Rev: 1.0
- Sistem Operasi : *Windows 2000 Professional* versi 5.0.2195 *Build 2195*
Windows ME versi 4.90.3000 *Build 3000*
- *Driver* VGA : nv4_disp.dll versi 6.13.10.2832
- *Driver* Suara : tridwave.sys (110520, 5.00.2128.1)
- *DirectX* : Versi 8.1 (4.08.01.0881)
Versi 9.0b (4.09.0000.0902)

1.6 Sistematika Penulisan

BAB I PENDAHULUAN

Bab ini berisi deskripsi mengenai latar belakang masalah, rumusan masalah, batasan masalah, tujuan penulisan, sistematika penulisan, metodologi penelitian, dan Ketentuan dalam penulisan.

BAB II LANDASAN TEORI

Bab ini berisi penjelasan mengenai teori yang digunakan penulis dalam proses pembuatan karya tulis ini.

BAB III ANALISIS dan RANCANGAN SISTEM

Bab ini berisi deskripsi mengenai analisis dan rancangan sistem yang dibuat.

BAB IV IMPLEMENTASI dan HASIL PROGRAM

Bab ini berisi deskripsi mengenai tahap penulisan program dan implementasi rancangan sistem ke sistem yang sesungguhnya.

BAB V KESIMPULAN dan SARAN

Bab ini berisi kesimpulan penulis berdasarkan rumusan masalah yang dihadapi dan saran yang diberikan penulis pada pembaca seputar pengembangan perangkat-lunak yang menjadi bahan penulisan ini.

DAFTAR PUSTAKA

LAMPIRAN

BAB II

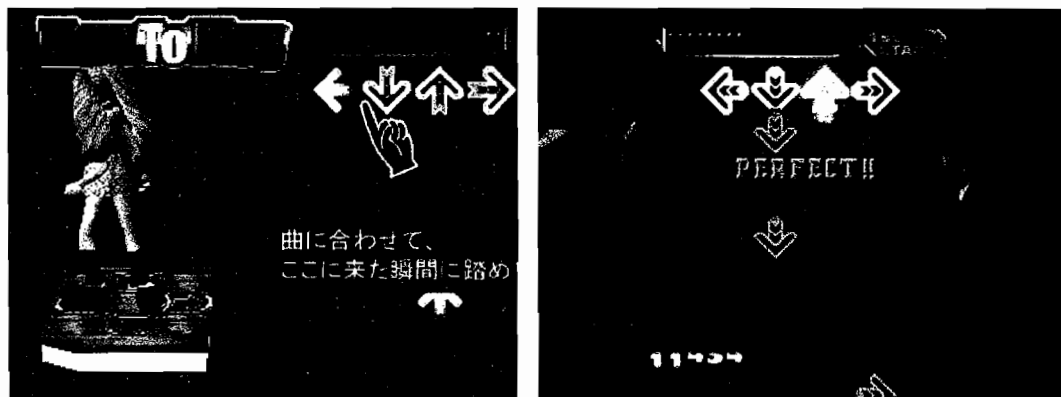
LANDASAN TEORI

2.1 *Dance Dance Revolution (DDR)*

DDR adalah salah satu dari sekian banyak program permainan yang tergolong musikal. DDR dikembangkan pertama kali oleh suatu perusahaan pembuat program permainan yang berasal dari Jepang, *Konami*. DDR diluncurkan pertama kali pada mesin *arcade* yang sampai saat ini masih banyak dijumpai di tempat - tempat hiburan.

DDR tersebut dimainkan dengan menggunakan pijakan kaki pada lantai yang telah disediakan. Pemain akan mendapatkan nilai setiap kali berhasil menginjak lantai yang tepat, sebaliknya pemain dapat gugur bila terlalu sering gagal menginjak lantai yang tepat.

Berikut adalah contoh layar permainan DDR dan cara memainkannya:



Gambar 2.1 Contoh layar permainan DDR dan cara memainkannya
(*snapshot* dari *DDR IV PlayStation*).

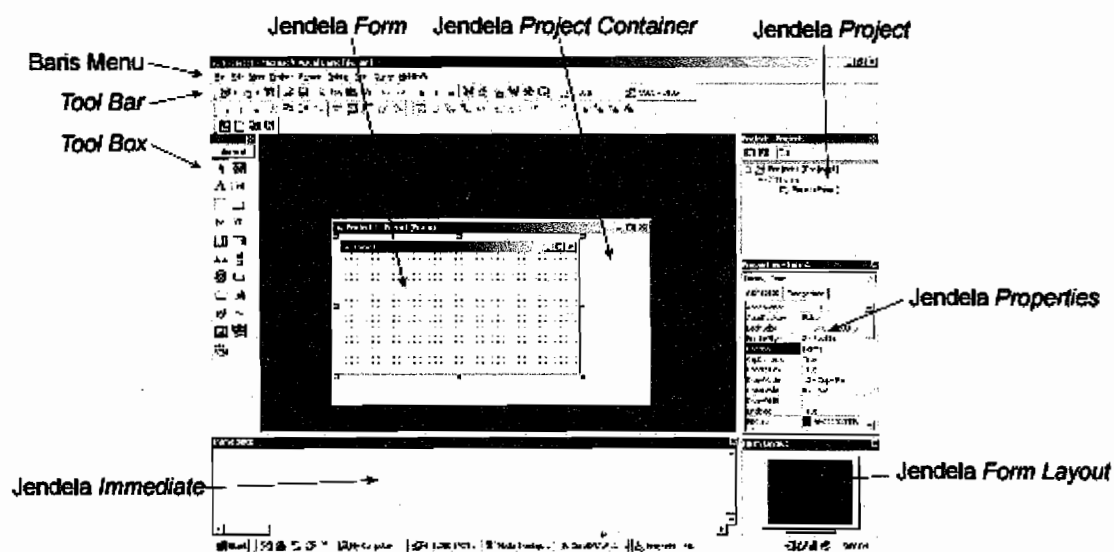
2.2 Microsoft Visual Basic 6.0 (VB6)

2.2.1 Pengenalan VB

VB merupakan bahasa pemrograman yang menggunakan konsep pemrograman “visual”. Konsep “visual” menawarkan begitu banyak kemudahan dalam merancang sistem (Tosin, 1997).

VB dikembangkan berdasarkan bahasa pemrograman BASIC. VB menawarkan fitur-fitur yang mempermudah *programmer* dalam membuat program yang berbasis *windows* dan menggunakan konsep pemrograman berorientasi objek.

Berikut adalah layar tampilan standar dari VB:



Gambar 2.2 Tampilan VB.

Keterangan lebih lanjut mengenai gambar di atas dapat dilihat pada Tabel

2.1 berikut ini:

Tabel 2.1 Bagian-bagian VB.

Nama Bagian	Keterangan
Baris menu	Terdiri dari perintah untuk memanggil <i>project</i> , menyimpan <i>project</i> , menambahkan <i>ActiveX</i> , <i>debugging</i> , format <i>form</i> , dan lain-lain. Bagian ini akan selalu digunakan.
<i>Tool bar</i>	Dilengkapi <i>icon</i> untuk mempermudah pengguna mengakses baris menu. Suatu keterangan mengenai <i>icon</i> akan muncul bila kursor <i>mouse</i> didiamkan di atas <i>icon</i> -nya.
<i>Tool box</i>	Seperti <i>tool bar</i> namun merupakan daftar <i>ActiveX object</i> yang dapat digunakan dalam program. <i>ActiveX</i> dapat ditambahkan dengan klik kanan pada <i>toolbox</i> , kemudian pilih <i>component</i> . Pada baris menu, pilih <i>Project-Component</i> , atau dengan menekan <i>Ctrl-T</i> .
Jendela <i>Immediate</i>	Untuk mencobakan baris program yang akan dituliskan, tanpa perlu mengeksekusi program sebenarnya, hasil dari baris program tersebut akan langsung muncul pada baris berikutnya.
Jendela <i>Project</i>	Menggunakan tampilan pohon, sehingga memudahkan untuk melihat bagian-bagian dari <i>project</i> , seperti daftar <i>form</i> , daftar modul, <i>report</i> , dll. klik kiri untuk mengubah jendela <i>form</i> .
Jendela <i>Properties</i>	Jendela ini menampilkan <i>properties</i> dari <i>ActiveX object</i> yang sedang aktif. Dengan jendela ini, perubahan terhadap <i>properties object</i> (seperti letak, warna, ukuran, <i>z order</i> , nama objek, dan lain-lain) dapat dilakukan.
Jendela <i>Form Layout</i>	Digunakan untuk dapat mengatur letak <i>form</i> pada layar, dengan cara klik dan <i>drag</i> , Untuk memudahkan, dapat juga ditampilkan garis bantu untuk layar dengan resolusi lain.
Jendela <i>Form</i>	Perubahan tampilan program termasuk untuk menambahkan, menghapus atau mengubah <i>ActiveX</i> yang digunakan dapat dilakukan. Secara <i>default</i> , objek akan diletakan dengan mengikuti <i>grid</i> (titik-titik bantu) yang ada.
Jendela <i>Project Container</i>	Jendela ini tidak selalu berupa tampilan <i>form</i> , jendela ini juga dapat menampilkan baris – baris program yang digunakan dalam <i>form</i> bersangkutan. Jendela ini dapat dibuka sebanyak-banyaknya.

2.2.2 Dasar Pemrograman VB

VB merupakan bahasa pemrograman tingkat tinggi, berarti bahwa bahasa yang digunakan hampir serupa dengan bahasa yang digunakan manusia. Seperti

pada pemrograman *BASIC*, VB juga bukan merupakan bahasa pemrograman yang *case-sensitive* (membedakan huruf besar dan huruf kecil).

a. Event (*Event Procedure*)

Event merupakan suatu bentuk interaksi antara pemakai dengan program pada saat program tersebut dijalankan.

Bentuk-bentuk interaksi yang dapat dilakukan pemakai terhadap program terdapat dalam *event*, seperti perintah-perintah yang dijalankan saat *form* dipanggil, pemakai meng-klik suatu objek, pemakai menekan suatu tombol, pemakai mengaktifkan suatu objek, pemakai menggerakkan *mouse*, dan lain sebagainya.

Event tidak perlu dibuat oleh *programmer*, setiap objek dalam VB telah memiliki *event* masing-masing. Dengan *parameter* berbeda-beda, *event* menyesuaikan kebutuhan *programmer* dalam pembuatan perangkat-lunaknya. *Event* juga disebut sebagai '*Proc*' dalam VB dan diperlakukan seperti suatu fungsi pada umumnya.

b. Properti (*Properties*)

Properti mempunyai peranan untuk memberikan atau mengubah suatu *setting* pada objek yang dibuat. Properti memberikan banyak kemungkinan *setting* pada suatu objek (Tosin, 2000).

Properti dapat menentukan nama, letak, ukuran, warna bahkan isi dari suatu objek. Penunjukan suatu properti dari suatu objek dilakukan dengan

memberikan tanda titik (.) setelah objek yang dimaksudkan kemudian menyebutkan properti yang dimaksudkan. Contoh:

```
Textbox1.text = "Nicko"
Picture1.left = 3000
```

c. Variabel dan Konstanta

Variabel dan konstanta digunakan dalam pemrograman dengan VB untuk menampung data sementara. Variabel dan konstanta tersebut lebih merupakan penamaan untuk suatu alamat dalam memori sistem. VB mengenal banyak tipe data, yaitu :

Tabel 2.2 Tipe-tipe variabel dan konstanta pada VB.

Tipe Data	Ukuran	Kisaran	Contoh
<i>Integer</i>	2 byte	-32.768 sampai 32.768	Dim Bird% Bird% = 37
<i>Long Integer</i>	4 byte	-2.147.483.648 sampai 2.147.483.647	Dim Loan& Loan& = 350000
<i>Single-precision</i>	4 byte	-3.402823E38 sampai 3.402823E38	Dim Price! Price! = 899.99
<i>Double-precision</i>	8 byte	-1.79769313486232D308 sampai	Dim Pi# Pi# = 3.1415926535
<i>Currency</i>	8 byte	-92233720368547735808 sampai	Dim Debt@ Debt@ = 7600300.50
<i>String</i>	1 byte per karakter	0 sampai 65.535	Dim Dog\$ Dog\$ = "pointer"
<i>Boolean</i>	2 byte	True atau False	Dim Flag as Boolean Flag = True
<i>Date</i>	8 byte	1 Januari 100 sampai 31 Desember 9999	Dim Birthday as Date Birthday = #3-1-63#
<i>Variant</i>	16 byte (untuk angka); 22 byte + 1 byte (untuk string)	Semua tipe data	Dim Total Total = 289.13 Dim Aku Aku = "Belajar"

Pada VB, deklarasi variabel dapat digantikan dengan suatu karakter tertentu diakhir nama variabel untuk menentukan tipe datanya, sehingga dalam VB deklarasi variabel berikut adalah sama :

- o Dim I as Integer
- o Dim I%

Sedangkan ukuran penyimpanan variabel dan konstanta dalam memori diukur dalam satuan *byte*, sehingga $8 \text{ bit} = 1 \text{ byte} = 1 \text{ karakter}$.

Dengan perintah 'Type', VB mengizinkan pemakainya untuk membuat sendiri tipe datanya. Hal ini sangat berguna untuk mengelompokkan data-data yang terbagi menjadi beberapa kategori. Contoh :

```
Type Employee
    Name As String
    DateOfBirth As Date
    HireDate As Date
End Type
```

Tipe data yang berkelompok tersebut digunakan dengan cara:

```
Dim ProductManager as Employee
ProductManager.Name = "Erick Cody"
ProductManager.DateOfBirth = #3-6-1970#
ProductManager.HireDate = #8-1-2001#
```

Hal tersebut tampak seperti pemanggilan properti dari objek karena VB menggunakan pemisah (*separator*) yang sama untuk hubungan antar objek dengan propertinya dan variabel data buatan dengan variabel komponennya.

Konstanta adalah serupa dengan variabel, namun nilai yang terkandung dalam konstanta tidak dapat diubah selama program berjalan, sedangkan variabel dapat berubah kapanpun selama program berjalan. Meskipun demikian konstanta

berguna saat digunakan dalam rumus matematis untuk nilai-nilai yang tidak berubah, seperti:

- $\pi = 3.1428571428571428571428571428571$
- 1 meter = 100 cm

Pendeklarasian konstanta serupa dengan pendeklarasian variabel, perbedaannya terletak pada perintah “Const” sebelum pendeklarasian. Contoh deklarasi konstanta:

- Const Dim I as Integer
- Const I = 100

d. If Then Else

Perintah ini adalah salah satu perintah yang berguna untuk percabangan dalam pemrograman. Percabangan berarti bahwa program dapat melakukan suatu pemeriksaan terhadap suatu kondisi tertentu dan kemudian mengambil tindakan berdasarkan kondisi yang ada. Kondisi yang diperlukan oleh perintah ini adalah kondisi benar (True) atau kondisi salah (False). Salah satu cara untuk memperoleh kondisi adalah dengan menggunakan operator-operator perbandingan. Sedangkan operator logika digunakan untuk dapat menggabungkan beberapa kondisi sekaligus untuk sebuah percabangan.

Tabel 2.3 Operator perbandingan pada VB.

Operator	Arti	Contoh
=	Sama dengan	10 = 10 adalah True
≠	Tidak sama dengan	10 ≠ 10 adalah False
>	Lebih besar dari	10 > 10 adalah False
<	Lebih kecil dari	10 < 10 adalah False
>=	Lebih besar dari atau sama dengan	10 >= 10 adalah True
<=	Lebih kecil dari atau sama dengan	10 <= 10 adalah True

Tabel 2.4 Operator logika pada VB.

Operator	Keterangan
And	Jika kedua ekspresi bernilai True, hasilnya akan True. Jika tidak, hasilnya False.
Or	Jika salah satu atau kedua ekspresi bernilai True, hasilnya bernilai True.
Not	Jika ekspresi bernilai True, hasilnya False. Jika ekspresi bernilai False, hasilnya True.
Xor	Jika salah satu ekspresi bernilai True, hasilnya True. Jika ekspresi bernilai True atau False,

Format dasar penulisan perintah ini adalah :

```

If (condition) Then
    (statement1)
[Else
    (statement2)]
End If

```

Condition adalah bagian yang menentukan pengambilan keputusan. Jika condition bernilai True, maka statement1 akan dijalankan, sebaliknya bila condition bernilai False, maka statement2 yang akan dieksekusi.

Perintah Else tidak harus mengikuti perintah If, perintah Else digunakan bila ada statement yang akan dijalankan bila condition bernilai False.

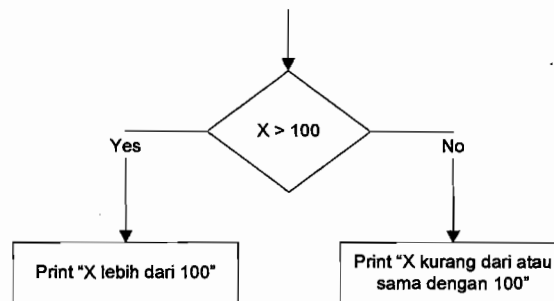
Contoh :

```

Dim X%
X = 1000
If X > 100 Then
    Print "X lebih dari 100"
Else
    Print "X kurang dari atau sama dengan 100"
End If

```

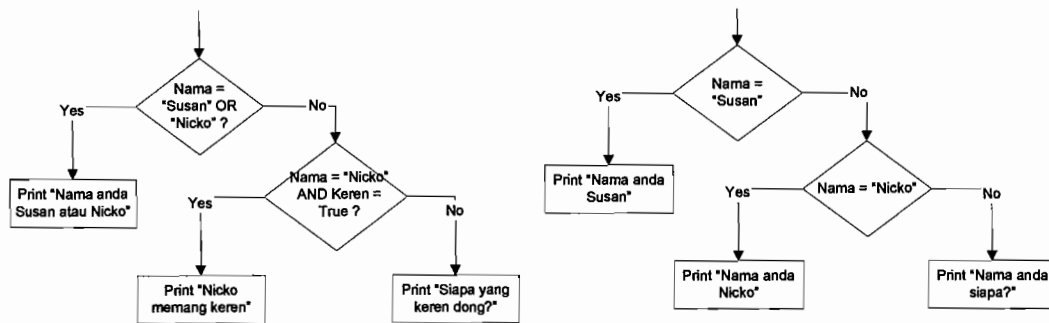
Akan menghasilkan : X lebih dari 100



Gambar 2.3 Diagram alir perintah If.

e. Select Case

Perintah ini juga digunakan dalam VB untuk percabangan dalam pemrograman. Percabangan dengan perintah ini sedikit berbeda dengan perintah If. Dengan perintah `Select Case`, *programmer* memperoleh keuntungan dengan semakin mudahnya membaca program dengan percabangan. Namun perintah ini tidak dapat melakukan pemeriksaan terhadap banyak kondisi yang berbeda sekaligus karena perintah ini hanya memeriksa kondisi sebuah ekspresi.



Perintah If

Perintah Select Case

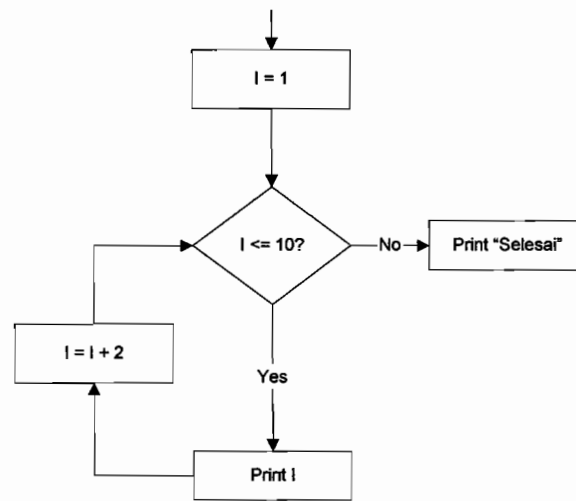
Gambar 2.4 Perbandingan diagram alir perintah If dan Select Case.

f. For Next

Perintah ini digunakan untuk membuat suatu perulangan dalam program. Perulangan berarti program melakukan suatu baris-baris perintah yang sama sebanyak yang ditentukan. Format perintah ini adalah :

```
For variabel = start to end [Step x]
    (Statement yang ingin diulang)
Next variable
```

Variabel adalah suatu variabel yang menampung nilai awal dari perulangan (start) dan jumlah perulangan yang telah dilakukan (berdasarkan nilai awal perulangan). Jika jumlah perulangan yang diminta telah dilakukan (variabel = end), maka perulangan akan dihentikan. Step adalah nilai yang ditambahkan pada variabel pada setiap perulangan. Hal ini dilakukan untuk mengubah nilai variabel agar perulangan dapat dihentikan. Jika step tidak disertakan, maka nilai variabel akan ditambah dengan nilai standar penambahannya, yaitu 1.



Gambar 2.5 Diagram alir perintah For Next.

g. Do Loop

Perintah ini juga dapat membuat suatu perulangan dalam program. Namun perintah ini menggunakan suatu pemeriksaan terhadap suatu kondisi (dapat berupa ekspresi yang kompleks) untuk menentukan akan perulangan dihentikan.

Sehingga *format* umum penulisan perintah ini adalah :

```

Do [While/Until] (condition)
    (Statement yang ingin diulang)
Loop
  
```

atau

```

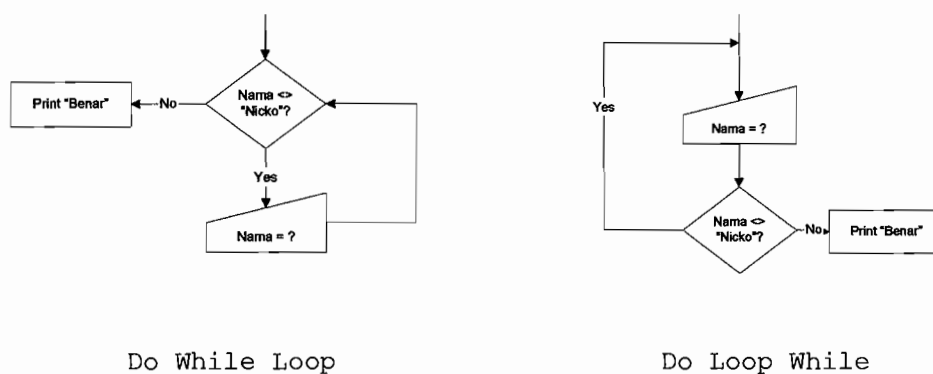
Do
    (Statement yang ingin diulang)
Loop [While/Until] (Condition)
  
```

Perintah `while/until` dapat diletakkan di akhir perintah atau di awal perintah. Jika `while/until` diletakkan di awal, maka sebelum melakukan perulangan yang pertama, komputer akan melakukan pemeriksaan kondisi terlebih dahulu, jika kondisi terpenuhi maka perulangan dapat dilakukan kemudian

melakukan pemeriksaan lagi sebelum melakukan perulangan berikutnya. Dengan cara ini dimungkinkan tidak terjadi perulangan sama sekali karena kondisinya tidak terpenuhi pada pemeriksaan pertamanya.

Namun bila perintah `while/until` diletakkan di akhir perintah, maka perulangan akan dilakukan terlebih dahulu sebelum pemeriksaan dilakukan. Akibatnya, minimal akan terjadi sebuah perulangan, kemudian perulangan berikutnya akan dilakukan bila kondisinya terpenuhi.

Perintah `while` dan `until` tidak dapat digunakan bersamaan. Jika menggunakan perintah `while`, maka suatu perulangan akan terus dilakukan selama kondisinya bernilai benar dan akan dihentikan saat kondisi bernilai salah. Sedangkan jika menggunakan perintah `until`, maka perulangan akan terus dilakukan selama kondisinya bernilai salah dan akan dihentikan saat kondisinya bernilai benar.



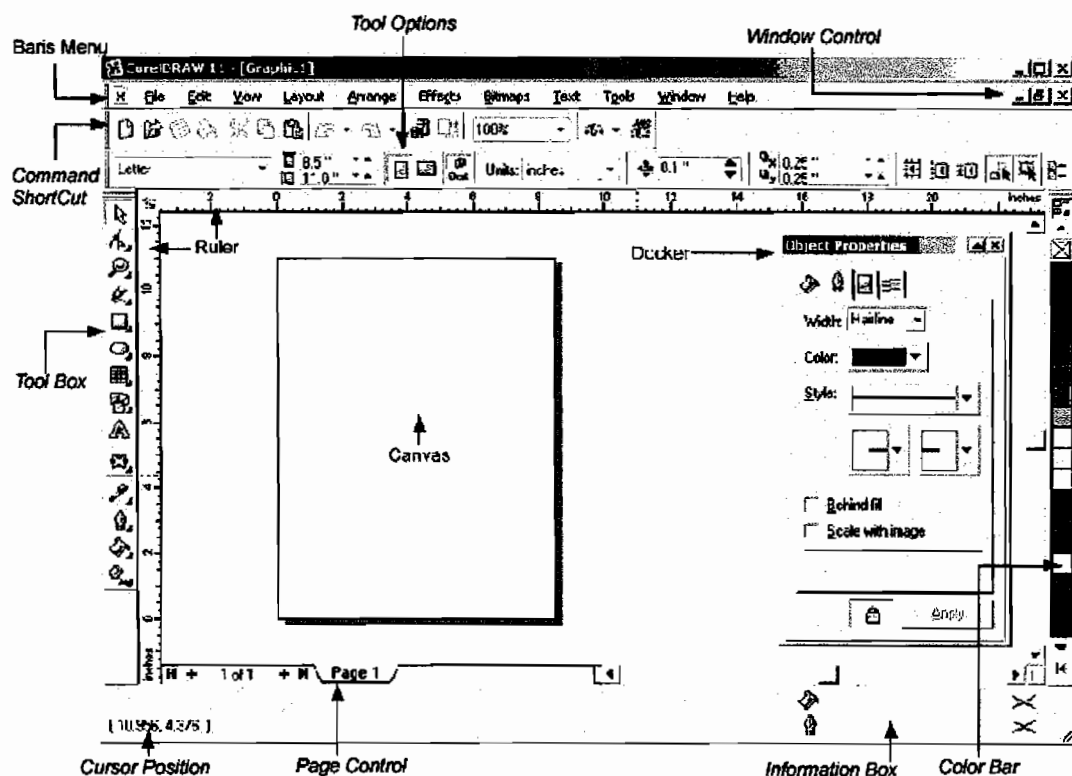
Gambar 2.6 Perbandingan perintah Do While Loop dan Do Loop While.

2.3 Corel Draw

Corel Draw merupakan salah satu perangkat-lunak grafis dengan vektor yang banyak digunakan dalam dunia komputer grafis. Saat ini *Corel Draw* telah mencapai versi 11.

Banyak *graphics designer* menggunakan *Corel Draw* untuk membuat desain. Bahkan: *Corel Draw* adalah pilihan yang terbaik untuk para desainer grafis dibanding program-program untuk membuat *page layout* lainnya (Maki, 2002).

Berikut adalah tampilan dasar dari *Corel Draw* 11:



Gambar 2.7 Tampilan *Corel Draw*.

Penjelasan mengenai gambar di atas dapat dilihat pada Tabel 2.5 berikut ini:

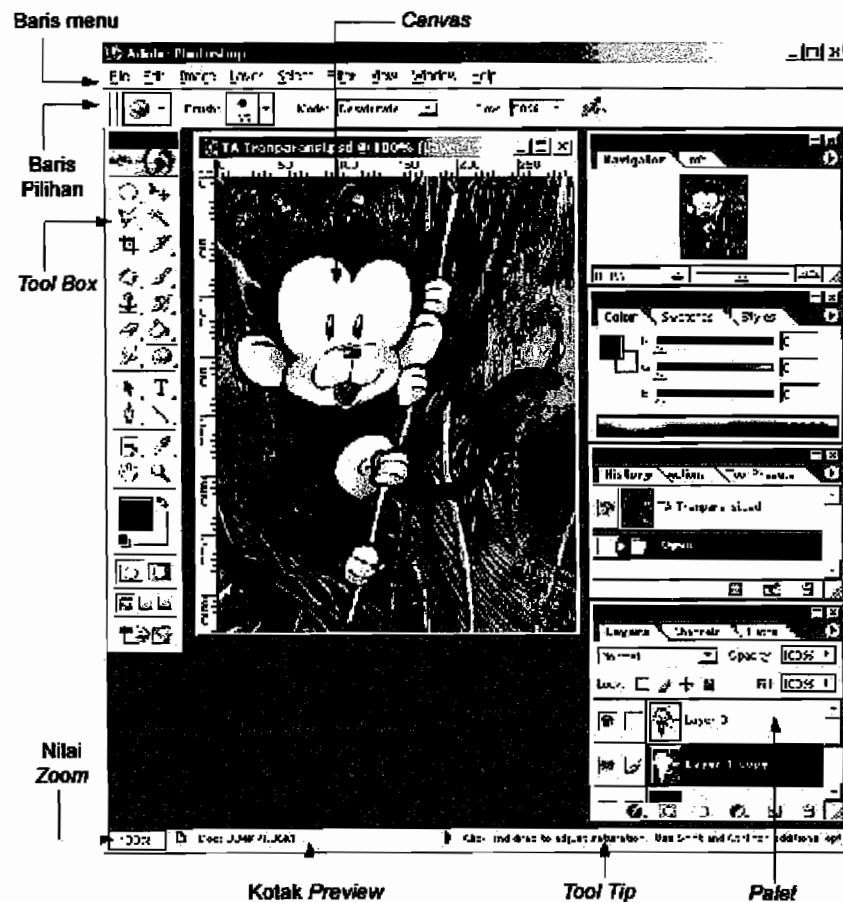
Tabel 2.5 Bagian-bagian *Corel Draw*.

Nama Bagian	Keterangan
Baris Menu	Berisi perintah tambahan, mulai dari membuka file, menyimpan, mengatur halaman, <i>print</i> , dan lain-lain.
<i>Command Shortcut</i>	Berisi ikon-ikon untuk operasi penting dari menu.
<i>Tool Box</i>	Berisi berbagai macam 'peralatan' yang digunakan untuk menggambar.
<i>Tool Options</i>	Isi bagian ini berubah-ubah sesuai dengan <i>tool</i> yang sedang digunakan.
<i>Canvas</i>	Tempat menggambar.
<i>Ruler</i>	Penggaris layar, digunakan untuk menyesuaikan objek dengan keadaan sebenarnya (jika dicetak).
<i>Docker</i>	Ekspansi dari beberapa perintah menu. <i>Docker</i> dapat dibuat 'melayang' (<i>floating</i>) atau tergabung dengan jendela utama (<i>docking</i>).
<i>Window Control</i>	Untuk mengatur jendela <i>canvas</i> (bukan jendela <i>Corel Draw</i>). Untuk bekerja dengan banyak dokumen.
<i>Page Control</i>	Mengatur halaman dari dokumen yang memiliki banyak halaman.
<i>Color Bar</i>	Tabel warna.
<i>Information Box</i>	Informasi dari objek yang aktif.
<i>Cursor Position</i>	Posisi kursor <i>mouse</i> .

2.4 Adobe Photoshop 7.0

Berbeda dengan *Corel Draw*, *Photoshop* merupakan perangkat-lunak grafis yang bertipe *bitmap*. *Graphics designer* juga banyak menggunakan perangkat-lunak ini untuk pengolahan gambar karena fasilitas dan kemudahan yang diberikan perangkat-lunak ini.

Photoshop bukanlah penyunting gambar biasa, tetapi merupakan program yang paling populer dan paling hebat yang ada di pasaran sekarang ini (McClelland, 2002).

Gambar 2.8 Tampilan *Adobe Photoshop 7.0*.

Penjelasan mengenai gambar di atas dapat dilihat pada Tabel 2.6 berikut ini:

Tabel 2.6 Bagian-bagian *Adobe Photoshop 7.0*.

Nama bagian	Keterangan
Baris menu	Berisi perintah tambahan, mulai dari membuka <i>file</i> , menyimpan, mengatur halaman, <i>print</i> , dan lain-lain.
Baris pilihan	Digunakan untuk mengubah perilaku <i>tool</i> yang sedang digunakan.
Tool box	<i>Tool</i> dasar dari <i>Photoshop</i> .
Nilai Zoom	Nilai perbesaran gambar yang tampak pada layar.
Canvas	Tempat menggambar.
Kotak Preview	Ukuran gambar dalam memori. Juga untuk melihat apakah gambar muat pada kertas, jika gambar dicetak.
Tool Tip	Keterangan singkat mengenai <i>tool</i> .
Palet	Jendela yang melayang berisi fasilitas dari <i>Photoshop</i> seperti <i>layers</i> , <i>history</i> , <i>color table</i> , dan lain-lain.

2.5 File Inisialisasi (INI File)

File inisialisasi atau *file* INI berisi nilai *setting* yang diperlukan oleh sistem *Windows*. *File* ini digunakan sejak versi *Windows* 3.xx, dan untuk menjaga stabilitas pada sistem *Windows* (untuk *Windows* 9x/NT) *file* ini tetap digunakan. Biasanya *setting Windows* terdapat pada *file System.ini* dan *Win.ini* pada direktori *Windows* maupun direktori sistem *Windows*. Untuk melihat *setting file* INI, digunakan *file utility Sysedit.exe* untuk *Windows* versi 95, 98, 98SE, dan *file Msconfig.exe* untuk *Windows* ME (Hadi, 2001). Format *file* INI dibagi atas:

```
[Section]
Key = Value
```

Keterangan:

```
[Section] = Nama bagian atau seksi
Key       = Nama kunci
Value     = Nilai dari nama kunci
```

Contoh:

```
[Keyboard]
Type = 4
```

2.6 Windows Application Programming Interface (API)

API adalah fungsi-fungsi yang terdapat pada inti sistem operasi dan dapat digunakan untuk pengembangan program lain yang berbasiskan pada sistem operasi tersebut. Dengan API, perangkat-lunak dapat melakukan hal-hal yang dapat dilakukan oleh sistem operasi seperti penanganan perangkat-keras (disket, perangkat tampilan/*display device*, *printer*, dan lain-lain), penanganan tampilan

antar-muka/GUI (*Graphical User Interface*), penanganan *file* (membuat direktori, menghapus *file*, memindahkan *file*, mengubah atribut *file*, dan lain-lain), dan lain-lain. Maka *programmer* dapat membuat program dengan lebih cepat dan mudah.

Windows adalah salah satu sistem operasi yang menyediakan API, dan kemudian dikenal dengan nama *Windows* API. Contoh sistem operasi lain yang juga menggunakan API adalah Linux. Namun karya tulis ini hanya akan membahas mengenai *Windows* API.

Windows API mulai diimplementasikan oleh *Microsoft* pada sistem operasi *Windows* 95 dan seterusnya hingga sistem operasi terbaru dari *Microsoft* yaitu *Windows* XP. Dengan menggunakan *Windows* API, *programmer* dapat membuat program yang berbasis sistem operasi *Windows* dengan lebih cepat dan mudah.

Dengan *Windows* API, perangkat-lunak yang berbasiskan pada sistem operasi *Windows* dapat melakukan berbagai hal, seperti telah disebutkan sebelumnya, bahkan termasuk penanganan *registry* (basisdata pada sistem operasi *Windows* yang berfungsi untuk menyimpan berbagai informasi konfigurasi dari sistem, seperti profil untuk setiap pengguna, informasi mengenai perangkat-keras sistem, informasi perangkat-lunak yang ter-*install*, dan lain-lain).

Microsoft sebagai pengembang dari *Windows* API, bermaksud untuk mempermudah pengembang perangkat-lunak untuk mengembangkan perangkat-lunak yang berbasiskan pada sistem operasinya. *Windows* API dapat digunakan pada sebagian besar bahasa pemrograman yang berbasis pada sistem operasi *Windows*, seperti *Microsoft* VB, *Microsoft* Visual C++, *Microsoft* VB.net,

Borland Delphi, dan lain-lain. Namun kebanyakan dari *Windows* API dikembangkan dengan bahasa *C/C++*, sehingga ada fungsi-fungsi dari *Windows* API yang memerlukan penyesuaian agar dapat digunakan dengan bahasa pemrograman yang digunakan, termasuk *Microsoft VB*.

Windows API dibuat berkelompok sesuai fungsinya, kebanyakan dikelompokkan pada suatu *file DLL (Dynamic Link Library)* yang berekstensi *.dll* dan berada pada direktori sistem dari sistem operasi *Windows* berada (pada umumnya '*C:\Windows\System*' pada sistem operasi *Windows 9x/ME/2000/XP* atau '*C:\WinNT\System32*' pada sistem operasi *Windows NT*). Berikut adalah daftar *library Windows* yang paling sering dan banyak digunakan untuk pemrograman dengan *Windows* API (Hadi, 2001):

Tabel 2.7 Daftar *file-file dll* yang sering digunakan.

Nama File DLL	Deskripsi File
Advapi32.dll	<i>Library</i> yang mendukung fungsi-fungsi keamanan dan rutin-rutin <i>registry</i> .
Comdlg32.dll	Standar kotak dialog <i>Windows</i> .
Gdi32.dll	Penanganan grafik <i>Windows</i> .
Kernel32.dll	Fungsi sistem operasi <i>Windows 32-bit</i>
Lz32.dll	Fungsi kompresi <i>file</i>
Mpr.dll	Fungsi <i>Intenet</i>
Netapi32.dll	Fungsi jaringan
Shell32.dll	<i>Library shell 32-bit</i>
User32.dll	Penanganan rutin <i>user interface</i>
Version.dll	Versi <i>Windows</i>
Winmm.dll	Fungsi-fungsi <i>multimedia Windows</i>
Winspool.drv	Fungsi-fungsi <i>printer spooler</i>

Seiring dengan perkembangan *Windows* API, dikenal dua tipe *Windows* API. Tipe-tipe tersebut adalah *Windows* API 16-bit dan *Windows* API 32-bit (*Win32* API). Terdapat perbedaan yang cukup banyak diantara keduanya, banyak fungsi dari *Windows* API versi lama (16-bit) telah disempurnakan pada versi 32-

bit, sehingga ada fungsi *Windows* API 16-bit yang tidak dapat dijalankan dengan *Win32* API. Perbedaan lain diantara keduanya adalah bahwa *Win32* API menggunakan penamaan fungsi yang bersifat *case-sensitive*, berarti *Windows* API 32-bit membedakan huruf besar dan huruf kecil untuk nama fungsinya (Hadi, 2001).

2.7 API pada VB

Windows API digunakan karena tidak semua perintah disediakan oleh VB. Meskipun VB telah menyediakan berbagai fasilitas untuk membantu *programmer* dalam menulis program, namun fasilitas tersebut masih terbatas sehingga menimbulkan kesulitan untuk mengimplementasikan ide *programmer*. Salah satu cara untuk melalui kesulitan ini adalah dengan menggunakan API yang telah disediakan oleh sistem operasi *Windows*. Contohnya adalah untuk mendapatkan bentuk *form* yang tidak kotak (seperti lingkaran), VB tidak menyediakan fungsi untuk melakukannya, sehingga diperlukan *Windows* API karena *Windows* API disertai dengan fungsi untuk membuat bagian dari *form* tidak tampak (tidak digunakan dalam program ini).

Berikut adalah *format* pemanggilan fungsi dll dengan menggunakan VB:

```
Declare Function NamaFungsi Lib "NamaLibrary" [Alias _
"AliasFungsi"] ([[ByVal/ByRef] variabel [as type][, _
[ByVal/ByRef] variabel [as type]]...) As FunctionType
```



Keterangan:

- NamaFungsi : nama dari fungsi yang ingin dipanggil. Jika fungsi merupakan *Win32* API, maka nama fungsi bersifat *case-sensitive*.
- NamaLibrary : nama *file library* yang menyimpan fungsi yang dimaksudkan, cukup dengan hanya menyebutkan mana *file* tanpa direktori karena secara *default* VB akan mengambil *file* tersebut dari direktori sistem dari sistem operasi *Windows* yang digunakan.
- AliasFungsi : nama alias dari fungsi yang dimaksudkan.
- Variabel : nama *parameter* dari fungsi.
- Type : tipe data dari *parameter*.
- FunctionType : tipe data nilai balik fungsi.

Format pemanggilan fungsi di atas akan menghasilkan fungsi yang memberikan nilai balik pada fungsi pemanggilnya. Jika nilai balik tidak diinginkan, maka *format* pemanggilan fungsi menjadi:

```
Declare Sub NamaFungsi Lib "NamaLibrary" [Alias
"AliasFungsi"] _ ([ByVal/ByRef] variabel [as type][,
[ByVal/ByRef] variabel [as _ type]]...)
```

Jika pemanggilan dilakukan dalam *form*, *module*, atau *class module*, sebelum perintah `declare` diberikan kata `Private` atau `Public` untuk mendefinisikan sifat fungsi tersebut. `Private` berarti fungsi tersebut hanya dapat dipanggil oleh fungsi/Sub dalam *form*, *module*, atau *class module* yang

mendeklarasikannya, sedangkan `Public` berarti fungsi tersebut dapat dipanggil oleh fungsi/Sub manapun dalam `project`.

Banyak fungsi yang disediakan oleh VB juga terdapat pada *Windows API*, sehingga mendapatkan hasil yang serupa (namun dikatakan bahwa *Windows API* memberikan waktu yang lebih singkat untuk memperoleh hasil yang serupa daripada menggunakan fasilitas yang disediakan VB). Tetapi pemanggilan fungsi dari *Windows API* menjadi masalah dan memungkinkan untuk timbul perilaku program yang tidak diinginkan. Untuk performa yang terbaik dan perilaku yang diinginkan, *programmer* disarankan untuk menggunakan fasilitas yang telah disediakan oleh VB jika disediakan.

2.7.1 Bit Block Transfer (BitBlt)

Perangkat-lunak permainan, seperti telah disebutkan sebelumnya, seringkali dinilai dari tampilan grafik yang dimilikinya. Namun tidak mungkin grafik yang baik tersebut didapatkan dengan menggabungkan bermacam-macam perintah grafik seperti `pset`, `line`, `circle` atau semacamnya. Maka proses pembuat grafik yang baik dapat dilakukan dengan perangkat-lunak grafis lain seperti *CorelDraw*, *Adobe Photoshop*, dan lain-lain. Dengan *Windows API*, VB memungkinkan penggunaan grafik tersebut dengan GDI (*Graphical Device Interface*). Sistem GDI adalah bagian dari *Win32 API* yang mengandung fungsi untuk menggambar dan memanipulasi tampilan visual dari *Windows* (Hakim, 2003). Salah satu API yang digunakan untuk penanganan grafik adalah `BitBlt`.

`Bitblt` adalah fungsi bagian *Windows* API yang secara cepat meng-*copy* *bitmap* (gambar) dari suatu `hDC` (*handle device context*) seperti *picturebox*, ke `hDC` yang lain (Hakim, 2003).

Karena `Bitblt` adalah fungsi dari *Win32* API, maka `Bitblt` memerlukan pendeklarasian sebelum dapat digunakan. Contoh pendeklarasian `Bitblt` (*case-sensitive*):

```
Declare Function Bitblt Lib "gdi32.dll" Alias "Bitblt" _
  (ByVal hDestDC as Long, ByVal x as Long, ByVal y as Long, _
  ByVal nWidth as Long, ByVal nHeight as Long, ByVal hSrcDC _
  as Long, ByVal xSrc as Long, ByVal ySrc as Long, ByVal _
  dwRop as Long) as Long
```

Parameter `hDestDC` adalah nama objek yang memiliki `hDC`, yang berfungsi sebagai `hDC` tujuan. Parameter `x` dan `y` adalah koordinat kartesian dari `hDC` sumber yang akan dipindahkan pada `hDC` tujuan. Kemudian `nWidth` dan `nHeight` adalah parameter lebar dan tinggi dari gambar yang akan dipindahkan pada `hDC` tujuan. Parameter `hSrcDC` adalah nama objek yang menjadi `hDC` sumber. Parameter `xSrc` dan `ySrc` adalah koordinat kartesian pada `hDC` sumber yang merupakan pojok kiri atas dari gambar akan dipindahkan. Sedangkan `dwRop` adalah *raster operation*.

Beberapa *raster operation* yang sering digunakan di antaranya:

- `vbDstInvert` : Inversi terhadap warna tujuan.
- `vbNotSrcCopy` : Menginversi sumber sebelum di-*copy* ke tujuan.
- `vbNotSrcErase` : Menginversi, hasil kombinasi sumber dan tujuan dengan OR.

- `vbSrcAnd` : Mengkombinasikan sumber dan tujuan dengan operator logika AND.
- `vbSrcCopy` : Meletakkan sumber ke tujuan secara langsung.
- `vbSrcErase` : Mengkombinasikan sumber yang diinversi dan tujuan dengan AND.
- `vbSrcInvert` : Mengkombinasikan sumber dan tujuan dengan operator logika XOR.
- `vbSrcPaint` : Mengkombinasikan sumber dan tujuan dengan operator logika OR.

Jika objek *picture box* digunakan maka properti *Autoredraw* dari objek *picture box* tersebut harus diset menjadi `True`. Jika tidak diset `True`, maka `Bitblt` tidak akan menampilkan apa-apa.

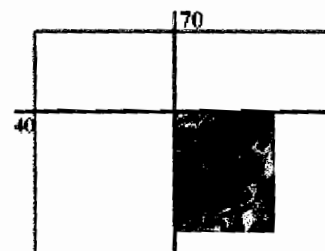
Contoh penggunaan `Bitblt`:

```
Bitblt Picture2.hDC, 70, 40, 50, 60, Picture1.hDC, 50, 20, _
vbSrcCopy
```

Akan menghasilkan tampilan seperti pada Gambar 2.7 berikut ini:



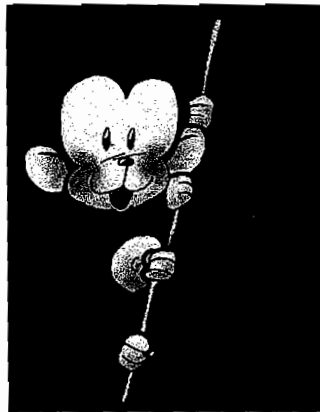
Picture box
Name : Picture1



Picture box
Name : Picture2

Gambar 2.9 Operasi `Bitblt`.

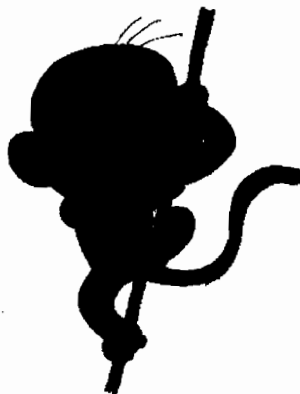
Pada kenyataannya sebuah gambar 2-D adalah sebuah matriks 2-D, dan karena matriks 2-D berbentuk kotak, maka gambar harus disimpan dalam bentuk kotak pula. Oleh karena itu, untuk membuat suatu gambar yang tidak lagi berbentuk kotak diperlukan dua buah gambar (selain gambar latar). Gambar pertama adalah gambar yang ingin ditampilkan dikenal sebagai *sprite*.



Gambar 2.10 Contoh *sprite*.

Sedangkan gambar kedua adalah gambar hitam putih yang menentukan bagian transparan dari gambar pertama disebut sebagai *mask*.

Pada umumnya *sprite* dan *mask* disatukan menjadi sebuah gambar, namun dapat juga dibuat terpisah.



Gambar 2.11 Contoh *mask*.

Metode yang digunakan dikenal dengan *masking*. Untuk ‘menyatukan’ gambar *sprite* dengan gambar latar diperlukan dua buah tahap. Pada tahap pertama dilakukan digunakan *raster operation* `vbSrcAnd` pada *mask* dan gambar latar. *Raster operation* ini menggunakan logika AND pada gambar latar dan *mask*. *Pixel* dari gambar latar (dalam biner) dioperasikan dengan piksel (*pixel*) pada *mask* (dalam biner), maka piksel dari gambar latar pada posisi yang dioperasikan akan ditampilkan bila hasil operasi adalah `True` dan ‘lubang’ berwarna hitam akan ditampilkan bila sebaliknya.

Sehingga, jika sebuah piksel pada gambar latar bernilai 11010011 dioperasikan dengan piksel yang berwarna hitam dari *mask* menjadi:

Gambar latar = 11010011
Mask = 00000000 AND
 Hasil = 00000000 (= Piksel berwarna hitam)

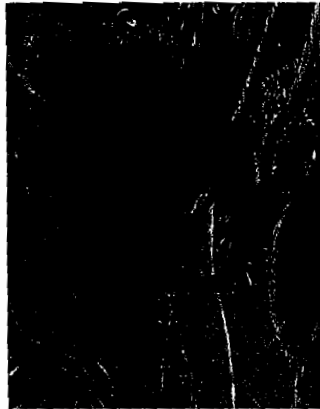
Sedangkan bila piksel dari gambar latar tersebut dioperasikan dengan piksel yang berwarna putih dari *mask*, perhitungannya menjadi:

Gambar latar = 11010011
Mask = 11111111 AND
 Hasil = 11010011 (= Piksel dari gambar latar)

Contoh perintah pada tahap ini adalah

```
Bitblt Background.hDC, 0, 0, 298, 382, Mask.hDC, 0, 0, _
vbSrcAnd
```


Contoh grafik yang diperoleh setelah tahap ini:



Gambar 2.12 Contoh gambar latar yang telah dioperasikan dengan *mask*.

Pada tahap berikutnya, gambar latar yang telah 'dilubangi' tersebut kembali dioperasikan dengan *sprite*. Namun kali ini *raster operation* yang digunakan bukan `vbSrcAnd` melainkan `vbSrcPaint`. Dengan *raster operation* ini, operasi yang dilakukan adalah operasi logika OR.

Sehingga jika piksel pada gambar latar bernilai 11010011 dioperasikan dengan piksel yang berwarna hitam dari *sprite* perhitungannya menjadi:

Gambar latar = 11010011
Sprite = 00000000 OR
 Hasil = 11010011 (= Piksel dari gambar latar)

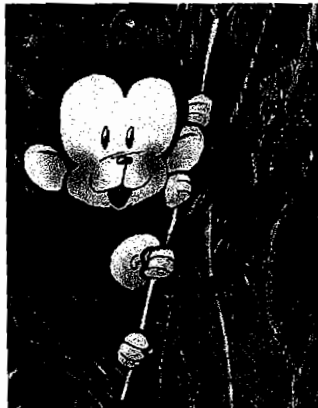
Kemudian jika piksel berwarna hitam dari gambar latar tersebut dioperasikan dengan piksel dari *sprite* yang ingin ditampilkan (misal piksel bernilai 00101110), maka perhitungannya menjadi:

Gambar latar = 00000000
Sprite = 00101110 OR
 Hasil = 00101110 (= Piksel *sprite* yang ingin ditampilkan)

Contoh perintah pada tahap ini adalah:

```
Bitblt Background.hDC, 0, 0, 298, 382, Sprit.hDC, 0, 0, _
vbSrcPaint
```

Contoh grafik yang diperoleh setelah tahap ini adalah:



Gambar 2.13 Contoh gambar latar yang telah dioperasikan dengan *sprite*.

Hasil akhir yang diperoleh adalah *sprite* yang tidak lagi berbentuk kotak, namun mengikuti bentuk *mask* yang ditentukan. Kemudian *sprite* dengan bagian transparan tersebut disatukan dengan gambar latar.

2.7.2 Menulis Nilai *String* ke dalam *File* INI (`WritePrivateProfileString`)

API ini digunakan untuk menuliskan suatu *string* ke dalam *file* INI. Jika *file* yang ditunjuk untuk ditulis tidak ada, maka API akan dengan otomatis membuat *file* yang baru. Begitu pula bila *section* tidak ditemukan, maka API akan membuat sebuah *section* yang baru. Fungsi ini akan mengembalikan nilai 0 bila terjadi kesalahan sedangkan bila berhasil, fungsi akan mengembalikan nilai selain nilai 0.

Fungsi ini dideklarasikan dengan:

```
Declare Function WritePrivateProfileString Lib "kernel32" _
Alias "WritePrivateProfileStringA" (ByVal _
lpApplicationName As String, ByVal lpKeyName As Any, ByVal _
lpString As Any, ByVal lpFileName As String) As Long
```

Keterangan:

- lpApplicationName : Nama Seksi
- lpKeyName : Nama Kunci
- lpString : *String* yang akan ditulis
- lpFileName : Nama *file* INI yang akan ditulis

Contoh penggunaan fungsi ini adalah:

```
Private Sub Command1_Click()
    Dim X as Byte
    X = WritePrivateProfileString("TEST", "Nilai", _
    "Tester Nilai String", "C:\Tester.INI")
    If X = 0 then
        MsgBox "Error: Data tidak bisa ditulis..."
    else
        MsgBox "Menulis data berhasil"
    End if
End Sub
```

Pada contoh tersebut, jika *file Tester.ini* belum ada, maka *file* tersebut akan dibuat pada *root* direktori dari drive C. Namun bila *file* tersebut sudah ada, fungsi akan mencari seksi (TEST). Jika seksi tidak ditemukan maka fungsi akan membuat seksi yang baru (TEST). Kemudian fungsi akan mencari nama kunci,

yaitu Nilai. Jika nama kunci tidak ditemukan, maka nama kunci akan dibuat. Kemudian fungsi akan menuliskan *string* yang ingin dituliskan pada *file* tersebut. Jika nama kunci ditemukan, maka fungsi akan mengubah nilai yang ada pada nama kunci tersebut. Sehingga hasil dari fungsi tersebut adalah sebuah *file* *Tester.ini* yang berisi (asumsi *file* belum ada sebelumnya):

```
[TEST]
  Nilai="Tester Nilai String"
```

2.7.3 Mengambil Nilai *String* dari *File* INI (*GetPrivateProfileString*)

Selain dapat menulis *file* INI, API juga disertai dengan fasilitas untuk mengambil nilai dari suatu *file* INI. Nilai yang dapat diambil dapat berupa nilai *integer* ataupun *string*. Untuk mengambil nilai *string* digunakan fungsi API *GetPrivateProfileString*. Jika pembacaan dari nama kunci dan seksi yang diinginkan berhasil, maka fungsi ini akan mengembalikan nilai *string*. Namun bila tidak berhasil, maka nilai yang dikembalikan adalah nilai dari *lpdefault*.

Pendeklarasian fungsi ini adalah:

```
Declare Function GetPrivateProfileString Lib "kernel32" _
  Alias "GetPrivateProfileStringA" (ByVal lpApplicationName _
  As String, ByVal lpKeyName As Any, ByVal lpDefault As _
  String, ByVal lpReturnedString As String, ByVal nSize As _
  Long, ByVal lpFileName As String) As Long
```

Keterangan:

- *lpApplicationName* : Nama Seksi
- *lpKeyName* : Nama Kunci
- *lpDefault* : Nilai *default* jika pembacaan tidak berhasil

- lpReturnedString : Nilai data *string* jika pembacaan berhasil
- nSize : Ukuran panjang *string* lpReturnedString
- lpFileName : Nama *file* INI yang akan ditulis

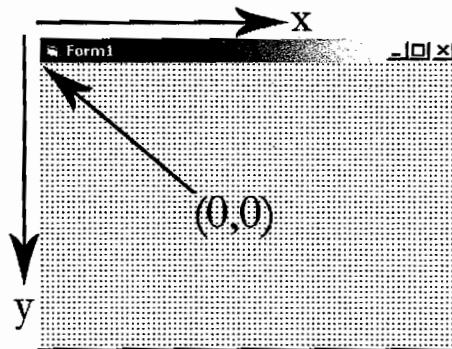
Contoh penggunaan fungsi ini adalah (Hadi, 2001):

```
Private Sub Command1_Click()
    Dim Buffer as String * 255
    Dim X as Long
    Dim SisINI, RetStr as String
    SisINI = "C:\Windows\System.INI"
    X = GetPrivateProfileString("boot", "scrnsave.exe", _
    "(error)", Buffer, 255, SisINI)
    RetStr = Left(Buffer, X)
    If RetStr = "(error)" Then
        MsgBox "Nilai Screen Saver tidak ditemukan"
    Else
        MsgBox "Screen Saver = " + RetStr
    End If
End Sub
```

2.8 Grafik pada VB

Grafik merupakan bagian yang sangat penting dalam program permainan komputer. Dengan grafik, manusia dapat berinteraksi dengan komputer. Bahkan seringkali tingkat ketertarikan manusia terhadap sebuah permainan ditentukan dari bagus atau tidaknya grafik yang digunakan dalam permainan.

VB juga mempunyai fasilitas yang cukup baik dalam menangani grafik. VB menggunakan koordinat kartesian yang sedikit berbeda, sisi vertikal dari layar disebut dengan sumbu y sedangkan sisi horizontal disebut sebagai sumbu x . Penunjukan suatu titik dalam VB menggunakan *format* yang sama dengan penunjukan suatu titik pada koordinat kartesian, yaitu dengan menyebutkan sumbu x dari titik terlebih dahulu kemudian diikuti dengan sumbu y dari titik tersebut. Sehingga pada sebuah *form* titik (0,0)-nya terdapat pada pojok kiri atas dari *form*, tidak termasuk bagian *title bar* dari *form* tersebut.



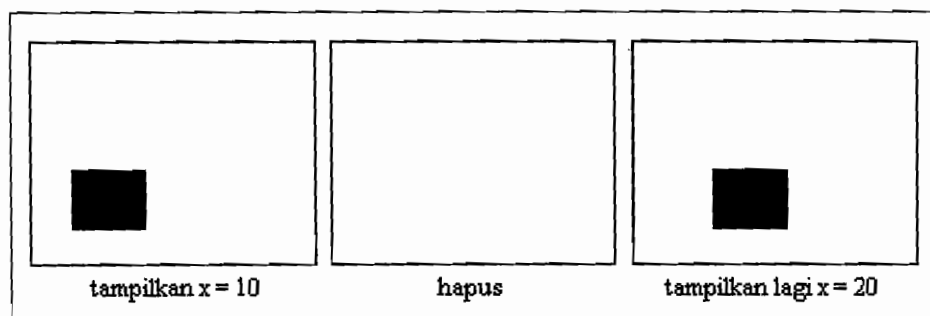
Gambar 2.14 Sistem koordinat VB.

Satuan standar dari VB adalah *twips* (1 *twips* = 1/1440 inci). Dengan menggunakan satuan ini, maka dengan resolusi berapapun suatu garis dengan panjang 1440 *twips* akan ditampilkan sepanjang 1 inci. Namun satuan *twips* mempunyai kelemahan yaitu perhitungan yang semakin rumit. Untuk itu VB juga menyediakan beberapa macam satuan lainnya, salah satunya adalah piksel. Dengan satuan piksel, resolusi layar akan sangat menentukan panjang dari suatu objek, contohnya: suatu garis dengan panjang 640 piksel akan terlihat panjang pada resolusi 640×480 namun akan terlihat pendek pada resolusi 1280×1024 (Hakim, 2003).

2.9 Dasar Animasi

Animasi mempunyai peranan yang sangat penting dalam pemrograman permainan. Dengan animasi, dapat dibuat seolah-olah objek pada layar monitor menjadi 'hidup', sehingga pemain dapat berinteraksi lebih baik dengan komputer yang digunakannya.

Terdapat banyak teknik animasi, namun teknik animasi yang paling dasar yang akan digunakan untuk membuat suatu objek terlihat bergerak pada layar monitor, yaitu dengan menampilkan gambar, menghapusnya, kemudian menampilkan kembali gambar yang sama pada lokasi yang berbeda atau dengan gambar yang berbeda. Dengan perhitungan yang baik, animasi akan ditampilkan tanpa terlihat 'patah-patah'. Istilah *sprite* juga digunakan untuk menunjuk pada objek yang 'digerakkan', sehingga dalam program permainan akan dijumpai banyak sekali *sprite*.



Gambar 2.15 Dasar animasi.

Meskipun disertai dengan proses penghapusan, namun *flicker* (layar berkedip) tidak terlihat pada layar. Karena terlalu cepat, mata manusia tidak dapat membedakan perubahan yang terjadi, bahkan animasi mungkin menjadi tidak

dapat dinikmati. Karenanya diperlukan suatu metode untuk memperlambat ‘gerakan’, yaitu dengan menggunakan waktu tunda.

Windows API juga disertai dengan fungsi untuk membuat waktu tunda, yaitu dengan menggunakan fungsi `GetTickCount`. Fungsi ini berfungsi untuk memperoleh waktu saat ini dari sistem. Untuk membuat waktu tunda pertamanya waktu saat ini disimpan kemudian dijalankan sebuah rutin perulangan yang akan berakhir jika waktu tunda yang diinginkan telah dicapai dengan membandingkan selisih waktu tunda sekarang dengan waktu tunda saat rutin dimulai.

Berikut adalah pendeklarasian fungsi `GetTickCount` beserta contoh implementasinya dalam menentukan waktu tunda:

```
Declare Function GetTickCount Lib "kernel32.dll" As Long
Sub delay(wait as Long)
    Dim lasttick as Long, currenttick as Long
    Lasttick = GetTickCount `ambil waktu terakhir
    Do
        Currenttick = GetTickCount
        Doevents `cek tugas lainnya
    Loop until (currenttick - lasttick) > wait
    `ulangi sampai waktunya sama dengan yang diinginkan.
End Sub
```

Semua animasi memiliki *frame*. *Frame* adalah istilah bagi per satuan gambar dalam sebuah animasi. Maka animasi memiliki satuan, yaitu fps (*frame per second*/jumlah *frame* per detik). Untuk dapat memperoleh animasi yang cukup halus, diperlukan animasi minimal 29,97 fps.

BAB III

ANALISIS dan RANCANGAN SISTEM

3.1 Analisis Sistem

Sistem merupakan perangkat-lunak yang bersifat *entertainment*, sehingga sistem dirancang untuk dipahami oleh penggunanya dalam waktu yang relatif singkat. Pengguna diharapkan untuk tidak mengalami kesulitan dalam mengoperasikan sistem, terutama karena sistem adalah perangkat-lunak yang bersifat menghibur dengan irama musik bukan 'memaksa' pengguna untuk banyak berpikir.

Kebanyakan perangkat-lunak permainan dirancang dengan grafik yang bagus dan dengan suara yang juga mendukung suasana, sehingga pengguna dapat lebih menikmati permainannya. Maka untuk hasil rancangan yang terbaik, digunakan perangkat-lunak profesional yang umum digunakan dalam pembuatan grafik yang baik, yaitu *CorelDRAW* 11 dan *Adobe Photoshop* 7.

Pertama-tama rancangan sistem dibuat dengan menggunakan *CorelDraw*, sehingga semua rancangan sistem sudah disertai dengan warna, tidak lagi hanya berupa ukuran dan letak tombol. Kemudian dengan menggunakan *Adobe Photoshop*, gambar hasil rancangan tersebut langsung dipilah - pilah sehingga memenuhi bentuk dan ukuran yang diinginkan saat penulisan perangkat-lunaknya dengan menggunakan VB.

Tampilan dan letak antarmuka dirancang agar tidak 'meniru' program aslinya untuk menghindari konflik.

3.2 Kebutuhan Perangkat-keras

Rancangan kebutuhan perangkat-keras minimum untuk dapat menjalankan program permainan ini adalah :

- Prosesor : Kompatibel IBM dengan teknologi MMX
- Memori : 64 MB
- Kartu Grafik : Dengan memori 4MB
- Kartu Suara : Apa saja
- *Harddisk* : 20 MB

3.3 Kebutuhan Perangkat-lunak

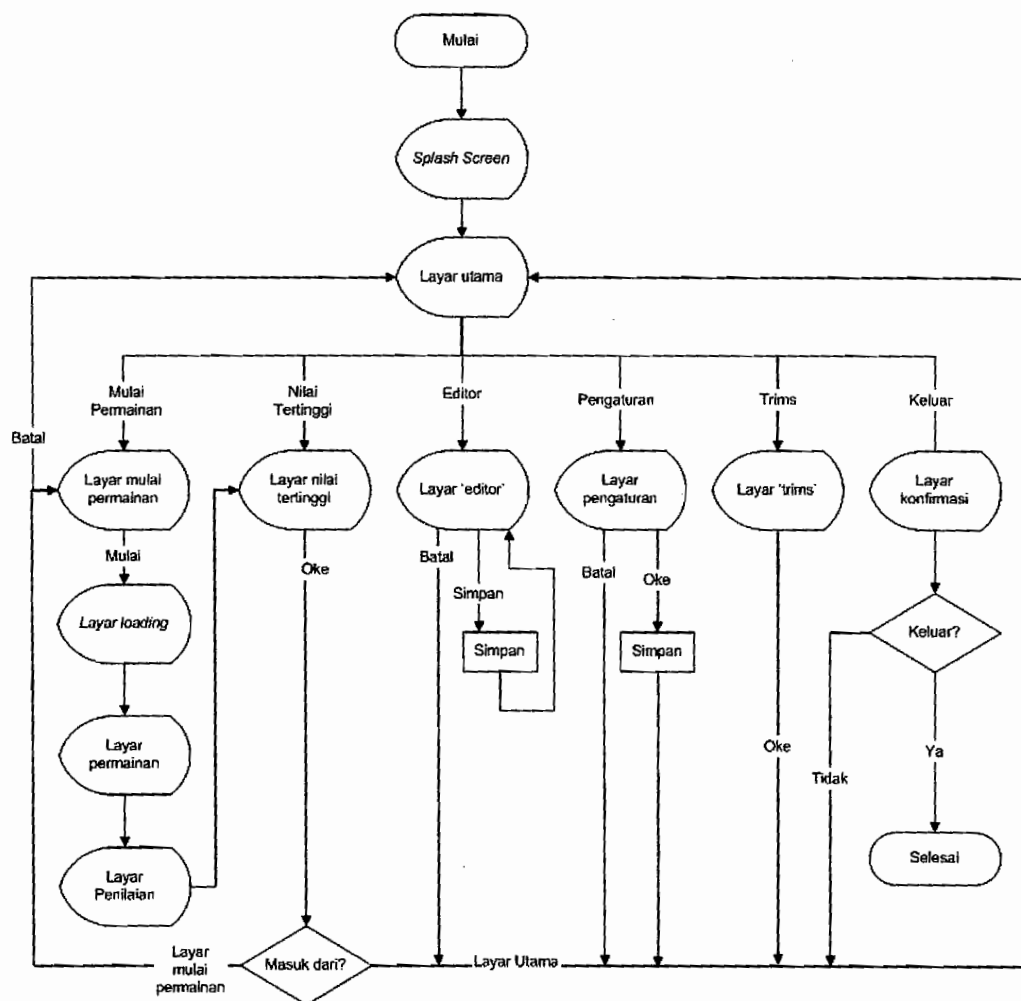
Rancangan kebutuhan perangkat-lunak minimum untuk dapat menjalankan program permainan ini adalah :

- Sistem Operasi : *Microsoft Windows 95/98/ME/2000/XP*
- *Driver* VGA : Kompatibel
- *Driver* Suara : Kompatibel
- Pengembangan : VB 6.0

3.4 Rancangan Sistem

Perangkat-lunak akan diawal dengan menampilkan splash screen. Kemudian pengguna akan dihadapkan dengan layar utama, yang akan memberikan akses ke hampir semua layar dalam perangkat-lunak, yaitu: layar 'mulai permainan', layar 'nilai tertinggi', layar 'editor', layar 'pengaturan', layar 'trims', dan konfirmasi jika pemain ingin menghentikan perangkat-lunak.

Berikut adalah diagram alir yang menjelaskan secara singkat mengenai rancangan pergantian layar yang akan ditampilkan bila pemain menekan tombol pada setiap layarnya:

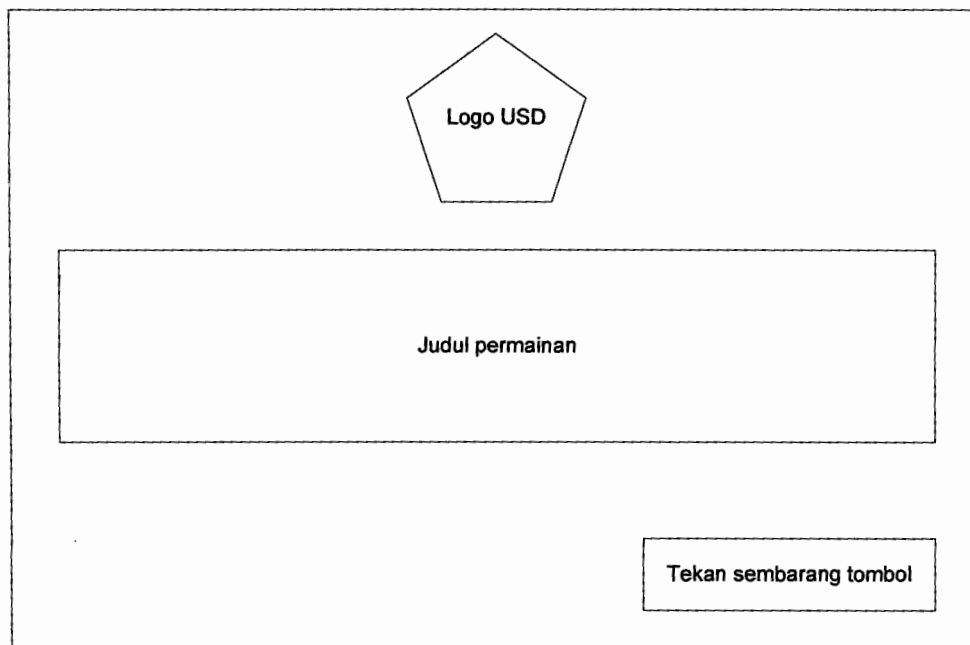


Gambar 3.1 Struktur program (diagram alir).

3.4.1 Layar *Splash Screen*

Layar ini adalah semacam layar perkenalan/sambutan kepada pemain, layar ini yang akan dilihat pemain saat pertama kali menjalankan program ini. Jika pemain tidak bersedia untuk menunggu, maka layar ini dapat dilewati dengan menekan tombol *Escape*. Kemudian pemain akan dihadapkan pada layar utama.

Berikut adalah rancangan layar *splash screen*:



Gambar 3.2 Rancangan tampilan *splash screen*.

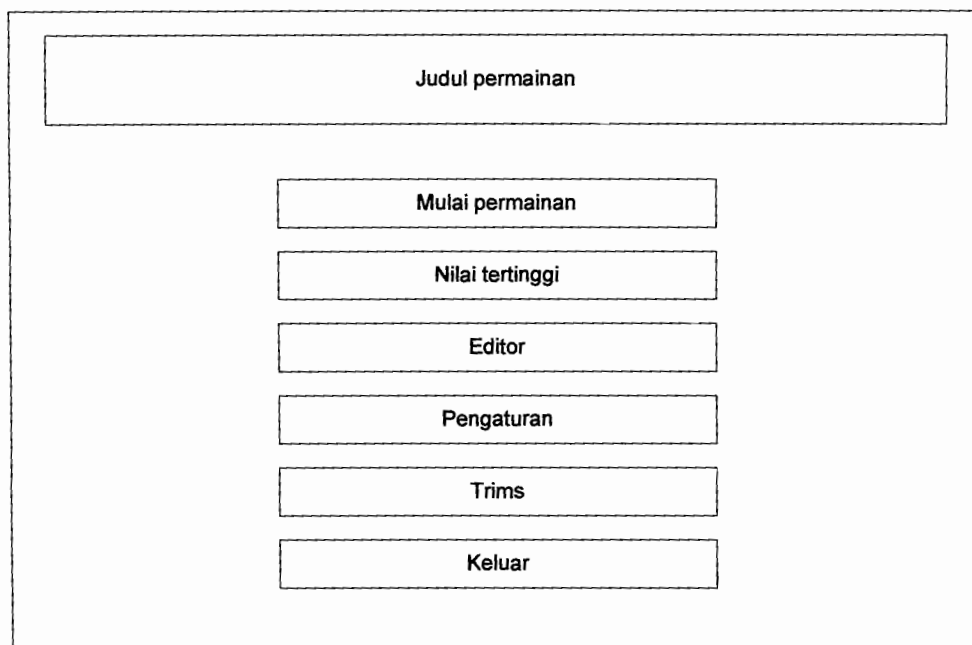
Keterangan Gambar 3.2:

- Logo USD : Letak logo USD.
- Judul permainan : Nama Judul perangkat-lunak.
- Tekan sembarang tombol merupakan pesan bagi pemain untuk menekan sembarang tombol untuk melanjutkan ke layar utama.

3.4.2 Layar Utama

Layar ini akan muncul beberapa saat setelah layar *splash screen*. Layar ini dirancang dengan menu yang dapat mengakses semua tampilan layar program, yaitu layar mulai permainan, layar nilai tertinggi, layar editor, layar pengaturan, layar trims, dan menu untuk keluar dari program. Menu ini dapat dilihat pada Gambar 3.3.

Berikut adalah rancangan tampilan layar utama:



Gambar 3.3 Rancangan tampilan layar utama.

Keterangan Gambar 3.3:

- Mulai permainan : Memulai permainan.
- Nilai tertinggi : Melihat nilai tertinggi yang telah diperoleh dari permainan sebelumnya.
- Editor : Mengubah dan membuat 'gerakan'.
- Pengaturan : Mengubah set program.

- Trims : Melihat catatan programmer.
- Keluar : Keluar dari permainan.

3.4.3 Layar Mulai Permainan

Pada layar ini, pemain diminta untuk terlebih dahulu menentukan lagu yang akan dimainkannya, tingkat kesulitan dan nama dari pemain. Kemudian, pemain dapat memulai permainannya dengan menekan tombol 'Mulai'.

Berikut adalah rancangan tampilan layar ini:

The diagram shows a game start screen layout. It consists of a large rectangular frame containing several smaller rectangular boxes. At the top is a box labeled 'Judul permainan'. Below it is a large box labeled 'Daftar lagu'. To the right of the 'Daftar lagu' box is a vertical stack of three boxes labeled 'Mudah', 'Standar', and 'Sulit', with the label 'Kesulitan' positioned above them. Below the 'Daftar lagu' box is a box labeled 'Nama pemain'. At the bottom right are two boxes labeled 'Mulai' and 'Batal'.

Gambar 3.4 Rancangan tampilan layar mulai permainan.

Keterangan Gambar 3.4:

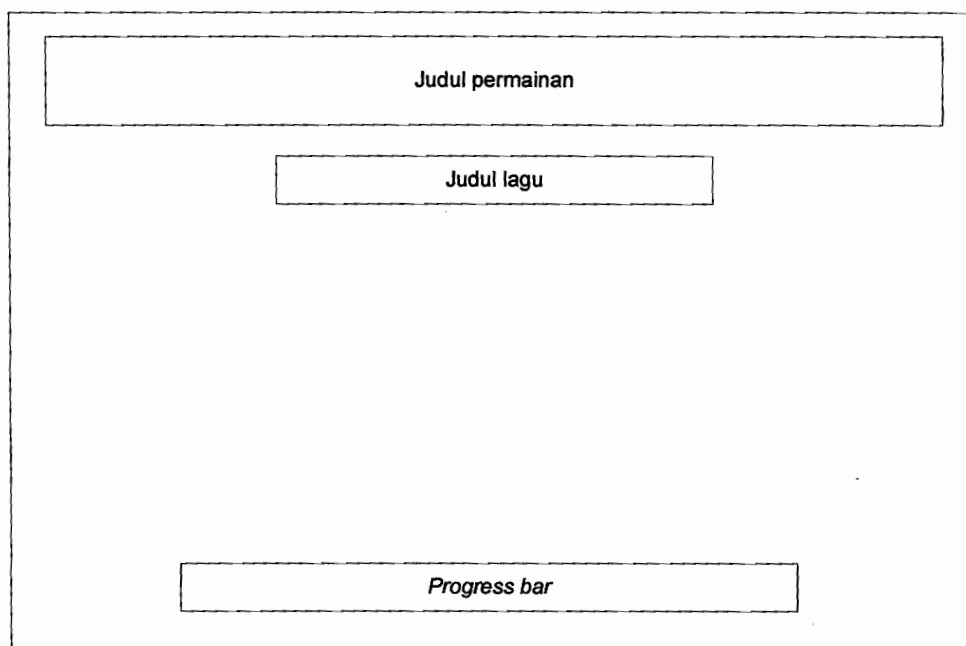
- Daftar lagu : Daftar lagu yang akan ada dalam *folder* MP3.
- Nama pemain : Memasukkan nama dari pemain, untuk dimasukkan dalam daftar nilai tertinggi.

- Kesulitan : Pemain dapat memilih tingkat kesulitan (mudah, standar, dan sulit) yang ingin dimainkan.
- Mulai : Tombol untuk memulai permainan.
- Batal : Tombol untuk kembali ke menu utama.

3.4.4 Layar *Loading*

Layar ini dirancang agar pemain dapat mengetahui posisi program dalam melakukan pembentukan terhadap data 'panah' yang dibutuhkan dalam program permainan.

Berikut adalah rancangan tampilan layar ini:



Gambar 3.5 Rancangan tampilan layar *loading*.

Keterangan Gambar 3.5:

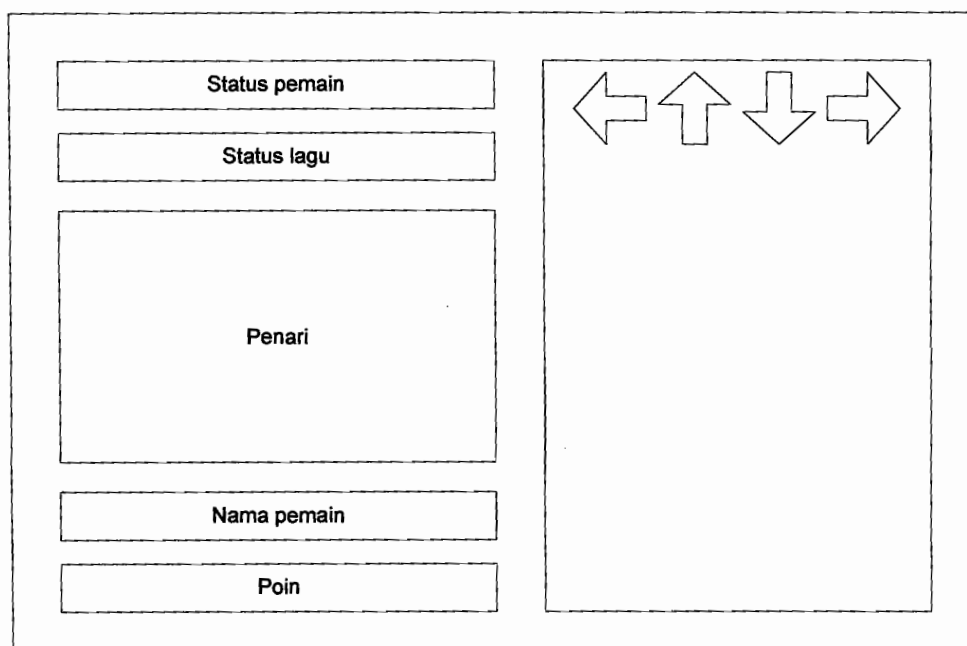
- Judul permainan : Nama perangkat-lunak.

- Judul lagu : Judul dari lagu yang akan dimainkan.
- *Progress bar* : Persentase posisi proses.

3.4.5 Layar Permainan

Layar ini adalah 'bermain', berdasarkan pada urutan 'panah' pemain mengikuti irama musik dan menekan tombol pada perangkat masukannya. Pada layar ini pemain dapat melihat statusnya selama permainan berlangsung, panjangnya lagu yang dimainkannya, nilai yang telah diperolehnya, nama pemain, dan sebuah animasi tarian.

Berikut adalah rancangan layar permainan:



Gambar 3.6 Rancangan tampilan layar permainan.

Keterangan Gambar 3.6:

- $\leftarrow \uparrow \downarrow \rightarrow$: Panah-panah' akan muncul di sini, pemain bermain dengan urutan 'panah' yang muncul dan irama lagu.

- Status pemain : Status pemain saat ini.
- Status lagu : Lamanya lagu, kosong bila baru dimulai, dan semakin penuh saat semakin mendekati akhir lagu.
- Penari : Animasi tarian ditampilkan di sini.
- Nama pemain : Nama pemain saat ini.
- Poin : Nilai yang telah diperoleh pemain dalam permainannya saat ini.

3.4.6 Layar Penilaian

Layar penilaian digunakan untuk memberitahukan pada pemain mengenai hasil permainan yang baru selesai dilakukannya. Layar ini akan memberikan peringkat bagi pemain dan memasukan nama dan nilai dari pemain ke dalam daftar nilai tertinggi bila pemain dapat masuk ke dalam peringkat 10 besar.

Berikut adalah rancangan tampilan layar ini:

The diagram shows a rectangular frame containing the following elements:

- Three horizontal input fields at the top: "Judul lagu", "Kesulitan", and "Nama pemain".
- A vertical stack of five buttons: "Sempurna", "Oke", "Nyaris", "Gagal", and "Lewat".
- A large rectangular box on the right side labeled "Nilai Huruf".
- A horizontal input field at the bottom center labeled "Total poin".
- A button at the bottom right labeled "Oke2".

Gambar 3.7 Rancangan tampilan layar penilaian.

Keterangan Gambar 3.7:

- Judul lagu : Judul lagu yang terakhir dimainkan pemain.
- Kesulitan : Tingkat kesulitan dari permainan terakhir.
- Nama pemain : Nama pemain terakhir.
- Sempuna, Oke, Nyaris, Gagal, dan Lewat adalah bagian yang akan menunjukkan persentase status masing-masing area dalam permainan terakhir.
- Total poin : Total nilai yang didapat dari permainan terakhir.
- Nilai Huruf : Nilai pemain secara garis besar.
- Oke2 : Tekan untuk melanjutkan ke layar selanjutnya.

3.4.7 Layar Nilai Tertinggi

Layar ini menampilkan nilai-nilai yang pernah diperoleh dari permainan sebelumnya. Pada layar ini, pemain dapat memilih lagu dan untuk setiap lagu terdapat pilihan tingkat kesulitan. Sehingga pemain dapat melihat nama - nama pemain yang pernah memainkan suatu lagu dan mendapatkan nilai peringkat 10 besar pada tingkat kesulitan tertentu. Hal ini dilakukan untuk 'menantang' pemain, sehingga pemain tetap memainkan program permainan ini dan berusaha mencetak namanya pada layar ini.

Berikut adalah rancangan tampilan layar ini:

Gambar 3.8 Rancangan tampilan layar nilai tertinggi.

Keterangan Gambar 3.8:

- Judul lagu : Daftar lagu.
- Kesulitan : Tingkat kesulitan yang ingin ditampilkan.
- 10 Besar : Daftar nama - nama dan nilai - nilai yang pernah diperoleh dari permainan sebelumnya berdasarkan lagu dan tingkat kesulitannya.
- Oke : Jika pemain masuk ke layar ini dari menu utama, maka pemain akan kembali ke menu utama jika menekan tombol ini. Jika pemain masuk ke layar ini setelah permainannya selesai, maka pemain akan dikembalikan ke layar mulai permainan.



3.4.8 Layar Editor

Layar ini difungsikan bagi pemain yang juga ingin mengubah atau membuat urutan 'panah'. Pemain dapat memilih lagu yang ingin diubahnya dengan memilih *file* MP3 dari lagu tersebut. Judul dari lagu dapat diubah pada bagian 'Judul Lagu'. Untuk setiap lagu, tiap tingkat kesulitan dapat memiliki urutan 'panah' yang berbeda. Maka jika pemain ingin mengubah urutan 'panah' pada tingkat kesulitan termudah, pemain terlebih dahulu memilih tingkat kesulitan termudah sebelum melakukan perubahan. Tiap tingkat kesulitan menggunakan cara yang sama untuk diubah. Pemain mengubah urutan 'panah' tersebut dari tabel di sebelah kiri layar yang dilengkapi dengan penunjuk waktu beserta tombol yang harus ditekan pemain.

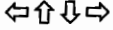
Berikut adalah rancangan tampilan layar ini:

Rancangan tampilan layar editor yang menunjukkan area kontrol panah dan daftar opsi lagu/kesulitan.

	Judul lagu
	File MP3 ...
	Kesulitan
	Mudah
	Standar
	Sulit
	Tes
	Reset
	Simpan
	Batal

Gambar 3.9 Rancangan tampilan layar editor.

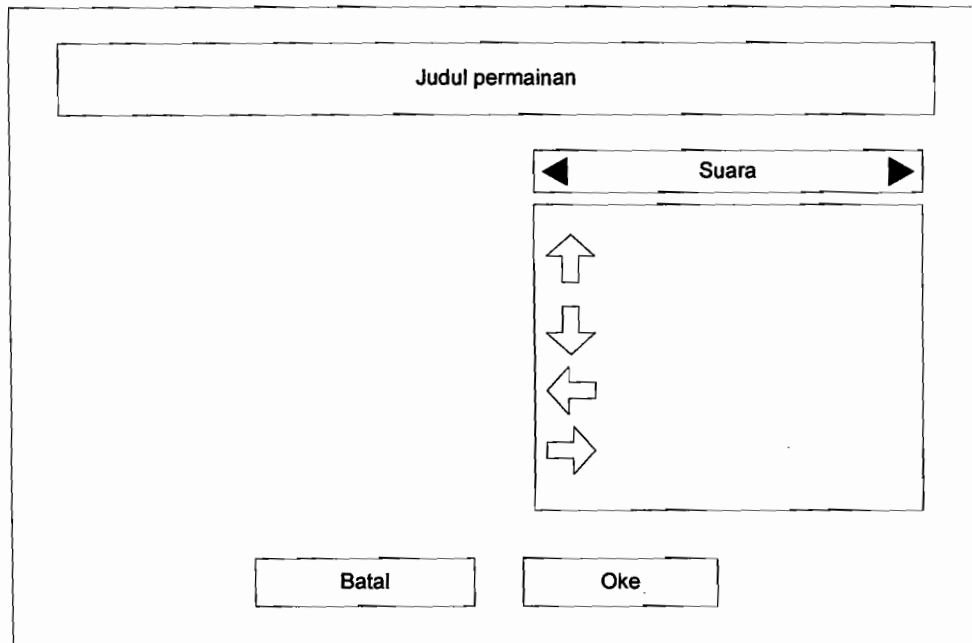
Keterangan Gambar 3.9:

-  : Tempat pemain mengubah urutan 'panah'.
- Judul lagu : Judul lagu yang ingin ditampilkan.
- File MP3 : *File* MP3 yang ingin ditambahkan dengan urutan 'panah'.
Pemain dapat menekan tombol '...' di sampingnya untuk *browse*.
- Kesulitan : Pemain dapat memilih tingkat kesulitan yang ingin diubah.
- Tes : Tombol untuk menguji perubahan urutan 'panah'.
- *Reset* : Mengembalikan keadaan seperti sebelum dilakukan perubahan/saat terakhir kali urutan 'panah' disimpan.
- Simpan : Menyimpan hasil perubahan.
- Batal : Kembali ke layar utama.

3.4.9 Layar Pengaturan

Layar ini menyediakan fasilitas pengaturan program permainan agar sesuai dengan keinginan pemain dalam hal tingkat *volume* suara dan tombol permainan. Hal ini tidak harus dilakukan oleh pemain (pemain tidak perlu melakukan pengaturan program berulang-ulang), karena program telah mempunyai nilai standar dan dapat menyimpan perubahan pengaturan yang dilakukan oleh pemain. Perubahan dilakukan dengan menekan tombol panah ◀ atau ▶ pada layar.

Berikut adalah rancangan tampilan layar ini:



Gambar 3.10 Rancangan tampilan layar pengaturan.

Keterangan Gambar 3.10:

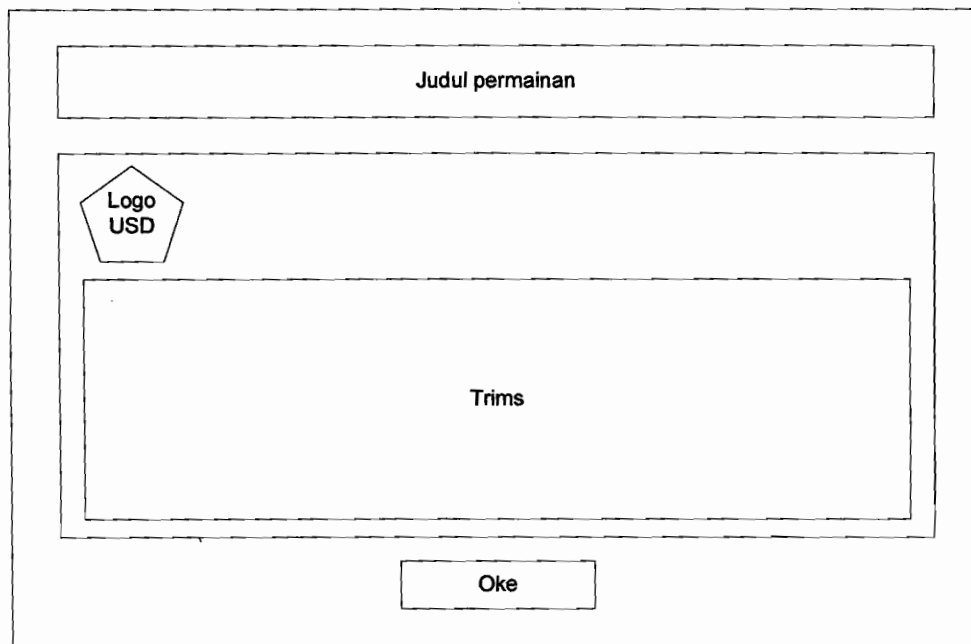
- Suara : Pemain dapat menyesuaikan tingkat *volume* suara.
- ↶↷↸↹ : Tombol yang mewakili 'panah' pada perangkat masukan yang dipilih pemain.
- Oke : Menyimpan hasil perubahan.
- Batal : Kembali ke layar utama.

3.4.10 Layar Trims

Layar ini digunakan oleh penulis untuk memberikan informasi pada pemain mengenai pembuatan program ini dan ucapan terima kasih penulis pada pihak-pihak yang telah mendukung penulis dalam pembuatan program permainan

ini. Pada layar ini juga disertakan hak yang diterima pemain atas program ini. Pemain dapat kembali ke layar utama dengan menekan tombol 'Oke'.

Berikut adalah rancangan tampilan layar ini:



Gambar 3.11 Rancangan tampilan trims.

Keterangan Gambar 3.11:

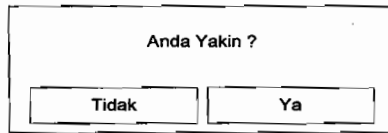
- Judul permainan : Nama perangkat-lunak.
- Logo USD : Letak logo USD.
- Trims : Layar gulung berisi informasi ucapan terima kasih.

3.4.11 Jendela Pendukung

3.4.11.1 Konfirmasi

Saat pemain menghendaki untuk mengakhiri program, maka pemain dikonfirmasi. Hal ini dilakukan agar pemain tidak keluar dari program secara tidak sengaja.

Rancangan tampilan layar konfirmasi adalah sebagai berikut :

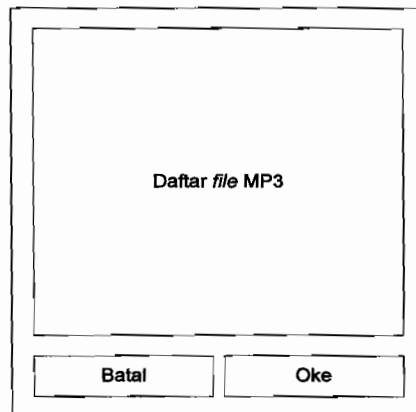


Gambar 3.12 Rancangan tampilan konfirmasi.

3.4.11.2 *Browse*

Jendela ini akan muncul ketika pemain memilih *file* MP3 yang *file* inisialisasinya ingin diubah dalam layar 'editor'.

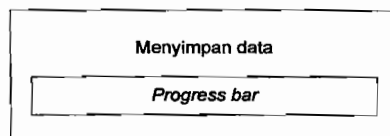
Berikut adalah rancangan layar *browse*:



Gambar 3.13 Rancangan tampilan jendela *browse*.

3.4.11.3 Menyimpan Data

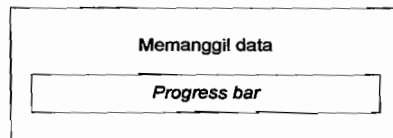
Jendela ini akan muncul ketika program sedang menyimpan data *file* INI.



Gambar 3.14 Rancangan tampilan saat menyimpan data.

3.4.11.4 Memanggil Data

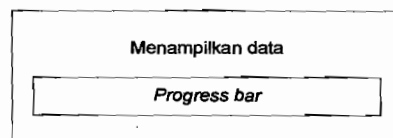
Jendela ini muncul ketika program sedang membaca data dari *file* INI.



Gambar 3.15 Rancangan tampilan saat memanggil data.

3.4.11.5 Menampilkan Data

Jendela ini muncul ketika program sedang menampilkan data dari *file* INI.



Gambar 3.16 Rancangan tampilan saat menampilkan data.

3.4.12 *File* Inisialisasi

File inisialisasi merupakan *file* tulang punggung bagi perangkat-lunak permainan ini karena *file* inisialisasi digunakan untuk menyimpan semua informasi mengenai suatu lagu. *File* inisialisasi dirancang untuk dapat menyimpan informasi mengenai:

- judul dari lagu
- daftar nilai tertinggi yang pernah diperoleh pemain pada setiap tingkat kesulitan dari suatu lagu
- urutan tombol yang harus ditekan pemain pada suatu waktu untuk setiap tingkat kesulitan.

File inialisasi dirancang memiliki nama *file* yang sama dengan nama *file* MP3nya, kecuali berbeda ekstensi.

Berikut adalah format rancangan *file* inialisasi yang digunakan:

```
[Keterangan]
Judul=Dance-Dance-Dance
[NilaiMudah]
n1=Nicko
s1=1000
[NilaiStandar]
n1=Nicko
s1=10000
[NilaiSulit]
n1=Nicko
s1=100000
[PanahMudah]
t0=0000
[PanahStandar]
t0=0100
[PanahSulit]
t0=1001
```

Bagian keterangan

Bagian nilai untuk tingkat kesulitan 'mudah', 'standar', dan 'sulit'.

Bagian kombinasi panah untuk tingkat kesulitan 'mudah', 'standar', dan 'sulit'.

Keterangan seksi:

- Keterangan : Informasi umum dari lagu
- NilaiMudah : Informasi nilai tertinggi pada tingkat kesulitan 'Mudah'.
- NilaiStandar : Informasi nilai tertinggi pada tingkat kesulitan 'Standar'.
- NilaiSulit : Informasi nilai tertinggi pada tingkat kesulitan 'Sulit'.
- PanahMudah : Informasi urutan kombinasi tombol yang harus ditekan pemain pada suatu saat untuk tingkat kesulitan 'Mudah'.
- PanahStandar : Informasi urutan kombinasi tombol yang harus ditekan pemain pada suatu saat untuk tingkat kesulitan 'Standar'.
- PanahSulit : Informasi urutan kombinasi tombol yang harus ditekan pemain pada suatu saat untuk tingkat kesulitan 'Sulit'.

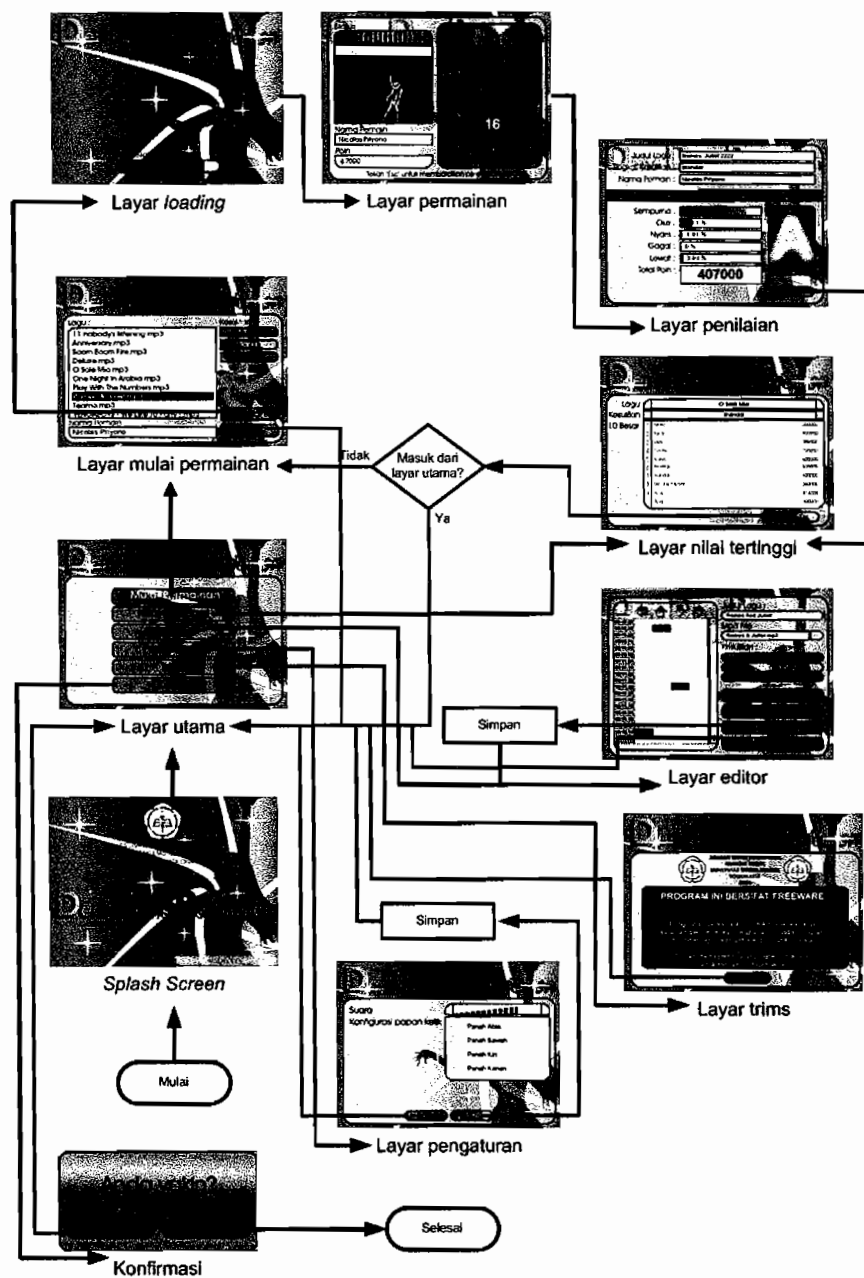
Keterangan kunci:

- Judul : Informasi judul dari suatu lagu
- nx : Bagian ini berulang sebanyak maksimal sepuluh kali untuk menunjukkan informasi 10 nama dari pemain yang menduduki peringkat ke-x dari daftar nilai tertinggi pada suatu tingkat kesulitan.
- sx : Bagian ini berulang sebanyak maksimal sepuluh kali untuk menunjukkan informasi 10 nilai dari pemain yang menduduki peringkat ke-x dari daftar nilai tertinggi pada suatu tingkat kesulitan.
- tx : Bagian ini berulang sebanyak panjangnya lagu untuk menunjukkan informasi urutan kombinasi tombol yang harus ditekan pemain pada saat ke-x dari suatu lagu.

BAB IV

IMPLEMENTASI dan HASIL PROGRAM

Berikut adalah diagram layar yang akan ditampilkan berdasarkan tombol yang ditekan oleh pemain:



Gambar 4.1 Peta layar dengan rancangan tampilan sistem.

4.1 Layar *Splash Screen*

Karena layar ini hanya merupakan pengenalan dengan pemain, maka layar ini ditampilkan untuk waktu yang singkat. Dalam waktu tertentu layar ini akan dihilangkan dan pemain akan langsung dihadapkan dengan layar berikutnya. Penanganan waktu tersebut dilakukan dengan menggunakan objek *timer* dari VB. objek *timer* yang berbeda digunakan untuk memberikan animasi teks berkedap-kedip, yaitu dengan membuat objek *picturebox* yang menyimpan gambar teks tersebut ditampilkan dan dihilangkan.

Berikut adalah tampilan implementasi layar *splash screen*:



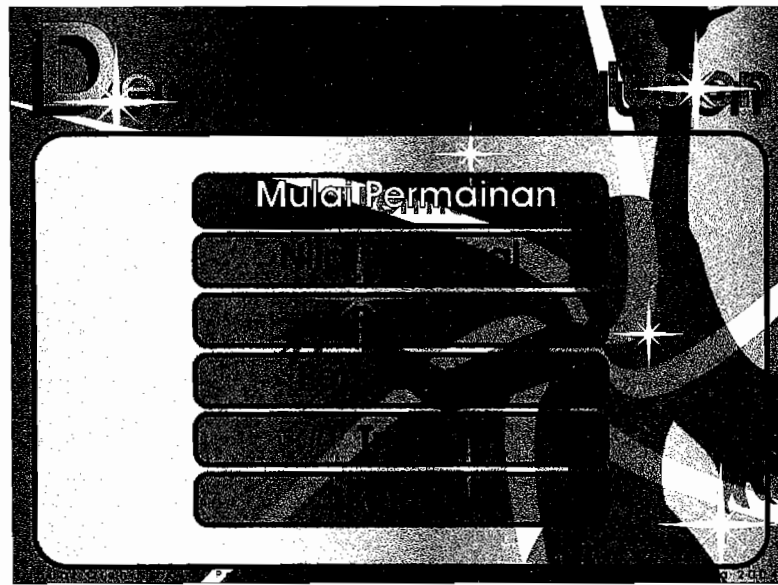
Gambar 4.2 Contoh tampilan implementasi layar *splash screen*.

4.2 Layar Utama

Sebenarnya layar utama adalah *form* yang pertama kali dipanggil, namun kemudian disembunyikan dan layar *splash screen* ditampilkan. Sehingga *form* ini melakukan beberapa hal penting seperti memanggil *file* `setting.ini`, mengubah

tampilan program menjadi *fullscreen*, dan melakukan pemeriksaan terhadap adanya kartu suara pada sistem.

Berikut adalah tampilan implementasi layar utama:



Gambar 4.3 Contoh tampilan implementasi layar utama.

Berikut adalah fungsi yang digunakan untuk melakukan pemeriksaan terhadap *file* *setting.ini*. Jika *file* tersebut tidak ditemukan, maka fungsi ini akan membuat *file* *setting.ini* dengan mode standar. Jika *file* tersebut ditemukan, maka fungsi ini akan memanggil nilai-nilai yang terdapat dalam *file* tersebut sebagai nilai dari variabel.

```
Private Sub CekSettingINI()
    Dim X As Byte
    Dim buffer As String * 255
    X = GetPrivateProfileString("Setting", "Nama", _
        "(error)(error)", buffer, 255, App.Path & "\setting.ini")
    NamaPemain = Left(buffer, X)

    If NamaPemain = "(error)(error)" Then
        'Tidak ada file setting.ini
        X = WritePrivateProfileString("Setting", "Nama", "USD", _
            App.Path & "\setting.ini")
        CekX X
        X = WritePrivateProfileString("Setting", "Volume", "70", _
            App.Path & "\Setting.ini")
    End If
End Sub
```

```

CekX X
X = WritePrivateProfileString("Setting", "Atas", "38", _
    App.Path & "\setting.ini")
CekX X
X = WritePrivateProfileString("Setting", "Bawah", "40", _
    App.Path & "\setting.ini")
CekX X
X = WritePrivateProfileString("Setting", "Kiri", "37", _
    App.Path & "\setting.ini")
CekX X
X = WritePrivateProfileString("Setting", "Kanan", "39", _
    App.Path & "\setting.ini")
CekX X
NamaPemain = "USD"
Volume = 500
Atas = vbKeyUp
Bawah = vbKeyDown
Kiri = vbKeyLeft
Kanan = vbKeyRight
Else
'ada file setting.ini
X = GetPrivateProfileString("Setting", "Volume", "70", _
    buffer, 3, App.Path & "\setting.ini")
Volume = Int(Val(Left(buffer, X)))
X = GetPrivateProfileString("Setting", "Atas", _
    Str(vbKeyUp), buffer, 5, App.Path & "\setting.ini")
Atas = Int(Val(Left(buffer, X)))
X = GetPrivateProfileString("Setting", "Bawah", _
    Str(vbKeyDown), buffer, 5, App.Path & "\setting.ini")
Bawah = Int(Val(Left(buffer, X)))
X = GetPrivateProfileString("Setting", "Kiri", _
    Str(vbKeyLeft), buffer, 5, App.Path & "\setting.ini")
Kiri = Int(Val(Left(buffer, X)))
X = GetPrivateProfileString("Setting", "Kanan", _
    Str(vbKeyRight), buffer, 5, App.Path & "\setting.ini")
Kanan = Int(Val(Left(buffer, X)))
End If
End Sub

```

Berikut adalah bagian program yang terdapat dalam *event load* dari *form*

layar utama :

```

ResolusiXAwal = GetDeviceCaps(hDC, HORZRES)
ResolusiYAwal = GetDeviceCaps(hDC, VERTRES)
BitPerPixelAwal = GetDeviceCaps(hDC, BITSPIXEL)
Init_FullScreen_DX Me

```

Ketiga baris pertama berfungsi untuk menyimpan resolusi awal sebelum program dijalankan. Hal ini dimaksudkan agar program dapat mengembalikan resolusi layar pada saat program berakhir. Baris berikutnya adalah pemanggilan

fungsi untuk mengubah resolusi layar menjadi mode *fullscreen*, yaitu dengan memanggil suatu fungsi bernama `Init_FullScreen_DX` Me.

Berikut adalah isi dari fungsi untuk mengubah mode *fullscreen*:

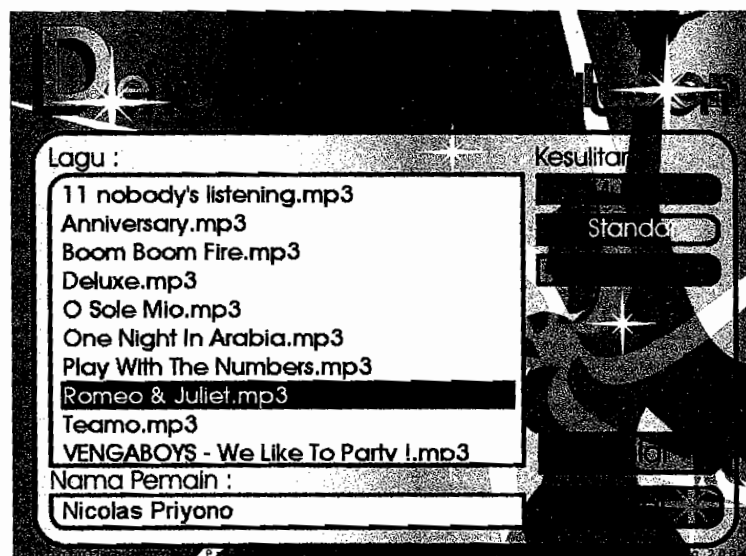
```
Public DX As New DirectX7
Public dd As DirectDraw7
Public ScreenWidth As Long
Public ScreenHeight As Long
Public ScreenDepth As Long

Sub Init_FullScreen_DX(Window As Form)
    Set dd = DX.DirectDrawCreate("")
    Call dd.SetCooperativeLevel(Window.hWnd, DDSCL_FULLSCREEN Or _
        DDSCL_EXCLUSIVE Or DDSCL_ALLOWREBOOT)
    Call dd.SetDisplayMode(DirectX.ScreenWidth, _
        DirectX.ScreenHeight, DirectX.ScreenDepth, 0, DDSDM_DEFAULT)
End Sub
```

4.3 Layar Mulai Permainan

Program akan memainkan lagu yang dipilih oleh pemain, sehingga pemain dapat memilih lagu sesuai dengan selernya. Pada layar ini juga pemain dapat memasukkan nama dan tingkat kesulitan sebelum permainan dimulai.

Berikut adalah tampilan implementasi dari layar mulai permainan:



Gambar 4.4 Contoh tampilan implementasi layar mulai permainan.

Daftar lagu menggunakan objek `FileListBox` dari Visual Basic. Namun objek ini tidak mempunyai properti untuk menghilangkan bingkainya padahal bingkai tersebut tidak diinginkan. Maka digunakan suatu objek tambahan untuk membuat bingkai dari `FileListBox` seakan-akan menghilang. Objek yang dimaksudkan adalah `Frame`. Maka `FileListBox` diletakkan di dalam `Frame`, namun posisi atas dan kiri dari `FileListBox` sedikit digeser ke kiri atas agar bingkai sisi kiri dan atas tidak terlihat. Sedangkan untuk membuat bingkai sisi kanan dan bawah tidak terlihat adalah dengan membuat lebar dan tinggi dari `FileListBox` menjadi lebih besar dari ukuran `Frame`. Sehingga tidak digunakan perintah sedikitpun untuk membuat bingkai dari `FileListBox` menghilang.

4.4 Layar Loading

Layar ini hanya menampilkan posisi proses yang sedang dilakukan oleh program. *Form* loading akan memanggil *file* INI yang ada untuk *file* MP3 yang ingin dimainkan, kemudian membentuk *array* sesuai dengan data yang terdapat pada *file* INI tersebut. Namun terdapat tambahan elemen sesuai dengan tingkat kesulitannya, hal ini dimaksudkan agar terjadi kecocokan kemunculan 'panah' dengan lagu.

Berikut adalah tampilan implementasi layar *loading*:



Gambar 4.5 Contoh tampilan implementasi layar *loading*.

Berikut adalah bagian fungsi yang digunakan untuk membentuk larik yang digunakan sebagai acuan untuk memunculkan 'panah':

```
totalpanah = 0
i = (waktutempuh * perdetik)
While i < PanjangLagu And Not batal
  DoEvents
  '+++ MENAMPILKAN PERSENTASE PROSES LOADING +++
  persentase = Int((i / PanjangLagu) * 289)
  BitBlt Status.hDC, 0, 0, persentase, 37, ButtonTemp.hDC, _
  0, 1, vbSrcCopy
  temp2 = Str(i)
  temp2 = "t" & Right(temp2, Len(temp2)-1)
  X = GetPrivateProfileString("Panah" & TempKesulitan, _
  temp2, "0", buffer, 255, App.Path & "\ini\" & temp)
  temp2 = Left(buffer, 4)
  '+++ NILAI STANDAR BAGI ATRIBUT PANAH +++
  With ArrayPanah(i)
    .Tipepanah1 = 0
    .Tipepanah2 = 0
    .Aktif = True
    .Over = False
    .Status = 0
    .Y = 480
    .waktu = 0.1 * i 'detik
  End With
  If InStr(temp2, "1") Then
    totalpanah = totalpanah + 1
  End If
End While
```

```

'+++ MENGUBAH ATRIBUT PANAH +++
For j = 1 To 4
  If Left(temp2, 1) = "1" Then
    With ArrayPanah(i)
      If .Tipepanah1 = 0 Then .Tipepanah1 = j _
      Else .Tipepanah2 = j
    End With
  End If
  temp2 = Right(temp2, 4-j)
Next j
'+++ REFRESHING PROGRESS BAR +++
Status.Refresh
Delay (2)
i = i + 1
Wend

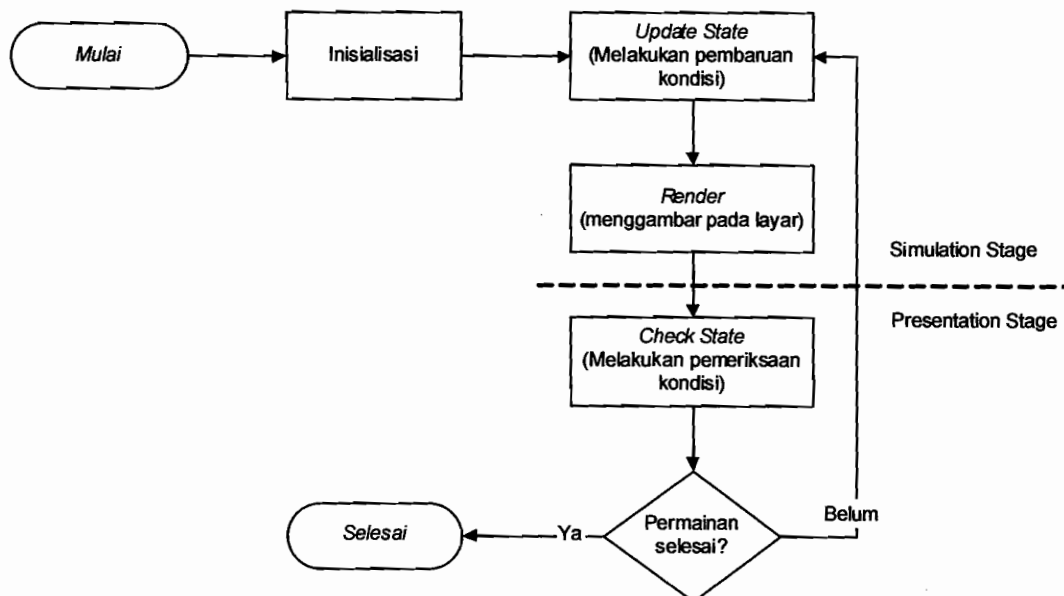
```

4.5 Layar Permainan

Layar ini mengalami beberapa perubahan dari rancangan awalnya untuk mendukung tampilan yang lebih baik. Layar ini merupakan layar utama dalam program, sehingga konsentrasi perintah program banyak terdapat pada *form* ini.

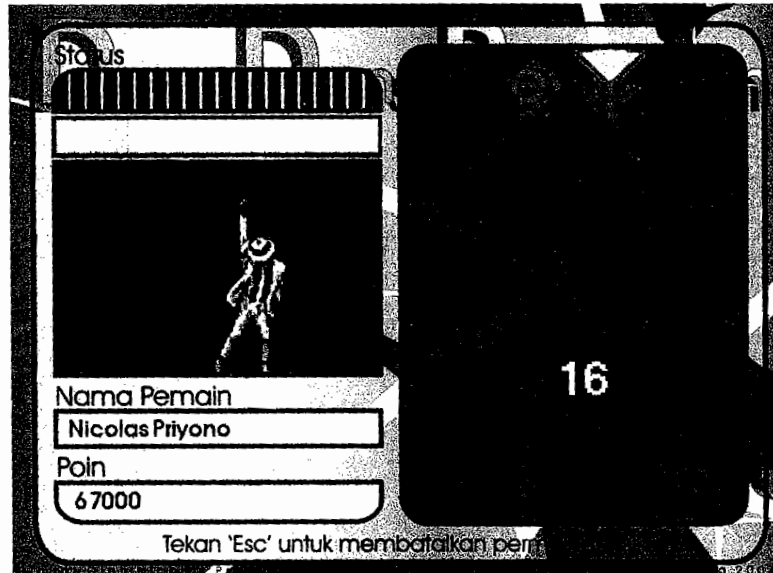
Untuk mendapatkan tampilan permainan yang diinginkan, digunakan suatu perulangan yang umum digunakan dalam pembuatan program permainan.

Berikut adalah diagram alir dari perulangan tersebut:



Gambar 4.6 Diagram alir algoritma yang digunakan (Majalah Level, 2003).

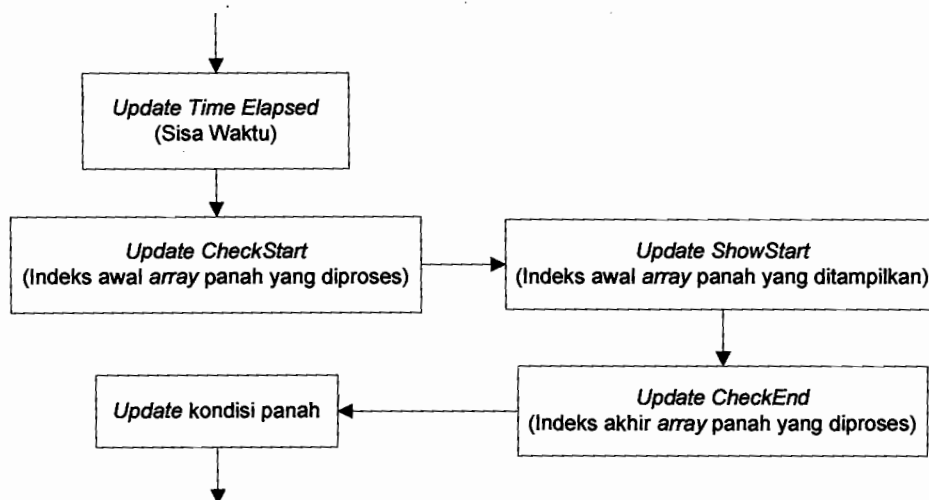
Berikut adalah tampilan implementasi layar permainan:



Gambar 4.7 Contoh tampilan implementasi layar permainan.

Pada tahap *update state*, dilakukan pembaruan kondisi permainan. Kondisi-kondisi tersebut adalah: memperbarui sisa waktu permainan, memperbarui indeks *array* panah yang akan diproses, dan memperbarui keadaan dari setiap elemen *array* panah tersebut.

Berikut adalah diagram alir untuk tahap *update state*:



Gambar 4.8 Diagram alir algoritma tahap *update state*.

Tahap *Render* berfungsi untuk melakukan perubahan terhadap tampilan layar. Tahap ini mengubah kondisi menjadi tampilan visual yang dapat dilihat pemain. Dalam setiap perulangannya, tahap ini melakukan banyak sekali perubahan pada layar untuk mendapatkan gambar yang diinginkan.

Untuk mendapatkan tampilan seperti pada Gambar 4.7, digunakan beberapa kali proses gambar pada layar dan menggunakan beberapa objek (*picturebox*) untuk mempermudah proses pemrograman baik yang tampak bagi pemain maupun yang tidak tampak. *Picturebox* yang tidak tampak dimaksudkan sebagai penampung sementara untuk diproses lebih lanjut. Sehingga terdapat 6 objek yang ditampilkan pada pemain, yaitu *form* permainan itu sendiri, *picturebox* status pemain, *picturebox* status lagu, *picturebox* penari, *picturebox* nama pemain, dan *picturebox* nilai pemain.

Gambar utama (gambar latar) diletakkan pada *form* permainan dan nama pemain hanya dilakukan satu kali. Sedangkan gambar status pemain, status lagu, nilai pemain, penari dan 'area panah' selalu berubah sehingga dilakukan beberapa kali peletakan gambar.

Untuk status pemain dan nilai pemain, dilakukan dua kali peletakan gambar, yaitu gambar untuk membersihkan objek dari keadaan sebelumnya dan gambar yang baru yang menunjukkan keadaan saat ini.

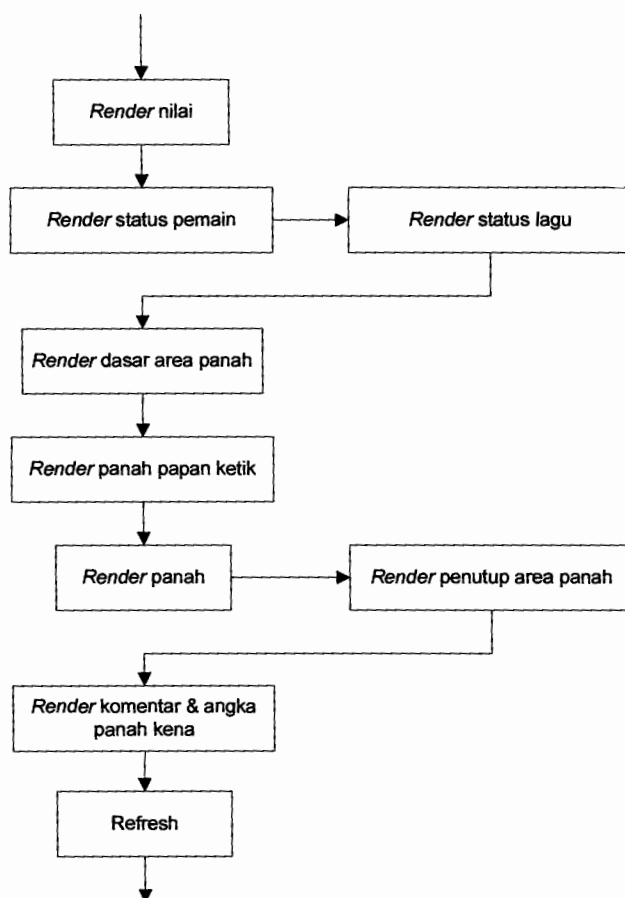
Untuk status lagu dan penari, dilakukan satu kali peletakan gambar, karena gambar yang baru tidak akan dipengaruhi oleh gambar pada keadaan sebelumnya.

Proses rumit adalah dalam menggambar 'area panah', karena terjadi banyak sekali peletakan gambar. Gambar-gambar tersebut adalah gambar latar

'area panah', gambar dari 'panah' tombol papan ketik, gambar dari *mask* 'panah', gambar 'panah', gambar penutup batas 'area panah' bagian atas, gambar penutup batas 'area panah' bagian bawah, gambar *mask* komentar, gambar komentar, dan gambar angka 'panah kena'. Khusus untuk gambar *mask* 'panah' dan gambar 'panah' dilakukan berulang-ulang sesuai dengan jumlah 'panah' yang ditampilkan pada layar.

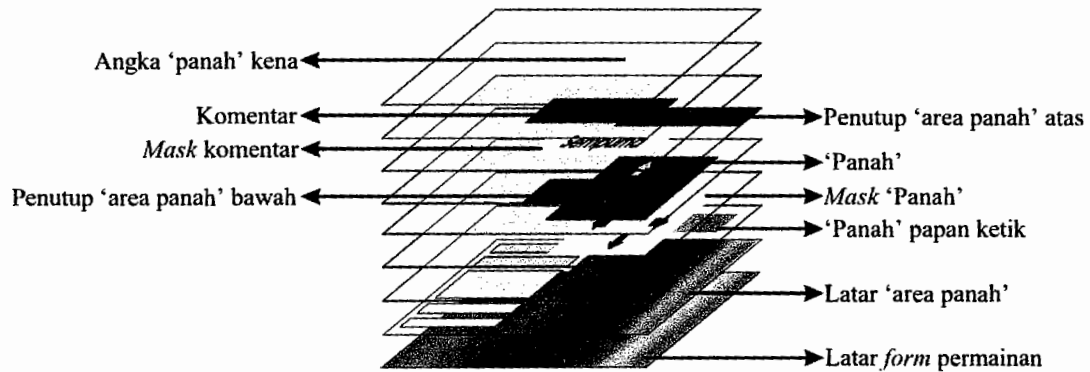
Windows API digunakan untuk 'menempelkan' gambar pada layar karena kecepataannya. Sehingga pemain tidak merasakan bahwa layar mengalami puluhan kali peletakan gambar untuk menghasilkan suatu animasi.

Berikut adalah diagram alir tahap *render*:



Gambar 4.9 Diagram alir algoritma tahap *render*.

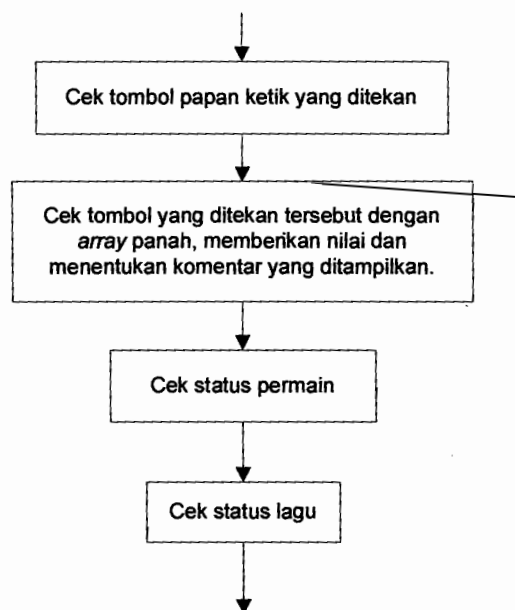
Berikut adalah ilustrasi peletakan gambar pada 'area panah':



Gambar 4.10 Ilustrasi peletakan gambar layar permainan.

Sedangkan pada tahap *check state*, dilakukan pemeriksaan tombol panah yang ditekan pemain terhadap panah yang terdapat pada *array*, memberikan nilai, menentukan komentar, dan memeriksa kondisi permainan berakhir.

Berikut adalah digram alir dari tahap *check state*:



Gambar 4.11 Diagram alir algoritma tahap *check state*.

Berikut merupakan bagian program dari proses *check state*, yang digunakan untuk menyimpan tombol papan ketik yang ditekan :

```

If Not KeyboardPressed Then
  If (GetKeyState(Kiri) And KEY_DOWN) Then
    LastKeyboardKey(0) = Kiri
  End If
  If (GetKeyState(Atas) And KEY_DOWN) Then
    LastKeyboardKey(0) = Atas
  End If
  If (GetKeyState(Bawah) And KEY_DOWN) Then
    LastKeyboardKey(0) = Bawah
  End If
  If (GetKeyState(Kanan) And KEY_DOWN) Then
    LastKeyboardKey(0) = Kanan
  End If
Else
  If LastKeyboardKey(1) = 0 Then
    If (GetKeyState(Kiri) And KEY_DOWN) Then
      If Kiri <> LastKeyboardKey(0) Then
        LastKeyboardKey(1) = Kiri
      End If
    End If
    If (GetKeyState(Atas) And KEY_DOWN) Then
      If Atas <> LastKeyboardKey(0) Then
        LastKeyboardKey(1) = Atas
      End If
    End If
    If (GetKeyState(Bawah) And KEY_DOWN) Then
      If Bawah <> LastKeyboardKey(0) Then
        LastKeyboardKey(1) = Bawah
      End If
    End If
    If (GetKeyState(Kanan) And KEY_DOWN) Then
      If Kanan <> LastKeyboardKey(0) Then
        LastKeyboardKey(1) = Kanan
      End If
    End If
  End If
End If
If LastKeyboardKey(0) <> 0 Or LastKeyboardKey(1) <> 0 Then
  KeyboardPressed = True
End If

```

Bagian program berikut digunakan untuk melakukan pemeriksaan terhadap tombol papan ketik yang ditekan dengan lokasi 'panah':

```

gain = 0
If Not (ArrayPanah(CekStart).Tipepanah1 = 0 And _
ArrayPanah(CekStart).Tipepanah2 = 0) And _
ArrayPanah(CekStart).Status = 0 Then
  'tipepanah1 tdk 0 dan tipepanah2 tdk 0
  If ArrayPanah(CekStart).Y <= Xlimit-tinggiPanah Then

```



```

'tidak menekan tombol sampai batas panah
ArrayPanah(CekStart).Status = 1
CommentNumber = 1
CommentTime = DefaultCommentTime
gain = gainlewat
If countsempurna > 10 Then
    PlaySound "oow"
End If
countsempurna = 0
countgagal = countgagal + 1
commentsound = False
ArrayPanah(CekStart).Over = True
ElseIf ArrayPanah(CekStart).Y >= Xlimit-(jarakoke +
jaraknyaris) And ArrayPanah(CekStart).Y <= Xlimit +
jaraksempurna + jarakoke + jaraknyaris + jarakgagal And
keyboardhitarrow = False Then
    'berada dalam hit are
    For j = 0 To 1
        Select Case LastKeyboardKey(j)
            Case Kiri
                panah(j) = 1
            Case Atas
                panah(j) = 2
            Case Bawah
                panah(j) = 3
            Case Kanan
                panah(j) = 4
            Case Else
                panah(j) = 0
        End Select
    Next j
    If ArrayPanah(CekStart).Tipepanah1 <> 0 And
ArrayPanah(CekStart).Tipepanah2 <> 0 Then
        'double arrow
        If (ArrayPanah(CekStart).Tipepanah1 = panah(0) And
ArrayPanah(CekStart).Tipepanah2 = panah(1)) Or
(ArrayPanah(CekStart).Tipepanah1 = panah(1) And
ArrayPanah(CekStart).Tipepanah2 = panah(0)) Then
            'sempurna
            If ArrayPanah(CekStart).Y >= Xlimit Then
                ArrayPanah(CekStart).Status = 5
                ArrayPanah(CekStart).Aktif = False
                poin = poin + hitsempurna
                gain = gainsempurna * 2
                countsempurna = countsempurna + 1
                countgagal = 0
            End If
            'Oke
            If ArrayPanah(CekStart).Y >= Xlimit +
jaraksempurna Or ArrayPanah(CekStart).Y <
Xlimit Then
                ArrayPanah(CekStart).Status = 4
                ArrayPanah(CekStart).Aktif = False
                poin = poin + hitoke
                gain = gainoke * 2
                countsempurna = countsempurna + 1
            End If
        End If
    End If
End If

```

```

        countgagal = 0
    End If
    'Nyaris
    If ArrayPanah(CekStart).Y >= Xlimit + _
    jaraksempurna + jarakoke Or _
    ArrayPanah(CekStart).Y < Xlimit-jarakoke Then
        ArrayPanah(CekStart).Status = 3
        poin = poin + hitnyaris
        gain = gainnyaris * 2
        If countsempurna > 10 Then
            PlaySound "oow"
        End If
        countsempurna = 0
        countgagal = 0
    End If
    'Gagal
    If ArrayPanah(CekStart).Y >= Xlimit + _
    jaraksempurna + jarakoke + jaraknyaris Or _
    ArrayPanah(CekStart).Y < Xlimit-jarakoke-_
    jaraknyaris Then
        ArrayPanah(CekStart).Status = 2
        poin = poin + hitgagal
        gain = gaingagal * 2
        If countsempurna > 10 Then
            PlaySound "oow"
        End If
        countsempurna = 0
        countgagal = countgagal + 1
    End If
    CommentTime = DefaultCommentTime
    CommentNumber = ArrayPanah(CekStart).Status
    CekStart = CekStart + 1
    keyboardhitarrow = True
    commentsound = False
End If
Else
    'single arrow
    If ArrayPanah(CekStart).Tipepanah1 = panah(0) Or _
    ArrayPanah(CekStart).Tipepanah1 = panah(1) Then
        'sempurna
        If ArrayPanah(CekStart).Y >= Xlimit Then
            ArrayPanah(CekStart).Status = 5
            ArrayPanah(CekStart).Aktif = False
            poin = poin + hitsempurna
            gain = gainsempurna
            countsempurna = countsempurna + 1
            countgagal = 0
        End If
        'Oke
        If ArrayPanah(CekStart).Y >= Xlimit + Or _
        ArrayPanah(CekStart).Y < Xlimit Then
            ArrayPanah(CekStart).Status = 4
            ArrayPanah(CekStart).Aktif = False
            poin = poin + hitoke
            gain = gainoke
            countsempurna = countsempurna + 1
        End If
    End If
End If

```

```

        countgagal = 0
    End If
    'Nyaris
    If ArrayPanah(CekStart).Y >= Xlimit + _
    jaraksempurna + jarakoke Or _
    ArrayPanah(CekStart).Y < Xlimit-jarakoke Then
        ArrayPanah(CekStart).Status = 3
        poin = poin + hitnyaris
        gain = gainnyaris
        If countsempurna > 10 Then
            PlaySound "oow"
        End If
        countsempurna = 0
        countgagal = 0
    End If
    'Gagal
    If ArrayPanah(CekStart).Y >= Xlimit + _
    jaraksempurna + jarakoke + jaraknyaris Or _
    ArrayPanah(CekStart).Y < Xlimit-jarakoke-__
    jaraknyaris Then
        ArrayPanah(CekStart).Status = 2
        poin = poin + hitgagal
        gain = gaingagal
        If countsempurna > 10 Then
            PlaySound "oow"
        End If
        countsempurna = 0
        countgagal = countgagal + 1
    End If
    CommentTime = DefaultCommentTime
    CommentNumber = ArrayPanah(CekStart).Status
    CekStart = CekStart + 1
    keyboardhitarrow = True
    commentsound = False
    End If
End If
End If
Else
    CekStart = CekStart + 1
End If
If CekStart > PanjangLagu Then CekStart = PanjangLagu
Statuspemain = Statuspemain + gain
CommentTime = CommentTime-1

```

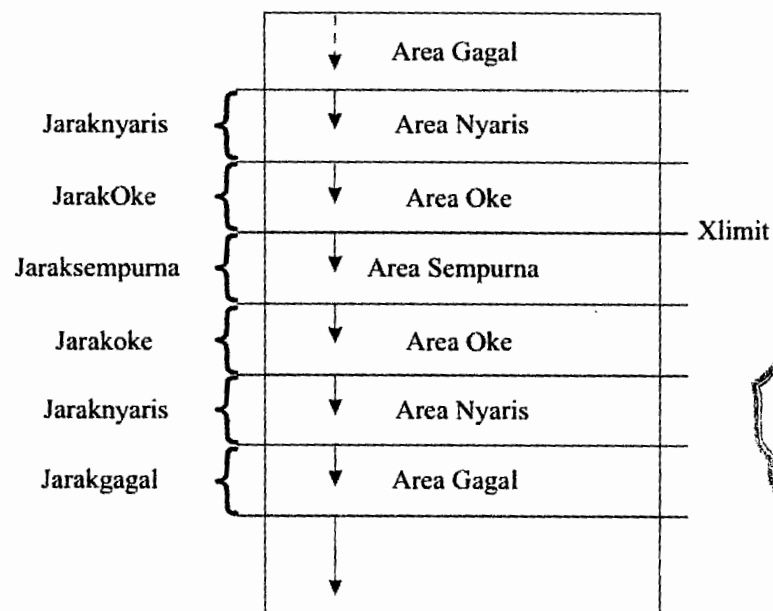
Pertama-tama diperiksa adanya 'panah' yang belum memiliki status, sehingga hanya 'panah' yang memiliki status yang akan diproses.

Kemudian diperiksa lokasi dari panah, jika melebihi batas permainan (keluar dari layar terlalu jauh), maka status 'panah' akan dianggap sebagai 'lewat'. Jika 'panah' belum masih dalam area permainan dan penekanan papan

ketik yang terakhir tidak mengenai 'panah', maka dilakukan suatu proses konversi dari penekanan tombol pada papan ketik ke model tipe panah.

Kemudian diperiksa apakah 'panah' yang muncul merupakan 'panah ganda' atau 'panah tunggal'. Jika tipe 'panah' cocok dengan tombol papan ketik yang ditekan dan 'panah' berada pada area 'panah' dapat ditekan, maka dilakukan pencocokan lokasi panah dengan nilai yang akan diperoleh pemain. Setiap tombol papan ketik ditekan pada saat terdapat 'panah' pada area 'panah' dapat ditekan, maka program akan mengakumulasi area posisi 'panah' untuk kemudian memberikan komentar dan angka 'panah kena', menambahkan nilai, dan mengubah status pemain.

Berikut adalah ilustrasi area yang digunakan dalam program untuk pemberian nilai:

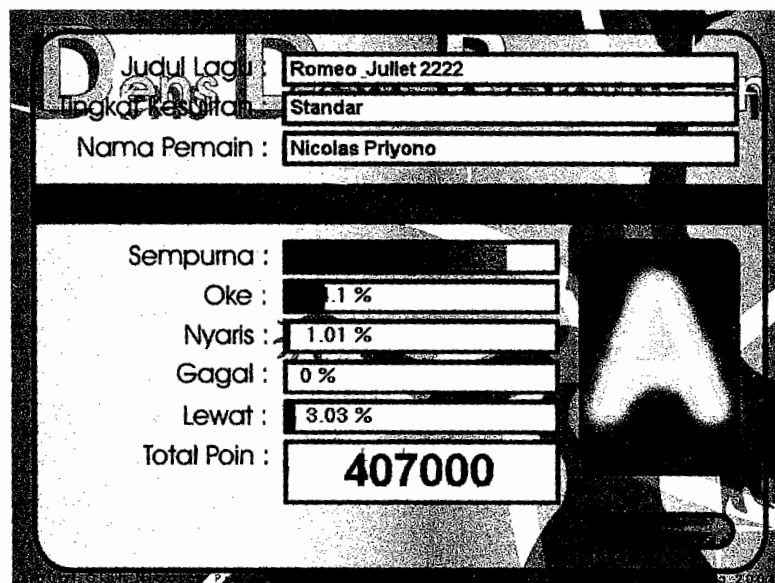


Gambar 4.12 Ilustrasi area.

4.6 Layar Penilaian

Karena layar ini berfungsi untuk menampilkan hasil permainan yang telah dilakukan, maka *form* ini melakukan perhitungan terhadap semua 'panah' yang ada. Kemudian menampilkan hasil perhitungannya pada layar sehingga pemain dapat mengetahui nilai yang diperoleh dan detilnya. Pada bagian akhir dari perhitungan, pemain akan berikan komentar dari program seperti suara yang menandakan keberhasilan ataupun kegagalan permainannya.

Berikut adalah tampilan implementasi layar penilaian:



Gambar 4.13 Contoh tampilan implementasi layar penilaian.

Berikut adalah fungsi yang digunakan untuk melakukan perhitungan tersebut:

```
Sub hitungsatusatu()
  While i < PanjangLagu And Not selesaihitung
    DoEvents
    If ArrayPanah(i).Status <> 0 Then
      BitBlt Bar(5-ArrayPanah(i).Status).hDC, 0, 0, 223, _
        23, ButtonTemp.hDC, 0, 0, vbSrcCopy
      Bar(5-ArrayPanah(i).Status).CurrentX = 10
      Bar(5-ArrayPanah(i).Status).CurrentY = 0
      Bar(5-ArrayPanah(i).Status).FontSize = 12
      Bar(5-ArrayPanah(i).Status).FontBold = True
```

```

Select Case ArrayPanah(i).Status
Case 1: 'lewat
    barlewat = barlewat + 1
    BitBlt Bar(4).hDC, 0, 0, _
        (barlewat * Bar(4).ScaleWidth \ totalpanah), 23, _
    ButtonTemp.hDC, 0, 24, vbSrcCopy
    Bar(5-ArrayPanah(i).Status).Print
    Left(Str((barlewat / totalpanah) * 100), 5) & " %"
Case 2: 'gagal
    bargagal = bargagal + 1
    BitBlt Bar(3).hDC, 0, 0, _
        (bargagal * Bar(3).ScaleWidth \ totalpanah), 23, _
    ButtonTemp.hDC, 0, 24, vbSrcCopy
    Bar(5-ArrayPanah(i).Status).Print
    Left(Str((bargagal / totalpanah) * 100), 5) & " %"
Case 3: 'nyaris
    barnyaris = barnyaris + 1
    BitBlt Bar(2).hDC, 0, 0, _
        (barnyaris * Bar(2).ScaleWidth \ totalpanah), 23, _
    ButtonTemp.hDC, 0, 24, vbSrcCopy
    Bar(5-ArrayPanah(i).Status).Print
    Left(Str((barnyaris / totalpanah) * 100), 5) & " %"
Case 4: 'oke
    baroke = baroke + 1
    BitBlt Bar(1).hDC, 0, 0, _
        (baroke * Bar(1).ScaleWidth \ totalpanah), 23, _
    ButtonTemp.hDC, 0, 24, vbSrcCopy
    Bar(5-ArrayPanah(i).Status).Print
    Left(Str((baroke / totalpanah) * 100), 5) & " %"
Case 5: 'sempurna
    barsempurna = barsempurna + 1
    BitBlt Bar(0).hDC, 0, 0, _
        (barsempurna * Bar(0).ScaleWidth \ totalpanah), _
    23, ButtonTemp.hDC, 0, 24, vbSrcCopy
    Bar(5-ArrayPanah(i).Status).Print
    Left(Str((barsempurna / totalpanah) * 100), 5) & _
    " %"
End Select
Bar(5-ArrayPanah(i).Status).Refresh
PlaySound2 "count"
Grading
End If
i = i + 1
Delay 4
Wend
Audiance
selesaihitung = True
End Sub

```

Berikut juga adalah fungsi untuk melakukan perhitungan yang sama, namun hanya akan dilakukan bila pemain menekan tombol *Escape* untuk melewati proses perhitungan dan langsung melihat hasilnya:

```

Sub hitungsemua()
  While i < PanjangLagu
    If ArrayPanah(i).Status <> 0 Then
      Select Case ArrayPanah(i).Status
        Case 1: 'lewat
          barlewat = barlewat + 1
        Case 2: 'gagal
          bargagal = bargagal + 1
        Case 3: 'nyaris
          barnyaris = barnyaris + 1
        Case 4: 'oke
          baroke = baroke + 1
        Case 5: 'sempurna
          barsempurna = barsempurna + 1
      End Select
    End If
    i = i + 1
  Wend
  For i = 0 To 4
    BitBlt Bar(i).hDC, 0, 0, 223, 23, ButtonTemp.hDC, 0, _
    0, vbSrcCopy
    Bar(i).CurrentX = 10
    Bar(i).CurrentY = 0
    Bar(i).FontSize = 12
    Bar(i).FontBold = True
  Next i

  'lewat
  BitBlt Bar(4).hDC, 0, 0, (barlewat * Bar(4).ScaleWidth \
totalpanah), 23, ButtonTemp.hDC, 0, 24, vbSrcCopy
  Bar(4).Print Left(Str((barlewat / totalpanah) * 100), 5) & _
  " %"
  'gagal
  BitBlt Bar(3).hDC, 0, 0, (bargagal * Bar(3).ScaleWidth \
totalpanah), 23, ButtonTemp.hDC, 0, 24, vbSrcCopy
  Bar(3).Print Left(Str((bargagal / totalpanah) * 100), 5) & _
  " %"
  'nyaris
  BitBlt Bar(2).hDC, 0, 0, (barnyaris * Bar(2).ScaleWidth \
totalpanah), 23, ButtonTemp.hDC, 0, 24, vbSrcCopy
  Bar(2).Print Left(Str((barnyaris / totalpanah) * 100), 5) & _
  " %"
  'oke
  BitBlt Bar(1).hDC, 0, 0, (baroke * Bar(1).ScaleWidth \
totalpanah), 23, ButtonTemp.hDC, 0, 24, vbSrcCopy
  Bar(1).Print Left(Str((baroke / totalpanah) * 100), 5) & " %"
  'sempurna
  BitBlt Bar(0).hDC, 0, 0, (barsempurna * Bar(0).ScaleWidth \
totalpanah), 23, ButtonTemp.hDC, 0, 24, vbSrcCopy
  Bar(0).Print Left(Str((barsempurna / totalpanah) * 100), _
  5) & " %"

  For i = 0 To 4
    Bar(i).Refresh
  Next i

```

```

Grading
Audiance
End Sub

```

Berikut fungsi yang digunakan untuk menyisipkan nama dan nilai pemain ke dalam daftar nilai tertinggi:

```

Sub simpanscore()
Dim i, j, X As Integer
Dim ditemukan As Boolean
ReDim MatriksNilai(9) As Nilai
Dim buffer As String * 255
Dim tempangka As String
Dim temp As String
Dim temp2 As String
'reset matriks nilai
For i = 0 To 9
MatriksNilai(i).Nama = ""
MatriksNilai(i).Nilai = 0
Next i

'menentukan tingkat kesulitan yang akan dipanggil
Select Case Variabel.Kesulitan
Case 0: 'Mudah
temp = "Mudah"
Case 1: 'Standar
temp = "Standar"
Case 2: 'Sulit
temp = "Sulit"
End Select

'memanggil file ini untuk disimpan dalam matriks
j = 0
X = 1
While j < 10 And X <> 0 'peringkat
tempangka = Str(j + 1)
If j + 1 < 10 Then tempangka = Right(tempangka, 1)
Else tempangka = Right(tempangka, 2)
X = GetPrivateProfileString("Nilai" & temp, "n" &
tempangka, "", buffer, 255, App.Path & "\INI\" &
Left>NamaFileMP3, Len>NamaFileMP3)-3) & "ini")
If X <> 0 Then
MatriksNilai(j).Nama = Left(buffer, X)
X = GetPrivateProfileString("Nilai" & temp, "s" &
tempangka, "", buffer, 255, App.Path & "\INI\" &
Left>NamaFileMP3, Len>NamaFileMP3)-3) & "ini")
MatriksNilai(j).Nilai = Val(Left(buffer, X))
If X = 0 Then
X = 1
End If
End If
j = j + 1
Wend

```



```

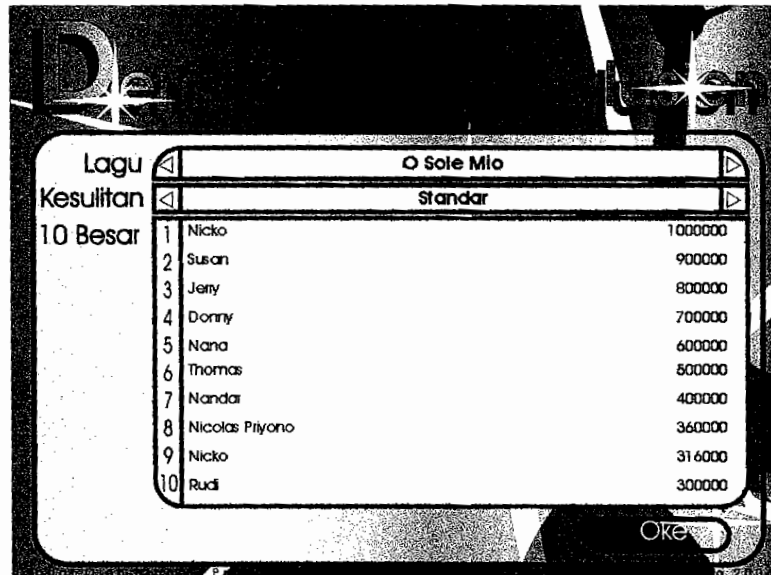
'mencari posisi pemain dalam daftar nilai tertinggi
j = j-1
i = 0
ditemukan = False
While i < j And Not ditemukan
    If MatriksNilai(i).Nilai < poin Then
        'menggeser posisi-posisi dibawahnya
        For X = j-1 To i Step -1
            MatriksNilai(X + 1) = MatriksNilai(X)
        Next X
        'menyisipkan nilai baru
        MatriksNilai(i).Nama = NamaPemain
        MatriksNilai(i).Nilai = poin
        ditemukan = True
    End If
    i = i + 1
Wend
'simpan file ini
For i = 0 To 9
    tempangka = Str(i + 1)
    If i + 1 < 10 Then tempangka = Right(tempangka, 1) _
    Else tempangka = Right(tempangka, 2)
    X = WritePrivateProfileString("Nilai" & temp, "n" & _
    tempangka, MatriksNilai(i).Nama, App.Path & "\INI\" & _
    Left>NamaFileMP3, Len>NamaFileMP3-3) & "ini")
    temp2 = Str(MatriksNilai(i).Nilai)
    X = WritePrivateProfileString("Nilai" & temp, "s" & _
    tempangka, temp2, App.Path & "\INI\" & Left>NamaFileMP3, _
    Len>NamaFileMP3-3) & "ini")
Next i
End Sub

```

4.7 Layar Nilai Tertinggi

Form ini akan menampilkan informasi yang terdapat dalam *file* INI. Informasi ditampilkan berdasarkan urutan *file* INI berdasarkan nama *file*-nya, tanpa melakukan pemeriksaan terhadap *file* MP3-nya. Sehingga ada kemungkinan *file* INI tanpa *file* MP3-nya.

Berikut adalah tampilan implementasi layar nilai tertinggi:



Gambar 4.14 Contoh tampilan implementasi layar nilai tertinggi.

Baris-baris perintah berikut digunakan untuk membangun suatu daftar

judul lagu berdasarkan *file* INI yang ada:

```
FileINI.Path = App.Path & "\INI"
Dim X As Integer
Dim buffer As String * 255
Dim i, j As Integer
Dim tempangka As String
ReDim Arrayjudul(FileINI.ListCount-1) As String
For i = 0 To FileINI.ListCount-1
    X = GetPrivateProfileString("Keterangan", "Judul", _
        "Tidak berjudul", buffer, 255, App.Path & "\INI\" & _
        FileINI.List(i))
    Arrayjudul(i) = Left(buffer, X)
Next i
```

Berikut ini juga baris-baris perintah yang digunakan untuk konversi judul

lagu menjadi indeks:

```
If Judullagu = "" Then Judullagu = Arrayjudul(0)
For i = 0 To FileINI.ListCount-1
    If Arrayjudul(i) = Judullagu Then
        FileINI.ListIndex = i
    End If
Next i
```

Fungsi yang digunakan untuk menampilkan daftar nilai yang terdapat pada

file INI adalah:

```

Sub TampilNilai()
'+++ MEMBUAT MATRIKS DAFTAR NILAI +++
ReDim MatriksNilai(9) As Nilai
Dim buffer As String * 255
Dim tempangka As String

'Kosongkan daftar nama dan nilai
For j = 0 To 9
    Peringkat(j).Caption = ""
    Nilai(j).Caption = ""
Next j

'menentukan tingkat kesulitan yang akan dipanggil _
ke dalam matriks
Select Case Variabel.Kesulitan
    Case 0: 'Mudah
        temp = "Mudah"
    Case 1: 'Standar
        temp = "Standar"
    Case 2: 'Sulit
        temp = "Sulit"
End Select

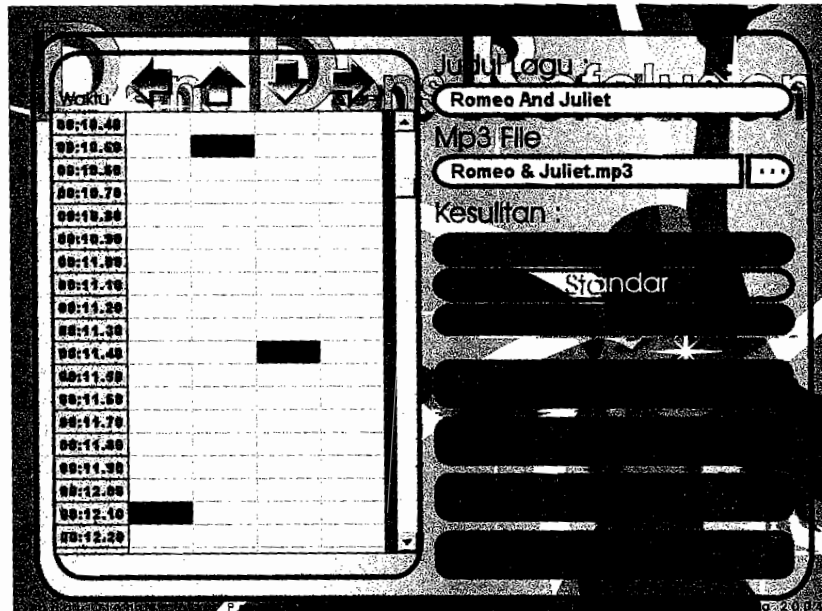
'memanggil file ini untuk disimpan dalam matriks
j = 1
X = 1
While j <= 10 And X <> 0 'peringkat
    tempangka = Str(j)
    If j < 10 Then tempangka = Right(tempangka, 1) _
    Else tempangka = Right(tempangka, 2)
    X = GetPrivateProfileString("Nilai" & temp, "n" & _
    tempangka, "", buffer, 255, App.Path & "\INI\" & _
    FileINI.List(FileINI.ListIndex))
    If X <> 0 Then
        MatriksNilai(j-1).Nama = Left(buffer, X)
        X = GetPrivateProfileString("Nilai" & temp, "s" & _
        tempangka, "", buffer, 255, App.Path & "\INI\" & _
        FileINI.List(FileINI.ListIndex))
        MatriksNilai(j-1).Nilai = Val(Left(buffer, X))
    End If
    j = j + 1
Wend

'menampilkan nilai berdasarkan matriks
For i = 0 To j-2
    Peringkat(i).Caption = MatriksNilai(i).Nama
    Nilai(i).Caption = MatriksNilai(i).Nilai
Next i

```

4.8 Layar Editor

Berikut tampilan implementasi dari layar editor:



Gambar 4.15 Contoh tampilan implementasi layar editor.

Pertama-tama pemain harus memilih lagu dengan menekan tombol dengan 3 buah titik (...). Setelah memilih lagu, pemain dapat melakukan perubahan terhadap tabel dan menyimpannya dalam file INI. Pemain dapat juga membuat perubahan terhadap tabel dengan menekan tombol papan ketik seperti pada waktu sedang bermain.

Fungsi yang digunakan untuk memanggil *file* MP3 dan melakukan pemeriksaan terhadap *file* INI-nya adalah:

```
Public Sub BukaMP3(mp3filename As String)
    Dim i, j As Integer
    'Cari panjang lagu dan tampilkan di tabel
    Mp3Object.SetSongName mp3filename
    Mp3Object.SetPath App.Path & "\mp3"
    Mp3Object.SetVolume Volume
    Mp3Object.StartPlayMP3
    PanjangLagu = 0
```

```

While PanjangLagu = 0
    Mp3Object.StartPlayMP3
    PanjangLagu = Mp3Object.MP3Duration
Wend
Mp3Object.StopPlayMP3

If Int(PanjangLagu) <= 3 * 60 Then
'tidak boleh lebih dari 3 menit
    MP3.Text = mp3filename
    Tabel.Clear
    Panjang = Int(PanjangLagu)
    Tabel.Rows = (Panjang * perdetik) + 1
    ProgressWindow.Kalimat = "Memanggil MP3"
    ProgressWindow.Show
    For i = 0 To Panjang-1
        detik = i Mod 60
        menit = Int(i / 60)
        If menit = 0 Then
            temp = "00"
        ElseIf menit <= 9 Then
            temp = "0" & Right(Str(menit), _
                Len(Str(menit)-1))
        Else
            temp = Right(Str(menit), Len(Str(menit)-1))
        End If
        If detik = 0 Then
            temp = temp & ":00"
        ElseIf detik = perdetik Then
            temp = temp & ":" & perdetik
        ElseIf detik <= perdetik-1 Then
            temp = temp & ":0" & detik
        Else
            temp = temp & ":" & detik
        End If
        For j = 0 To perdetik
            If j <> 0 Then
                Tabel.TextMatrix((i * perdetik) + j, 0) = _
                    temp & "." & Right(Str(j * perdetik), 2)
            Else
                Tabel.TextMatrix((i * perdetik) + j, 0) = _
                    temp & ".00"
            End If
        Next j
        ProgressWindow.RefreshProgressBar _
            Int((i / Panjang-1) * 100)
    Next i
    Unload ProgressWindow

    ReDim MatriksMudah(Tabel.Rows, 4)
    ReDim MatriksStandar(Tabel.Rows, 4)
    ReDim MatriksSulit(Tabel.Rows, 4)

    'Memeriksa file ini berdasarkan file MP3
    Dim AdaINI As Boolean
    temp = Left(MP3.Text, Len(MP3.Text)-4)
    temp = temp & ".ini"

```

```

AdaINI = False
FileINI.Path = App.Path & "\ini"
For i = 0 To FileINI.ListCount-1
    If FileINI.List(i) = temp Then
        AdaINI = True
    End If
Next i

'Kalau file ini ada, maka loading ini file
Dim buffer As String * 255
If AdaINI = True Then
    Dim X As Byte
    X = GetPrivateProfileString("keterangan", "judul", _
    "Belum ada", buffer, 255, App.Path & "\ini\" & temp)
    Judul.Text = Left(buffer, X)

    'Pindahkan file ini menjadi matriks mudah
    Dim temp2 As String
    ProgressWindow.Kalimat = "Memanggil INI"
    ProgressWindow.Show
    For i = 0 To Tabel.Rows-1
        temp2 = Str(i)
        temp2 = "t" & Right(temp2, Len(temp2)-1)
        X = GetPrivateProfileString("PanahMudah", _
        temp2, "0", buffer, 255, App.Path & _
        "\ini\" & temp)
        temp2 = Left(buffer, 4)
        For j = 1 To 4
            If Left(temp2, 1) = "1" Then
                MatriksMudah(i, j) = True
            Else
                MatriksMudah(i, j) = False
            End If
            temp2 = Right(temp2, 4-j)
        Next j
        ProgressWindow.RefreshProgressBar _
        Int(i / (Tabel.Rows * 3) * 100)
    Next i

    'Pindahkan file ini menjadi matriks standar
    For i = 0 To Tabel.Rows-1
        temp2 = Str(i)
        temp2 = "t" & Right(temp2, Len(temp2)-1)
        X = GetPrivateProfileString("PanahStandar", _
        temp2, "0", buffer, 255, App.Path & _
        "\ini\" & temp)
        temp2 = Left(buffer, 4)
        For j = 1 To 4
            If Left(temp2, 1) = "1" Then
                MatriksStandar(i, j) = True
            Else
                MatriksStandar(i, j) = False
            End If
            temp2 = Right(temp2, 4-j)
        Next j
        ProgressWindow.RefreshProgressBar _

```

```

        Int((i + Tabel.Rows) / (Tabel.Rows * 3) * 100)
    Next i
    'Pindahkan file ini menjadi matriks sulit
    For i = 0 To Tabel.Rows-1
        temp2 = Str(i)
        temp2 = "t" & Right(temp2, Len(temp2)-1)
        X = GetPrivateProfileString("PanahSulit", _
            temp2, "0", buffer, 255, App.Path & _
            "\ini\" & temp)
        temp2 = Left(buffer, 4)
        For j = 1 To 4
            If Left(temp2, 1) = "1" Then
                MatriksSulit(i, j) = True
            Else
                MatriksSulit(i, j) = False
            End If
            temp2 = Right(temp2, 4-j)
        Next j
        ProgressWindow.RefreshProgressBar _
            Int((i + Tabel.Rows + Tabel.Rows) / _
            (Tabel.Rows * 3) * 100)
    Next i
    Unload ProgressWindow

    PanggilMatriks
Else
    Judul.Text = Left(temp, Len(temp)-4)
End If
Else
    MP3.Text = tempmp3filename
    DurasiLebih.Show
End If
End Sub

```

Berikut adalah fungsi untuk menampilkan matriks 'panah' ke dalam tabel:

```

Private Sub PanggilMatriks()
    ProgressWindow.Kalimat = "Menampilkan"
    ProgressWindow.Show
    Select Case Kesulitan
    Case 0: 'Mudah
        For i = 0 To Tabel.Rows-1
            For j = 1 To Tabel.Cols-1
                Tabel.Row = i
                Tabel.Col = j
                If MatriksMudah(i, j) = True Then
                    Tabel.CellBackColor = vbRed
                Else
                    Tabel.CellBackColor = vbWhite
                End If
            Next j
            ProgressWindow.RefreshProgressBar _
                Int((i / Tabel.Rows) * 100)
        Next i
    Case 1: 'Standar

```

```

For i = 0 To Tabel.Rows-1
  For j = 1 To Tabel.Cols-1
    Tabel.Row = i
    Tabel.Col = j
    If MatriksStandar(i, j) = True Then
      Tabel.CellBackColor = vbRed
    Else
      Tabel.CellBackColor = vbWhite
    End If
  Next j
  ProgressWindow.RefreshProgressBar _
  Int((i / Tabel.Rows) * 100)
Next i
Case 2: 'Sulit
  For i = 0 To Tabel.Rows-1
    For j = 1 To Tabel.Cols-1
      Tabel.Row = i
      Tabel.Col = j
      If MatriksSulit(i, j) = True Then
        Tabel.CellBackColor = vbRed
      Else
        Tabel.CellBackColor = vbWhite
      End If
    Next j
    ProgressWindow.RefreshProgressBar _
    Int((i / Tabel.Rows) * 100)
  Next i
End Select
Unload ProgressWindow
End Sub

```

Fungsi untuk menset ulang matriks dan menampilkannya pada tabel

adalah:

```

If tes = False And MP3.Text <> "" Then
  ProgressWindow.Kalimat = "Reset"
  ProgressWindow.Show
  Select Case Kesulitan
  Case 0: 'Mudah
    For i = 0 To Tabel.Rows-1
      For j = 1 To Tabel.Cols-1
        MatriksMudah(i, j) = False
      Next j
      ProgressWindow.RefreshProgressBar _
      Int((i / (Tabel.Rows-1)) * 100)
    Next i
  Case 1: 'Standar
    For i = 0 To Tabel.Rows-1
      For j = 1 To Tabel.Cols-1
        MatriksStandar(i, j) = False
      Next j
      ProgressWindow.RefreshProgressBar _
      Int((i / (Tabel.Rows-1)) * 100)
    Next i
  End Select

```



```

Case 2: 'Mudah
    For i = 0 To Tabel.Rows-1
        For j = 1 To Tabel.Cols-1
            MatriksSulit(i, j) = False
        Next j
        ProgressWindow.RefreshProgressBar _
            Int((i / (Tabel.Rows-1)) * 100)
    Next i
End Select
Unload ProgressWindow
PanggilMatriks
Else
    PlaySound "error"
End If

```

Berikut adalah baris-baris perintah untuk menyimpan pada *file* INI:

```

If tes = False And Tabel.Rows > 0 And matriksberubah Then
    Dim X As Byte
    Dim temp2 As String
    'Tulis Judul
    ProgressWindow.Kalimat = "Menyimpan"
    ProgressWindow.Show
    X = WritePrivateProfileString("Keterangan", "Judul", _
        Judul.Text, App.Path & "\ini\" & Left(MP3.Text, _
        Len(MP3.Text)-4) & ".ini")
    CekX X
    Select Case Kesulitan
    Case 0: 'Tulis Panah Mudah
        For i = 0 To Tabel.Rows-1
            temp = ""
            For j = 1 To Tabel.Cols-1
                If MatriksMudah(i, j) = True Then
                    temp = temp & "1"
                Else
                    temp = temp & "0"
                End If
            Next j
            temp2 = Str(i)
            temp2 = "t" & Right(temp2, Len(temp2)-1)
            X = WritePrivateProfileString("PanahMudah", _
                temp2, temp, App.Path & "\ini\" & Left(MP3.Text, _
                Len(MP3.Text)-4) & ".ini")
            CekX X
            ProgressWindow.RefreshProgressBar _
                Int(i-10 / (Tabel.Rows * 3) * 100)
        Next i
        For i = 1 To 10
            temp2 = Str(i)
            temp2 = "n" & Right(temp2, Len(temp2)-1)
            X = WritePrivateProfileString("NilaiMudah", _
                temp2, "Belum ada", App.Path & "\ini\" & _
                Left(MP3.Text, Len(MP3.Text)-4) & ".ini")
            CekX X
            temp2 = Str(i)
            temp2 = "s" & Right(temp2, Len(temp2)-1)

```

```

X = WritePrivateProfileString("NilaiMudah", _
temp2, "0", App.Path & "\ini\" & Left(MP3.Text, _
Len(MP3.Text)-4) & ".ini")
CekX X
ProgressWindow.RefreshProgressBar _
Int(i + (Tabel.Rows-1) / (Tabel.Rows * 3) * 100)
Next i
Case 1: 'Tulis Panah Standar
For i = 0 To Tabel.Rows-1
temp = ""
For j = 1 To Tabel.Cols-1
If MatriksStandar(i, j) = True Then
temp = temp & "1"
Else
temp = temp & "0"
End If
Next j
temp2 = Str(i)
temp2 = "t" & Right(temp2, Len(temp2)-1)
X = WritePrivateProfileString("PanahStandar", _
temp2, temp, App.Path & "\ini\" & Left(MP3.Text, _
Len(MP3.Text)-4) & ".ini")
CekX X
ProgressWindow.RefreshProgressBar _
Int(i-10 / (Tabel.Rows * 3) * 100)
Next i
For i = 1 To 10
temp2 = Str(i)
temp2 = "n" & Right(temp2, Len(temp2)-1)
X = WritePrivateProfileString("NilaiStandar", _
temp2, "Belum ada", App.Path & "\ini\" & _
Left(MP3.Text, Len(MP3.Text)-4) & ".ini")
CekX X
temp2 = Str(i)
temp2 = "s" & Right(temp2, Len(temp2)-1)
X = WritePrivateProfileString("NilaiStandar", _
temp2, "0", App.Path & "\ini\" & Left(MP3.Text, _
Len(MP3.Text)-4) & ".ini")
CekX X
ProgressWindow.RefreshProgressBar _
Int(i + (Tabel.Rows-1) / (Tabel.Rows * 3) * 100)
Next i
Case 2: 'Tulis Panah Sulit
For i = 0 To Tabel.Rows-1
temp = ""
For j = 1 To Tabel.Cols-1
If MatriksSulit(i, j) = True Then
temp = temp & "1"
Else
temp = temp & "0"
End If
Next j
temp2 = Str(i)
temp2 = "t" & Right(temp2, Len(temp2)-1)
X = WritePrivateProfileString("PanahSulit", _
temp2, temp, App.Path & "\ini\" & _

```

```

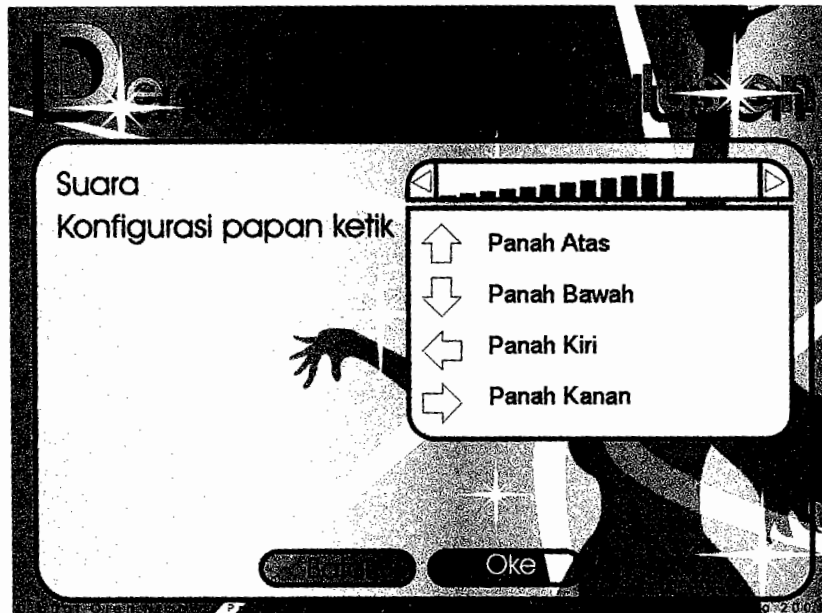
        Left(MP3.Text, Len(MP3.Text)-4) & ".ini")
        CekX X
        ProgressWindow.RefreshProgressBar _
        Int(i-10 / (Tabel.Rows * 3) * 100)
    Next i
    For i = 1 To 10
        temp2 = Str(i)
        temp2 = "n" & Right(temp2, Len(temp2)-1)
        X = WritePrivateProfileString("NilaiSulit", _
        temp2, "Belum ada", App.Path & "\ini\" & _
        Left(MP3.Text, Len(MP3.Text)-4) & ".ini")
        CekX X
        temp2 = Str(i)
        temp2 = "s" & Right(temp2, Len(temp2)-1)
        X = WritePrivateProfileString("NilaiSulit", _
        temp2, "0", App.Path & "\ini\" & _
        Left(MP3.Text, Len(MP3.Text)-4) & ".ini")
        CekX X
        ProgressWindow.RefreshProgressBar _
        Int(i + (Tabel.Rows-1) / (Tabel.Rows * 3) * 100)
    Next i
    End Select
    matriksberubah = False
    Unload ProgressWindow
Else
    PlaySound "error"
End If

```

4.9 Layar Pengaturan

Perubahan tinggi rendahnya suara permainan dan tombol papan ketik dalam permainan dapat diatur dari layar ini. Namun perubahannya hanya akan terjadi jika pemain menutup layar ini dengan menekan tombol 'Oke'. Jika tidak, maka perubahan yang terjadi pada layar ini tidak akan disimpan.

Berikut adalah tampilan implementasi dari layar pengaturan:



Gambar 4.16 Contoh tampilan implementasi layar pengaturan.

Berikut adalah bagian program yang digunakan untuk memeriksa status

tombol papan ketik yang ditekan:

```

KeyAscii = 0
For i = 0 To 255
  DoEvents
  If (GetAsyncKeyState(i) And &H8001) <> 0 Then
    KeyAscii = i
  End If
Next i

If Papanketik(KeyAscii) <> "" Then
  If KeyAscii = vbKeyEscape Then
    PlaySound "clickout"
    Select Case yangdiubah
      Case 1: 'Atas
        Label(yangdiubah-1).Caption = Papanketik(Atas)
      Case 2: 'Bawah
        Label(yangdiubah-1).Caption = Papanketik(Bawah)
      Case 3: 'Kiri
        Label(yangdiubah-1).Caption = Papanketik(Kiri)
      Case 4: 'Kanan
        Label(yangdiubah-1).Caption = Papanketik(Kanan)
    End Select
    yangdiubah = 0
    Timer1.Enabled = False
  ElseIf yangdiubah > 0 Then

```

```

Select Case yangdiubah
Case 1: 'Atas
    Atas = KeyAscii
Case 2: 'Bawah
    Bawah = KeyAscii
Case 3: 'Kiri
    Kiri = KeyAscii
Case 4: 'Kanan
    Kanan = KeyAscii
End Select
Label(yangdiubah-1).Caption = Papanketik(KeyAscii)
yangdiubah = 0
Timer1.Enabled = False
End If
End If

```

4.10 Layar Trims

Layar ini diubah sedikit dari bentuk rancangan semula sehingga memiliki animasi teks menggulung ke atas. Hal ini dilakukan karena informasi yang ingin disampaikan lebih banyak dari pada informasi pada saat perancangan. Untuk dapat melakukan animasi tersebut, diperlukan tiga kali peletakan gambar. Gambar pertama adalah gambar latar dari *form*, diikuti dengan gambar *mask* dari teks dan gambar teks pada posisi terakhir.

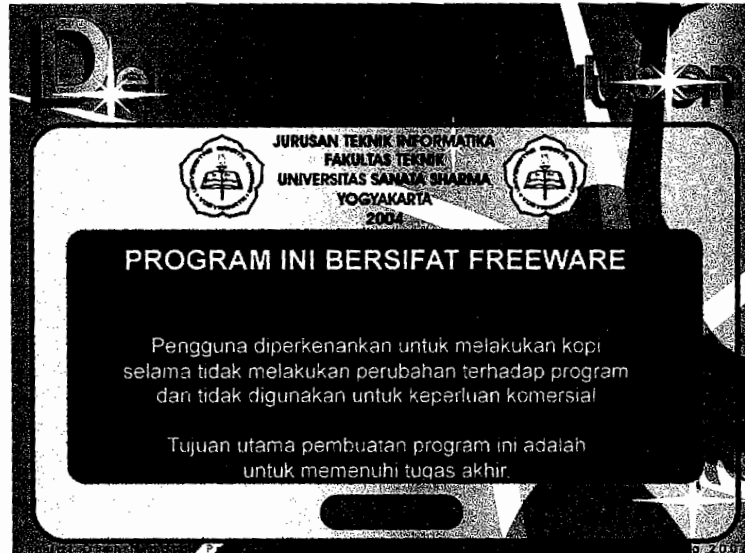
Berikut adalah bagian program yang terdapat dalam *event timer* dari objek *timer* untuk melakukan animasi tersebut:

```

Private Sub Timer1_Timer()
    DoEvents
    '+++ MEMBUAT TEKS BERGERAK KE ATAS +++
    Me.Picture = LoadPicture(App.Path & "\images\trims.bmp")
    BitBlt Me.hDC, 30, 201, 582, 213, Tekstemp.hDC, 582, Y, _
        vbSrcAnd 'Mask
    BitBlt Me.hDC, 30, 201, 582, 213, Tekstemp.hDC, 0, Y, _
        vbSrcPaint 'Teks
    Me.Refresh
    Y = Y + 2
    If Y > 1805 Then Y = 0
End Sub

```

Berikut adalah tampilan implementasi dari layar trims:



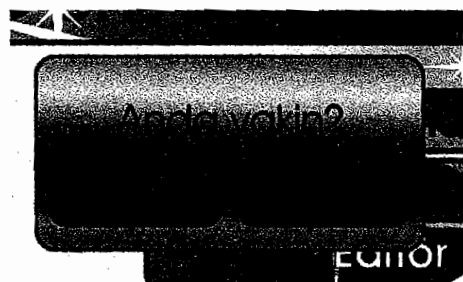
Gambar 4.17 Contoh tampilan impementasi layar trims.

4.11 Jendela Pendukung

Penampilan jendela pendukung memerlukan bentuk jendela yang berbeda dengan layar-layar sebelumnya. Ujung-ujung dari jendela pendukung yang berbentuk bulat memerlukan penanganan khusus. Untuk melakukannya diperlukan suatu metode untuk mengubah bentuk *form region*, yang dilakukan dengan menggunakan modul *Skining*.

4.11.1 Konfirmasi

Berikut tampilan implementasi dari jendela konfirmasi:



Gambar 4.18 Contoh tampilan implementasi jendela konfirmasi.

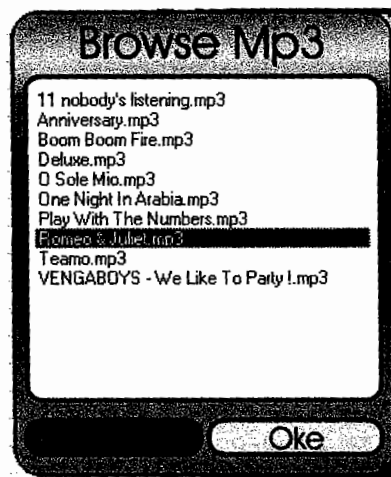
Bagian program yang digunakan untuk menyimpan data pada *file* *setting.ini* saat pemain menyetujui akan keluar dari program adalah sebagai berikut:

```
Dim X As Byte
Dim temp2 As String
X = WritePrivateProfileString("Setting", "Nama", NamaPemain, _
App.Path & "\setting.ini")
CekX X
temp2 = Str(Volume)
X = WritePrivateProfileString("Setting", "Volume", temp2, _
App.Path & "\Setting.ini")
CekX X
temp2 = Str(Atas)
X = WritePrivateProfileString("Setting", "Atas", temp2, _
App.Path & "\setting.ini")
CekX X
temp2 = Str(Bawah)
X = WritePrivateProfileString("Setting", "Bawah", temp2, _
App.Path & "\setting.ini")
CekX X
temp2 = Str(Kiri)
X = WritePrivateProfileString("Setting", "Kiri", temp2, _
App.Path & "\setting.ini")
CekX X
temp2 = Str(Kanan)
X = WritePrivateProfileString("Setting", "Kanan", temp2, _
App.Path & "\setting.ini")
CekX X
```

4.11.2 Browse

Setelah pemain memilih lagu yang ingin diubah pada layar 'editor', *form* ini akan melakukan pemanggilan fungsi `BukaMP3` dari *form editor* dengan parameter berupa nama *file* MP3 yang ingin diubah pemain.

Berikut ini tampilan implementasi *browse*:



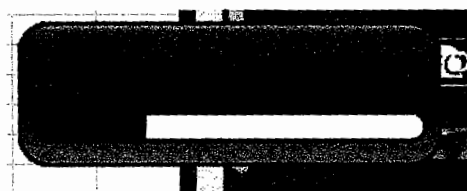
Gambar 4.19 Contoh tampilan implementasi *browse*.

4.12 Menampilkan, Menyimpan, dan Membaca Data

Saat diimplementasikan, ketiga tampilan tersebut dibentuk menjadi sebuah *form*, sehingga mempermudah proses pemrograman.

Berikut ini tampilan implementasi dari jendela ini pada saat membaca file

INI:



Gambar 4.20 Contoh tampilan implementasi saat membaca file INI.

4.13 Modul dan Kelas

Keterangan mengenai fungsi dari modul dan kelas yang digunakan dalam program.

4.13.1 Modul `DirectX`

Modul ini digunakan untuk menyimpan hal-hal yang berhubungan dengan *DirectX*, terutama untuk keperluan mengubah resolusi layar. Fungsi yang terdapat pada modul ini adalah fungsi `Init_FullScreen_DX` dan `Exit_DX`.

4.13.2 Modul `md1OnTop`

Modul ini digunakan untuk mengaktifkan status suatu *form* menjadi “*always on top*” dan juga menonaktifkannya.

4.13.3 Modul `PublicSub`

Modul ini digunakan untuk menyimpan Sub atau Fungsi yang bersifat `Public`, sehingga semua *form* dapat menggunakannya. Fungsi yang terdapat dalam modul ini adalah `CekX` (untuk memeriksa status file INI), `Playsound` dan `Playsound2` (untuk memainkan bunyi), `Papanketik` (Untuk konversi nilai integer dari tombol papan ketik menjadi ASCII), dan `Delay` (untuk menunda sistem sejenak).

4.13.4 Modul `skinning`

Modul ini menyimpan fungsi, variabel, konstanta dan deklarasi *Windows* API yang digunakan untuk keperluan transparansi *form*, sehingga dapat membuat bentuk *form* yang berbeda.

4.13.5 Modul `Variabel`

Modul ini berisikan variabel, konstanta, dan `type` yang bersifat `public`, sehingga semua *form* dapat mengaksesnya.

4.13.6 Modul `WindowsAPI`

Modul ini mengumpulkan deklarasi fungsi *Windows* API dan konstanta *Windows* API yang bersifat `public`, sehingga semua *form* dapat menggunakannya.

4.13.7 Kelas `MP3`

Kelas ini merupakan pemutar *file* MP3 yang merupakan *file* suara. Untuk mendukung program, kelas ini disertai dengan fungsi-fungsi seperti: `SetSongName` (untuk menentukan nama *file*), `SetPath` (untuk menentukan letak *folder* dari *file* suara), `SetVolume` (untuk mengubah tinggi rendah suara), `StartPlayMP3` (untuk mulai memainkan *file* suara), `StopPlayMP3` (untuk berhenti memainkan *file* suara), `GetPosition` (untuk memberikan posisi suara saat ini dalam satuan detik), `GetPercentposition` (untuk memberikan posisi suara saat ini dalam persen) dan `MP3Duration` (untuk memberikan panjang *file* suara dalam satuan detik).

BAB V

KESIMPULAN dan SARAN

5.1 Kesimpulan

Kesimpulan dari pembuatan perangkat-lunak permainan ini adalah:

1. Salah satu cara untuk mendapatkan mode *fullscreen* adalah dengan mengimplementasikan *DirectX*.
2. Animasi sederhana dibuat dengan melakukan penghapusan suatu gambar dan menggambarnya kembali dengan keadaan gambar yang berbeda dari sebelumnya. Untuk hal ini digunakan *Windows API* untuk melakukan pemindahan sebagian ataupun seluruh gambar secara cepat.
3. Posisi musik yang dimainkan digunakan sebagai patokan saat yang tepat untuk menampilkan 'panah' yang diinginkan. Kemudian pemain berusaha menyelaraskan 'panah' yang muncul dengan menggunakan piranti masukan.
4. *File* inisialisasi (*.ini) dapat digunakan untuk menyimpan data program. Dengan *Windows API*, pembuatan, penulisan dan pembacaan file inisialisasi menjadi lebih mudah dilakukan. Namun file inisialisasi standar tidak dapat mempunyai ukuran lebih dari 64 KB, hal ini yang membuat program hanya dapat memuat lagu yang panjangnya 3 menit.
5. *Windows API* digunakan untuk mendeteksi tombol dari papan ketik yang ditekan. Pemberian nilai dilakukan bila ada suatu 'panah' pada suatu area yang tipenya cocok dengan tombol papan ketik.

6. Beberapa *ActiveX* yang telah disediakan VB dapat dibuat tidak berbingkai dan tidak berkesan timbul, namun ada juga yang tidak dapat.
7. Efek separuh transparan diimplementasikan dengan menggunakan teknik *masking*. Dengan teknik ini diperlukan 2 buah gambar untuk memberikan efek separuh transparan tersebut, kedua gambar tersebut adalah gambar asli dan *mask*.

5.2 Saran

Saran yang dapat diberikan penulis sehubungan dengan pengembangan perangkat-lunak permainan ini adalah:

1. Program saat ini hanya mendukung resolusi 640×480 piksel. Sehingga masih dapat dikembangkan untuk resolusi yang lebih tinggi.
2. Implementasi *DirectX* pada program masih sangat sedikit. Karena penggunaan *DirectX* dapat membuat program permainan menjadi lebih baik, maka program akan sangat baik bila dikembangkan dengan mengimplementasikan *DirectX*.
3. Program masih menggunakan algoritma yang sederhana untuk menyesuaikan 'panah' dengan lagu yang diputar, perbaikan algoritma ini dapat membuat permainan menjadi lebih mengasyikan.
4. Program hanya dapat menggunakan papan ketik sebagai piranti masukannya, program dapat dikembangkan untuk menggunakan piranti masukan yang lain seperti, *joystick* ataupun *dancing carpet*.

5. Program dapat dikembangkan untuk dimainkan dengan lawan main berupa komputer atau pemain lainnya.
6. Program hanya mendukung format suara MP3, sehingga pengembangan dukungan terhadap format sura yang lain dapat lebih membuat pemain leluasa memilih lagu.



DAFTAR PUSTAKA

- Hadi, R, 2001, "Pemrograman Windows API dengan Microsoft Visual Basic", PT. Elex Media Komputindo, Jakarta.
- Hakim, L, 2003, "Pemrograman Game dengan Visual Basic", Andi Offset, Yogyakarta.
- Halvorson, M, 2000, "Step By Step Microsoft Visual Basic 6.0 Professional", PT. Elex Media Komputindo, Jakarta.
- Maki, T, 2002, "Mastering Computer Graphic Untuk Pemula", Nexx Media, Jakarta.
- McClelland, D, 2002, "Deke McClelland's Look & Learn Photoshop", PT. Elex Media Komputindo, Jakarta.
- Tosin, R dan Riberu, T, dkk, 1997, "Cara Mudah Belajar Visual Basic 4.0", Dinastindo, Jakarta.
- Level International Computer Gaming Magazine, nomor 10, Oktober 2003.

LAMPIRAN

Lampiran A. Daftar Istilah

Nama istilah	Arti
<i>Adapter</i>	Perangkat-keras dengan fungsi tertentu, seperti <i>VGA adapter</i> .
<i>API</i>	= <i>Application Programming Interface</i> . Fungsi yang disertakan bersama dengan sistem operasi untuk membantu <i>programmer</i> membuat perangkat-lunak yang berbasis pada sistem operasi tersebut.
<i>Bitmap</i>	Gambar digital dengan yang terdiri dari gabungan titik-titik berwarna untuk membentuk suatu objek tertentu.
<i>Cursor</i>	Petunjuk pada layar monitor yang menunjukkan perintah yang dapat diterima sistem, misalnya perintah untuk klik, ketik, mengubah ukuran layar, atau sistem yang sedang tidak dapat menerima perintah.
<i>DirectX</i>	Perangkat-lunak yang berfungsi untuk meningkatkan performa sistem dalam hal <i>Multimedia</i> .
<i>Entertainment</i>	Hiburan
<i>Internet</i>	Gabungan jaringan-jaringan komputer yang sangat luas sehingga menghubungkan seluruh dunia.
<i>Joystick</i>	Perangkat masukan berupa tongkat dengan beberapa tombol, digunakan untuk mengoperasikan program permainan (biasanya permainan yang mengutamakan kecepatan dan ketepatan).
<i>Keyboard</i>	Perangkat masukan yang berupa papan dengan tombol-tombol berupa huruf, angka, simbol dan tombol fungsi.
Mesin <i>Arcade</i>	Mesin komputer yang diubah bentuknya sehingga berpenampilan lebih menarik dan hanya memiliki sebuah fungsi permainan, biasanya dilengkapi dengan perangkat masukan yang mendukung permainannya.
<i>Mouse</i>	Perangkat masukan yang berfungsi sebagai alat penunjuk (<i>pointing device</i>), digunakan dengan cara menggerakkan tangan untuk menggerakkan <i>mouse</i> sehingga <i>cursor</i> bergerak.
<i>Multimedia</i>	Jenis perangkat-lunak yang bersifat <i>entertainment</i> .
<i>Pixel (Picture Element)</i>	Piksel, Satuan titik terkecil pada layar/gambar digital.
<i>Prorammer</i>	Seseorang atau kelompok orang yang secara langsung melakukan penulisan perintah bagi komputer.
<i>Resolution</i>	Resolusi, menunjukkan jumlah <i>pixel</i> pada layar monitor.
<i>Sound Effect</i>	Efek suara, digunakan untuk mendukung suasana yang diinginkan.
<i>Vector</i>	Gambar digital dengan perhitungan matematis tertentu untuk membentuk gambar, sehingga perbesaran gambar tidak mempengaruhi kualitas gambar. Tidak lagi tersusun atas titik-titik seperti <i>bitmap</i> .
Windows	Sistem operasi komputer berbasis kompatibel IBM yang paling banyak digunakan di dunia.

Lampiran B. Ketentuan dalam Penulisan

Istilah dalam tulisan dan artinya:

Garis bawah (_)	Baris berikutnya adalah baris yang sama, dilakukan karena lebar kertas lebih kecil dari panjang kalimat. Digunakan untuk penulisan perintah pemrograman yang panjang. Tanda ' _ ' adalah standar bahasa pemrograman <i>Visual Basic</i> untuk melanjutkan baris perintah yang panjang pada baris berikutnya, untuk mempermudah pembacaan perintah.
Tanda petik tunggal dengan warna abu-abu	Teks di belakang tanda ini merupakan keterangan, bukan merupakan perintah program. Dimaksudkan sebagai penjelas maksud program.
Nama dan tahun setelah kalimat, dalam tanda kurung	Kalimat/paragraf merupakan kutipan, lihat daftar pustaka untuk keterangan lebih lanjut mengenai sumber.

Format tulisan dan artinya:

<i>Cetak miring</i>	Kata serapan dari bahasa asing (bukan istilah dalam bahasa Indonesia).
Font Courier New	Perintah dalam bahasa pemrograman <i>Visual Basic</i> .
Font Courier New dengan warna abu-abu	Keterangan program

