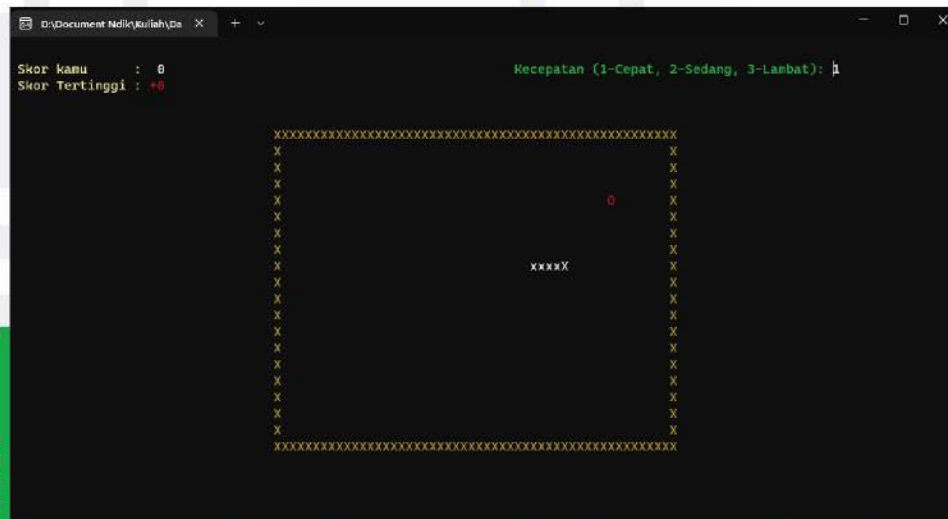
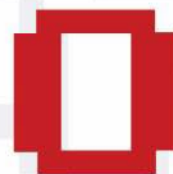




U10 Asem Portofolio



MOHAMMAD FARID HENDIANTO
2200018401
KELAS I



SCAN DISINI



<https://bit.ly/uloasem>

KATA PENGANTAR

Puji dan syukur yang tak terhingga saya panjatkan ke hadirat Allah SWT, yang telah memberikan kemudahan dan kesempatan kepada saya untuk dapat menyelesaikan proyek aplikasi game Ulo Asem. Saya merasa sangat bersyukur karena telah diberikan kemampuan dan kekuatan untuk menyelesaikan proyek ini tepat pada waktunya.

Tidak lupa saya sampaikan terima kasih yang tak terhingga kepada Bapak Ali Tarmuji, S.T., M.Cs. Teknik Informatika, selaku dosen pembimbing yang telah memberikan bimbingan dan arahan yang sangat membantu saya dalam mengerjakan pembuatan aplikasi ini. Saya merasa sangat beruntung telah diberikan kesempatan untuk belajar dan mengikuti bimbingan dari Bapak.

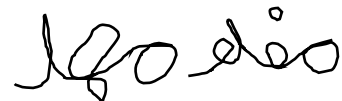
Terimakasih juga kepada teman-teman saya yang mensupport saya dalam pembuatan aplikasi ini, serta tidak lupa juga kepada orang tua saya yang selalu memberikan dukungan dan semangat kepada saya sehingga saya dapat menyelesaikan tugas ini.

Meskipun saya berusaha semaksimal mungkin dalam menyusun portofolio ini, saya sadar bahwa masih banyak kekurangan dan kelemahan yang terdapat pada aplikasi ini. Oleh karena itu, saya mengajak pembaca untuk memberikan kritik dan saran yang membangun kepada saya, agar aplikasi yang akan datang dapat menjadi lebih baik lagi. Kritik konstruktif sangat saya harapkan demi kesempurnaan aplikasi selanjutnya.

Akhir kata, saya berharap portofolio ini dapat memberikan gambaran yang baik tentang aplikasi game Ulo Asem yang saya buat, dan dapat menjadi referensi yang bermanfaat bagi pembaca. Saya juga berharap agar pembuatan aplikasi game dapat berjalan dengan lebih lancar dan sukses di masa yang akan datang.

Yogyakarta, Januari 2023

Penyusun



Mohammad Farid Hendianto

SURAT PENGESAHAN

Yogyakarta, 25 Januari 2023

Nomor : 01/SP/Universitas Ahmad Dahlan/FTI/Jurusan Informatika/2022

Kepada Yth,

Dekan Fakultas Teknologi Industri

Universitas Ahmad Dahlan

Dengan Hormat,

Saya yang bertanda tangan di bawah ini :

Nama : Mohammad Farid Hendianto

NIM : 2200018401

Jurusan : Informatika

Fakultas : Fakultas Teknologi Industri

Universitas : Universitas Ahmad Dahlan

Dengan ini menyatakan bahwa aplikasi yang berjudul "Ulo Asem" merupakan hasil karya saya sendiri dan tidak melanggar hak cipta atau hak kekayaan intelektual milik pihak lain. Aplikasi ini dibuat untuk memenuhi tugas dasar sistem komputer dan sebagai portofolio aplikasi saya.

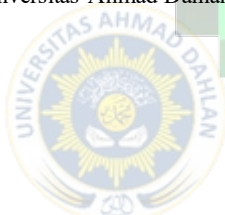
Demikian surat pengesahan ini dibuat dengan sebenarnya dan atas perhatiannya saya ucapkan terima kasih.

Mengetahui.

Dekan Fakultas Teknologi Industri
Universitas Ahmad Dahlan

Dosen Pembimbing

Pembuat Aplikasi.



Sunardi, S.T., M.T., Ph.D.



Ali Tarmuji, S.T., M.Cs



Mohammad Farid Hendianto

DAFTAR ISI

DAFTAR ISI	4
A. DESKRIPSI DAN RUANG LINGKUP PROYEK	5
1) DESKRIPSI PROYEK	5
2) RUANG LINGKUP PROYEK	8
B. DAFTAR SELURUH SPESIFIKASI APLIKASI	9
C. RANCANGAN ANTARMUKA	11
D. DIAGRAM PROSES APLIKASI	15
E. PENJELASAN KODING / SKRIP PROGRAM	19
1) INISIALISASI VARIABEL	19
2) PROSEDUR PROSES UTAMA	25
F. TAMPILAN YANG DIHASILKAN APLIKASI	49
H. ANALISIS Pengerjaan Proyek	55
1) TINJAUAN DARI SISI WAKTU	55
2) KETERCAPAIAN SPESIFIKASI	57
3) BIAYA YANG DIPERLUKAN	58
4) KENDALA YANG DIHADAPI	58
5) TANTANGAN MASA DEPAN	59

A. DESKRIPSI DAN RUANG LINGKUP PROYEK

1) DESKRIPSI PROYEK



Gambar 1 Logo Ulo Asem. (Sumber: Penulis)

Aplikasi game Ulo Asem adalah sebuah aplikasi yang dibuat menggunakan bahasa pemrograman Assembly, yaitu MASM. Aplikasi ini ditujukan untuk membantu para pemula dalam mempelajari bahasa Assembly dengan cara yang menyenangkan.

Aplikasi game Ulo Asem ini diberi nama dengan mengambil inspirasi dari bahasa Jawa yang merupakan salah satu bahasa daerah di Indonesia. Kata 'ulo' dalam bahasa Jawa merupakan salah satu sinonim untuk kata 'ular', sedangkan 'asem' merupakan kepanjangan dari bahasa Assembly yang merupakan bahasa pemrograman yang digunakan dalam pembuatan aplikasi ini. Sehingga, penamaan aplikasi game Ulo Asem ini merupakan gabungan dari kata 'ular' dan 'Assembly', yang diharapkan dapat menjadi salah satu sarana belajar bahasa Assembly yang menyenangkan bagi para pemula.

Dalam aplikasi game Ulo Asem ini, pengguna dapat memainkan game tersebut dengan menggunakan tombol panah pada keyboard untuk mengatur arah gerakan ular. Ular tersebut akan bergerak mengikuti arah yang ditunjukkan oleh pengguna, dan akan terus bergerak hingga menabrak tembok atau tubuh sendiri. Selain itu, setiap kali ular berhasil memakan sebuah apel, maka skor akan bertambah dan panjang ular akan bertambah.

Aplikasi ini dilengkapi dengan fitur-fitur seperti tampilan dinding, papan skor, pemilihan kecepatan Ulo, pembuatan acak apel, dan pergerakan Ulo yang dapat dikontrol dengan keyboard. Selain itu, saat game berakhir, aplikasi game Ulo Asem akan menampilkan skor akhir yang telah diraih oleh pengguna. Skor tersebut

akan ditampilkan sesuai dengan jumlah apel yang telah dimakan oleh ular. Selain itu, aplikasi game ini juga menyediakan fitur reset yang dapat digunakan oleh pengguna untuk mengulang permainan dari awal.

Dalam proyek pembuatan aplikasi ini, saya telah berusaha untuk menyertakan dokumentasi yang lengkap dan jelas sebagai referensi bagi pengguna aplikasi ini. Dokumentasi tersebut berisi informasi mengenai cara penggunaan aplikasi, kode program yang digunakan, serta gambaran dari alur proses pembuatan aplikasi. Hal ini dilakukan agar pengguna dapat dengan mudah memahami dan menggunakan aplikasi ini dengan baik. Selain itu, dokumentasi ini juga dapat digunakan sebagai bahan referensi bagi para mahasiswa/i Fakultas Teknologi Industri Universitas Ahmad Dahlan Jurusan Informatika yang ingin mempelajari bahasa Assembly melalui aplikasi game Ulo Asem ini.

Dokumentasi tersebut juga mencakup informasi mengenai arsitektur aplikasi, serta detail tentang masing-masing fitur yang terdapat pada aplikasi ini. Seperti fitur tampilan dinding, papan skor, pemilihan kecepatan Ulo, pembuatan acak apel, dan pergerakan Ulo yang dapat dikontrol dengan keyboard. Selain itu, dokumentasi ini juga menjelaskan proses pembuatan aplikasi ini, mulai dari perancangan, implementasi, hingga pengujian aplikasi.

Selain itu, dalam proyek pembuatan aplikasi ini, saya juga berusaha untuk menyertakan dokumentasi yang lengkap dan jelas sebagai referensi bagi pengguna aplikasi ini. Dokumentasi tersebut berisi informasi mengenai cara penggunaan aplikasi, kode program yang digunakan, serta gambaran dari algoritma yang digunakan dalam pembuatan aplikasi ini.

Aplikasi ini dikembangkan dengan menggunakan bahasa pemrograman Assembly yaitu MASM. Aplikasi ini ditujukan untuk membantu para pemula dalam mempelajari bahasa Assembly dengan cara yang menyenangkan. Dalam aplikasi game Ulo Asem ini, pengguna dapat memainkan game tersebut dengan menggunakan tombol panah pada keyboard untuk mengatur arah gerakan ular. Ular tersebut akan bergerak mengikuti arah yang ditunjukkan oleh pengguna, dan akan terus bergerak hingga menabrak tembok atau tubuh sendiri. Selain itu, setiap kali ular berhasil memakan sebuah apel, maka skor akan bertambah dan panjang ular akan bertambah.

Aplikasi ini juga dilengkapi dengan beberapa fitur yang dapat membantu dalam proses belajar Assembly. Fitur-fitur tersebut seperti tampilan dinding, papan skor, pemilihan kecepatan Ulo, pembuatan acak apel, dan pergerakan Ulo yang dapat dikontrol dengan keyboard. Selain itu, saat game berakhir, aplikasi game Ulo Asem akan menampilkan skor akhir yang telah diraih oleh pengguna. Skor tersebut akan ditampilkan sesuai dengan jumlah apel yang telah dimakan oleh ular. Selain itu, aplikasi game ini juga menyediakan fitur reset yang dapat digunakan oleh pengguna untuk mengulang permainan dari awal.

Dalam proyek pembuatan aplikasi ini, saya telah berusaha sebaik mungkin untuk menyelesaikan proyek ini sesuai dengan tujuan dan ruang lingkup yang telah ditentukan. Proses pembuatan aplikasi ini meliputi tahap-tahap seperti perencanaan, pengembangan, pengujian dan evaluasi. Pada tahap perencanaan, saya melakukan riset dan analisis terhadap game-game serupa yang sudah ada sebagai acuan dalam pembuatan aplikasi ini. Pada tahap pengembangan, saya mulai menulis kode program dengan menggunakan bahasa Assembly dan menggunakan software MASM sebagai compiler. Pada tahap pengujian, saya melakukan uji coba terhadap aplikasi yang telah dibuat untuk mengecek apakah aplikasi tersebut dapat berjalan dengan baik dan sesuai dengan yang diharapkan. Pada tahap evaluasi, saya melakukan evaluasi terhadap aplikasi yang telah dibuat dan melakukan perbaikan-perbaikan yang diperlukan sesuai dengan hasil pengujian.

Dalam proyek pembuatan aplikasi ini, saya juga berusaha untuk menyertakan fitur-fitur yang bermanfaat bagi pengguna. Salah satu fitur yang disertakan adalah pemilihan kecepatan Ulo. Pengguna dapat memilih kecepatan gerakan ular sesuai dengan preferensi mereka. Fitur ini dapat membantu pengguna dalam menyesuaikan tingkat kesulitan game sesuai dengan kemampuan mereka. Selain itu, saya juga menyertakan fitur pembuatan acak apel. Fitur ini dapat membuat permainan lebih menantang karena apel yang ditampilkan tidak selalu muncul di tempat yang sama.

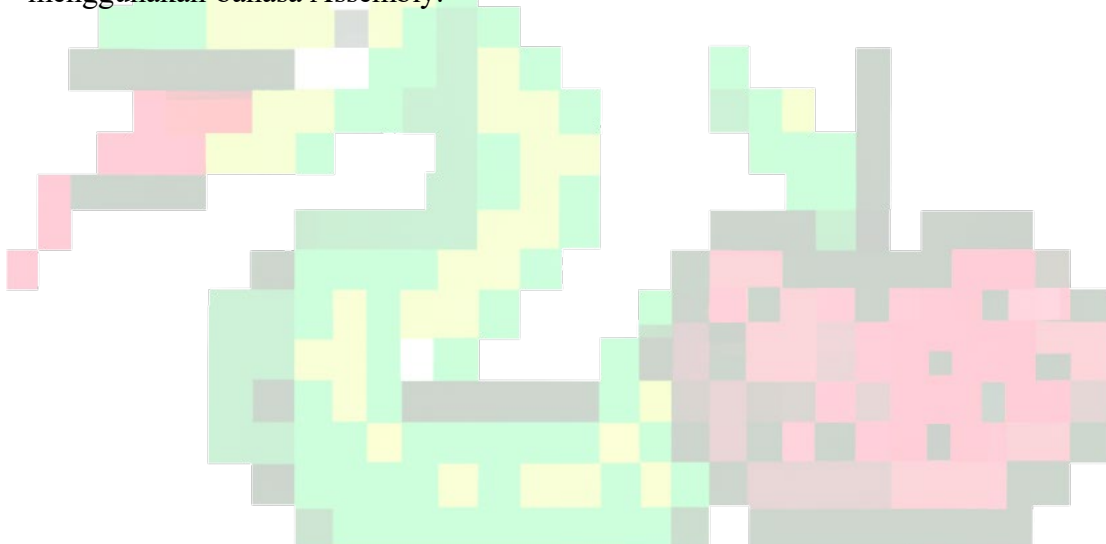
Selain itu, saya juga menyertakan fitur tampilan dinding. Fitur ini dapat membuat permainan lebih menarik karena dinding yang ditampilkan dapat diubah sesuai dengan preferensi pengguna. Pengguna dapat memilih dari berbagai macam warna dinding yang tersedia. Selain itu, saya juga menyertakan fitur papan skor. Fitur ini dapat membantu pengguna dapat melihat skor yang telah diraih serta high score yang pernah dicapai.

Dalam proyek pembuatan aplikasi ini, saya juga berusaha untuk menyertakan dokumentasi yang lengkap dan jelas sebagai referensi bagi pengguna aplikasi ini. Dokumentasi tersebut berisi informasi mengenai cara penggunaan aplikasi, kode program yang digunakan, serta gambaran dari algoritma yang digunakan dalam pembuatan aplikasi. Dokumentasi ini diharapkan dapat membantu pengguna dalam mengerti cara kerja aplikasi serta membantu dalam proses troubleshooting jika terjadi masalah.

Secara keseluruhan, aplikasi game Ulo Asem ini diharapkan dapat menjadi media belajar yang menyenangkan bagi para pemula yang ingin mempelajari bahasa Assembly. Aplikasi ini diharapkan dapat membantu para pemula dalam memahami cara kerja bahasa Assembly serta dapat meningkatkan kemampuan logika dan problem solving mereka. Saya berharap aplikasi ini dapat diterima dengan baik oleh pengguna dan dapat bermanfaat bagi mereka.

2) RUANG LINGKUP PROYEK

Ruang lingkup proyek ini adalah pembuatan sebuah aplikasi game Ulo Asem yang ditujukan untuk semua pengguna yang ingin bermain game tersebut. Aplikasi ini dapat dijalankan di sistem operasi Windows dengan menggunakan bahasa Assembly. Tujuan dari pembuatan aplikasi ini adalah untuk menghasilkan sebuah game yang dapat menghibur pengguna serta dapat digunakan sebagai media belajar Assembly bagi mahasiswa/i Fakultas Teknologi Industri Universitas Ahmad Dahlan Jurusan Informatika. Aplikasi ini dilengkapi dengan fitur-fitur seperti tampilan dinding, papan skor, pemilihan kecepatan Ulo, pembuatan acak apel, dan pergerakan Ulo yang dapat dikontrol dengan keyboard. Batasan dari aplikasi ini adalah hanya dapat digunakan di sistem operasi Windows dan ditulis dengan menggunakan bahasa Assembly.



B. DAFTAR SELURUH SPESIFIKASI APLIKASI

Untuk dapat menjalankan aplikasi game Ulo Asem, diperlukan spesifikasi sistem yang minimal sebagai berikut:

Sistem Operasi	: Windows XP 32 bit atau yang lebih tinggi
Processor	: 512 Mhz atau lebih tinggi
Memory	: 32 MB atau lebih tinggi
Ukuran File	: 39 KB
Storage	: 1 MB ruang hardisk yang tersedia
Sound card	: Terdapat soundcard untuk menjalankan audio
Graphics	: Graphic card yang mendukung penampilan warna

Pastikan bahwa semua perangkat lunak yang diperlukan, seperti sistem operasi dan aplikasi pendukung, telah terinstal dengan benar pada komputer Anda agar game Ulo Asem dapat berjalan dengan lancar. Jika ada perangkat lunak yang kurang atau tidak terinstal dengan benar, maka kemungkinan game Ulo Asem akan mengalami error atau tidak dapat dijalankan sama sekali. Oleh karena itu, pastikan untuk melakukan instalasi semua perangkat lunak yang diperlukan dengan benar agar game dapat berjalan dengan lancar.

Selain spesifikasi sistem yang diperlukan untuk menjalankan aplikasi game Ulo Asem, aplikasi ini juga memiliki beberapa spesifikasi aplikasi yang perlu diperhatikan.

- Aplikasi ini ditulis menggunakan bahasa pemrograman Assembly yaitu MASM.
- Aplikasi ini hanya dapat digunakan pada sistem operasi Windows.
- Aplikasi ini dilengkapi dengan fitur-fitur seperti tampilan dinding, papan skor, pemilihan kecepatan Ulo, pembuatan acak apel, dan pergerakan Ulo yang dapat dikontrol dengan keyboard.
- Aplikasi ini dapat digunakan untuk menghibur pengguna serta dapat digunakan sebagai media belajar Assembly bagi mahasiswa/i Fakultas Teknologi Industri Universitas Ahmad Dahlan Jurusan Informatika.
- Aplikasi ini memiliki batasan hanya dapat digunakan di sistem operasi Windows dan ditulis dengan menggunakan bahasa Assembly.

Semua spesifikasi aplikasi ini harus diperhatikan agar aplikasi dapat berjalan dengan baik dan sesuai dengan tujuan pembuatannya. Jika ada spesifikasi yang tidak sesuai, maka aplikasi mungkin tidak dapat dijalankan atau mengalami error. Namun, selalu ada kemungkinan untuk dapat meningkatkan atau menambah fitur-fitur yang dapat membuat pengalaman bermain game Ulo Asem semakin

menyenangkan bagi pengguna. Selain itu, jika terdapat bug atau masalah dalam aplikasi, saya akan segera melakukan perbaikan untuk mengatasinya secepat mungkin agar aplikasi dapat digunakan dengan baik oleh pengguna.

Selain spesifikasi sistem yang diperlukan, aplikasi game Ulo Asem juga dilengkapi dengan beberapa fitur yang akan membuat pengalaman bermain game semakin menyenangkan bagi pengguna. Fitur-fitur tersebut antara lain:

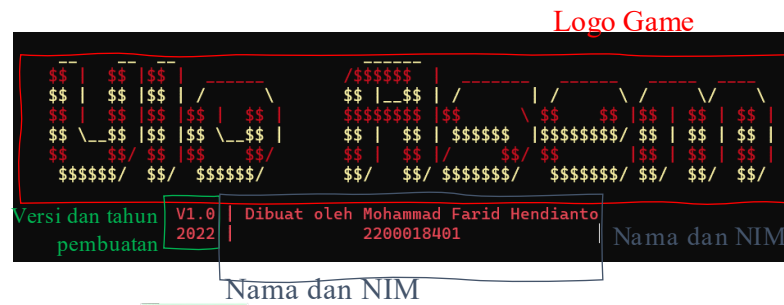
- Tampilan dinding: Aplikasi ini menyediakan tampilan dinding yang akan membuat permainan semakin menyenangkan bagi pengguna.
- Papan skor: Fitur papan skor pada aplikasi game Ulo Asem menampilkan skor secara real-time dari skor yang telah didapatkan saat dimainkan. Skor tersebut ditampilkan dalam bentuk angka yang mudah dibaca dan dipahami oleh pengguna. Selain itu, fitur ini juga menampilkan skor tertinggi dari permainan sebelum-sebelumnya. Skor tertinggi ini diperoleh dari hasil permainan saat program dijalankan dan ditampilkan sebagai acuan bagi pengguna untuk dapat meningkatkan skor yang telah didapatkan. (Skor tertinggi yang ditampilkan hanya berlaku untuk permainan saat program) dijalankan. Pemilihan kecepatan Ulo: Pengguna dapat memilih kecepatan gerakan ular sesuai dengan keinginan mereka.
- Pembuatan acak apel: Aplikasi ini menyediakan pembuatan acak apel yang akan membuat permainan
- Fitur reset: Untuk mengulang permainan dari awal

Dalam proyek pembuatan aplikasi ini, saya telah berusaha untuk menyertakan spesifikasi aplikasi yang sesuai dengan studi kasus yang ada. Namun, selalu ada kemungkinan untuk dapat meningkatkan atau menambah fitur-fitur yang ada pada aplikasi ini, tergantung dari hasil evaluasi dan umpan balik dari pengguna. Selain spesifikasi sistem yang diperlukan, aplikasi ini juga menyertakan fitur-fitur seperti tampilan dinding, papan skor, pemilihan kecepatan Ulo, pembuatan acak apel, dan pergerakan Ulo yang dapat dikontrol dengan keyboard. Selain itu, aplikasi ini juga dilengkapi dengan fitur reset yang dapat digunakan oleh pengguna untuk mengulang permainan dari awal.

Dengan spesifikasi dan fitur yang disertakan dalam aplikasi ini, diharapkan dapat memberikan pengalaman yang menyenangkan bagi pengguna serta dapat digunakan sebagai media belajar bahasa Assembly bagi mahasiswa/i Fakultas Teknologi Industri Universitas Ahmad Dahlan Jurusan Informatika.

C. RANCANGAN ANTARMUKA

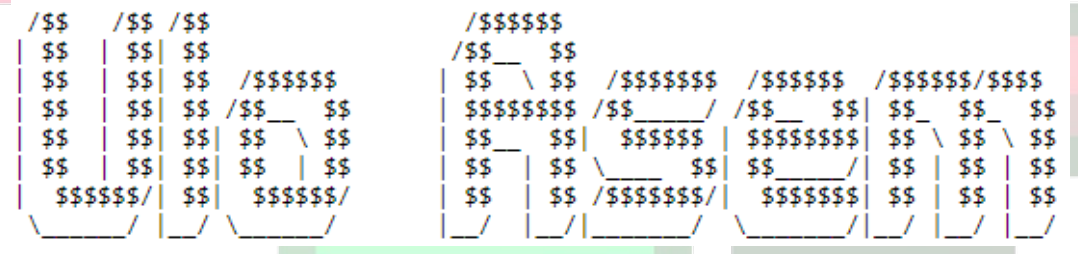
Dalam rancangan antarmuka aplikasi Ulo Asem, terdapat 4 tampilan utama yang akan ditampilkan kepada pengguna.



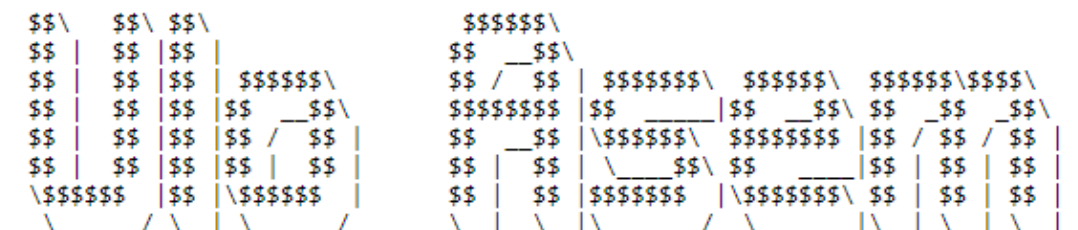
Gambar 2 Tampilan Awal Aplikasi Ulo Asem (Sumber: Penulis)

Tampilan pertama adalah tampilan awal yang menampilkan logo game Ulo Asem dengan menggunakan ASCII Art. Logo Ulo Asem dalam berbentuk tulisan ini akan ditampilkan dalam bentuk animasi. Selain itu, tampilan ini juga akan menampilkan informasi mengenai versi dan tahun pembuatan program Ulo Asem serta nama dan NIM pembuat aplikasi.

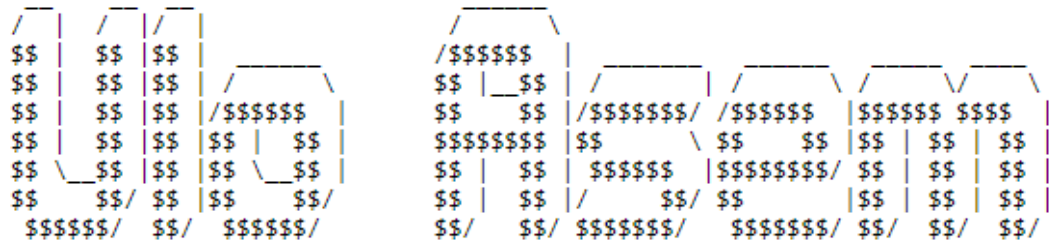
Berikut adalah rancangan tulisan Logo Ulo Asem yang ingin dianimasikan



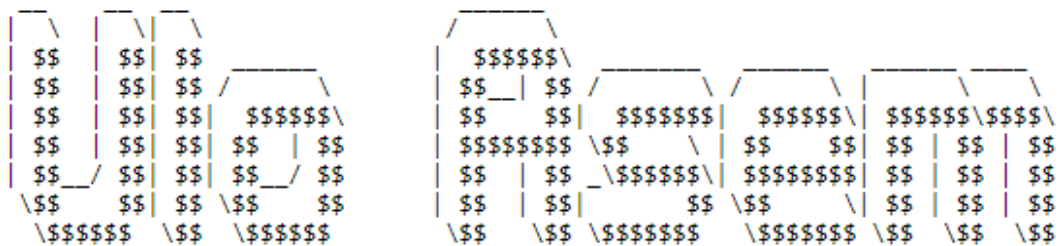
Gambar 3 Frame 1 logo Ulo Asem (Sumber: Penulis)



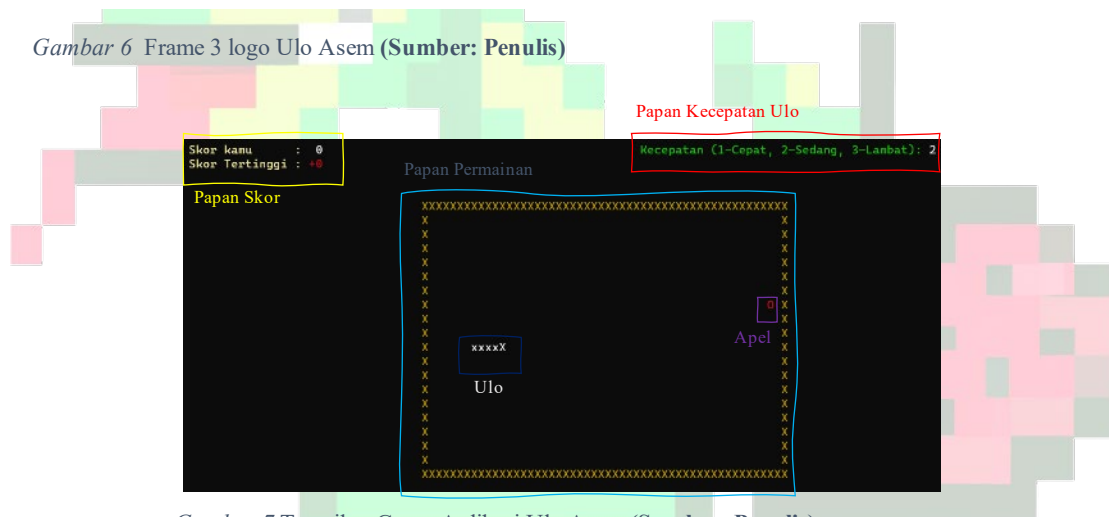
Gambar 4 Frame 2 logo Ulo Asem (Sumber: Penulis)



Gambar 5 Frame 3 logo Ulo Asem (Sumber: Penulis)



Gambar 6 Frame 3 logo Ulo Asem (Sumber: Penulis)



Gambar 7 Tampilan Game Aplikasi Ulo Asem (Sumber: Penulis)

Tampilan kedua adalah tampilan game, yang merupakan inti dari aplikasi ini. Tampilan ini akan menampilkan papan skor yang berisi skor yang sedang berlangsung di permainan, serta skor tertinggi dari permainan sebelumnya. Selain itu, tampilan ini juga akan menampilkan papan kecepatan ulo yang menunjukkan kecepatan ulo saat ini, serta papan permainan yang berisikan dinding, ulo, dan apel yang ditampilkan secara acak di dalam dinding permainan.

Gambar
Tengkorak dan
pesan Game Over



Gambar 8 Tampilan Game Over Aplikasi Ulo Asem (Sumber: Penulis)

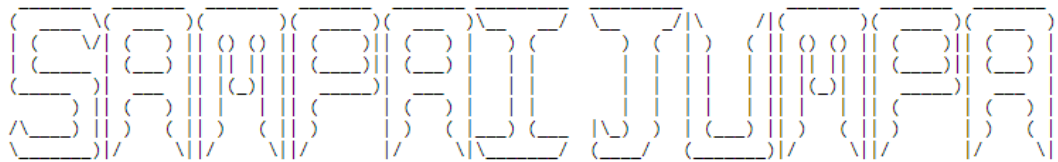
Tampilan ketiga adalah tampilan GameOver, yang akan ditampilkan ketika permainan berakhir. Tampilan ini akan menampilkan pesan GameOver dengan tulisan dan ASCII Art, serta menampilkan skor yang didapatkan selama permainan. Selain itu, tampilan ini juga akan menampilkan skor tertinggi yang pernah didapatkan selama permainan. Dengan tampilan antarmuka yang sederhana dan intuitif, pengguna akan dengan mudah mengoperasikan aplikasi Ulo Asem dan memahami cara bermain game.



Gambar 9 Tampilan Keluar Aplikasi Ulo Asem (Sumber: Penulis)

Tampilan terakhir adalah Tampilan Keluar adalah tampilan yang ditampilkan ketika aplikasi Ulo Asem ditutup. Di tampilan ini, terdapat pesan yang ditampilkan dengan ASCII Art bertuliskan "SAMPAI JUMPA" yang menandakan bahwa pengguna telah selesai menggunakan aplikasi.

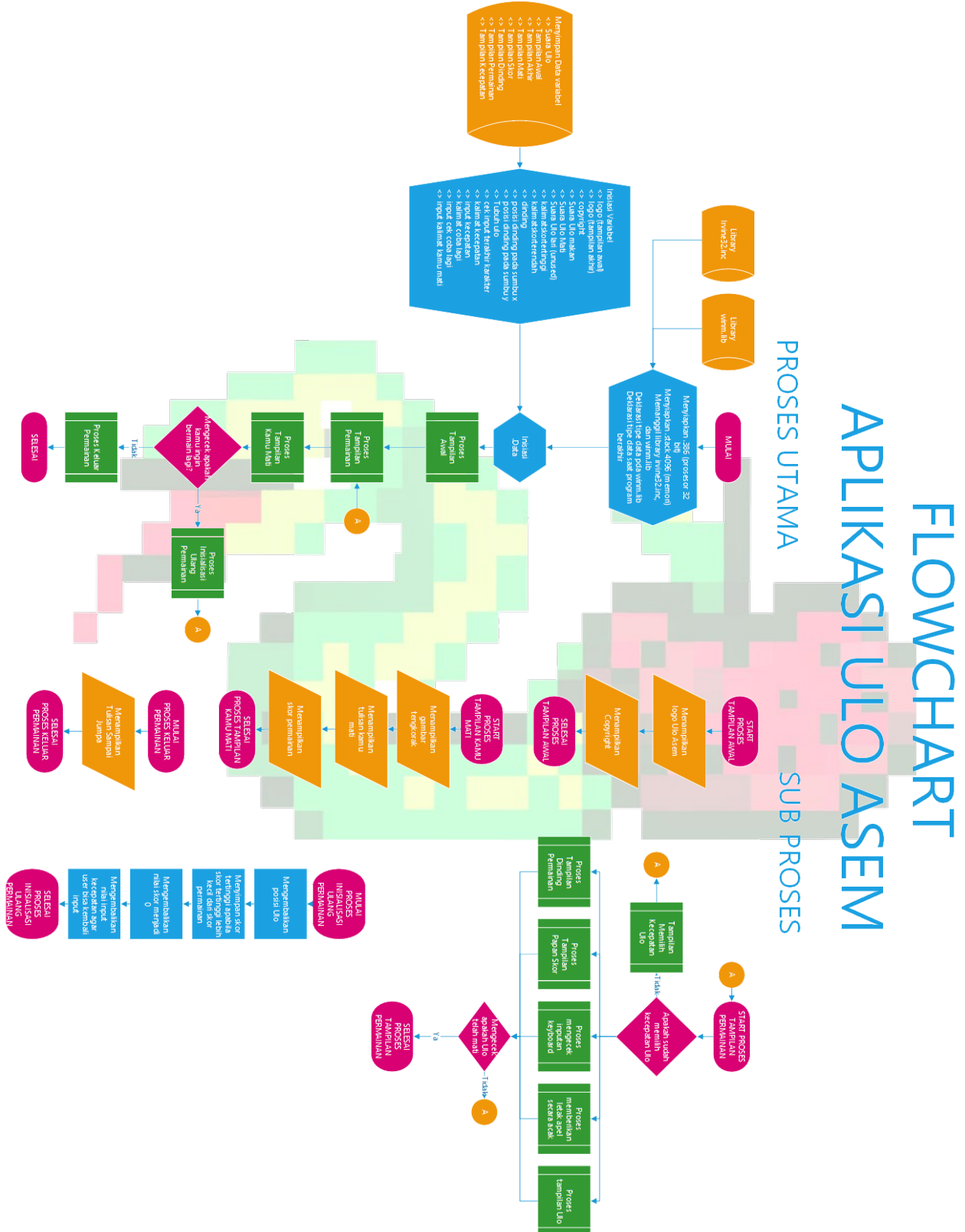
Berikut adalah rancangan tulisan SAMPAI JUMPA



Gambar 10 Rancangan tulisan "SAMPAI JUMPA" dengan ASCII ART



D. DIAGRAM PROSES APLIKASI



Gambar 11 Flowchart Ulo Asem (**Sumber: Penulis**)

Berikut adalah alur program aplikasi Ulo Asem

1. Memuat dan menampilkan interface permainan
 - Menggunakan library Irvine32 untuk memanggil Prosedur clrscr yang akan membersihkan layar
 - Memanggil Prosedur TampilanDinding untuk menggambar tembok pada layar permainan
 - Memanggil Prosedur TampilanPapanSkor untuk menampilkan papan skor pada layar permainan
 - Memanggil Prosedur MemilihKecepatanUlo untuk memberikan pemain pilihan kecepatan ulo
 - Memulai loop yang akan menampilkan ulo dengan panjang awal 5 segmen dengan Prosedur TampilanPemain
 - Memanggil Prosedur Randomize dari library Irvine32 untuk mengacak posisi apel yang akan muncul pada layar
 - Memanggil Prosedur BuatAcakApel untuk menentukan posisi apel yang akan muncul
 - Memanggil Prosedur TampilanApel untuk menampilkan apel pada layar permainan.
 - Loop akan terus berjalan dan menunggu input dari pemain sampai pemain memutuskan untuk keluar dari permainan dengan menekan tombol x atau sampai pemain mati.
2. Meminta input kecepatan Ulo dari pemain

Pada bagian ini, aplikasi akan meminta pemain untuk memasukkan kecepatan Ulo yang diinginkan pada awal permainan. Input yang dimasukkan akan digunakan sebagai parameter untuk menentukan seberapa cepat gerakan Ulo pada permainan. Setelah pemain memasukkan kecepatan yang diinginkan, aplikasi akan melanjutkan ke langkah berikutnya.

3. Menampilkan Ulo dengan panjang 5 segmen

Pada tahap ini, Ulo akan ditampilkan di layar dengan panjang 5 segmen. Setiap segmen ditampilkan dengan menggunakan koordinat x dan y yang disimpan dalam array xPosisi dan yPosisi. Posisi awal Ulo ditentukan secara acak di layar.

4. Menampilkan Apel secara acak di layar

Pada poin ini, aplikasi akan menampilkan apel secara acak di layar. Apel akan muncul di posisi yang tidak sama dengan posisi Ulo. Proses ini akan terus berulang setiap kali apel dimakan oleh Ulo.

5. Meminta input gerakan dari pemain

Pada poin ini, program akan meminta input dari pemain tentang arah gerakan Ulo. Input yang diperbolehkan adalah arah atas, bawah, kiri, dan kanan.

Setelah input diterima, program akan memperbarui posisi Ulo sesuai dengan arah yang dipilih pemain.

6. Memeriksa apakah input yang dimasukkan valid atau tidak

Dalam tahap ini, program akan memeriksa apakah input gerakan yang dimasukkan pemain merupakan salah satu dari empat arah yang tersedia, yaitu ke atas, ke bawah, ke kiri, atau ke kanan. Jika input yang dimasukkan valid, maka program akan melanjutkan ke tahap selanjutnya. Namun jika input yang dimasukkan tidak valid, program akan mengabaikan input tersebut dan terus menunggu input yang valid dari pemain.

7. Menggerakkan Ulo sesuai dengan input yang dimasukkan oleh pemain

Dalam tahap ini, program akan memeriksa apakah input yang diberikan pemain valid atau tidak. Jika input yang diberikan pemain valid, maka Ulo akan bergerak sesuai dengan arah yang diberikan pemain. Jika input yang diberikan pemain tidak valid, maka Ulo akan terus bergerak ke arah yang sama seperti sebelumnya. Misalnya, jika pemain memberikan input untuk bergerak ke kiri, tetapi Ulo sedang berada di ujung kanan layar, maka Ulo tidak akan bergerak ke kiri dan akan terus bergerak ke arah yang sama seperti sebelumnya.

8. Memeriksa apakah Ulo telah memakan Apel

Untuk tahap ini, program akan memeriksa apakah koordinat Ulo sama dengan koordinat Apel. Jika sama, maka Ulo akan memakan Apel dan panjang Ulo akan bertambah 1 segmen. Sebaliknya, jika tidak sama, maka Ulo tidak akan memakan Apel dan tidak ada perubahan pada panjang Ulo.

9. Jika Ulo telah memakan Apel, maka tambahkan 1 segmen pada Ulo dan menampilkan Apel baru secara acak

Tahap ini merupakan bagian dari permainan Ulo Asem di mana setelah Ulo berhasil memakan Apel, maka panjang Ulo akan ditambah sebanyak 1 segmen. Selain itu, Apel yang telah dimakan juga akan dihapus dan digantikan dengan Apel baru yang akan ditampilkan secara acak di layar permainan. Ini bertujuan untuk menjaga keseruan permainan, sehingga pemain harus terus-menerus mengontrol gerakan Ulo untuk memakan Apel yang muncul di layar.

10. Memeriksa apakah Ulo telah menabrak tembok atau menabrak dirinya sendiri

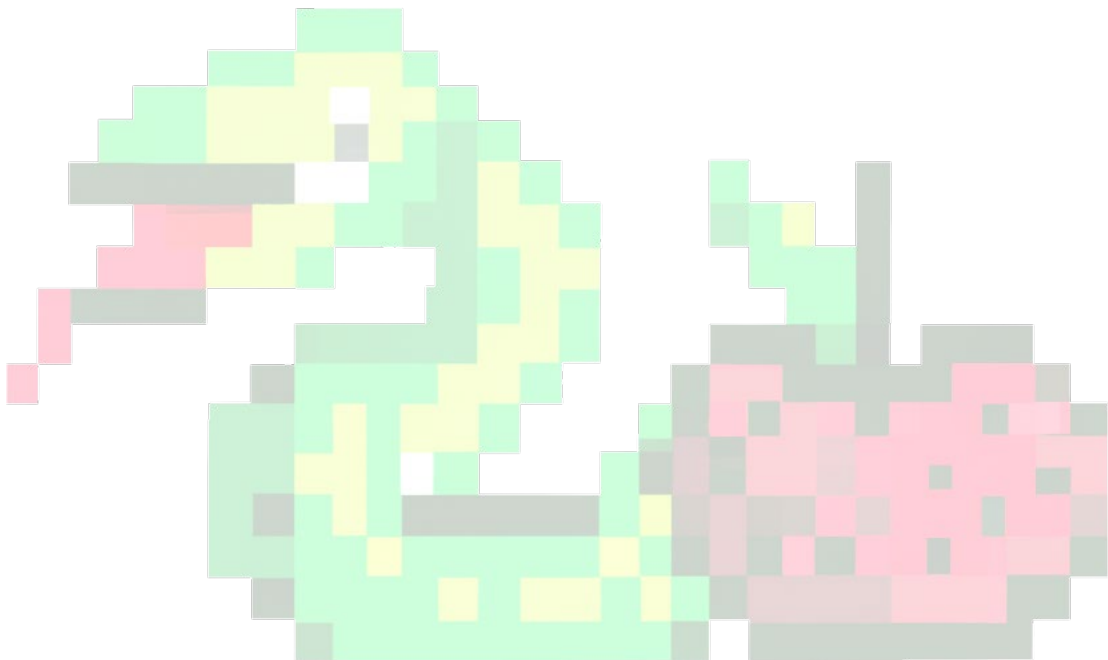
Tahap ini adalah tahap yang mengecek apakah Ulo telah menabrak tembok atau menabrak dirinya sendiri. Proses ini dilakukan dengan memeriksa posisi Ulo apakah sudah melewati batas layar atau menyentuh salah satu segmen tubuhnya sendiri. Jika kondisi tersebut terpenuhi, maka permainan akan dihentikan dan Ulo dinyatakan mati.

11. Jika Ulo telah menabrak tembok atau menabrak dirinya sendiri, maka permainan berakhir dan menampilkan skor akhir

Apabila Ulo telah menabrak tembok atau menabrak dirinya sendiri, maka permainan berakhir dan menampilkan skor akhir. Tahap ini bertujuan untuk menghentikan permainan jika kondisi tersebut terjadi. Skor akhir akan ditampilkan pada layar sebagai hasil dari permainan yang telah dimainkan oleh pemain.

12. Meminta input dari pemain untuk memainkan lagi atau keluar dari permainan

Jika pemain memilih untuk memainkan lagi, maka ulangi tahap 2 sampai 11. Jika pemain memilih untuk keluar dari permainan, maka program akan selesai dan menampilkan pesan "Sampai Jumpa".



E. PENJELASAN KODING/ SKRIP PROGRAM

1) INISIALISASI VARIABEL

Inisialisasi variabel dalam kode aplikasi Ulo Asem adalah proses pembuatan dan pemberian nilai awal pada variabel yang digunakan dalam aplikasi. Variabel yang diinisialisasi dapat berupa tipe data seperti integer, string, atau boolean. Pada kode aplikasi Ulo Asem, variabel yang diinisialisasi dapat digunakan untuk menyimpan data seperti input dari pengguna, hasil perhitungan, atau status dari aplikasi. Inisialisasi variabel juga dapat dilakukan pada saat aplikasi pertama kali dijalankan untuk mengatur kondisi awal dari aplikasi.

```
1  .386
2  .model flat, stdcall
3  .stack 4096
4  ExitProcess PROTO, dwExitCode: DWORD
5  INCLUDE Irvine32.inc
6  includelib Winmm.lib
7  PlaySound PROTO,
8      pszSound:PTR BYTE,
9      hmod:DWORD,
10     fdwSound:DWORD
```

Gambar 12 Cuplikan kodingan ke-1 Aplikasi Ulo Asem (Sumber: Penulis)

Ini adalah cuplikan koding dari bagian awal dari program Ulo Asem yang ditulis dalam bahasa Assembly. Pada bagian ini, terdapat beberapa perintah yang digunakan untuk mengatur model, stack, dan library yang digunakan dalam program.

Perintah ".386" digunakan untuk mengatur versi processor yang digunakan dalam program. Pada kasus ini, program ini dioptimalkan untuk digunakan pada processor dengan arsitektur 386 atau lebih tinggi.

Perintah ".model flat, stdcall" digunakan untuk mengatur model memory yang digunakan dalam program. "Flat" mengindikasikan bahwa program akan menggunakan memory model yang tidak terbagi-bagi, sedangkan "stdcall" mengindikasikan bahwa program akan menggunakan standar pemanggilan prosedur.

Perintah ".stack 4096" digunakan untuk mengatur ukuran stack yang digunakan dalam program. Dalam hal ini, ukuran stack yang digunakan adalah 4096 byte.

Perintah "ExitProcess PROTO, dwExitCode: DWORD" digunakan untuk mendeklarasikan prototipe fungsi ExitProcess yang digunakan untuk keluar dari

program. Parameter "dwExitCode" digunakan untuk menentukan kode keluar yang akan dikembalikan saat program keluar.

Perintah "INCLUDE Irvine32.inc" digunakan untuk memasukkan library Irvine32 yang berisi berbagai fungsi yang dapat digunakan dalam pemrograman assembly. Library ini dikembangkan oleh Dr. Kenneth L. Irvine dan menyediakan berbagai fungsi yang berguna untuk pemrograman assembly, seperti fungsi untuk menangani input-output, operasi matematika, dan lain-lain.

Perintah "includelib Winmm.lib" digunakan untuk memasukkan library Windows Multimedia (Winmm.lib) yang digunakan untuk mengelola audio dan musik. Dalam koding di atas, library ini digunakan untuk mengelola suara yang digunakan dalam aplikasi Ulo Asem.

Perintah "PlaySound PROTO, pszSound:PTR BYTE, hmod:DWORD, fdwSound:DWORD" digunakan untuk mendeklarasikan prototipe dari fungsi PlaySound. Fungsi ini digunakan untuk memutar suara yang ditentukan oleh parameter pszSound. Parameter hmod dan fdwSound digunakan untuk menentukan modul yang digunakan dan flag yang digunakan untuk mengontrol cara suara diputar.

```
12 .data
13 ; Mengatur suara
14 SuaraUloMakan BYTE "DeviceConnect",0
15 SuaraUloMati BYTE "DeviceDisconnect",0
16 SuaraUloLari BYTE "DeviceFail",0
17
18 SND_ALIAS DWORD 000000h
```

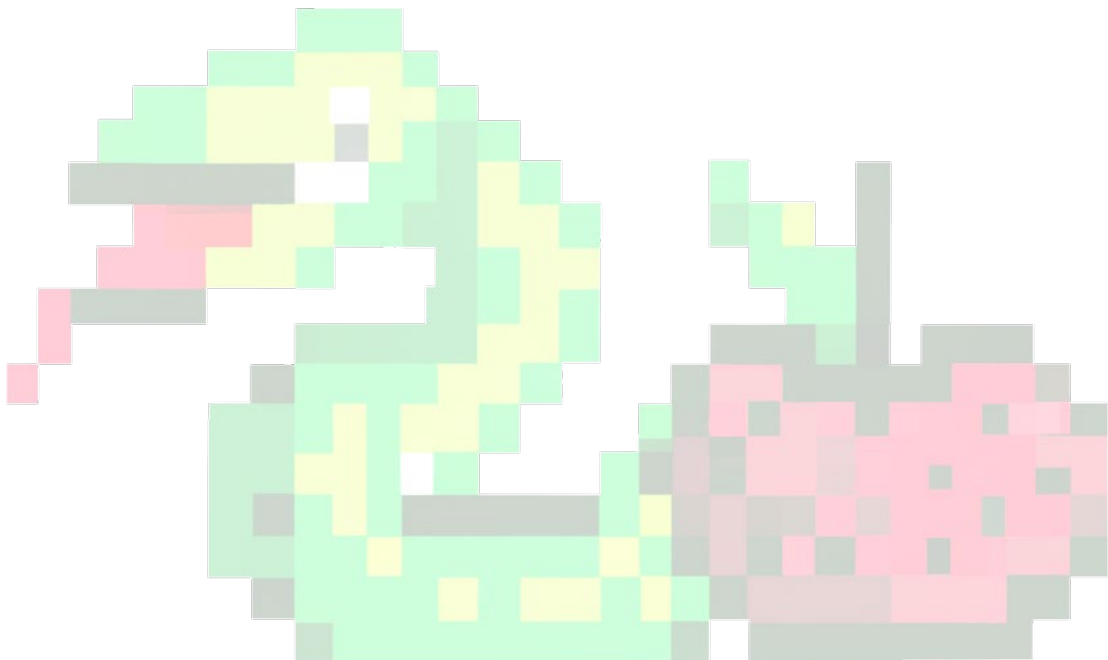
Gambar 13 Cuplikan kodingan ke-2 Aplikasi Ulo Asem (Sumber: Penulis)

Perintah "SuaraUloMakan BYTE "DeviceConnect",0" digunakan untuk mendefinisikan suara yang akan diputar ketika ular berhasil memakan apel dalam permainan. Perintah "SuaraUloMati BYTE "DeviceDisconnect",0" digunakan untuk mendefinisikan suara yang akan diputar ketika ular mati dalam permainan. Perintah "SuaraUloLari BYTE "DeviceFail",0" digunakan untuk mendefinisikan suara yang akan diputar ketika ular bergerak dalam permainan. Perintah "SND_ALIAS DWORD 000000h" digunakan untuk mengatur suara yang akan diputar menjadi suara yang telah didefinisikan sebelumnya.

Perintah "INCLUDE Irvine32.inc" digunakan untuk memasukkan library Irvine32 yang berisi berbagai fungsi yang dapat digunakan dalam program Assembly. Perintah "includelib Winmm.lib" digunakan untuk memasukkan library Winmm yang berisi fungsi-fungsi multimedia yang dapat digunakan dalam program Assembly, seperti fungsi untuk memutar suara. Perintah "PlaySound

PROTO," digunakan untuk mendefinisikan prototipe dari fungsi PlaySound yang akan digunakan dalam program untuk memutar suara.

Perintah "includelib Winmm.lib" digunakan untuk memasukkan library Winmm yang berisi fungsi-fungsi multimedia seperti pemutaran suara. Fungsi PlaySound digunakan untuk memutar suara sesuai dengan event yang terjadi dalam permainan, seperti Ulo makan, Ulo mati, dan Ulo bergerak. Variabel SuaraUloMakan, SuaraUloMati, dan SuaraUloLari digunakan untuk menyimpan nama suara yang akan dimainkan. Variabel SND_ALIAS digunakan untuk menentukan tipe suara yang akan dimainkan, yaitu suara sistem.



[illegible]

Gambar 14 Cuplikan kodingan ke-3 Aplikasi Ulo Asem (Sumber: Penulis)

Bagian koding yang ditunjukkan di sini adalah bagian yang menangani tampilan logo dari aplikasi Ulo Asem. Ditunjukkan dengan variabel yang diawali dengan "logo_", yang masing-masing menyimpan sebuah string yang ditampilkan sebagai ASCII art. Setiap variabel tersebut menyimpan string yang digunakan untuk menampilkan sebuah frame dari logo Ulo Asem. Di sini, terdapat 4 frame yang digunakan dalam menampilkan logo Ulo Asem, yang ditampilkan secara bergantian untuk memberikan efek animasi pada logo tersebut.

Bagian tersebut menampilkan juga variabel untuk menyimpan string copyright dari aplikasi yang dibuat. Pada bagian ini, ditampilkan informasi bahwa aplikasi ini

adalah versi 1.0 dan dibuat oleh seseorang dengan nama Mohammad Farid Hendianto, serta tahun pembuatan aplikasi yaitu 2022 dan nomor identitas pembuatnya yaitu 2200018401.

Selain terdapat ascii art yang digunakan untuk menampilkan logo game Ulo Asem pada tampilan awal dan copyright, terdapat juga tampilan untuk game over, dan tampilan akhir. Ascii art digunakan untuk menampilkan gambar dengan menggunakan karakter-karakter dari tabel ASCII. Di sini, telah didefinisikan beberapa variabel yang berisi baris-baris dari karakter ASCII yang digunakan untuk menampilkan logo Ulo Asem, pesan "Game Over", dan pesan "Sampai Jumpa" pada tampilan akhir. Di samping itu, juga didefinisikan ascii art dari tengkorak sebagai tampilan game over.

Tampilan game over menggunakan ASCII art yang menggambarkan sebuah tengkorak. ASCII art ini dibentuk dari beberapa baris yang ditentukan dalam variabel tengkorak_1 sampai tengkorak_12. Baris-baris tersebut digabungkan untuk membentuk gambar tengkorak yang akan ditampilkan saat pesan "Game Over" ditampilkan.

```

91
92     xDinding BYTE 52 DUP("X"),0
93
94     strSkor BYTE "Skor kamu      : ",0
95     Skor BYTE 0
96     strSkorTertinggi BYTE "Skor Tertinggi : ",0
97     SkorTertinggi BYTE 0

```

Gambar 15 Cuplikan kodingan ke-4 Aplikasi Ulo Asem (Sumber: Penulis)

Bagian ini adalah deklarasi variabel untuk digunakan dalam aplikasi. Variabel xDinding digunakan untuk menampilkan dinding permainan dengan karakter "X" sebanyak 52 kali. Variabel strSkor digunakan untuk menampilkan teks "Skor kamu: " saat menampilkan skor saat ini, sedangkan variabel Skor digunakan untuk menyimpan nilai skor saat ini. Variabel strSkorTertinggi digunakan untuk menampilkan teks "Skor Tertinggi: " saat menampilkan skor tertinggi, dan variabel SkorTertinggi digunakan untuk menyimpan nilai skor tertinggi.

```

98
99     strCobaLagi BYTE "Coba Lagi? 1=Ya, 0=Tidak",0
100     SalahInput BYTE "Input Salah",0
101     strKamumati BYTE "KAMU MATI ",0
102     strtampilanskor BYTE " point(s)",0
103     blank BYTE "

```

Gambar 16 Cuplikan kodingan ke-5 Aplikasi Ulo Asem (Sumber: Penulis)

Pada bagian kode ini digunakan beberapa variabel untuk menyimpan string yang akan ditampilkan di aplikasi. Variabel strSkor digunakan untuk menyimpan string "Skor kamu : ", dan digunakan saat aplikasi menampilkan skor yang didapatkan saat bermain. Variabel Skor digunakan untuk menyimpan skor yang didapatkan saat bermain. Variabel strSkorTertinggi digunakan untuk menyimpan string "Skor Tertinggi : ", yang digunakan saat aplikasi menampilkan skor tertinggi yang pernah didapatkan. Variabel SkorTertinggi digunakan untuk menyimpan skor tertinggi yang pernah didapatkan. Variabel strCobaLagi digunakan untuk

menyimpan string "Coba Lagi? 1=Ya, 0=Tidak", yang akan ditampilkan saat aplikasi menanyakan kepada pengguna apakah ingin mencoba lagi. Variabel SalahInput digunakan untuk menyimpan string "Input Salah", yang akan ditampilkan saat pengguna memasukkan input yang salah. Variabel strKamumati digunakan untuk menyimpan string "KAMU MATI", yang akan ditampilkan saat Ulo mati. Variabel strtampilanskor digunakan untuk menyimpan string " point(s)", yang digunakan untuk menampilkan skor pada saat gameover. Variabel ini digunakan untuk memperjelas bahwa skor yang didapat dalam permainan adalah dalam bentuk poin, dan digunakan setelah skor didapat ditampilkan. Variabel blank digunakan untuk mengatur jarak pada saat menampilkan skor dan pesan pada saat gameover. Variabel ini digunakan untuk menjaga agar tampilan tetap rapi dan teratur. Variabel blank digunakan untuk menyimpan string " ", yang digunakan untuk mengosongkan tampilan pada saat menampilkan skor dan skor tertinggi.

```

105     Ulo BYTE "X", 104 DUP("x")
106
107     xPosisi BYTE 45,44,43,42,41, 100 DUP(?)
108     yPosisi BYTE 15,15,15,15,15, 100 DUP(?)
109
110     xPosisiDinding BYTE 34,34,85,85
111     yPosisiDinding BYTE 5,24,5,24
112
113     xApelPos BYTE ?
114     yApelPos BYTE ?
115
116     inputChar BYTE "+"
117     TerakhirInputKarakter BYTE ?
118
119     strKecepatan BYTE "Kecepatan (1-Cepat, 2-Sedang, 3-Lambat): ",0
120     Kecepatan     DWORD 0
121

```

Gambar 17 Cuplikan kodingan ke-6 Aplikasi Ulo Asem (Sumber: Penulis)

Variabel "Ulo" digunakan untuk menyimpan string "X" yang digunakan sebagai tampilan dari ular. Kemudian diikuti oleh 104 karakter "x", yang digunakan untuk mengisi bagian tubuh ular yang lain.

Variabel xPosisi dan yPosisi digunakan untuk menyimpan posisi dari setiap bagian tubuh ular dalam koordinat x dan y. Kemudian diikuti oleh 100 karakter "?", yang digunakan sebagai placeholder untuk posisi tubuh ular yang akan datang.

Variabel xPosisiDinding dan yPosisiDinding digunakan untuk menyimpan posisi dinding dari permainan dalam koordinat x dan y. xPosisiDinding menyimpan posisi dinding dari AtasKiri, BawahKiri, AtasKanan, dan BawahKanan, sementara yPosisiDinding menyimpan posisi dinding dari Atas, Bawah.

Variabel xApelPos dan yApelPos digunakan untuk menyimpan posisi apel dalam koordinat x dan y.

Variabel `inputChar` digunakan untuk menyimpan karakter input dari user saat permainan berlangsung. Variabel ini diinisialisasi dengan karakter "+" yang digunakan sebagai tanda permainan baru.

Variabel `TerakhirInputKarakter` digunakan untuk menyimpan karakter input terakhir dari user.

Variabel `strKecepatan` digunakan untuk menyimpan string "Kecepatan (1-Cepat, 2-Sedang, 3-Lambat): " yang digunakan untuk menampilkan pilihan kecepatan permainan pada saat permainan baru dimulai. Variabel `Kecepatan` digunakan untuk menyimpan nilai kecepatan yang dipilih oleh user.

Kecepatan adalah variabel yang digunakan untuk menyimpan nilai yang menentukan kecepatan gerak dari Ulo. Nilai ini akan ditentukan oleh user melalui input. Nilai yang dapat ditentukan oleh user adalah 1, 2, dan 3, yang masing-masing menunjukkan kecepatan cepat, sedang, dan lambat. Setelah user memasukkan input, nilai akan disimpan di dalam variabel `Kecepatan` untuk digunakan dalam perhitungan gerak Ulo.

2) PROSEDUR PROSES UTAMA

Pada bagian kode ini, digunakan `Prosedur TampilanAwal` yang digunakan untuk menampilkan logo permainan yang ditampilkan saat awal permainan. Pertama, dilakukan pemanggilan fungsi `Gotoxy` yang digunakan untuk menentukan posisi cursor saat menuliskan teks. Kemudian, dilakukan pemanggilan fungsi `WriteString` untuk menuliskan teks pada posisi cursor yang telah ditentukan. Selain itu, digunakan juga fungsi `SetTextColor` untuk menentukan warna teks yang akan dituliskan. Setelah semua teks dituliskan, dilakukan delay selama 250 milisecond sebelum menampilkan logo yang kedua. Proses ini dilakukan berulang hingga semua teks yang ditampilkan di logo permainan ditampilkan. Setelah itu, dilakukan pemanggilan fungsi untuk menampilkan string `copyright_a` dan `copyright_b` yang menyimpan informasi tentang pembuat dan tahun pembuatan permainan. Kemudian, dilakukan delay selama 250 milisecond sebelum melanjutkan ke proses selanjutnya.

122	.code	196	mov dl,25	274	mov dl,25
123	Utana PROC	197	mov dh,6	275	mov dh,7
124	TampilanAwal PROC	198	call Gotoxy	276	call Gotoxy
125	mov dl,25	199	mov edx, OFFSET logo_a_1	277	mov edx, OFFSET logo_b_2
126	mov dh,6	200	mov eax, yellow	278	mov eax, red
127	call Gotoxy	201	call SetTextColor	279	call SetTextColor
128	mov edx, OFFSET logo_a_0	202	call WriteString	280	call WriteString
129	mov eax, yellow	203		281	
130	call SetTextColor	204	mov dl,25	282	mov dl,25
131	call WriteString	205	mov dh,7	283	mov dh,8
132		206	call Gotoxy	284	call Gotoxy
133	mov dl,25	207	mov edx, OFFSET logo_b_1	285	mov edx, OFFSET logo_c_2
134	mov dh,7	208	mov eax, red	286	mov eax, yellow
135	call Gotoxy	209	call SetTextColor	287	call SetTextColor
136	mov edx, OFFSET logo_b_0	210	call WriteString	288	call WriteString
137	mov eax, red	211		289	
138	call SetTextColor	212	mov dl,25	290	mov dl,25
139	call WriteString	213	mov dh,8	291	mov dh,9
140		214	call Gotoxy	292	call Gotoxy
141	mov dl,25	215	mov edx, OFFSET logo_c_1	293	mov edx, OFFSET logo_e_2
142	mov dh,8	216	mov eax, yellow	294	mov eax, red
143	call Gotoxy	217	call SetTextColor	295	call SetTextColor
144	mov edx, OFFSET logo_c_0	218	call WriteString	296	call WriteString
145	mov eax, yellow	219		297	
146	call SetTextColor	220	mov dl,25	298	mov dl,25
147	call WriteString	221	mov dh,9	299	mov dh,10
148		222	call Gotoxy	300	call Gotoxy
149	mov dl,25	223	mov edx, OFFSET logo_e_1	301	mov edx, OFFSET logo_f_2
150	mov dh,9	224	mov eax, red	302	mov eax, yellow
151	call Gotoxy	225	call SetTextColor	303	call SetTextColor
152	mov edx, OFFSET logo_e_0	226	call WriteString	304	call WriteString
153	mov eax, red	227		305	
154	call SetTextColor	228	mov dl,25	306	mov dl,25
155	call WriteString	229	mov dh,10	307	mov dh,11
156		230	call Gotoxy	308	call Gotoxy
157	mov dl,25	231	mov edx, OFFSET logo_f_1	309	mov edx, OFFSET logo_g_2
158	mov dh,10	232	mov eax, yellow	310	mov eax, red
159	call Gotoxy	233	call SetTextColor	311	call SetTextColor
160	mov edx, OFFSET logo_f_0	234	call WriteString	312	call WriteString
161	mov eax, yellow	235		313	
162	call SetTextColor	236	mov dl,25	314	mov dl,25
163	call WriteString	237	mov dh,11	315	mov dh,12
164		238	call Gotoxy	316	call Gotoxy
165	mov dl,25	239	mov edx, OFFSET logo_g_1	317	mov edx, OFFSET logo_h_2
166	mov dh,11	240	mov eax, red	318	mov eax, yellow
167	call Gotoxy	241	call SetTextColor	319	call SetTextColor
168	mov edx, OFFSET logo_g_0	242	call WriteString	320	call WriteString
169	mov eax, red	243		321	
170	call SetTextColor	244	mov dl,25	322	mov dl,37
171	call WriteString	245	mov dh,12	323	mov dh,14
172		246	call Gotoxy	324	call Gotoxy
173	mov dl,25	247	mov edx, OFFSET logo_h_1	325	mov edx, OFFSET copyright_a
174	mov dh,12	248	mov eax, yellow	326	mov eax, red (red*2)
175	call Gotoxy	249	call SetTextColor	327	call SetTextColor
176	mov edx, OFFSET logo_h_0	250	call WriteString	328	call WriteString
177	mov eax, yellow	251		329	mov dl,37
178	call SetTextColor	252	mov dl,37	330	mov dh,15
179	call WriteString	253	mov dh,14	331	call Gotoxy
180		254	call Gotoxy	332	mov edx, OFFSET copyright_b
181	mov dl,37	255	mov edx, OFFSET copyright_a	333	call WriteString
182	mov dh,14	256	mov eax, red (red*2)	334	mov eax,250
183	call Gotoxy	257	call SetTextColor	335	call Delay
184	mov edx, OFFSET copyright_a	258	call WriteString	336	
185	mov eax, red (red*2)	259		337	
186	call SetTextColor	260	mov dl,15	338	mov dl,25
187	call WriteString	261	call Gotoxy	339	mov dh,6
188	mov dl,37	262	mov edx, OFFSET copyright_b	340	call Gotoxy
189	mov dh,15	263	call WriteString	341	mov edx, OFFSET logo_a_3
190	call Gotoxy	264	mov eax,250	342	mov eax, yellow
191	mov edx, OFFSET copyright_b	265	call Delay	343	call SetTextColor
192	call WriteString	266		344	call WriteString
193	mov eax,250	267	mov dl,25	345	
194	call Delay	268	mov dh,6	346	mov dl,25
195		269	call Gotoxy	347	mov dh,7
		270	mov edx, OFFSET logo_a_2	348	call Gotoxy
		271	mov eax, yellow	349	mov edx, OFFSET logo_b_3
		272	call SetTextColor	350	mov eax, red
		273	call WriteString	351	call SetTextColor
				352	call WriteString
				353	
				354	mov dl,25
				355	mov dh,8
				356	call Gotoxy
				357	mov edx, OFFSET logo_c_3
				358	mov eax, yellow
				359	call SetTextColor
				360	call WriteString
				361	

361		448	mov dl,25	534	
362	mov dl,25	449	mov dh,11	535	mov dl,37
363	mov dh,9	450	call Gotoxy	536	mov dh,14
364	call Gotoxy	451	mov edx, OFFSET logo_g_0	537	call Gotoxy
365	mov edx, OFFSET logo_e_3	452	mov eax, red	538	mov edx, OFFSET copyright_a
366	mov eax, red	453	call SetTextColor	539	mov eax, red (red*2)
367	call SetTextColor	454	call WriteString	540	call SetTextColor
368	call WriteString	455		541	call WriteString
369		456	mov dl,25	542	mov dl,37
370	mov dl,25	457	mov dh,12	543	mov dh,15
371	mov dh,10	458	call Gotoxy	544	call Gotoxy
372	call Gotoxy	459	mov edx, OFFSET logo_h_0	545	mov edx, OFFSET copyright_b
373	mov edx, OFFSET logo_f_3	460	mov eax, yellow	546	call WriteString
374	mov eax, yellow	461	call SetTextColor	547	mov eax,250
375	call SetTextColor	462	call WriteString	548	call Delay
376	call WriteString	463		549	
377		464	mov dl,37	550	mov dl,25
378	mov dl,25	465	mov dh,14	551	mov dh,6
379	mov dh,11	466	call Gotoxy	552	call Gotoxy
380	call Gotoxy	467	mov edx, OFFSET copyright_a	553	mov edx, OFFSET logo_a_2
381	mov edx, OFFSET logo_g_3	468	mov eax, red (red*2)	554	mov eax, yellow
382	mov eax, red	469	call SetTextColor	555	call SetTextColor
383	call SetTextColor	470	call WriteString	556	call WriteString
384	call WriteString	471	mov dl,37	557	
385		472	mov dh,15	558	mov dl,25
386	mov dl,25	473	call Gotoxy	559	mov dh,7
387	mov dh,12	474	mov edx, OFFSET copyright_b	560	call Gotoxy
388	call Gotoxy	475	call WriteString	561	mov edx, OFFSET logo_b_2
389	mov edx, OFFSET logo_h_3	476	mov eax,250	562	mov eax, red
390	mov eax, yellow	477	call Delay	563	call SetTextColor
391	call SetTextColor	478		564	call WriteString
392	call WriteString	479	mov dl,25	565	
393		480	mov dh,6	566	mov dl,25
394	mov dl,37	481	call Gotoxy	567	mov dh,8
395	mov dh,14	482	mov edx, OFFSET logo_a_1	568	call Gotoxy
396	call Gotoxy	483	mov eax, yellow	569	mov edx, OFFSET logo_c_2
397	mov edx, OFFSET copyright_a	484	call SetTextColor	570	mov eax, yellow
398	mov eax, red (red*2)	485	call WriteString	571	call SetTextColor
399	call SetTextColor	486		572	call WriteString
400	call WriteString	487	mov dl,25	573	
401	mov dl,37	488	mov dh,7	574	mov dl,25
402	mov dh,15	489	call Gotoxy	575	mov dh,9
403	call Gotoxy	490	mov edx, OFFSET logo_b_1	576	call Gotoxy
404	mov edx, OFFSET copyright_b	491	mov eax, red	577	mov edx, OFFSET logo_e_2
405	call WriteString	492	call SetTextColor	578	mov eax, red
406	mov eax,250	493	call WriteString	579	call SetTextColor
407	call Delay	494		580	call WriteString
408	mov dl,25	495	mov dl,25	581	
409	mov dh,6	496	mov dh,8	582	mov dl,25
410	call Gotoxy	497	call Gotoxy	583	mov dh,10
411	mov edx, OFFSET logo_a_0	498	mov edx, OFFSET logo_c_1	584	call Gotoxy
412	mov eax, yellow	499	mov eax, yellow	585	mov edx, OFFSET logo_f_2
413	call SetTextColor	500	call SetTextColor	586	mov eax, yellow
414	call WriteString	501	call WriteString	587	call SetTextColor
415		502		588	call WriteString
416	mov dl,25	503	mov dl,25	589	
417	mov dh,7	504	mov dh,9	590	mov dl,25
418	call Gotoxy	505	call Gotoxy	591	mov dh,11
419	mov edx, OFFSET logo_b_0	506	mov edx, OFFSET logo_e_1	592	call Gotoxy
420	mov eax, red	507	mov eax, red	593	mov edx, OFFSET logo_g_2
421	call SetTextColor	508	call SetTextColor	594	mov eax, red
422	call WriteString	509	call WriteString	595	call SetTextColor
423		510		596	call WriteString
424	mov dl,25	511	mov dl,25	597	
425	mov dh,8	512	mov dh,10	598	mov dl,25
426	call Gotoxy	513	call Gotoxy	599	mov dh,12
427	mov edx, OFFSET logo_c_0	514	mov edx, OFFSET logo_f_1	600	call Gotoxy
428	mov eax, yellow	515	mov eax, yellow	601	mov edx, OFFSET logo_h_2
429	call SetTextColor	516	call SetTextColor	602	mov eax, yellow
430	call WriteString	517	call WriteString	603	call SetTextColor
431		518		604	call WriteString
432	mov dl,25	519	mov dl,25	605	
433	mov dh,9	520	mov dh,11	606	mov dl,37
434	call Gotoxy	521	call Gotoxy	607	mov dh,14
435	mov edx, OFFSET logo_e_0	522	mov edx, OFFSET logo_g_1	608	call Gotoxy
436	mov eax, red	523	mov eax, red	609	mov edx, OFFSET copyright_a
437	call SetTextColor	524	call SetTextColor	610	mov eax, red (red*2)
438	call WriteString	525	call WriteString	611	call SetTextColor
439		526		612	call WriteString
440	mov dl,25	527	mov dl,25	613	mov dl,37
441	mov dh,10	528	mov dh,12	614	mov dh,15
442	call Gotoxy	529	call Gotoxy	615	call Gotoxy
443	mov edx, OFFSET logo_f_0	530	mov edx, OFFSET logo_h_1	616	mov edx, OFFSET copyright_b
444	mov eax, yellow	531	mov eax, yellow	617	call WriteString
445	call SetTextColor	532	call SetTextColor	618	mov eax,250
446	call WriteString	533	call WriteString	619	call Delay

621	mov dl,25	715	mov dl,25	819	mov dl,37
622	mov dh,6	716	mov dl,25	820	mov dh,14
623	call Gotoxy	717	mov dh,9	821	call Gotoxy
624	mov edx, OFFSET logo_a_3	718	call Gotoxy	822	mov edx, OFFSET copyright_a
625	mov eax, yellow	719	mov edx, OFFSET logo_e_8	823	mov eax, red (red*2)
626	call SetTextColor	720	mov eax, red	824	call SetTextColor
627	call WriteString	721	call SetTextColor	825	call WriteString
628		722	call WriteString	826	mov dl,37
629	mov dl,25	723		827	mov dh,15
630	mov dh,7	724	mov dl,25	828	call Gotoxy
631	call Gotoxy	725	mov dh,10	829	mov edx, OFFSET copyright_b
632	mov edx, OFFSET logo_b_3	726	call Gotoxy	830	call WriteString
633	mov eax, red	727	mov edx, OFFSET logo_f_8	831	mov eax,250
634	call SetTextColor	728	mov eax, yellow	832	call Delay
635	call WriteString	729	call SetTextColor	833	
636		730	call WriteString	834	mov dl,25
637	mov dl,25	731		835	mov dh,6
638	mov dh,8	732	mov dl,25	836	call Gotoxy
639	call Gotoxy	733	mov dh,11	837	mov edx, OFFSET logo_a_2
640	mov edx, OFFSET logo_c_3	734	call Gotoxy	838	mov eax, yellow
641	mov eax, yellow	735	mov edx, OFFSET logo_g_8	839	call SetTextColor
642	call SetTextColor	736	mov eax, red	840	call WriteString
643	call WriteString	737	call SetTextColor	841	
644		738	call WriteString	842	mov dl,25
645	mov dl,25	739		843	mov dh,7
646	mov dh,9	740	mov dl,25	844	call Gotoxy
647	call Gotoxy	741	mov dh,12	845	mov edx, OFFSET logo_b_2
648	mov edx, OFFSET logo_e_3	742	call Gotoxy	846	mov eax, red
649	mov eax, red	743	mov edx, OFFSET logo_h_8	847	call SetTextColor
650	call SetTextColor	744	mov eax, yellow	848	call WriteString
651	call WriteString	745	call SetTextColor	849	
652		746	call WriteString	850	mov dl,25
653	mov dl,25	747		851	mov dh,8
654	mov dh,10	748	mov dl,37	852	call Gotoxy
655	call Gotoxy	749	mov dh,14	853	mov edx, OFFSET logo_c_2
656	mov edx, OFFSET logo_f_3	751	call Gotoxy	854	mov eax, yellow
657	mov eax, yellow	752	mov edx, OFFSET copyright_a	855	call SetTextColor
658	call SetTextColor	753	mov eax, red (red*2)	856	call WriteString
659	call WriteString	754	call SetTextColor	857	
660		755	call WriteString	858	mov dl,25
661	mov dl,25	756	mov dl,37	859	mov dh,9
662	mov dh,11	757	mov dh,15	860	call Gotoxy
663	call Gotoxy	758	call Gotoxy	861	mov edx, OFFSET logo_e_2
664	mov edx, OFFSET logo_g_3	759	mov edx, OFFSET copyright_b	862	mov eax, red
665	mov eax, red	760	call WriteString	863	call SetTextColor
666	call SetTextColor	761	mov eax,250	864	call WriteString
667	call WriteString	762	call Delay	865	
668		763	mov dl,25	866	mov dl,25
669	mov dl,25	764	mov dh,6	867	mov dh,10
670	mov dh,12	765	call Gotoxy	868	call Gotoxy
671	call Gotoxy	766	mov edx, OFFSET logo_a_1	869	mov edx, OFFSET logo_f_2
672	mov edx, OFFSET logo_h_3	767	mov eax, yellow	870	mov eax, yellow
673	mov eax, yellow	768	call SetTextColor	871	call SetTextColor
674	call SetTextColor	769	call WriteString	872	call WriteString
675	call WriteString	770		873	
676		771	mov dl,25	874	mov dl,25
677	mov dl,37	772	mov dh,7	875	mov dh,11
678	mov dh,14	773	call Gotoxy	876	call Gotoxy
679	call Gotoxy	774	mov edx, OFFSET logo_b_1	877	mov edx, OFFSET logo_g_2
680	mov edx, OFFSET copyright_a	775	mov eax, red	878	mov eax, red
681	mov eax, red (red*2)	776	call SetTextColor	879	call SetTextColor
682	call SetTextColor	777	call WriteString	880	call WriteString
683	call WriteString	778		881	
684	mov dl,37	779	mov dl,25	882	mov dl,25
685	mov dh,15	780	mov dh,8	883	mov dh,12
686	call Gotoxy	781	call Gotoxy	884	call Gotoxy
687	mov edx, OFFSET copyright_b	782	mov edx, OFFSET logo_c_1	885	mov edx, OFFSET logo_h_2
688	call WriteString	783	mov eax, yellow	886	mov eax, yellow
689	mov eax,250	784	call SetTextColor	887	call SetTextColor
690	call Delay	785	call WriteString	888	call WriteString
691		786		889	
692	mov dl,25	787	mov dl,25	890	mov dl,37
693	mov dh,6	788	mov dh,9	891	mov dh,14
694	call Gotoxy	789	call Gotoxy	892	call Gotoxy
695	mov edx, OFFSET logo_a_8	790	mov edx, OFFSET logo_e_1	893	mov edx, OFFSET copyright_a
696	mov eax, yellow	791	mov eax, red	894	mov eax, red (red*2)
697	call SetTextColor	792	call SetTextColor	895	call SetTextColor
698	call WriteString	793	call WriteString	896	call WriteString
699		794		897	mov dl,37
700	mov dl,25	795	mov dl,25	898	mov dh,15
701	mov dh,7	796	mov dh,10	899	call Gotoxy
702	call Gotoxy	797	call Gotoxy	900	mov edx, OFFSET copyright_b
703	mov edx, OFFSET logo_b_8	798	mov edx, OFFSET logo_f_1	901	call WriteString
704	mov eax, red	799	mov eax, yellow	902	mov eax,250
705	call SetTextColor	800	call SetTextColor	903	call Delay
706	call WriteString	801	call WriteString	904	
707		802		905	mov dl,25
708	mov dl,25	803	mov dl,25	906	mov dh,6
709	mov dh,8	804	mov dh,11	907	call Gotoxy
710	call Gotoxy	805	call Gotoxy	908	mov edx, OFFSET logo_a_3
711	mov edx, OFFSET logo_c_8	806	mov edx, OFFSET logo_g_1	909	mov eax, yellow
712	mov eax, yellow	807	mov eax, red	910	call SetTextColor
713	call SetTextColor	808	call SetTextColor	911	call WriteString
714	call WriteString	809	call WriteString	912	


```

913    mov dl,25
914    mov dh,7
915    call Gotoxy
916    mov edx, OFFSET logo_b_3
917    mov eax, red
918    call SetTextColor
919    call WriteString
920
921    mov dl,25
922    mov dh,8
923    call Gotoxy
924    mov edx, OFFSET logo_c_3
925    mov eax, yellow
926    call SetTextColor
927    call WriteString
928
929    mov dl,25
930    mov dh,9
931    call Gotoxy
932    mov edx, OFFSET logo_e_3
933    mov eax, red
934    call SetTextColor
935    call WriteString
936
937    mov dl,25
938    mov dh,10
939    call Gotoxy
940    mov edx, OFFSET logo_f_3
941    mov eax, yellow
942    call SetTextColor
943    call WriteString
944
945    mov dl,25
946    mov dh,11
947    call Gotoxy
948    mov edx, OFFSET logo_g_3
949    mov eax, red
950    call SetTextColor
951    call WriteString
952
953    mov dl,25
954    mov dh,12
955    call Gotoxy
956    mov edx, OFFSET logo_h_3
957    mov eax, yellow
958    call SetTextColor
959    call WriteString
960
961    mov dl,37
962    mov dh,14
963    call Gotoxy
964    mov edx, OFFSET copyright_a
965    mov eax, red (red*2)
966    call SetTextColor
967    call WriteString
968    mov dl,37
969    mov dh,15
970    call Gotoxy
971    mov edx, OFFSET copyright_b
972    call WriteString
973    mov eax, 250
974    call Delay
975
976    mov eax, white
977    call SetTextColor
978    call clrscr
979    TampilanAwal ENDP

```

Gambar 18 Cuplikan kodingan ke-7 Aplikasi Ulo Asem (Sumber: Penulis)

Pada bagian kode ini, digunakan Prosedur TampilanAwal yang digunakan untuk menampilkan logo dan copyright pada saat permainan dimulai.

Syntax yang digunakan pada kode ini adalah:

- mov: digunakan untuk memindahkan data dari register atau memory ke register lain
- dl, dh: digunakan untuk menentukan posisi x dan y dari cursor saat menulis string
- call: digunakan untuk memanggil Prosedur atau function yang telah didefinisikan sebelumnya, seperti Gotoxy, SetTextColor, dan WriteString
- OFFSET: digunakan untuk mengambil alamat memory dari suatu variable
- eax: register yang digunakan untuk menyimpan data
- yellow, red (red*2): digunakan untuk memberikan warna pada string yang ditulis
- Delay: digunakan untuk memberikan delay selama 250 milisecond

Secara garis besar, kode ini digunakan untuk menampilkan logo pada saat permainan dimulai dan memberikan warna pada logo dan copyright. Kemudian menggunakan call Gotoxy untuk menentukan posisi dari cursor, lalu menggunakan call WriteString untuk menuliskan string pada posisi yang telah ditentukan. Dan di akhir menggunakan Delay untuk memberikan delay selama 250 milisecond.

Pada bagian kode ini, digunakan beberapa prosedur yang digunakan untuk menampilkan logo dan copyright pada saat awal permainan.

Prosedur TampilanAwal digunakan untuk menampilkan logo sebelum permainan dimulai. Pada prosedur ini, digunakan beberapa perintah seperti mov, call, dan delay untuk mengatur posisi dan tampilan logo.

Pertama, digunakan perintah mov untuk mengatur posisi cursor pada koordinat (dl, dh) dan digunakan perintah call Gotoxy untuk menempatkan cursor pada posisi tersebut. Kemudian, digunakan perintah mov edx, OFFSET untuk menyimpan alamat dari setiap baris logo yang akan ditampilkan. Setelah itu, digunakan perintah call WriteString untuk menampilkan logo pada posisi yang telah ditentukan sebelumnya.

Selain itu, digunakan perintah call SetTextColor untuk mengubah warna tulisan logo dan perintah delay untuk menunda waktu sebelum logo ditampilkan.

Setelah logo ditampilkan, copyright ditampilkan dengan cara yang sama dengan logo yaitu dengan mengatur posisi cursor, menyimpan alamat dari setiap baris copyright, menampilkan copyright dan mengubah warna tulisan.

Setelah proses penampilan logo dan copyright selesai, prosedur TampilanAwal akan selesai juga.

3) Prosedur Proses Utama

```

981     Permainan::
982         call TampilanDinding
983         call TampilanPapanSkor
984         call MemilihKecepatanUlo
985
986         mov esi,0
987         mov ecx,5
988     TampilanUlo:
989         call TampilanPemain
990         inc esi
991     loop TampilanUlo
992         call Randomize
993         call BuatAcakApel
994         call TampilanApel

```

Gambar 19 Cuplikan kodingan ke-8 Aplikasi Ulo Asem (Sumber: Penulis)

Pada bagian kode ini, terdapat beberapa prosedur yang dijalankan yaitu TampilanDinding, TampilanPapanSkor, MemilihKecepatanUlo, TampilanUlo, dan TampilanApel. Prosedur TampilanDinding digunakan untuk menampilkan tembok pada layar, TampilanPapanSkor digunakan untuk menampilkan papan skor pada layar, MemilihKecepatanUlo digunakan untuk memberikan pemain pilihan untuk memilih kecepatan Ulo, TampilanUlo digunakan untuk menampilkan Ulo dengan panjang 5, dan

TampilanApel digunakan untuk menampilkan Apel pada layar. Pada TampilanUlo, digunakan loop dengan instruksi loop TampilanUlo dan digunakan mov esi,0 dan mov ecx,5 sebagai counter untuk mengatur jumlah panjang Ulo yang akan ditampilkan. Kemudian digunakan pula prosedur Randomize dan BuatAcakApel yang digunakan untuk mengacak posisi Apel yang akan ditampilkan.

```

996     PermainanLoop:
997         mov dl,106
998         mov dh,1
999         call Gotoxy
1000
1001         ; mendapatkan input key dari pemain
1002         call ReadKey
1003         jz noKey
1004         processInput:
1005         mov bl, inputChar
1006         mov TerakhirInputKarakter, bl
1007         mov inputChar, al
1008
1009         noKey:
1010         cmp inputChar, "x"
1011         je KeluarPermainan
1012
1013         cmp inputChar, "w"
1014         je CekAtas
1015
1016         cmp inputChar, "s"
1017         je CekBawah
1018
1019         cmp inputChar, "a"
1020         je CekKiri
1021
1022         cmp inputChar, "d"
1023         je CekKanan
1024         jne PermainanLoop
1025
1026         ; Cek keyboard (w,a,s,d) untuk bisa bergerak
1027         CekBawah:
1028         cmp TerakhirInputKarakter, "w"
1029         je JanganGantiArah
1030         mov cl, yPosisiDinding[1]
1031         dec cl
1032         cmp yPosisi[0], cl
1033         jnl PindahBawah
1034         je Mati
1035
1036         CekKiri:
1037         cmp TerakhirInputKarakter, "a"
1038         je JanganPergikiri
1039         cmp TerakhirInputKarakter, "d"
1040         je JanganGantiArah
1041         mov cl, xPosisiDinding[0]
1042         inc cl
1043         cmp xPosisi[0], cl
1044         jg PindahKiri
1045         je Mati
1046
1047         CekKanan:
1048         cmp TerakhirInputKarakter, "a"
1049         je JanganGantiArah
1050         mov cl, xPosisiDinding[2]
1051         dec cl
1052         cmp xPosisi[0], cl
1053         jnl PindahKanan
1054         je Mati
1055
1056         CekAtas:
1057         cmp TerakhirInputKarakter, "s"
1058         je JanganGantiArah
1059         mov cl, yPosisiDinding[0]
1060         inc cl
1061         cmp yPosisi, cl
1062         jg PindahAtas
1063         je Mati
1064
1065     PindahAtas:
1066         mov eax, Kecepatan
1067         add eax, Kecepatan
1068         call delay
1069         mov esi, 0
1070         call PerbaruiPemain
1071         mov ah, yPosisi[esi]
1072         mov al, xPosisi[esi]
1073         dec yPosisi[esi]
1074         call TampilanPemain
1075         call Tampilantubuh
1076         call CekUlo
1077
1078     PindahBawah:
1079         mov eax, Kecepatan
1080         add eax, Kecepatan
1081         call delay
1082         mov esi, 0
1083         call PerbaruiPemain
1084         mov ah, yPosisi[esi]
1085         mov al, xPosisi[esi]
1086         inc yPosisi[esi]
1087         call TampilanPemain
1088         call Tampilantubuh
1089         call CekUlo
1090
1091     PindahKiri:
1092         mov eax, Kecepatan
1093         call delay
1094         mov esi, 0
1095         call PerbaruiPemain
1096         mov ah, yPosisi[esi]
1097         mov al, xPosisi[esi]
1098         dec xPosisi[esi]
1099         call TampilanPemain
1100         call Tampilantubuh
1101         call CekUlo
1102
1103     PindahKanan:
1104         mov eax, Kecepatan
1105         call delay
1106         mov esi, 0
1107         call PerbaruiPemain
1108         mov ah, yPosisi[esi]
1109         mov al, xPosisi[esi]
1110         inc xPosisi[esi]
1111         call TampilanPemain
1112         call Tampilantubuh
1113         call CekUlo
1114
1115     CekApel::
1116         mov esi, 0
1117         mov bl, xPosisi[0]
1118         cmp bl, xApelPos
1119         jne PermainanLoop
1120         mov bl, yPosisi[0]
1121         cmp bl, yApelPos
1122         jne PermainanLoop
1123         call MemakanApel
1124
1125     jmp PermainanLoop

```

Gambar 20 Cuplikan kodingan ke-9 Aplikasi Ulo Asem (Sumber: Penulis)

Pada bagian kode ini, digunakan label `PermainanLoop` untuk melakukan perulangan yang akan terus berjalan selama permainan berlangsung. Pada dalam perulangan tersebut, digunakan beberapa instruksi seperti `"call ReadKey"` untuk membaca input dari keyboard yang diberikan pemain, `"cmp inputChar,"x"` untuk mengecek apakah pemain ingin keluar dari permainan dengan menekan tombol x, dan `"cmp inputChar,"w"` untuk mengecek apakah pemain ingin bergerak ke atas. Selanjutnya, ada juga instruksi seperti `"CekBawah: cmp TerakhirInputKarakter,"w"` yang digunakan untuk mengecek apakah pemain baru saja bergerak ke atas sebelumnya, sehingga tidak bisa bergerak ke bawah langsung setelah itu.

Selain itu, terdapat juga `"CekKiri"` dan `"CekKanan"` yang digunakan untuk mengecek apakah pemain ingin bergerak ke kiri atau kanan. Pada setiap perintah gerakan, digunakan juga perintah seperti `"mov cl, yPosisiDinding[1]"` untuk mengambil posisi dinding dari bawah dan `"cmp yPosisi[0],cl"` untuk membandingkan posisi Ulo dengan posisi dinding. Jika posisi Ulo sama atau kurang dari posisi dinding, maka Ulo akan mati dan permainan akan berakhir. Selain itu, jika pemain menekan tombol x, maka permainan akan segera berakhir dan akan ditampilkan skor dan opsi untuk mencoba lagi.

Prosedur `CekAtas`, `CekBawah`, `CekKiri`, dan `CekKanan` digunakan untuk mengecek apakah Ulo akan bergerak ke atas, bawah, kiri, atau kanan. Pada setiap Prosedur, terdapat pengecekan untuk menentukan apakah Ulo akan mati atau tidak. Jika Ulo akan mati, maka program akan langsung pindah ke Prosedur `Mati`. Jika Ulo tidak akan mati, maka program akan pindah ke Prosedur `PindahAtas`, `PindahBawah`, `PindahKiri`, atau `PindahKanan`.

Prosedur `CekAtas`, `CekBawah`, `CekKiri`, dan `CekKanan` juga digunakan untuk mencegah Ulo berbalik arah secara tiba-tiba. Misalnya, jika pemain baru saja bergerak ke atas dan segera menekan tombol ke bawah, maka Ulo tidak akan bergerak ke bawah dan tetap bergerak ke atas. Ini dilakukan dengan mengecek variabel `TerakhirInputKarakter` yang menyimpan input terakhir dari pemain.

Prosedur `TampilanDinding` digunakan untuk menampilkan dinding pada permainan. Posisi dinding ditentukan oleh `xPosisiDinding` dan `yPosisiDinding` yang sudah didefinisikan sebelumnya. Prosedur ini akan menggunakan loop untuk menampilkan dinding secara berulang-ulang sesuai dengan jumlah dinding yang ditentukan dalam `xPosisiDinding` dan `yPosisiDinding`. Setiap kali loop dijalankan, posisi dinding akan diupdate dan ditampilkan pada layar. Juga, Prosedur `TampilanPapanSkor` digunakan untuk menampilkan skor dan skor tertinggi pada permainan, yang diperoleh dari variabel `Skor` dan `SkorTertinggi` yang sudah didefinisikan sebelumnya.

```
1133      JanganGantiArah:
1134      mov inputChar, bl
1135      jmp noKey
1136
1137      JanganPergiKiri:
1138      mov inputChar, "+"
1139      jmp PermainanLoop
1140
1141      Mati::
1142      call KamuMati
1143
1144      BermainLagi::
1145      call InisialisasiUlangPermainan
```

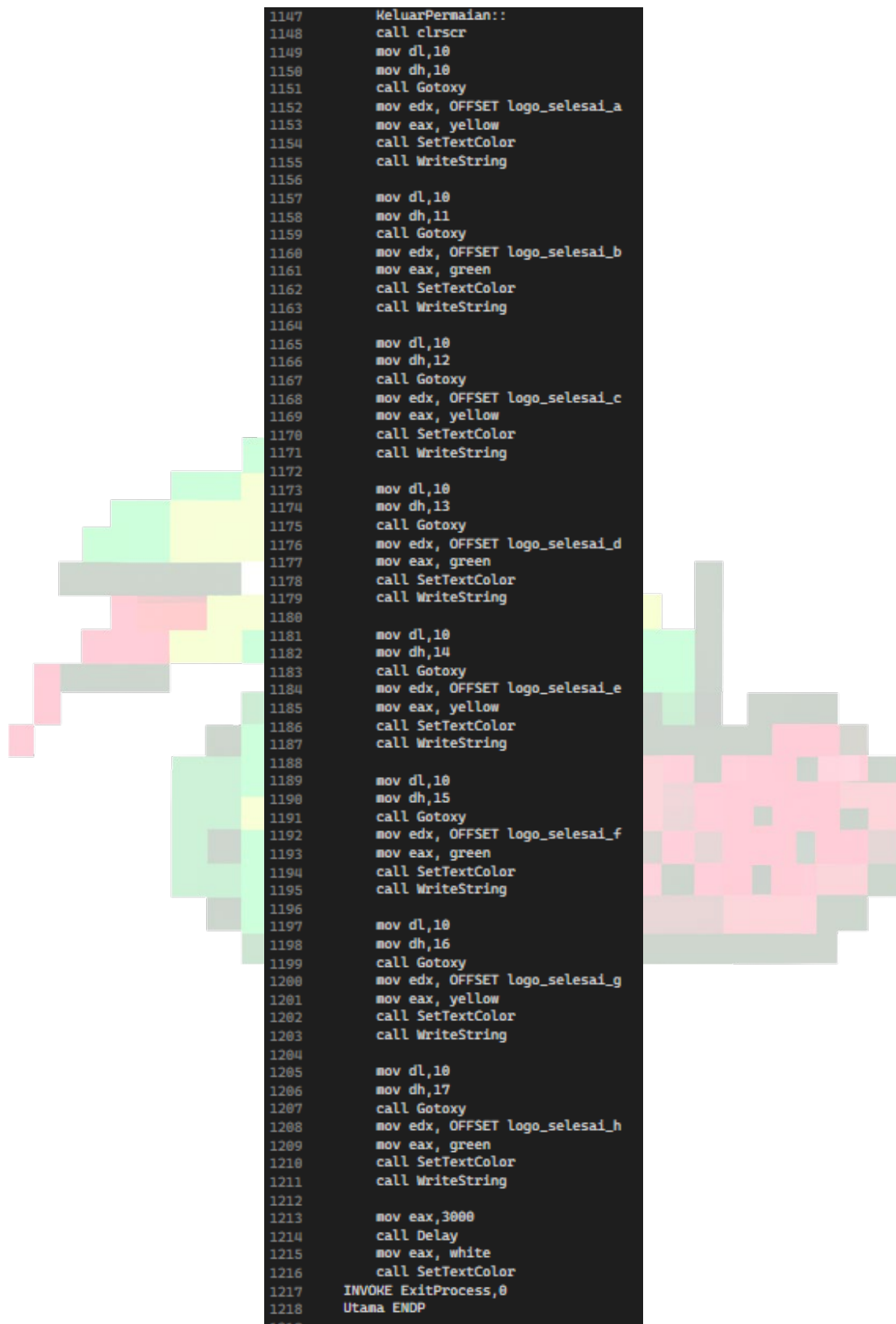
Gambar 21 Cuplikan kodingan ke-10 Aplikasi Ulo Asem (Sumber: Penulis)

Procedure `JanganGantiArah` digunakan untuk mencegah pemain dari mengubah arah gerak Ulo saat permainan sedang berlangsung. Jika pemain mencoba untuk mengubah arah gerak Ulo dengan menekan tombol yang berlawanan dengan arah yang sedang diterapkan, maka input akan diabaikan dan Ulo akan tetap bergerak dalam arah yang sama.

Procedure `JanganPergiKiri` digunakan untuk mencegah Ulo untuk bergerak ke Kiri pada saat permainan baru dimulai. Jika pemain mencoba untuk bergerak ke Kiri pada saat permainan baru dimulai, maka input akan diabaikan dan Ulo akan tetap bergerak ke Kanan.

Procedure `Mati` digunakan untuk menampilkan pesan "Kamu Mati" pada layar dan mengakhiri permainan.

Procedure `BermainLagi` digunakan untuk menginisialisasi ulang semua variabel dan mengulangi permainan dari awal.



Gambar 22 Cuplikan kodingan ke-11 Aplikasi Ulo Asem (Sumber: Penulis)

Kode tersebut digunakan untuk menampilkan logo "Sampai Jumpa" pada saat permainan selesai. Logo tersebut ditentukan oleh beberapa variabel string yang diberi nama `logo_selesai_a` sampai `logo_selesai_h`. Setiap variabel tersebut diisi dengan karakter yang digunakan untuk membentuk tulisan "Sampai Jumpa" yang terdiri dari beberapa baris.

Posisi Gotoxy yang digunakan untuk menentukan koordinat posisi tulisan yang ingin ditampilkan. Dalam kode ini, posisi ditentukan dengan menentukan nilai `dl` dan `dh`. Dalam setiap perintah Gotoxy, `dl` digunakan untuk menentukan posisi horizontal (koordinat x) dan `dh` digunakan untuk menentukan posisi vertikal (koordinat y).

Setelah posisi ditentukan, perintah `WriteString` digunakan untuk menampilkan tulisan yang ditentukan oleh variabel `edx` yang diisi dengan offset dari variabel yang digunakan untuk membentuk logo.

Perintah ini diikuti dengan perintah `SetTextColor` untuk menentukan warna tulisan yang akan ditampilkan. Setiap baris tulisan memiliki warna yang berbeda yaitu yellow, green, dan white.

Proses yang dilakukan pada kode tersebut adalah menampilkan logo selesai yang terdiri dari 8 baris yang ditulis dengan ASCII "Sampai Jumpa" pada posisi (10,10) sampai (10,17) di layar console.

Setiap baris yang ditampilkan memiliki offset yang berbeda yaitu `logo_selesai_a`, `logo_selesai_b`, `logo_selesai_c`, dst. Offset digunakan untuk menentukan lokasi dari variabel atau label dalam memori.

Setelah semua baris ditampilkan, warna teks akan diubah menjadi hijau dan kuning dengan menggunakan perintah `SetTextColor` dan digunakan delay selama 3000 mili second. Kemudian program akan dihentikan dengan perintah `ExitProcess`.


```

1220
1221     TampilanDinding PROC
1222         mov dl,xPosisiDinding[0]
1223         mov dh,yPosisiDinding[0]
1224         call Gotoxy
1225         mov edx,OFFSET xDinding
1226         mov eax, brown
1227         call SetTextColor
1228         call WriteString
1229
1230         ; Tampilan dinding atas
1231
1232         mov dl,xPosisiDinding[1]
1233         mov dh,yPosisiDinding[1]
1234         call Gotoxy
1235         mov edx,OFFSET xDinding
1236         call WriteString
1237
1238         mov dl, xPosisiDinding[2]
1239         mov dh, yPosisiDinding[2]
1240         mov eax,"X"
1241         inc yPosisiDinding[3]
1242
1243     L11:
1244         call Gotoxy
1245         call WriteChar
1246         inc dh
1247         cmp dh, yPosisiDinding[3]
1248         jl L11
1249
1250         mov dl, xPosisiDinding[0]
1251         mov dh, yPosisiDinding[0]
1252         mov eax,"X"
1253     L12:
1254         call Gotoxy
1255
1256         call WriteChar
1257         inc dh
1258         cmp dh, yPosisiDinding[3]
1259         jl L12
1260         ret
1261
1262     TampilanDinding ENDP

```

Gambar 23 Cuplikan kodingan ke-12 Aplikasi Ulo Asem (Sumber: Penulis)

Pada bagian kode di atas, ada sebuah prosedur bernama "TampilanDinding" yang digunakan untuk menampilkan dinding pada layar. Prosedur ini menggunakan beberapa perintah Gotoxy dan WriteString yang digunakan untuk mengatur posisi dan menuliskan karakter "X" sebagai simbol dinding.

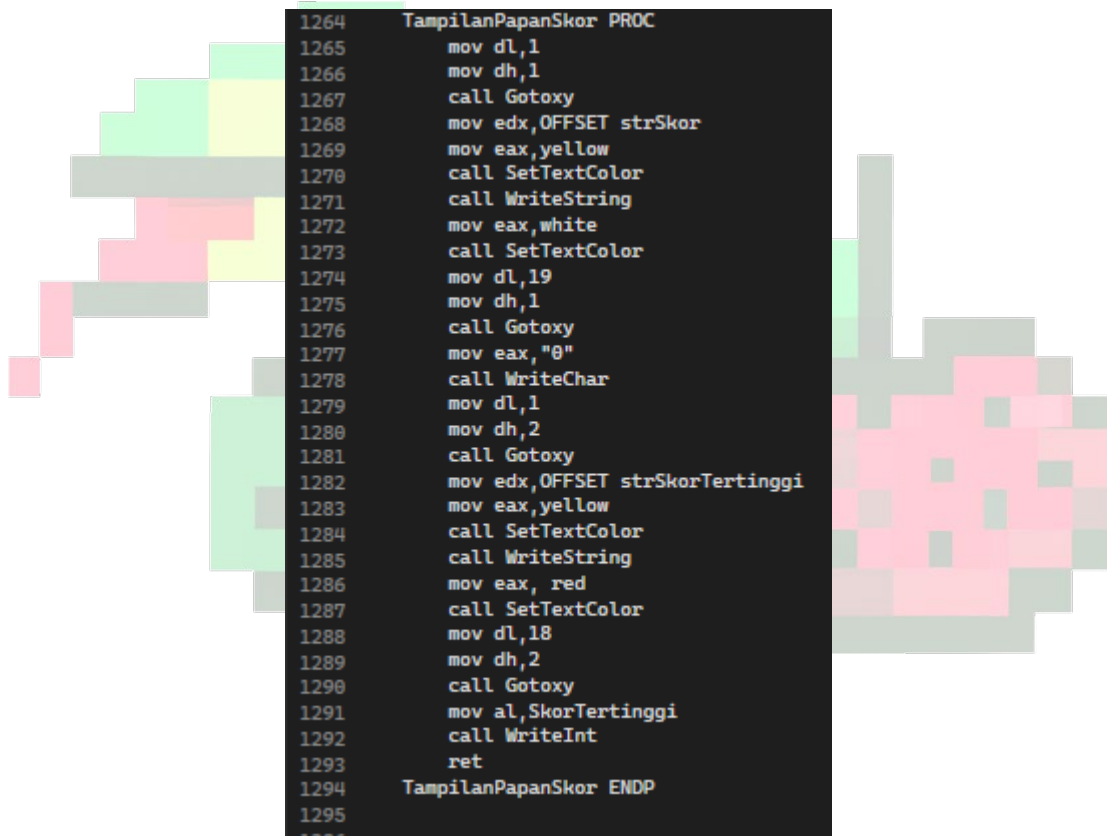
Pertama, perintah Gotoxy digunakan untuk mengatur posisi cursor di koordinat xPosisiDinding[0], yPosisiDinding[0]. Kemudian, perintah SetTextColor digunakan untuk memberikan warna pada dinding. Setelah itu, perintah WriteString digunakan untuk menuliskan string "X" sebagai simbol dinding.

Selanjutnya, perintah Gotoxy digunakan lagi untuk mengatur posisi cursor di koordinat xPosisiDinding[1], yPosisiDinding[1], dan perintah WriteString digunakan untuk menuliskan string "X" sebagai simbol dinding pada posisi tersebut.

Setelah itu, perintah Gotoxy digunakan untuk mengatur posisi cursor di koordinat xPosisiDinding[2], yPosisiDinding[2], dan perintah WriteChar digunakan untuk menuliskan karakter "X" sebagai simbol dinding. Kemudian, perintah loop digunakan untuk mengulang proses ini sampai posisi y cursor mencapai yPosisiDinding[3].

Kemudian, perintah Gotoxy digunakan lagi untuk mengatur posisi cursor di koordinat xPosisiDinding[0], yPosisiDinding[0], dan perintah WriteChar digunakan untuk menuliskan karakter "X" sebagai simbol dinding. Kemudian, perintah loop digunakan untuk mengulang proses ini sampai posisi y cursor mencapai yPosisiDinding[3].

Pada akhir prosedur, perintah ret digunakan untuk mengakhiri prosedur dan kembali ke alamat yang memanggil prosedur.



Gambar 24 Cuplikan kodingan ke-13 Aplikasi Ulo Asem (Sumber: Penulis)

Kutipan kode di atas adalah prosedur yang digunakan untuk menampilkan papan skor pada permainan. Prosedur ini dimulai dengan memanggil fungsi Gotoxy untuk menentukan posisi kursor pada koordinat (1,1) dan mencetak string "Skor: " pada posisi tersebut. Kemudian, ditampilkan juga nilai skor saat ini yang diinisialisasi dengan 0. Selanjutnya, posisi kursor dipindahkan ke koordinat (1,2) dan mencetak string "Skor Tertinggi: " dan nilai skor tertinggi yang disimpan dalam

variabel SkorTertinggi. Setelah itu, warna teks diatur kembali menjadi default (putih) dan prosedur selesai dengan perintah ret.

```

1297     MemilihKecepatanUlo PROC
1298         mov edx,0
1299         mov dl,65
1300         mov dh,1
1301         call Gotoxy
1302         mov edx,OFFSET strKecepatan
1303         mov eax,lightGreen
1304         call SetTextColor
1305         call WriteString
1306         mov eax,white
1307         call SetTextColor
1308         mov esi, 40
1309         mov eax,0
1310         call readInt
1311         cmp ax,1
1312         jl invalidKecepatan
1313         cmp ax, 3
1314         jg invalidKecepatan
1315         mul esi
1316         mov Kecepatan, eax
1317         ret
1318
1319     invalidKecepatan:
1320         mov dl,105
1321         mov dh,1
1322         call Gotoxy
1323         mov edx, OFFSET SalahInput
1324         call WriteString
1325         mov ax, 1500
1326         call delay
1327         mov dl,105
1328         mov dh,1
1329         call Gotoxy
1330         mov edx, OFFSET blank
1331         call writeString
1332         call TampilanPapanSkor
1333         call MemilihKecepatanUlo
1334         ret
1335     MemilihKecepatanUlo ENDP
1336

```

Gambar 25 Cuplikan kodingan ke-14 Aplikasi Ulo Asem (Sumber: Penulis)

Pada prosedur MemilihKecepatanUlo, digunakan untuk meminta input dari pemain untuk memilih kecepatan Ulo yang diinginkan. Pada bagian awal, digunakan perintah Gotoxy untuk menempatkan cursor pada posisi tertentu di layar dan kemudian mencetak string "Memilih Kecepatan : " menggunakan prosedur WriteString. Kemudian, digunakan perintah readInt untuk membaca input dari pemain dan menyimpannya ke dalam variabel eax. Selanjutnya, digunakan perintah cmp untuk membandingkan nilai yang diterima dari input pemain dengan 1 dan 3. Jika input pemain di luar jangkauan ini, maka program akan jump ke label invalidKecepatan. Pada label ini, digunakan perintah Gotoxy untuk menempatkan cursor pada posisi tertentu dan mencetak string "Input Salah" menggunakan prosedur WriteString. Kemudian, digunakan perintah delay untuk menunda program selama 1,5 detik sebelum menghapus pesan error dan memanggil prosedur MemilihKecepatanUlo lagi. Setelah input yang valid diterima, program akan

mengalikan nilai input dengan 40 (yang merupakan perbedaan milisecond setiap level kecepatan) dan menyimpan nilai tersebut ke dalam variabel Kecepatan. Kemudian, prosedur akan diakhiri dengan perintah ret.

```

1337     TampilanPemain PROC
1338         mov dl,xPosisi[esi]
1339         mov dh,yPosisi[esi]
1340         call Gotoxy
1341         mov dl, al
1342         mov al, Ulo[esi]
1343         call WriteChar
1344         mov al, dl
1345         ret
1346     TampilanPemain ENDP

```

Gambar 26 Cuplikan kodingan ke-15 Aplikasi Ulo Asem (Sumber: Penulis)

Bagian ini adalah prosedur yang digunakan untuk menampilkan karakter pemain pada posisi xPosisi[esi] dan yPosisi[esi]. Prosedur ini menggunakan fungsi Gotoxy untuk menentukan posisi dari karakter pemain. Kemudian, karakter pemain disimpan dalam register al dan ditampilkan menggunakan fungsi WriteChar. Setelah itu, register al di-restore ke nilai sebelumnya sebagai sementara register. Prosedur ini kemudian akan mengembalikan kontrol ke program utama setelah selesai dijalankan.

```

1348     PerbaruiPemain PROC
1349         mov dl,xPosisi[esi]
1350         mov dh,yPosisi[esi]
1351         call Gotoxy
1352         mov dl, al
1353         mov al, " "
1354         call WriteChar
1355         mov al, dl
1356         ret
1357     PerbaruiPemain ENDP

```

Gambar 27 Cuplikan kodingan ke-16 Aplikasi Ulo Asem (Sumber: Penulis)

Kode di atas adalah prosedur yang digunakan untuk menghapus posisi sebelumnya dari pemain sebelum menampilkan posisi baru dari pemain. Prosedur ini dikenal sebagai "PerbaruiPemain".

Pertama, prosedur ini mengambil posisi x dan y dari pemain yang disimpan dalam array xPosisi dan yPosisi yang ditunjuk oleh register esi. Kemudian, prosedur memanggil prosedur Gotoxy untuk mengatur posisi cursor pada posisi x dan y yang diambil.

Kemudian, prosedur menyimpan sementara nilai dari register al ke dl. Lalu, register al diisi dengan karakter "spasi" untuk menghapus posisi sebelumnya dari pemain. Prosedur WriteChar digunakan untuk mencetak karakter "spasi" pada posisi x dan y yang diambil.

Setelah itu, nilai dari register al diisi kembali dengan nilai yang disimpan sementara di register dl dengan perintah "mov al, dl" sebelum prosedur diakhiri dengan perintah "ret". Dengan demikian, tampilan Pemain di posisi (xPosisi, yPosisi) akan dihapus dan siap untuk digerakkan ke posisi baru.

```

1359      TampilanApel PROC
1360          mov eax,red (red * 0)
1361          call SetTextColor
1362          mov dl,xApelPos
1363          mov dh,yApelPos
1364          call Gotoxy
1365          mov al,"O"
1366          call WriteChar
1367          mov eax,white (black * 16)
1368          call SetTextColor
1369          ret
1370      TampilanApel ENDP

```

Gambar 28 Cuplikan kodingan ke-17 Aplikasi Ulo Asem (Sumber: Penulis)

Prosedur "TampilanApel" digunakan untuk menampilkan simbol "O" sebagai representasi dari buah apel. Pertama, warna teks diganti menjadi merah dengan memanggil fungsi "SetTextColor" dan passing nilai "red" (red * 0) ke dalamnya. Kemudian, posisi cursor ditempatkan di (xApelPos, yApelPos) dengan memanggil fungsi "Gotoxy" dan passing nilai dari xApelPos dan yApelPos ke dalamnya. Setelah itu, simbol "O" ditampilkan pada posisi cursor saat ini dengan memanggil fungsi "WriteChar" dan passing nilai "O" ke dalamnya. Terakhir, warna teks dikembalikan menjadi hitam dan putih dengan memanggil fungsi "SetTextColor" dan passing nilai "white" (black * 16) ke dalamnya.

```

1372      BuatAcakApel PROC
1373          mov eax,49
1374          call RandomRange
1375          add eax, 35
1376          mov xApelPos,al
1377          mov eax,17
1378          call RandomRange
1379          add eax, 6
1380          mov yApelPos,al
1381
1382          mov ecx, 5
1383          add cl, Skor
1384          mov esi, 0
1385      CekApelxPosisi:
1386          movzx eax, xApelPos
1387          cmp al, xPosisi[esi]
1388          je CekApelyPosisi
1389      Melanjutkanloop:
1390          inc esi
1391          loop CekApelxPosisi
1392          ret
1393      CekApelyPosisi:
1394          movzx eax, yApelPos
1395          cmp al, yPosisi[esi]
1396          jne Melanjutkanloop
1397          call BuatAcakApel
1398      BuatAcakApel ENDP

```

Gambar 29 Cuplikan kodingan ke-18 Aplikasi Ulo Asem (Sumber: Penulis)

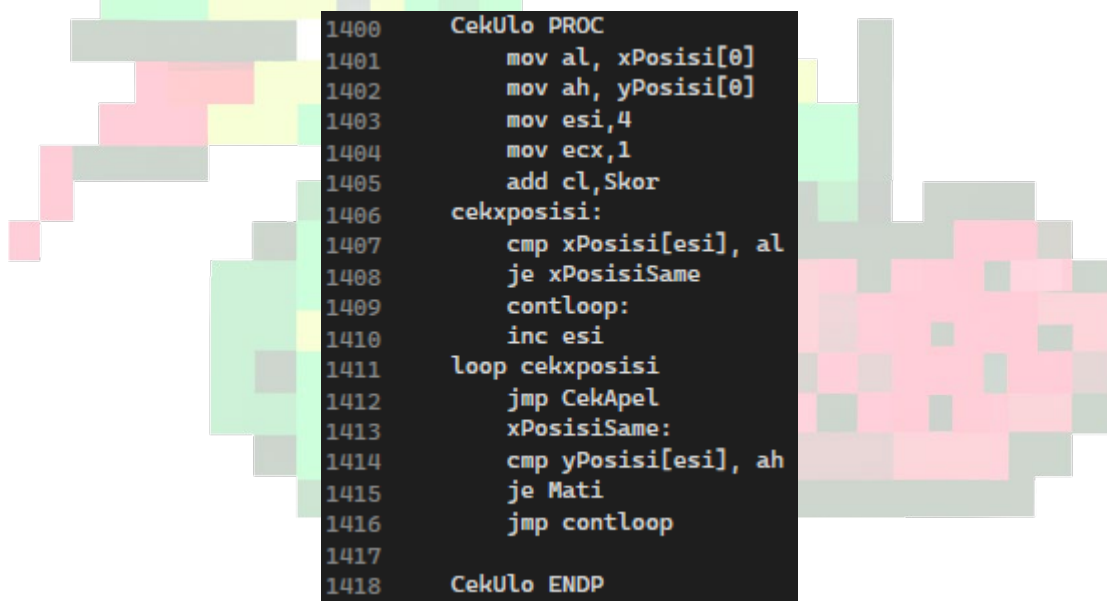
Kode di atas adalah prosedur untuk mengacak posisi buah apel pada layar. Pertama, fungsi RandomRange dipanggil dengan argumen 49, yang digunakan untuk menghasilkan angka acak dari 0 hingga 49. Kemudian, hasil dari fungsi tersebut ditambahkan dengan 35, sehingga rentang angka yang dihasilkan menjadi 35 hingga 84. Angka acak ini kemudian disimpan dalam register AL, yang digunakan sebagai posisi x dari buah apel.

Selanjutnya, fungsi RandomRange dipanggil lagi dengan argumen 17, yang digunakan untuk menghasilkan angka acak dari 0 hingga 17. Kemudian, hasil dari fungsi tersebut ditambahkan dengan 6, sehingga rentang angka yang dihasilkan menjadi 6 hingga 23. Angka acak ini kemudian disimpan dalam register AL, yang digunakan sebagai posisi y dari buah apel.

Setelah posisi x dan y dari buah apel ditentukan, kode mengecek apakah posisi x dan y dari buah apel sama dengan posisi x dan y dari ular. Jika posisi x dan y dari buah apel sama dengan posisi x dan y dari ular, maka prosedur BuatAcakApel akan dipanggil kembali untuk menentukan koordinat buah apel yang baru dan tidak ditempatkan pada posisi yang sama dengan ular. Namun, jika posisi x dan y dari buah apel tidak sama dengan posisi x dan y dari ular, maka kode akan kembali ke

perulangan dan melanjutkan untuk mengecek posisi x dan y dari segmen ular selanjutnya.

Setelah posisi x dan y dari buah apel diacak, kode kemudian memasukkan nilai 5 pada register ecx dan menambahkannya dengan skor. Kemudian, register esi diatur sebagai 0. Lalu, ada perulangan yang disebut "CekApelxPosisi" yang digunakan untuk memeriksa apakah posisi x dari buah apel sama dengan posisi x dari Ulo pada indeks esi yang saat ini diperiksa. Jika sama, kode akan melompat ke perintah "CekApelyPosisi" yang digunakan untuk memeriksa apakah posisi y dari buah apel sama dengan posisi y dari Ulo pada indeks esi yang saat ini diperiksa. Jika tidak sama, kode akan kembali ke perintah "Melanjutkanloop" dan menambahkan 1 pada esi, sehingga indeks berikutnya akan diperiksa dalam perulangan. Jika posisi x dan y dari buah apel sama dengan posisi x dan y dari Ulo pada indeks esi yang saat ini diperiksa, maka kode akan memanggil prosedur "BuatAcakApel" lagi untuk mengacak posisi buah apel yang baru. Setelah perulangan selesai, kode akan kembali ke prosedur yang memanggil prosedur "BuatAcakApel".



```

1400  CekUlo PROC
1401      mov al, xPosisi[0]
1402      mov ah, yPosisi[0]
1403      mov esi, 4
1404      mov ecx, 1
1405      add cl, Skor
1406      cekxposisi:
1407          cmp xPosisi[esi], al
1408          je xPosisiSame
1409          contloop:
1410              inc esi
1411          loop cekxposisi
1412          jmp CekApel
1413      xPosisiSame:
1414          cmp yPosisi[esi], ah
1415          je Mati
1416          jmp contloop
1417
1418  CekUlo ENDP

```

Gambar 30 Cuplikan kodingan ke-19 Aplikasi Ulo Asem (Sumber: Penulis)

Prosedur CekUlo digunakan untuk mengecek apakah ular (snake) yang digerakkan oleh pemain menabrak tubuhnya sendiri. Hal ini dilakukan dengan membandingkan posisi x dan y dari kepala Ulo (indeks 0) dengan posisi x dan y dari segmen-segmen tubuh Ulo lainnya (indeks 4 hingga Skor+1).

Pertama, kode menyimpan posisi x dan y dari kepala Ulo (indeks 0) ke dalam register AL dan AH. Kemudian, kode mengatur register ESI sebagai 4 (indeks 4 dari segmen tubuh Ulo) dan menambahkan skor saat ini ke register ECX. Kemudian, kode mengeksekusi loop dengan perintah "loop cekxposisi" yang akan diulang sesuai dengan skor saat ini. Dalam loop ini, kode membandingkan posisi x

dari Ulo pada indeks ESI dengan posisi x dari kepala Ulo. Jika posisi x sama, maka kode akan membandingkan posisi y dari Ulo pada indeks ESI dengan posisi y dari kepala Ulo. Jika kedua posisi x dan y sama, maka Ulo mati (menjalankan prosedur Mati). Jika posisi x atau posisi y tidak sama, loop akan dilanjutkan untuk membandingkan posisi Ulo selanjutnya.

Jika loop selesai tanpa Ulo mati, maka kode akan melompat ke prosedur CekApel untuk mengecek apakah Ulo telah menabrak buah.

```

1420  Tampilantubuh PROC
1421      mov ecx, 4
1422      add cl, Skor
1423      CetakTubuhPerulangan:
1424
1425      inc esi
1426      call PerbaruiPemain
1427      mov dl, xPosisi[esi]
1428      mov dh, yPosisi[esi]
1429      mov yPosisi[esi], ah
1430      mov xPosisi[esi], al
1431      mov al, dl
1432      mov ah, dh
1433      call TampilanPemain
1434      cmp esi, ecx
1435      jl CetakTubuhPerulangan
1436      ret
1437  Tampilantubuh ENDP

```

Gambar 31 Cuplikan kodingan ke-20 Aplikasi Ulo Asem (Sumber: Penulis)

Kode di atas adalah prosedur yang digunakan untuk mencetak tubuh dari Ulo (ular) pada game. Prosedur ini memiliki nama Tampilantubuh.

Pada baris pertama, register ECX diinisialisasi dengan 4 dan kemudian ditambah dengan skor. Ini digunakan untuk menentukan jumlah perulangan yang diperlukan untuk mencetak tubuh Ulo dan ekornya.

Pada baris ke-3, ada perulangan dengan nama CetakTubuhPerulangan. Dalam perulangan ini, register ESI diincrement (ESI akan digunakan sebagai indeks untuk mengakses elemen dari array xPosisi dan yPosisi). Kemudian, prosedur PerbaruiPemain dipanggil.

Pada baris ke-5 dan 6, register DL dan DH diisi dengan xPosisi[esi] dan yPosisi[esi], yaitu posisi saat ini dari segmen tubuh Ulo. Kemudian, pada baris ke-7 dan 8, posisi baru ditentukan untuk segmen tubuh Ulo dengan mengubah xPosisi[esi] dan yPosisi[esi]. Pada baris ke-9 dan 10, posisi lama dikembalikan ke register AL dan AH. Kemudian, prosedur TampilanPemain dipanggil untuk mencetak segmen tubuh Ulo pada posisi baru.

Pada baris ke-11, register ESI dibandingkan dengan ECX. Jika ESI kurang dari ECX, maka perulangan CetakTubuhPerulangan akan diulang. Setelah

perulangan selesai, prosedur akan mengembalikan kontrol ke pemanggil dengan perintah `ret`.

```

1439 MemakanApel PROC
1440     ; Ulo sedang memakan apel
1441     INVOKE PlaySound, OFFSET SuaraUloMakan, NULL, SND_ALIAS
1442     inc Skor
1443     mov ebx,4
1444     add bl, Skor
1445     mov esi, ebx
1446     mov ah, yPosisi[esi-1]
1447     mov al, xPosisi[esi-1]
1448     mov xPosisi[esi], al
1449     mov yPosisi[esi], ah
1450
1451     cmp xPosisi[esi-2], al
1452     jne Ceky
1453
1454     cmp yPosisi[esi-2], ah
1455     jl incy
1456     jg decy
1457     incy:
1458     inc yPosisi[esi]
1459     jmp Melanjutkan
1460     decy:
1461     dec yPosisi[esi]
1462     jmp Melanjutkan
1463
1464     Ceky:
1465     cmp yPosisi[esi-2], ah
1466     jl incx
1467     jg decx
1468     incx:
1469     inc xPosisi[esi]
1470     jmp Melanjutkan
1471     decx:
1472     dec xPosisi[esi]
1473
1474     Melanjutkan:
1475     call TampilanPemain
1476     call BuatAcakApel
1477     call TampilanApel
1478
1479     mov dl,18
1480     mov dh,1
1481     call Gotoxy
1482     mov al,Skor
1483     call WriteInt
1484     ret
1485 MemakanApel ENDP

```

Gambar 32 Cuplikan kodingan ke-21 Aplikasi Ulo Asem (Sumber: Penulis)

Kode di atas merupakan bagian dari sebuah program yang menangani logika untuk "memakan apel" dalam sebuah permainan ular. Prosedur ini disebut "MemakanApel".

Pertama, prosedur ini memanggil suara Ulo yang sedang makan dengan menggunakan "INVOKE PlaySound, OFFSET SuaraUloMakan, NULL, SND_ALIAS". Kemudian, skor ditambah satu dengan menggunakan "inc Skor".

Lalu, prosedur ini menggunakan "mov ebx,4" dan "add bl, Skor" untuk menentukan posisi dari ekor baru dari ular. Kemudian, posisi dari ekor baru diatur sama dengan posisi dari ekor lama dengan menggunakan "mov esi, ebx", "mov ah, yPosisi[esi-1]" dan "mov al, xPosisi[esi-1]".

Selanjutnya, prosedur ini mengecek apakah ekor lama dan segmen ulo sebelum itu berada pada sumbu x atau y dengan "cmp xPosisi[esi-2], al" dan "jne Ceko". Jika ekor lama dan segmen ulo sebelum itu berada pada sumbu y, prosedur ini mengecek apakah ekor yang baru harus di atas atau di bawah ekor yang lama dengan "cmp yPosisi[esi-2], ah" dan "jl incy" atau "jg decy". Jika ekor yang baru harus di atas ekor yang lama, prosedur ini menambahkan 1 pada posisi y dari ekor baru dengan "inc yPosisi[esi]" dan jika ekor yang baru harus di bawah ekor yang lama, prosedur ini mengurangi 1 pada posisi y dari ekor baru dengan "dec yPosisi[esi]".

Jika ekor lama dan segmen ulo sebelum itu berada pada sumbu x, prosedur ini mengecek apakah ekor yang baru harus di kanan atau di kiri dari ekor yang lama dengan "cmp yPosisi[esi-2], ah" dan "jl incx" atau "jg decx". Jika ekor yang baru harus di kanan ekor yang lama, prosedur ini menambahkan 1 pada posisi x dari ekor baru dengan "inc xPosisi[esi]" dan jika ekor yang baru harus di kiri ekor yang lama, prosedur ini mengurangi 1 pada posisi x dari ekor baru dengan "dec xPosisi[esi]".

Setelah itu, prosedur ini memanggil prosedur TampilanPemain untuk menampilkan ular di layar, memanggil prosedur BuatAcakApel untuk generate posisi apel yang baru secara acak, dan memanggil prosedur TampilanApel untuk menampilkan apel di layar. Kemudian, prosedur ini menggunakan instruksi mov dl, 18 dan mov dh, 1 untuk mengatur posisi cursor di layar, dan memanggil prosedur Gotoxy untuk menempatkan cursor di posisi yang ditentukan. Selanjutnya, prosedur ini menggunakan instruksi mov al, Skor untuk memindahkan nilai skor ke register AL, dan memanggil prosedur WriteInt untuk menulis skor pada layar. Prosedur ini kemudian menyelesaikan dengan instruksi ret untuk kembali ke program utama.

1488	KamuMati PROC	1558	mov dl,48
1489	INVOKE PlaySound, OFFSET SuaraUloMati, NULL, SND_ALIAS	1559	mov dh,9
1490	mov eax, 1000	1560	call Gotoxy
1491	call delay	1561	mov eax,red
1492	Call ClrScr	1562	call SetTextColor
1493		1563	mov edx, OFFSET tengkorak_9
1494	mov dl,48	1564	call WriteString
1495	mov dh,1	1565	
1496	call Gotoxy	1566	mov dl,48
1497	mov eax,lightred	1567	mov dh,10
1498	call SetTextColor	1568	call Gotoxy
1499	mov edx, OFFSET tengkorak_1	1569	mov eax,lightred
1500	call WriteString	1570	call SetTextColor
1501		1571	mov edx, OFFSET tengkorak_10
1502	mov dl,48	1572	call WriteString
1503	mov dh,2	1573	
1504	call Gotoxy	1574	mov dl,48
1505	mov eax,red	1575	mov dh,11
1506	call SetTextColor	1576	call Gotoxy
1507	mov edx, OFFSET tengkorak_2	1577	mov eax,red
1508	call WriteString	1578	call SetTextColor
1509		1579	mov edx, OFFSET tengkorak_11
1510	mov dl,48	1580	call WriteString
1511	mov dh,3	1581	
1512	call Gotoxy	1582	mov dl,48
1513	mov eax,lightred	1583	mov dh,12
1514	call SetTextColor	1584	call Gotoxy
1515	mov edx, OFFSET tengkorak_3	1585	mov eax,lightred
1516	call WriteString	1586	call SetTextColor
1517		1587	mov edx, OFFSET tengkorak_12
1518	mov dl,48	1588	call WriteString
1519	mov dh,4	1589	
1520	call Gotoxy	1590	
1521	mov eax,red	1591	mov dl, 57
1522	call SetTextColor	1592	mov dh, 13
1523	mov edx, OFFSET tengkorak_4	1593	call Gotoxy
1524	call WriteString	1594	mov eax, red
1525		1595	call SetTextColor
1526	mov dl,48	1596	mov edx, OFFSET strKamuMati
1527	mov dh,5	1597	call WriteString
1528	call Gotoxy	1598	
1529	mov eax,red	1599	mov dl, 56
1530	call SetTextColor	1600	mov dh, 15
1531	mov edx, OFFSET tengkorak_5	1601	call Gotoxy
1532	call WriteString	1602	mov eax, brown
1533		1603	call SetTextColor
1534	mov dl,48	1604	movzx eax, Skor
1535	mov dh,6	1605	call WriteInt
1536	call Gotoxy	1606	mov eax, yellow
1537	mov eax,lightred	1607	call SetTextColor
1538	call SetTextColor	1608	mov edx, OFFSET strtampilanskor
1539	mov edx, OFFSET tengkorak_6	1609	call WriteString
1540	call WriteString	1610	
1541		1611	mov dl, 50
1542	mov dl,48	1612	mov dh, 19
1543	mov dh,7	1613	call Gotoxy
1544	call Gotoxy	1614	mov eax, lightgreen
1545	mov eax,red	1615	call SetTextColor
1546	call SetTextColor	1616	mov edx, OFFSET strCobaLagi
1547	mov edx, OFFSET tengkorak_7	1617	call WriteString
1548	call WriteString	1618	
1549		1619	mov eax, white
1550	mov dl,48	1620	call SetTextColor
1551	mov dh,8	1621	mov al,Skor
1552	call Gotoxy	1622	cmp al,SkorTertinggi
1553	mov eax,lightred	1623	jle MencobaLagi
1554	call SetTextColor	1624	mov SkorTertinggi, al
1555	mov edx, OFFSET tengkorak_8		
1556	call WriteString		

```

1626      MencobaLagi:
1627      mov dl, 54
1628      mov dh, 20
1629      call Gotoxy
1630      mov eax, white
1631      call SetTextColor
1632      call ReadInt
1633      cmp al, 1
1634      je BermainLagi
1635      cmp al, 0
1636      je KeluarPermainan
1637
1638      mov dh, 17
1639      call Gotoxy
1640      mov edx, OFFSET SalahInput
1641      call WriteString
1642      mov dl, 56
1643      mov dh, 19
1644      call Gotoxy
1645      mov edx, OFFSET blank
1646      call WriteString
1647      jmp MencobaLagi
1648      KamuMati ENDP

```

Gambar 33 Cuplikan kodingan ke-22 Aplikasi Ulo Asem (Sumber: Penulis)

Kode diatas adalah prosedur yang ditujukan untuk menampilkan tampilan ketika pemain kalah dalam permainan. Pertama, prosedur ini memanggil prosedur PlaySound untuk memainkan suara "SuaraUloMati", lalu menghentikan program selama 1000 milidetik dengan memanggil prosedur delay. Kemudian, prosedur ClrScr dipanggil untuk membersihkan layar.

Setelah itu, terdapat beberapa perintah yang digunakan untuk menampilkan gambar tengkorak pada layar dengan menggunakan prosedur Gotoxy, SetTextColor, dan WriteString. Posisi dan warna dari setiap baris tengkorak ditentukan dengan mengatur nilai dl, dh, dan eax sebelum memanggil prosedur tersebut.

Selanjutnya, prosedur menampilkan pesan "Kamu Mati" dan skor akhir pemain dengan menggunakan prosedur Gotoxy, SetTextColor dan WriteString. Kemudian, prosedur menampilkan pilihan "Coba Lagi?" dan meminta input dari pemain untuk memutuskan apakah ingin bermain lagi atau keluar dari permainan. Pertama, posisi cursor ditetapkan pada baris 20 dan kolom 54 menggunakan prosedur Gotoxy. Kemudian, warna teks diubah menjadi putih menggunakan prosedur SetTextColor. Input diterima dari pemain menggunakan prosedur ReadInt dan disimpan di register AL. Kemudian, input dibandingkan dengan 1 untuk memutuskan apakah pemain ingin bermain lagi atau 0 jika pemain ingin keluar dari permainan. Jika input sesuai dengan kondisi yang ditentukan, maka prosedur akan mengarahkan ke prosedur BermainLagi atau KeluarPermainan. Jika input tidak sesuai dengan kondisi yang ditentukan, maka akan ditampilkan pesan "Salah Input" dan input sebelumnya akan dibersihkan. Kemudian, prosedur menampilkan pilihan "Coba Lagi?" dan meminta input dari pemain untuk memutuskan apakah ingin bermain lagi atau keluar dari permainan. Jika pemain memasukkan input 1, maka

permainan akan diulang dari awal. Jika pemain memasukkan input 0, maka permainan akan berakhir. Jika pemain memasukkan input yang salah, maka akan ditampilkan pesan "Salah Input" dan pemain akan diminta untuk memasukkan input lagi. Kemudian, prosedur akan membandingkan skor saat ini dengan skor tertinggi yang pernah didapatkan sebelumnya dan menyimpan skor tertinggi jika skor saat ini lebih tinggi.

```

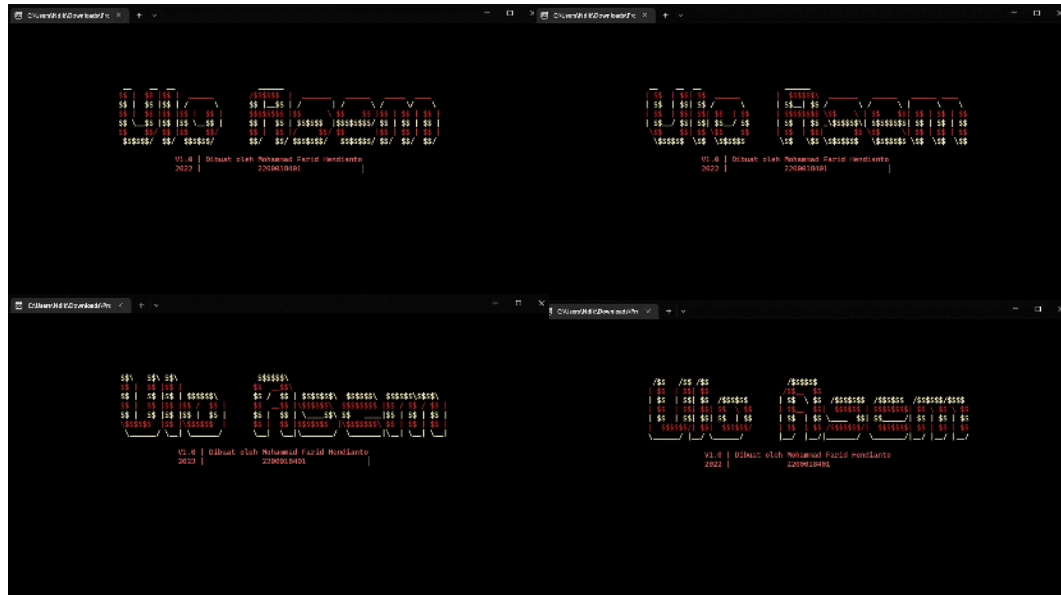
1650      InisialisasiUlangPermainan PROC
1651          mov xPosisi[0], 45
1652          mov xPosisi[1], 44
1653          mov xPosisi[2], 43
1654          mov xPosisi[3], 42
1655          mov xPosisi[4], 41
1656          mov yPosisi[0], 15
1657          mov yPosisi[1], 15
1658          mov yPosisi[2], 15
1659          mov yPosisi[3], 15
1660          mov yPosisi[4], 15
1661          mov Skor, 0
1662          mov TerakhirInputKarakter, 0
1663          mov inputChar, "+"
1664          dec yPosisiDinding[3]
1665          Call ClrScr
1666          jmp Permainan
1667      InisialisasiUlangPermainan ENDP
1668      END Utama

```

Gambar 34 Cuplikan kodingan ke-22 Aplikasi Ulo Asem (Sumber: Penulis)

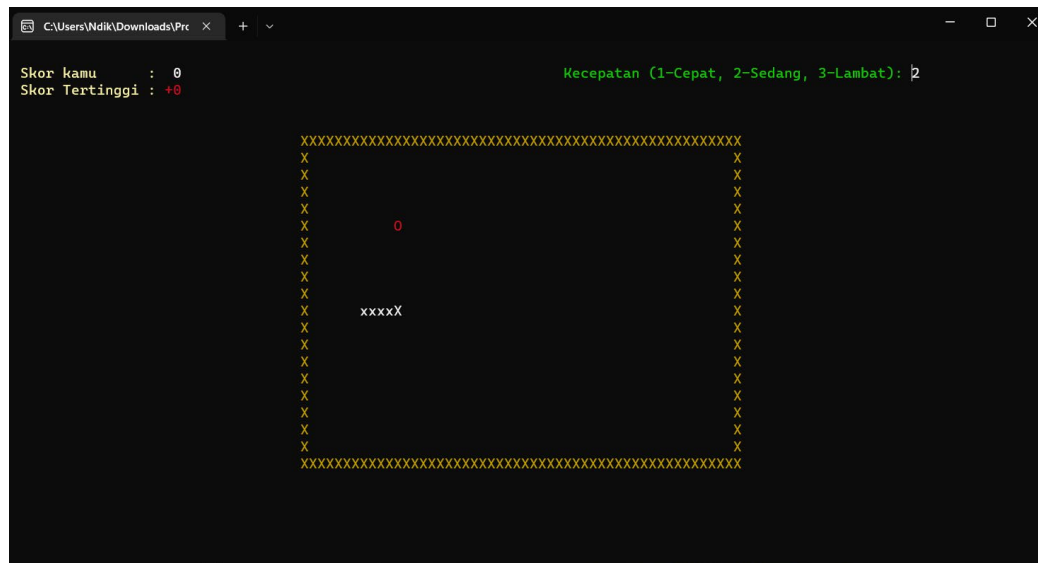
InisialisasiUlangPermainan adalah prosedur yang digunakan untuk mengatur ulang permainan ke kondisi awal. Pada bagian ini, kode tersebut mengatur posisi awal dari objek Ulo dan posisi dinding ke posisi awal, mengatur skor menjadi 0, mengatur inputChar dan TerakhirInputKarakter ke posisi awal, dan membersihkan layar. Kemudian, program akan menjump ke prosedur Permainan untuk memulai permainan kembali.

F. TAMPILAN YANG DIHASILKAN APLIKASI



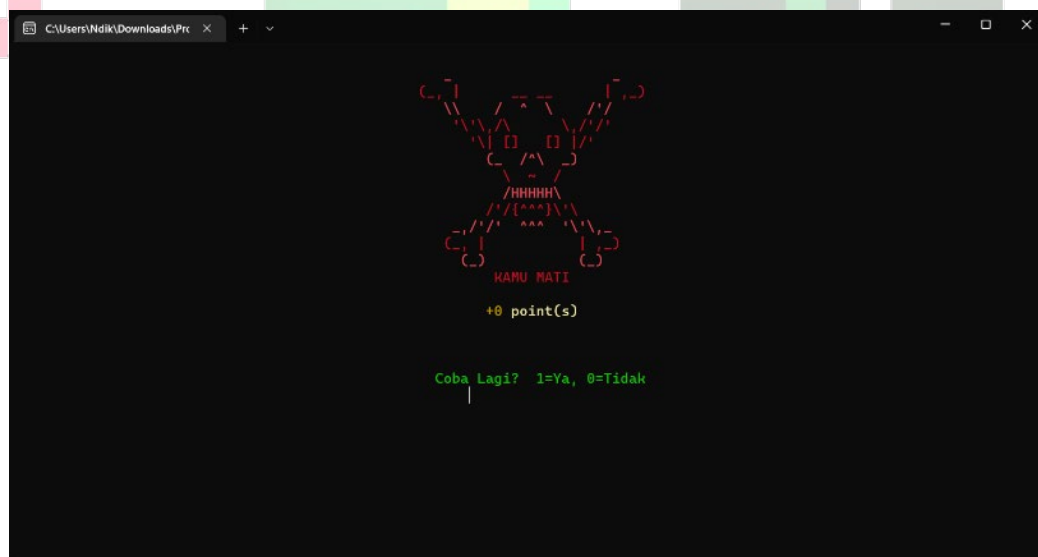
Gambar 35 4 frame untuk menampilkan animasi dalam tampilan saat aplikasi dijalankan. (Sumber: Penulis)

Dalam aplikasi Ulo Asem, tampilan pertama yang akan ditampilkan kepada pengguna adalah tampilan awal yang menampilkan logo game Ulo Asem dengan menggunakan ASCII Art. Logo Ulo Asem dalam berbentuk tulisan “Ulo Asem” akan ditampilkan dalam bentuk animasi. Selain itu, tampilan ini juga akan menampilkan informasi mengenai versi dan tahun pembuatan program Ulo Asem serta nama dan NIM pembuat aplikasi.



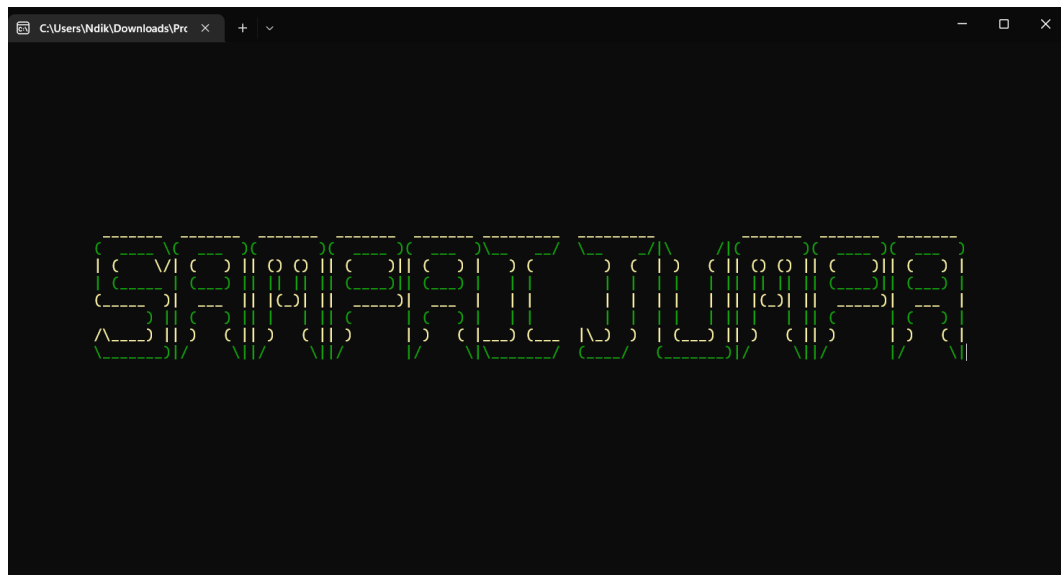
Gambar 36 Tampilan game.aplikasi Ulo Asem . (Sumber: Penulis)

Tampilan kedua adalah tampilan game, yang merupakan inti dari aplikasi ini. Tampilan ini akan menampilkan papan skor yang berisi skor yang sedang berlangsung di permainan, serta skor tertinggi dari permainan sebelumnya. Selain itu, tampilan ini juga akan menampilkan papan kecepatan ulo yang menunjukkan kecepatan ulo saat ini, serta papan permainan yang berisikan dinding, ulo, dan apel yang ditampilkan secara acak di dalam dinding permainan.



Gambar 37 Tampilan gameover.aplikasi Ulo Asem . (Sumber: Penulis)

Tampilan ketiga adalah tampilan GameOver, yang akan ditampilkan ketika permainan berakhir. Tampilan ini akan menampilkan pesan GameOver dengan tulisan dan ASCII Art, serta menampilkan skor yang didapatkan selama permainan. Selain itu, tampilan ini juga akan menampilkan skor tertinggi yang pernah didapatkan selama permainan.



Gambar 38 Tampilan keluar.aplikasi Ulo Asem . (Sumber: Penulis)

Tampilan terakhir adalah Tampilan Keluar adalah tampilan yang ditampilkan ketika aplikasi Ulo Asem ditutup. Di tampilan ini, terdapat pesan yang ditampilkan dengan ASCII Art bertuliskan "SAMPAI JUMPA" yang menandakan bahwa pengguna telah selesai menggunakan aplikasi.

Semua tampilan aplikasi walaupun berbentuk teks, bisa menampilkan warna juga.

G. STATUS UNGGAH DI GITHUB

File proyek aplikasi Ulo Asem sudah terupload ke dalam github. Dari source kode ulo.asm, dokumentasi, hingga linked library yang dibutuhkan seperti Irvine32.

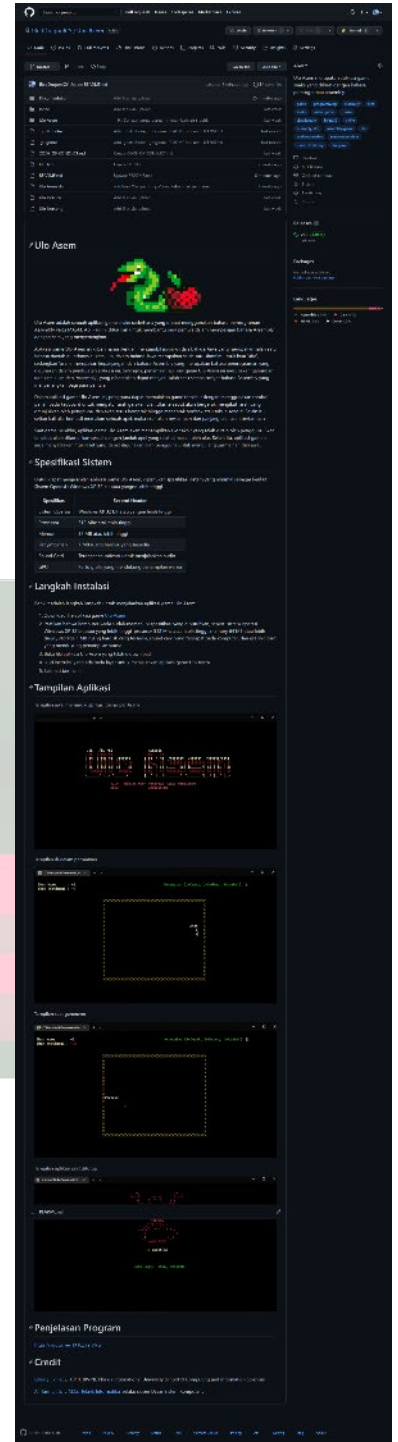
Link Github : github.com/IRedDragonICY/Ulo-Asem
Shortlink : bit.ly/uloasem



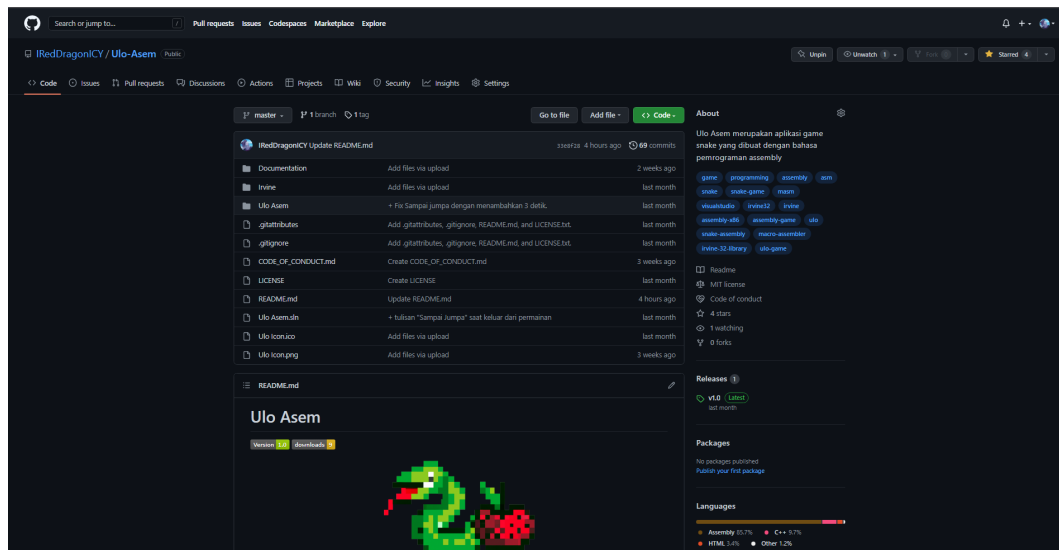
Gambar 40 QR kode untuk redirect ke link proyek github . (Sumber: Penulis)

Di dalam link proyek github Ulo Asem, di dalam readme sudah dijelaskan penjelasan aplikasi secara ringkas seperti deskripsi aplikasi, spesifikasi sistem, langkah instalasi, cuplikan tampilan aplikasi berbentuk gif. Penjelasan program ke youtube dan ucapan terimakasih.

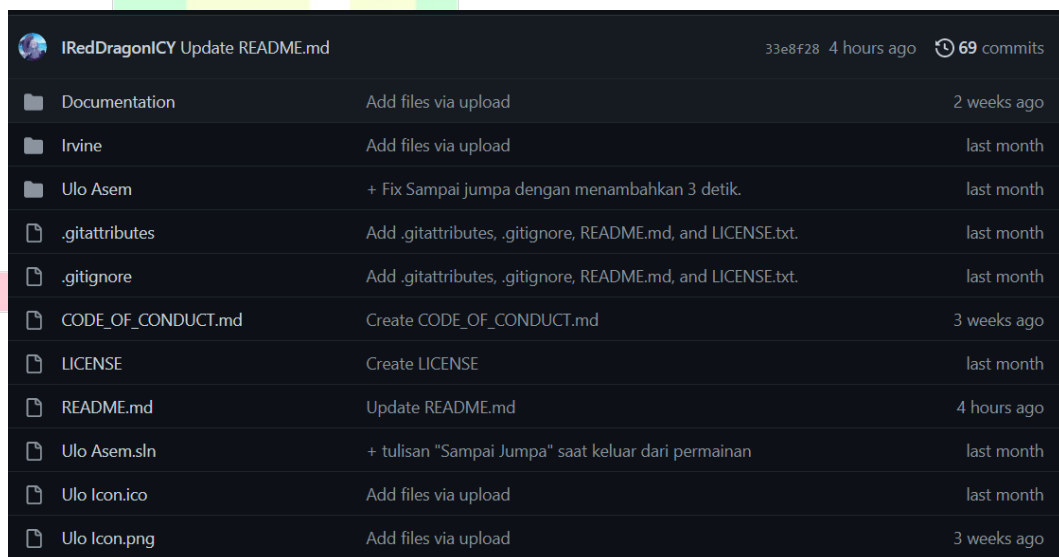
Selain itu, pada tabs releases sudah menampilkan aplikasi Ulo Asem yang siap di download. Di releases ditampilkan versi Ulo Asem ke v1.0.



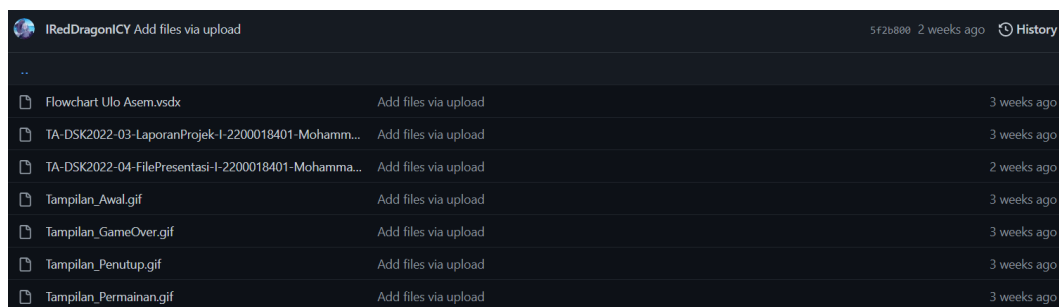
Gambar 39 Tampilan Proyek Ulo Asem di Github . (Sumber: Penulis)



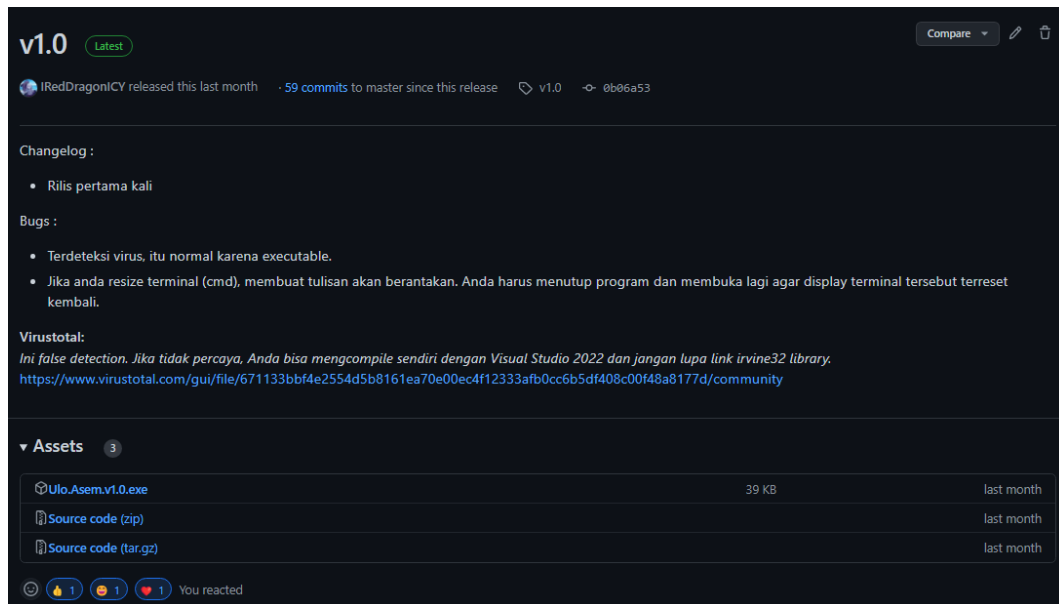
Gambar 41 Halaman depan proyek Ulo Asem di github (Sumber: Penulis)



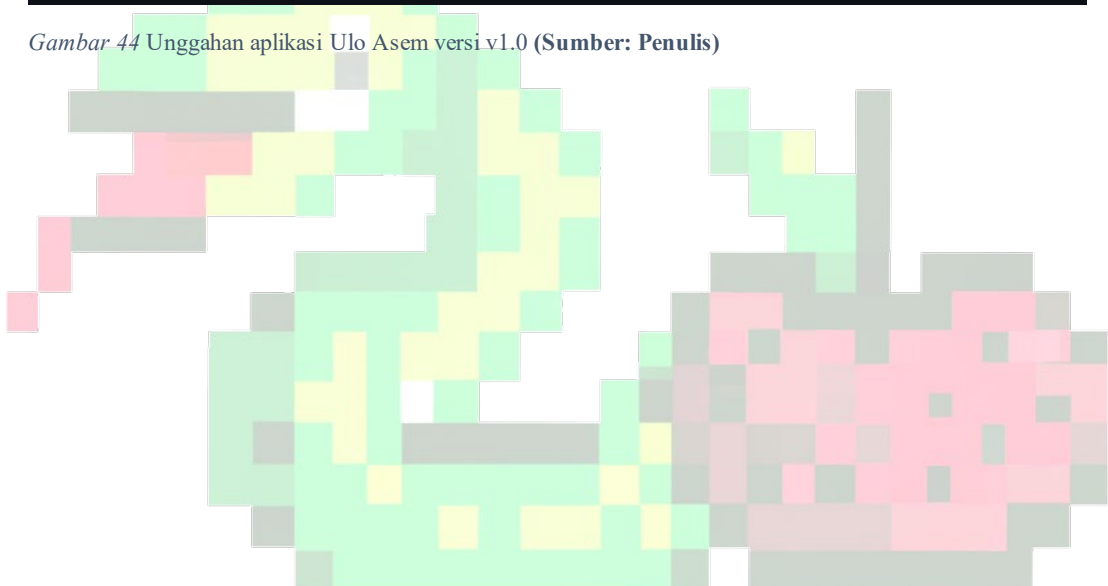
Gambar 42 Tampilan file proyek di github (Sumber: Penulis)



Gambar 43 Tampilan file dokumentasi proyek di github (Sumber: Penulis)



Gambar 44 Unggahan aplikasi Ulo Asem versi v1.0 (Sumber: Penulis)



1) TINJAUAN DARI SISI WAKTU

[illegible]

	: Pembuatan akun github
	: Pemilihan judul projek aplikasi
	: Pembuatan link projek aplikasi di github
	: Pembuatan aplikasi
	: Uji Coba Aplikasi
	: Pembuatan laporan aplikasi
	: Presentasi Aplikasi
	: Bukti unggah Aplikasi
	: Portofolio Aplikasi

Sebelum membuat aplikasi Ulo Asem, ada beberapa tahap yang harus dilalui. Pertama, pembuatan akun di GitHub. GitHub adalah platform yang digunakan untuk pengembangan perangkat lunak yang menyediakan layanan hosting untuk repositori Git. Ini memungkinkan tim pengembang untuk berkolaborasi dalam pengembangan proyek, melacak perubahan kode, dan mengelola versi. Karena saya sudah memiliki akun GitHub, maka tugas untuk pembuatan akun tidak memakan banyak waktu, cukup 1 jam untuk menyelesaikan dan mengumpulkan bukti bahwa sudah memiliki akun GitHub.

Kemudian, minggu depan dilakukan pemilihan judul proyek aplikasi menggunakan bahasa assembly untuk tugas akhir mata kuliah Dasar Sistem Komputer. Diberikan waktu 1 minggu untuk menyelesaikan. Untuk mencari topik yang cocok, tidak terlalu banyak contoh proyek yang sudah dibuat dalam bahasa assembly, mungkin dikarenakan merupakan bahasa yang paling bawah. Bahasa assembly yang tidak hanya TASM (Turbo Assembler), yang diajarkan dalam mata kuliah Dasar Sistem Komputer, melainkan ada MASM (Macro Assembler) yang dibuat. Saya mencari contoh proyek yang menggunakan bahasa assembly MASM. Saya mencoba untuk membuat tantangan membuat game menggunakan bahasa assembly. Saya kepikiran membuat game sederhana, yaitu game snake. Setelah di

cek di forum E-learning, ternyata belum ada yang membuat aplikasi game snake menggunakan assembly di forum E-learning sehingga saya memilih topik tersebut untuk dijadikan sebagai proyek tugas akhir mata kuliah Dasar Sistem Komputer. Menentukan topik judul proyek aplikasi membutuhkan sekitar 3 hari untuk mendapatkan topik yang sesuai.

Setelah menemukan judul proyek aplikasi yang cocok, kemudian saya membuat proyek di GitHub untuk menyimpan file-file yang dipakai dalam proyek saya sebagai source code. Saya juga tidak lupa untuk mengundang dosen saya, Bapak Ali Tarmuji, S.T., M.Cs sebagai collaborator untuk memantau proyek saya. Membuat proyek di GitHub tidak memerlukan waktu yang lama, hanya memerlukan sekitar 1 hari untuk membuat dan menyertakan bukti bahwa proyek di GitHub sudah dibuat di E-learning. Setelah wadah untuk menyimpan kode sudah dibuat (proyek di GitHub), kemudian saya mengintegrasikan dengan git yang sudah memiliki fitur di Visual Studio. Setelah itu, saya juga mencari library yang cocok digunakan pada bahasa Assembly bertipe MASM. Salah satu library yang saya temukan adalah library Irvine yang cocok digunakan pada bahasa Assembly bertipe MASM. Karena judul yang sudah saya pilih sudah fix, maka saya akan langsung juga membuat kode untuk membuat aplikasi Ulo Asem.

Setelah menemukan judul proyek aplikasi yang cocok, kemudian saya membuat proyek di GitHub untuk menyimpan file-file yang dipakai dalam proyek saya sebagai source code. Tidak lupa saya mengundang dosen saya, Bapak Ali Tarmuji, S.T., M.Cs sebagai collaborator untuk memantau proyek saya. Membuat proyek di GitHub tidak memerlukan waktu yang lama, hanya memerlukan 1 hari untuk membuat dan penyertaan bukti dimana proyek di GitHub sudah dibuat di E-learning.

Setelah wadah untuk menyimpan kode sudah dibuat (proyek di GitHub), kemudian saya mengintegrasikan dengan Git yang sudah ada fitur di Visual Studio. Setelah itu saya juga mencari library yang cocok digunakan pada bahasa Assembly bertipekan MASM. Salah satu library yang saya temukan adalah library Irvine yang cocok digunakan bahasa assembly bertipe MASM. Karena judul yang sudah saya pilih sudah fix, maka saya akan langsung juga membuat kode untuk membuat aplikasi Ulo Asem. Saya membuat kerangka untuk membuat game snake sekitar 2 minggu, dan finalisasi sekitar 4 hari. Saat finalisasi tersebut, saya menambah fitur seperti memberi warna, hingga debugging apabila terjadi bug, berlangsung bersamaan dengan fase uji coba aplikasi.

Selanjutnya, setelah aplikasi selesai dibuat, saya langsung membuat laporan aplikasi Ulo Asem selama 1 hari full, dan pada hari itu juga 1 jam 30 menit disisihkan untuk melakukan presentasi aplikasi Ulo Asem. Presentasi tersebut di upload ke dalam YouTube selama 1 jam dikarenakan file yang cukup besar dan kecepatan internet yang tidak terlalu cepat.

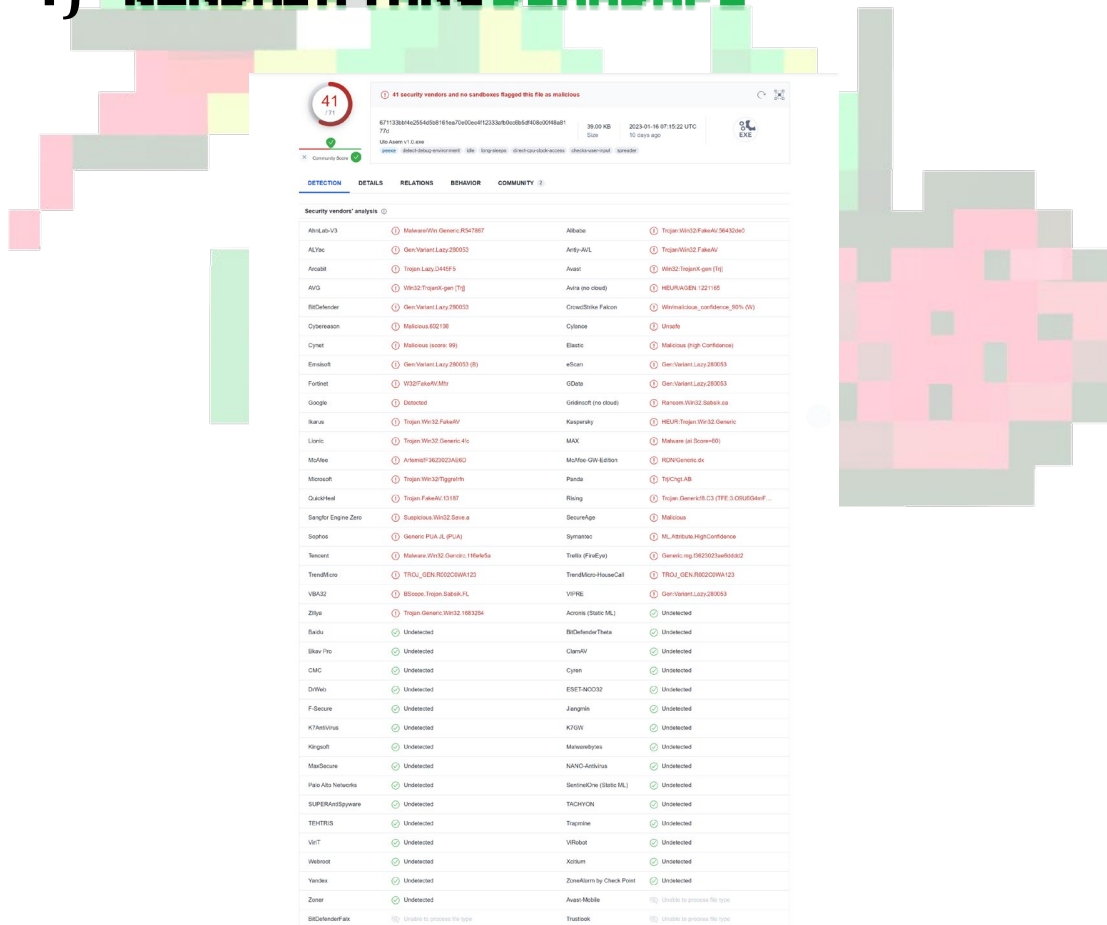
Terakhir, untuk pembuatan portofolio aplikasi diperkirakan selesai selama 4 hari. Saya memulai pembuatan portofolio aplikasi dan membuat pilihan waktu saat kosong di saat Ujian Akhir Semester 1.

2) KETERCAPAIAN SPESIFIKASI

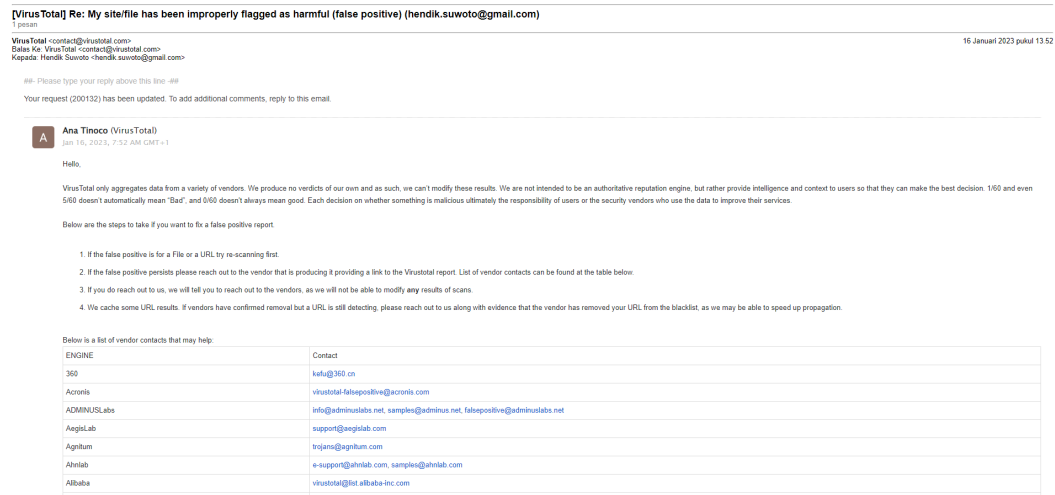
Untuk dapat menjalankan aplikasi game Ulo Asem, diperlukan spesifikasi sistem yang minimal sebagai berikut: sistem operasi Windows XP 32 bit atau yang lebih tinggi, prosessor 512 Mhz atau lebih tinggi, memory 32 MB atau lebih tinggi, ukuran file 39 KB, storage 1 MB ruang hardisk yang tersedia, sound card yang terdapat untuk menjalankan audio, dan graphic card yang mendukung penampilan warna. Pastikan bahwa semua perangkat lunak yang diperlukan, seperti sistem operasi dan aplikasi pendukung, telah terinstal dengan benar pada komputer Anda agar game Ulo Asem dapat berjalan dengan lancar. Selain spesifikasi sistem yang diperlukan, aplikasi game Ulo Asem juga dilengkapi dengan beberapa fitur yang akan membuat pengalaman bermain game semakin menyenangkan bagi pengguna. Fitur-fitur tersebut antara lain tampilan dinding, papan skor, pemilihan kecepatan Ulo, pembuatan acak apel, dan pergerakan Ulo yang dapat dikontrol dengan keyboard. Tampilan dinding pada aplikasi ini akan membuat permainan semakin menyenangkan bagi pengguna, sementara fitur pemilihan kecepatan Ulo akan memungkinkan pengguna untuk mengatur kecepatan gerakan ular sesuai dengan keinginan mereka. Pembuatan acak apel juga merupakan fitur yang akan membuat permainan semakin menantang dan tidak monoton. Pergerakan Ulo yang dapat dikontrol dengan keyboard akan membuat permainan semakin mudah dimainkan dan dikendalikan oleh pengguna. Fitur reset juga merupakan fitur yang dapat digunakan oleh pengguna untuk mengulang permainan dari awal. Keseluruhan, aplikasi game Ulo Asem ini telah dirancang dengan baik untuk dapat digunakan pada sistem operasi Windows dan ditulis dengan menggunakan bahasa Assembly. Spesifikasi system yang dibutuhkan untuk menjalankan aplikasi ini cukup rendah, sehingga dapat digunakan pada komputer dengan spesifikasi yang rendah. Selain itu, game ini juga dilengkapi dengan fitur-fitur yang menarik seperti grafis yang baik, suara yang menyenangkan, dan kontrol yang mudah digunakan. Meskipun game ini dirancang untuk sistem operasi Windows, aplikasi ini juga dapat digunakan pada sistem operasi lain dengan menggunakan software penerjemah. Aplikasi ini juga dilengkapi dengan dokumentasi yang lengkap yang memudahkan pengguna dalam menggunakannya.

Biaya yang dibutuhkan dalam pengembangan aplikasi Ulo Asem sangat minim atau bahkan tidak ada sama sekali. Hal ini dikarenakan saya menggunakan Visual Studio 2022 Insiders Community Edition, yang merupakan perangkat lunak gratis dan tersedia untuk digunakan tanpa biaya. Selain itu, saya juga menggunakan library Irvine yang merupakan open source yang tidak berbayar.

4) KENDALA YANG DIHADAPI



Gambar 45 Aplikasi Ulo Asem yang terdeteksi virus pada Virustotal (Sumber: Penulis)



Gambar 46 Bertanya Customer Service pada Virustotal untuk melaporkan false detection terhadap aplikasi Ulo Asem (Sumber: Penulis)

Salah satu kendala yang dihadapi oleh aplikasi Ulo Asem adalah terdeteksi oleh virustotal sebanyak 41 dari 71 provider. Hal ini mungkin dikarenakan aplikasi ini menggunakan library dari windows seperti winmm, dan menggunakan interrupt processor secara langsung dengan bahasa assembly. Bahasa assembly merupakan bahasa pemrograman yang jarang dipelajari karena lebih detail dan memerlukan pemahaman yang lebih dalam mengenai sistem operasi dan arsitektur komputer. Selain itu, bahasa assembly juga memerlukan lebih banyak waktu dan usaha untuk dapat membuat program yang efisien dan stabil.

5) TANTANGAN MASA DEPAN

Tantangan masa depan dalam membuat portofolio aplikasi Ulo Asem adalah menghilangkan false detection yang dapat muncul dari customer service Virus Total. Hal ini dapat dilakukan dengan cara re-scan dan menghubungi vendor yang menghasilkan false detection, serta menyediakan link dari laporan Virustotal. Selain itu, untuk meningkatkan pengalaman bermain dan menarik minat pengguna, akan ditambahkan fitur-fitur baru seperti mode frenzy (apel lebih banyak muncul), mode banyak rintangan seperti batu, fitur lainnya seperti memberikan warna pada ular, tampilan menu, tampilan option, grafis yang lebih memukau, dan maintenance proyek yang lebih lanjut. Namun, diperlukan kerja sama dengan vendor dan komunitas untuk menjamin keamanan dan kualitas aplikasi ini.