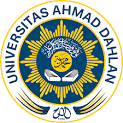
**LAPORAN PEMBUATAN APLIKASI**

“**Dino Runner Assembly**”

Diajukan untuk memenuhi salah satu tugas mata kuliah Dasar Sistem Komputer

Yang diampu oleh :

*Ali Tarmuji, S.T., M. Cs. Teknik Informatika*



**Disusun Oleh:**

**NAMA : M. MILKY GAZURA**

**NIM : 2200018396**

**KELAS : I**

**UNIVERSITAS AHMAD DAHLAN FAKULTAS TEKNOLOGI INDUSTRI**

**PROGRAM STUDI INFORMATIKA TAHUN 2022/2023**

**KATA PENGANTAR**

Puji dan syukur Saya panjatkan kepada Allah Ta’ala karena karunianya Saya dapat menyelesaikan Laporan Pembuatan Aplikasi Dino Runner Assembly ini. Laporan ini dapat diselesaikan tepat pada waktunya tidak terlepas dari bantuan dan dukungan dari berbagai pihak, yakni Bapak Ali Tarmuji ,S.T.,M.Cs., selaku dosen pengajar, dan teman-teman kelas i dari Prodi Informatika. Untuk itu Saya ucapkan terima kasih atas kontribusi bantuan dalam berbagai bentuk.

Saya menyadari bahwa masih ada kekurangan baik dari segi penyusunan kalimat maupun tata bahasa yang saya gunakan terlepas keterbatasan saya selaku seorang penuntut ilmu yang masih butuh banyak bimbingan. Oleh karena itu dengan tangan terbuka saya menerima berbagai kritik dan saran dari para pembaca agar saya dapat memperbaiki laporan yang saya susun.

Demikian semoga laporan Pembuatan Aplikasi Dino Runner Assembly ini bisa diterima sebagai ide atau gagasan yang menambah kekayaan intelektual dalam berbagai bidang. Semoga laporan Pembuatan Aplikasi Dino Runner Assembly saya ini dapat bermanfaat bagi pembaca dan juga untuk diri Saya sendiri.

Yogyakarta, 31 Desember 2022

M. Milky Gazura

**DAFTAR ISI**

Daftar Isi

[**BAB I** 4](#_Toc123500255)

[**PENDAHULUAN** 4](#_Toc123500256)

[**1.1 Latar Belakang** 4](#_Toc123500257)

[**1.2 Rumusan Masalah** 4](#_Toc123500258)

[**1.3 Tujuan Laporan** 4](#_Toc123500259)

[**BAB II** 5](#_Toc123500260)

[**2.1 Deskripsi Aplikasi Dino Runner Assembly** 5](#_Toc123500261)

[**2.2 Cara Kerja Aplikasi Dino Runner Assembly** 5](#_Toc123500262)

[**2.3 Desain atau Sketsa Antarmuka** 8](#_Toc123500263)

[**2.4 Kode Pada Program Dino Runner Assembly** 8](#_Toc123500264)

[**2.5 Penjelasan Pada Setiap Kode Program** 25](#_Toc123500265)

[**2.6 Pengujian Program Dino Runner Assembly** 34](#_Toc123500266)

[**BAB III** 40](#_Toc123500267)

[**PENUTUP** 40](#_Toc123500268)

# **BAB I**

# **PENDAHULUAN**

# **1.1 Latar Belakang**

Assembly adalah bahasa pemrograman tingkat tinggi yang digunakan untuk menulis kode yang akan dijalankan oleh komputer atau perangkat lainnya. Bahasa ini merupakan bagian dari keluarga bahasa pemrograman yang disebut bahasa mesin, yang merupakan bahasa yang paling dekat dengan bahasa yang digunakan oleh perangkat keras komputer.

Assembly menggunakan mnemonik untuk menyatakan instruksi yang akan dijalankan oleh komputer, yang kemudian akan diterjemahkan menjadi bahasa mesin yang dapat dipahami oleh komputer. Mnemonik ini merupakan kode singkat yang mudah diingat oleh programmer, yang mewakili instruksi yang lebih kompleks dalam bahasa mesin.

Assembly sangat efisien dalam penggunaan sumber daya komputer, karena kode yang ditulis dalam bahasa ini langsung diterjemahkan menjadi bahasa mesin yang dapat dijalankan oleh komputer. Namun, bahasa ini juga cukup sulit dipelajari karena programmer harus memahami cara kerja perangkat keras komputer secara mendetail, serta harus menulis kode dengan sangat teliti agar tidak terjadi kesalahan.

Assembly sering digunakan dalam pengembangan sistem operasi, device driver, dan aplikasi yang membutuhkan kecepatan eksekusi yang tinggi. Namun, bahasa ini juga dapat digunakan untuk mengembangkan aplikasi lainnya, tergantung pada kebutuhan dan tujuan yang ingin dicapai oleh programmer.

Assembly adalah bahasa pemrograman yang efisien dalam penggunaan sumber daya komputer, namun cukup sulit dipelajari karena programmer harus memahami cara kerja perangkat keras komputer secara mendetail. Bahasa ini sering digunakan dalam pengembangan sistem operasi, device driver, dan aplikasi yang membutuhkan kecepatan eksekusi yang tinggi, namun dapat juga digunakan untuk mengembangkan aplikasi lainnya. Pada kesempatan kali ini saya akan membuat Program Aplikasi DINORASEM(Dino Runner Assembly) Versi sederhana dalam Bahasa Assembly.

# **1.2 Rumusan Masalah**

Adapun rumusan masalah dalam laporan ini adalah bagaimana Proses-proses pembuatan Aplikasi Dino Runner Assembly?

# **1.3 Tujuan Laporan**

Adapun Tujuan dalam laporan ini adalah untuk mengetahui bagaimana Proses-proses pembuatan .Aplikasi Dino Runner Assembly.

# **BAB II**

**PEMBAHASAN MATERI**

# **2.1 Deskripsi Aplikasi Dino Runner Assembly**

Dino Runner Assembly adalah sebuah permainan edukatif yang dirancang untuk mengajarkan anak-anak cara membuat bentuk dinosaurus dengan menggunakan sejumlah blok yang tersedia. Permainan ini ditujukan untuk anak-anak usia di atas 5 tahun dan bertujuan untuk meningkatkan kemampuan berpikir logis dan mengembangkan keterampilan motorik halus mereka.

Permainan ini menyediakan sejumlah blok dengan bentuk yang berbeda-beda, yang harus digunakan oleh pemain untuk membangun bentuk dinosaurus yang diinginkan. Pemain harus memikirkan cara terbaik untuk menyusun blok-blok tersebut agar terbentuk menjadi bentuk dinosaurus yang tepat.

Selain itu, permainan ini juga menyediakan sejumlah tantangan yang harus diatasi oleh pemain, seperti membuat dinosaurus dengan ukuran tertentu atau membangun dinosaurus dengan struktur yang kompleks. Hal ini akan menantang kemampuan pemain untuk berpikir secara logis dan mencari solusi yang tepat untuk menyelesaikan tantangan tersebut.

Permainan ini dapat dimainkan secara individual atau dalam kelompok, sehingga pemain juga dapat belajar bekerja sama dan membantu satu sama lain dalam menyelesaikan tantangan yang dihadapi.

Dino Assembly adalah permainan yang menyenangkan dan menantang yang dapat mengasah kemampuan berpikir logis dan keterampilan motorik halus anak-anak, serta mengajarkan nilai-nilai kerja sama dan saling membantu.

# **2.2 Cara Kerja Aplikasi Dino Runner Assembly**

1. Pada bagian .MODEL SMALL, programmer menentukan bahwa model memori yang akan digunakan adalah model memori kecil.
2. Bagian STACK SEGMENT PARA STACK kemudian menentukan bahwa segment memori yang akan digunakan sebagai stack adalah segmen dengan nama STACK.
3. Bagian STACK ENDS kemudian menandakan bahwa deklarasi segment stack sudah selesai.
4. Bagian .DATA kemudian menyimpan data yang akan digunakan oleh kode. Data tersebut bisa berupa nilai-nilai konstanta, seperti lebar game (GAME\_WIDTH), atau variabel-variabel, seperti posisi dinosaurus (DINO\_X dan DINO\_Y).
5. Bagian .CODE kemudian menyimpan instruksi-instruksi yang akan dijalankan oleh komputer.
6. Instruksi-instruksi yang terdapat pada bagian .CODE tersebut terdiri dari berbagai macam perintah, seperti perintah untuk mengeprint string (MOV AH, 09H dan INT 21H), perintah untuk memindahkan cursor ke posisi tertentu (MOV AH, 02H dan INT 10H), dan perintah untuk memeriksa apakah terdapat perubahan waktu (MOV AH, 2CH dan INT 21H).
7. Alur kerja pada kode di atas dapat dilihat dari bagaimana instruksi-instruksi tersebut dijalankan secara berurutan. Pertama-tama, kode akan memanggil prosedur CLEAR\_SCREEN untuk membersihkan layar.
8. Kemudian, kode akan memeriksa apakah terdapat perubahan waktu dengan memanggil prosedur TIME\_CHECK.
9. Jika terdapat perubahan waktu, kode kemudian akan memanggil prosedur COLLISION\_CHECK untuk memeriksa apakah terdapat tabrakan antara dinosaurus dengan objek lainnya.
10. Jika terdapat tabrakan, maka kode akan menjalankan prosedur GAMEOVER\_GAM yang akan mencetak kata "GAMEOVER" ke layar dan meminta input dari pemain.
11. Jika pemain memasukkan input "w", maka kode akan memanggil prosedur RESET\_GAME untuk mengulang permainan. Jika tidak, maka permainan akan berakhir.
12. Jika tidak terdapat tabrakan, maka kode akan terus memanggil prosedur CACTUS\_MOVE, BIRD\_MOVE, dan DINO\_MOVE untuk memindahkan objek-objek tersebut di layar.
13. Kemudian, kode akan memanggil prosedur CLEAR\_SCREEN untuk membersihkan layar sebelum mengeprint objek-objek tersebut kembali ke layar.
14. Setelah itu, kode akan kembali ke label TIME\_CHECK untuk memeriksa apakah terdapat perubahan waktu lagi. Proses ini akan terus berlangsung sampai pemain mengalami tabrakan atau memutuskan untuk mengakhiri permainan.
15. Alur kerja pada kode di atas akan terus diulang sampai permainan berakhir atau pemain memutuskan untuk mengakhiri permainan.

Berikut ini adalah flowchart yang menggambarkan alur kerja pada kode Assembly di atas:

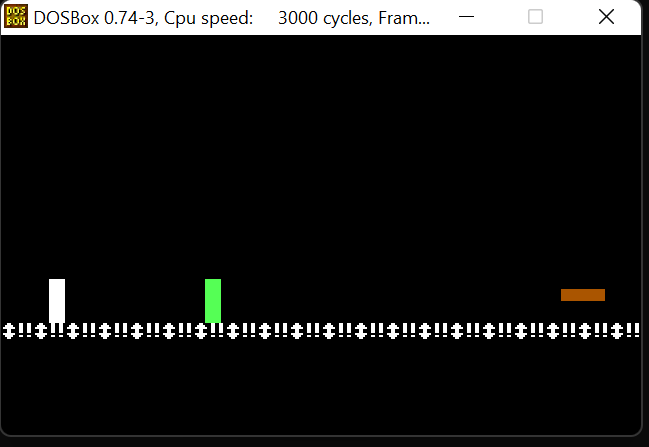
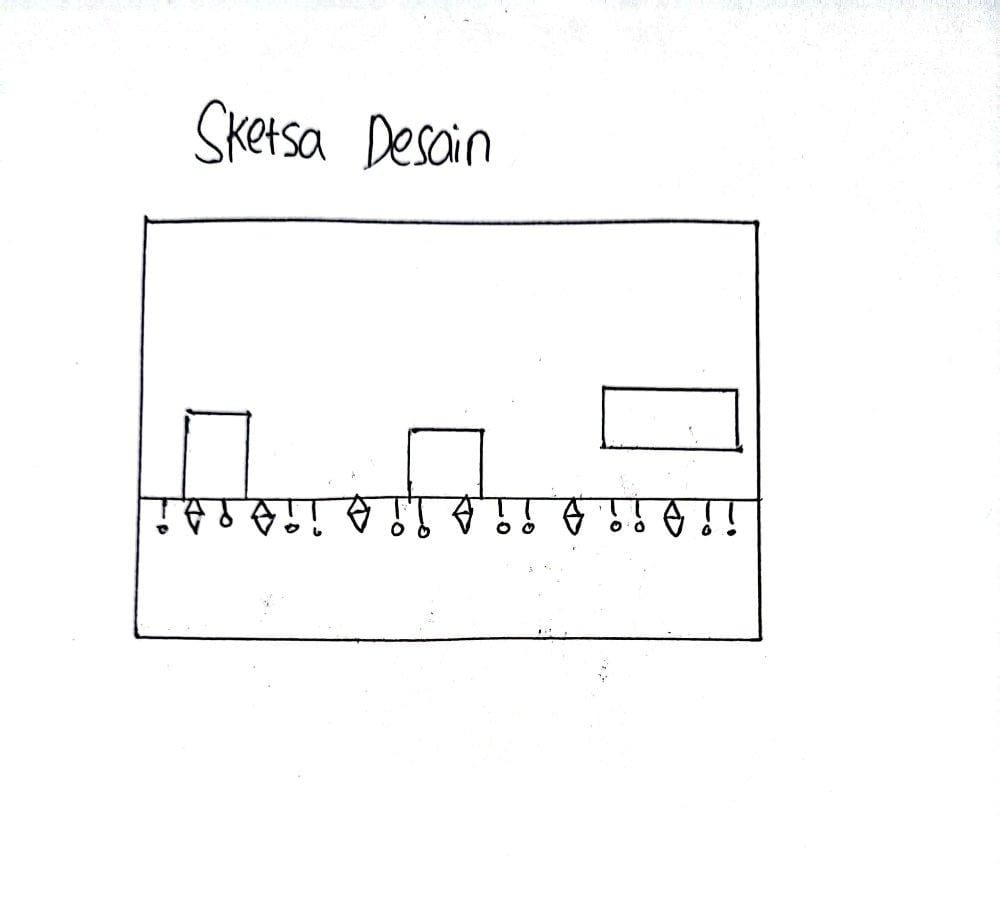
1. Mulai
2. Panggil prosedur CLEAR\_SCREEN
3. Pergi ke label TIME\_CHECK
4. Panggil prosedur TIME\_CHECK
5. Periksa apakah terdapat perubahan waktu
6. Jika tidak terdapat perubahan waktu, kembali ke label TIME\_CHECK
7. Jika terdapat perubahan waktu, panggil prosedur COLLISION\_CHECK
8. Periksa apakah terdapat tabrakan
9. Jika tidak terdapat tabrakan, lanjut ke langkah 10
10. Panggil prosedur CACTUS\_MOVE, BIRD\_MOVE, dan DINO\_MOVE
11. Panggil prosedur CLEAR\_SCREEN
12. Panggil prosedur GROUND\_DRAW
13. Periksa apakah terdapat perubahan waktu sebesar 20ms
14. Jika tidak terdapat perubahan waktu, kembali ke label TIME\_CHECK
15. Jika terdapat perubahan waktu, dan kembali ke label TIME\_CHECK
16. Jika terdapat tabrakan, panggil prosedur GAMEOVER\_GAM
17. Periksa apakah pemain memasukkan input "w"
18. Jika tidak, tutup permainan
19. Jika ya, panggil prosedur RESET\_GAME dan kembali ke langkah 2
20. Selesai

Flowchart dalam gambar :



# **2.3 Desain atau Sketsa Antarmuka**

# Aplikasi Dino Runner Assembly memiliki antarmuka yang sederhana dan mudah digunakan. Pengguna hanya mengklik tombol “w” untuk lompat, dan tombol “s” untuk menunduk. Apabila menabrak suatu rintangan pengguna bisa mengklik tombol “w” untuk kembali bermain.

****

# **2.4 Kode Pada Program Dino Runner Assembly**

.MODEL SMALL

STACK SEGMENT PARA STACK

DB 2048 DUP(" ")

STACK ENDS

.DATA

;LEBAR GAME

GAME\_WIDTH DW 140H

;DATA DINOSAURUS

DINO\_Y DW 90H

DINO\_X DW 20H

DINO\_X\_SIZE DW 8H

DINO\_Y\_SIZE DW ?

;DATA CACTUS

CACTUS\_X DW 140H

CACTUS\_Y DW 90H

CACTUS\_X\_SIZE DW 8H

CACTUS\_Y\_SIZE DW 16H

CACTUS\_SPEED DW 05H

;DATA BURUNG

BIRD\_X DW 200H

BIRD\_Y DW 85H

BIRD\_X\_SIZE DW 16H

BIRD\_Y\_SIZE DW 6H

;VARIABEL WAKTU

TIME\_VAR DB 05H

;FLAGS

FLAG1 DW 0H

FLAG2 DW 0H

EXIT\_FLAG DW 0H

;VARIABEL SCORE

SCORE\_TIME DW 0H

SCORE1 DB 0H

SCORE2 DB 0H

SCORE3 DB 0H

SCORE4 DB 0H

;STRING UNTUK DIPRINT

DISPLAY\_TEXT1 DB '0','$'

DISPLAY\_TEXT2 DB '0','$'

DISPLAY\_TEXT3 DB '0','$'

DISPLAY\_TEXT4 DB '0','$'

DISPLAY\_TEXTH1 DB '0','$'

DISPLAY\_TEXTH2 DB '0','$'

DISPLAY\_TEXTH3 DB '0','$'

DISPLAY\_TEXTH4 DB '0','$'

GAMEOVER\_TEXT DB 'GAMEOVER','$'

.CODE

;MACRO KETIKA GAMEOVER UNTUK MENGEPRINT KATA GAMEOVER

GAMEOVER\_GAM MACRO

;MENGGERAKKAN CURSOR UNTUK MENGEPRINT PADA LOKASI TERTENTU

MOV AH,02H

MOV BH,00H

MOV DH,01H

MOV DL,01H

INT 10H

;MENGEPRINT TEXT

MOV AH,09H

LEA DX,GAMEOVER\_TEXT

INT 21H

;MEMINTA INPUT

MOV AH,00H

INT 16H

CMP AL,77h

JNE EXITS

CALL RESET\_GAME

JMP GAMEON

ENDM

main PROC near

CALL CLEAR\_SCREEN

TIME\_CHECK:

;CEK WAKTU SISTEM

MOV AH,2CH

INT 21H

;APAKAH ADA PERUBAHAN WAKTU

CMP DL,TIME\_VAR

JE TIME\_CHECK

;APAKAH ADA PERUBAHAN WAKTU SEBESAR 20ms

INC SCORE\_TIME

CMP SCORE\_TIME,5H

JNE NO\_SCORE

MOV SCORE\_TIME,00H

;APABILA TIDAK ADA PERMAINAN SCORE MENGECEK APAKAH ADA TABRAKAN ANTARA DINO DAN OBJEK LAINNYA

NO\_SCORE:

CALL COLLISION\_CHECK

CMP EXIT\_FLAG,00H

JNE GAMEOVER

JMP GAMEON

;KETIKA COLLISION CHECK BENAR MAKA AKAN MENJALANKAN GAMEOVER

GAMEOVER:

GAMEOVER\_GAM

;PROSES KETIKA GAME MASIH DIMAINKAN MENGEPRINT OBJEK KE LAYAR

GAMEON:

CALL CACTUS\_MOVE

CALL BIRD\_MOVE

MOV TIME\_VAR,DL

CALL CLEAR\_SCREEN

CALL GROUND\_DRAW

CALL DINO\_MOVE

CMP FLAG2,1H

JE NUNDUK

CALL DINO\_DRAW\_STAND

JMP EXIT3

;KETIKA ADA INPUT 's'MAKA POSISI DINO AKAN MENUNDUK

NUNDUK:

CALL DINO\_DRAW\_CROUCH

MOV FLAG2,00H

;KETIKA TIDAK MENUNDUK

EXIT3:

CALL CACTUS\_DRAW

CALL BIRD\_DRAW

JMP TIME\_CHECK

EXITS:

RET

main ENDP

;DIGUNAKAN UNTUK MENGGAMBAR TANAH

GROUND\_DRAW PROC near

MOV DL,00H

CHAR1\_LOOP:

MOV AH,02h

MOV BH,00H

MOV DH,12H

INT 10H

MOV AH,09H

MOV AL,12H

MOV BH,00H

MOV BL,0FH

MOV CX,01H

INT 10H

INC DL

CHAR2\_LOOP:

MOV AH,02h

MOV BH,00H

MOV DH,12H

INT 10H

MOV AH,09H

MOV AL,13H

MOV BH,00H

MOV BL,0FH

MOV CX,01H

INT 10H

INC DL

CMP DL,28H

JNE CHAR1\_LOOP

RET

GROUND\_DRAW ENDP

;MENGGAMBAR DINO KETIKA BERDIRI

DINO\_DRAW\_STAND PROC near

MOV DINO\_Y\_SIZE, 16H

MOV CX,DINO\_X

MOV DX,DINO\_Y

SUB CX,DINO\_X\_SIZE

SUB DX,DINO\_Y\_SIZE

DINO\_DRAWB\_H:

MOV AH,0CH

MOV AL,0FH

MOV BH,00H

INT 10H

INC CX

CMP CX,DINO\_X

JNE DINO\_DRAWB\_H

DINO\_DRAWB\_V:

MOV CX,DINO\_X

SUB CX,DINO\_X\_SIZE

INC DX

CMP DX,DINO\_Y

JNE DINO\_DRAWB\_H

RET

DINO\_DRAW\_STAND ENDP

;MENGGAMBAR DINO KETIKA MENUNDUK

DINO\_DRAW\_CROUCH PROC near

MOV DINO\_Y\_SIZE, 8H

MOV CX,DINO\_X

MOV DX,DINO\_Y

SUB CX,DINO\_X\_SIZE

SUB DX,DINO\_Y\_SIZE

DINO\_DRAWBC\_H:

MOV AH,0CH

MOV AL,0FH

MOV BH,00H

INT 10H

INC CX

CMP CX,DINO\_X

JNE DINO\_DRAWBC\_H

DINO\_DRAWBC\_V:

MOV CX,DINO\_X

SUB CX,DINO\_X\_SIZE

INC DX

CMP DX,DINO\_Y

JNE DINO\_DRAWBC\_H

MOV CX,DINO\_X

MOV DX,DINO\_Y

SUB DX,8H

ADD CX,06H

DINO\_DRAWH\_HEAD:

MOV AH,0CH

MOV AL,0FH

MOV BH,00H

INT 10H

DEC CX

CMP CX,DINO\_X

JNE DINO\_DRAWH\_HEAD

DINO\_DRAWV\_HEAD:

MOV CX,DINO\_X

ADD CX,06H

INC DX

MOV BX,DINO\_Y

SUB BX,5H

CMP DX,BX

MOV BX,00H

JNE DINO\_DRAWH\_HEAD

RET

DINO\_DRAW\_CROUCH ENDP

;MENGGERAKKAN DINO

DINO\_MOVE PROC NEAR

;CEK APAKAH KEY TERTEKAN

CMP DINO\_Y,90H

JNE MOVE\_DINO\_UP

MOV AH, 01

INT 16H

JZ EXIT\_KEY

;CEK KEY MANA YANG TERTEKAN (INPUT DISIMPAN KE AL)

MOV AH,00H

INT 16H

CMP AL,77h ;APAKAH W TERTEKAN

JE MOVE\_DINO\_UP

CMP AL,73H ;APAKAH S TERTEKAN

JE MOVE\_DINO\_DOWN

JNE EXIT\_KEY

;PROSES KETIKA DINO LONCAT

MOVE\_DINO\_UP:

MOV AH, 01

INT 16H

JZ MOVEUP1

;CEK KEY MANA YANG TERTEKAN (INPUT DISIMPAN KE AL)

MOV AH,00H

INT 16H

CMP AL,73h ;APAKAH W TERTEKAN

JE MOVE\_DINO\_DOWN

MOVEUP1:

MOV FLAG2,00H

MOV AX,FLAG1

CMP AX,01H

JE MOVEDOWN

;PROSES KETIKA DINO BERGERAK KE ATAS

MOVEUP:

SUB DINO\_Y,05H

MOV BX,DINO\_Y

CMP DINO\_Y,40H

JNL EXIT\_KEY

INC AX

MOV FLAG1,AX

;PROSES KETIKA DINO BERGERAK KE BAWAH (GRAVITASI)

MOVEDOWN:

ADD DINO\_Y,05H

CMP DINO\_Y,90H

JNG EXIT\_KEY

MOV DINO\_Y,90H

MOV AX,FLAG1

DEC AX

MOV FLAG1,AX

JMP EXIT\_KEY

;PROSES KETIKA DINO MENUNDUK

MOVE\_DINO\_DOWN:

MOV FLAG2,01H

MOV DINO\_Y,90H

JMP EXIT\_KEY

EXIT\_KEY:

RET

DINO\_MOVE ENDP

;PROSES MENGGAMBAR KAKTUS

CACTUS\_DRAW PROC near

MOV CX,CACTUS\_X

MOV DX,CACTUS\_Y

SUB DX,CACTUS\_Y\_SIZE

SUB CX,CACTUS\_X\_SIZE

CACTUS\_DRAW\_H:

MOV AH,0CH

MOV AL,0AH

MOV BH,00H

INT 10H

INC CX

CMP CX,CACTUS\_X

JNE CACTUS\_DRAW\_H

CACTUS\_DRAW\_V:

MOV CX,CACTUS\_X

SUB CX,CACTUS\_X\_SIZE

INC DX

CMP DX,CACTUS\_Y

JNE CACTUS\_DRAW\_H

RET

CACTUS\_DRAW ENDP

;PROSES MENGGERAKKAN CACTUS

CACTUS\_MOVE PROC near

;MENGECEK APAKAH KAKTUS SUDAH DIUJUNG SEBELAH KIRI

CMP CACTUS\_X,08H

JNG RESET

;MENGGERAKKAN KAKTUS DENGAN KECEPATAN TERTENTU

MOV AX, CACTUS\_SPEED

SUB CACTUS\_X,AX

JMP LANJUT

RESET:

MOV AX,GAME\_WIDTH

MOV CACTUS\_X,AX

LANJUT:

RET

CACTUS\_MOVE ENDP

;MENGGAMBAR BURUNG

BIRD\_DRAW PROC near

MOV CX,BIRD\_X

CMP CX,140H

JG EXIT1

MOV DX,BIRD\_Y

SUB DX,BIRD\_Y\_SIZE

SUB CX,BIRD\_X\_SIZE

BIRD\_DRAW\_H:

MOV AH,0CH

MOV AL,06H

MOV BH,00H

INT 10H

INC CX

CMP CX,BIRD\_X

JNE BIRD\_DRAW\_H

BIRD\_DRAW\_V:

MOV CX,BIRD\_X

SUB CX,BIRD\_X\_SIZE

INC DX

CMP DX,BIRD\_Y

JNE BIRD\_DRAW\_H

EXIT1:

RET

BIRD\_DRAW ENDP

;MENGGERAKKAN BURUNG

BIRD\_MOVE PROC near

;MENGECEK APAKAH BURUNG SUDAH DIPOJOK KIRI

CMP BIRD\_X,08H

JNG RESET1

;MENGGERAKKAN BURUNG DENGAN KECEPATAN SAMA DENGAN KAKTUS

MOV AX, CACTUS\_SPEED

SUB BIRD\_X,AX

JMP LANJUT2

RESET1:

MOV AX,GAME\_WIDTH

MOV BIRD\_X,AX

LANJUT2:

RET

BIRD\_MOVE ENDP

COLLISION\_CHECK PROC near

;CEK DINO CACTUS X

MOV AX,CACTUS\_X

SUB AX,CACTUS\_X\_SIZE

CMP DINO\_X,AX

JNGE BIRD\_CHECK

MOV AX, DINO\_X

SUB AX,DINO\_X\_SIZE

CMP CACTUS\_X,AX

JNGE BIRD\_CHECK

;CEK DINO-CACTUS Y

MOV AX,CACTUS\_Y

SUB AX,CACTUS\_Y\_SIZE

CMP DINO\_Y,AX

JL BIRD\_CHECK

INC EXIT\_FLAG

JMP EXIT4

BIRD\_CHECK:

;CEK DINO-BIRD X

MOV AX,BIRD\_X

SUB AX,BIRD\_X\_SIZE

CMP DINO\_X,AX

JNGE EXIT4

MOV AX,DINO\_X

SUB AX,DINO\_X\_SIZE

CMP BIRD\_X,AX

JNGE EXIT4

;CEK DINO-BIRD Y

MOV AX,BIRD\_Y

SUB AX,BIRD\_Y\_SIZE

CMP DINO\_Y,AX

JL EXIT4

MOV AX, DINO\_Y

SUB AX, DINO\_Y\_SIZE

CMP AX, BIRD\_Y

JG EXIT4

INC EXIT\_FLAG

EXIT4:

RET

COLLISION\_CHECK ENDP

;MENGATUR ULANG SEMUA PARAMETER KE KONDISI AWAL

RESET\_GAME PROC NEAR

MOV DINO\_Y, 90H

MOV DINO\_X, 20H

MOV CACTUS\_X,140H

MOV CACTUS\_Y, 90H

MOV BIRD\_X, 200H

MOV BIRD\_Y, 85H

MOV FLAG1, 0H

MOV FLAG2, 0H

MOV EXIT\_FLAG, 0H

MOV SCORE\_TIME, 0H

MOV SCORE1 ,0H

MOV SCORE2 ,0H

MOV SCORE3 ,0H

MOV SCORE4 ,0H

MOV DISPLAY\_TEXT1, 30H

MOV DISPLAY\_TEXT2, 30H

MOV DISPLAY\_TEXT3, 30H

MOV DISPLAY\_TEXT4, 30H

MOV AX,00H

MOV DX,00H

MOV CX,00H

MOV BX,00H

RET

RESET\_GAME ENDP

;MENGATUR SCREEN DENGAN BACKGROUND HITAM (DIGUNAKAN UNTUK MEMBENTUK EFEK PENGHAPUSAN OBJEK DILAYAR)

CLEAR\_SCREEN PROC near

MOV AH,00h

MOV AL,13h

INT 10h

MOV AH,0BH

MOV BH,00H

MOV BL,00H

INT 10h

RET

CLEAR\_SCREEN ENDP

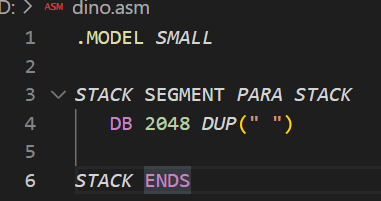
.STARTUP

call main

END

# 

# **2.5 Penjelasan Pada Setiap Kode Program**



kode di atas merupakan bagian dari kode program yang ditulis dalam 2048 assembly. Bagian ini memperkenalkan sebuah segment 2048 “STACK” dan menyatakan bahwa segment tersebut merupakan tipe “PARA STACK”.

Segment “STACK” merupakan bagian dari memori 2048 yang digunakan sebagai tempat menyimpan data sementara saat program dijalankan. Data yang disimpan di stack ini biasanya berupa nilai-nilai yang dibutuhkan oleh program, seperti nilai, nilai fungsi, dll.

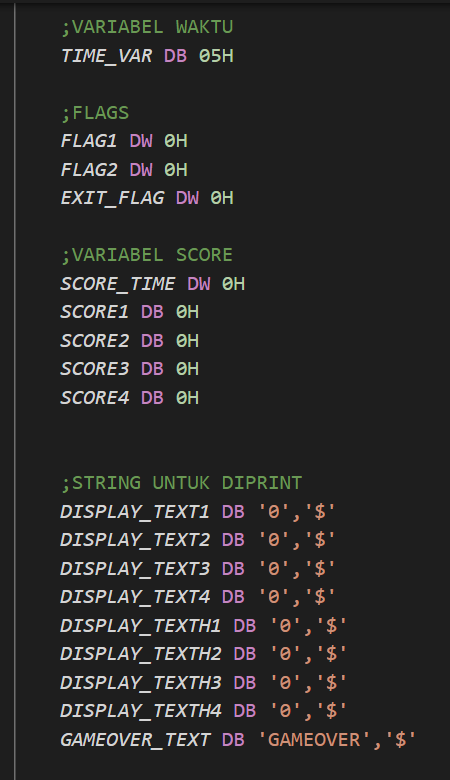
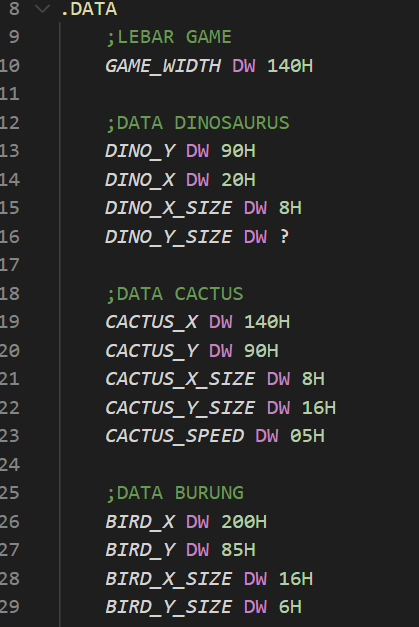
Segment “STACK” juga merupakan tempat dimana program menyimpan informasi tentang lokasi saat mengeksekusi fungsi. Saat program memanggil suatu fungsi, program akan menyimpan informasi tentang lokasi di stack, sehingga setelah fungsi tersebut selesai dijalankan, program dapat digunakan ke lokasi yang tepat.

“.MODEL SMALL” merupakan perintah yang digunakan untuk memberitahu assembler tentang model memori yang akan digunakan dalam program. Ada beberapa model memori yang tersedia, di antaranya adalah “SMALL”, “COMPACT”, “MEDIUM”, “LARGE”, dan “HUGE”.

Model memori “SMALL” menyatakan bahwa program yang ditulis hanya akan menggunakan kecil dari memori. Dengan kata lain, program yang ditulis hanya akan menggunakan sejumlah kecil memori yang tersedia. Model memori ini biasanya digunakan untuk program-program yang tidak terlalu besar atau yang tidak membutuhkan banyak memori.

Bagian “DB 2048 DUP(“ “)” menyatakan bahwa segment STACK terdiri dari 2048 byte data yang diinisialisasi dengan karakter spasi. Jumlah byte yang disimpan di stack ini bisa berbeda-beda tergantung kebutuhan program. Misalnya, jika program membutuhkan lebih banyak tempat untuk menyimpan data, maka jumlah byte yang disediakan di stack bisa ditambah. Sebaliknya, jika program tidak membutuhkan banyak tempat penyimpanan, maka jumlah byte yang disediakan di stack bisa dikurangi.

Secara umum, tujuan dari bagian kode di atas adalah untuk menyediakan sebuah segment memori yang dapat digunakan oleh program sebagai tempat penyimpanan sementara saat program dijalankan. Segment tersebut disebut “STACK” dan terdiri dari 2048 byte data yang diinisialisasi dengan karakter spasi.

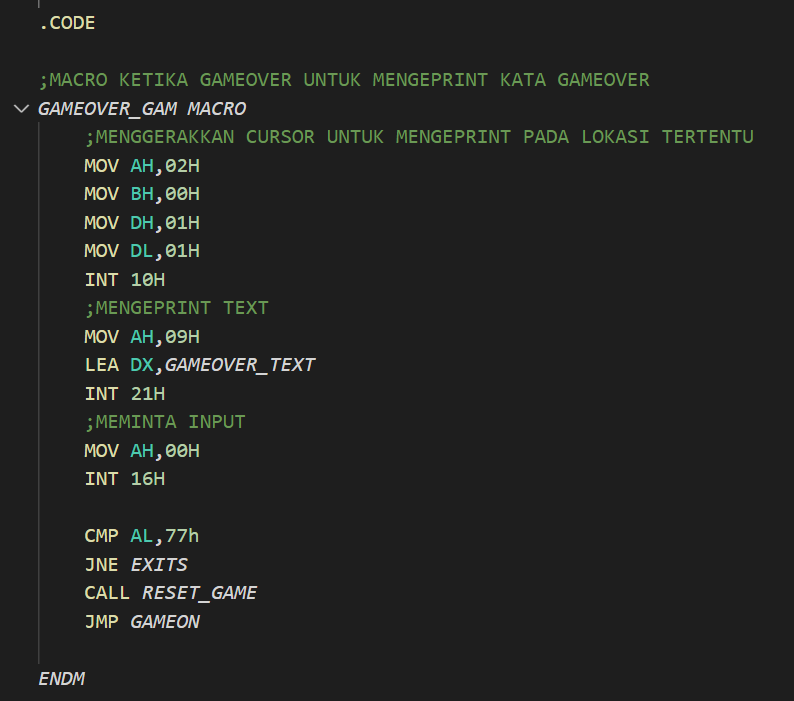


fungsi dari setiap program yang tercantum dalam kode tersebut:

1. LEBAR GAME: Berisi informasi mengenai lebar layar game.
2. DATA DINOSAURUS: Berisi informasi mengenai posisi, ukuran, dan lainnya mengenai dinosaurus dalam game.
3. DATA CACTUS: Berisi informasi mengenai posisi, ukuran, dan kecepatan cactus dalam game.
4. DATA BURUNG: Berisi informasi mengenai posisi, ukuran, dan lainnya mengenai burung dalam game.
5. VARIABEL WAKTU: Berisi informasi mengenai nilai dari waktu dalam game.
6. FLAGS: Berisi informasi mengenai flags yang akan digunakan dalam game.
7. VARIABEL SCORE: Berisi informasi mengenai skor yang diperoleh dalam game.
8. STRING UNTUK DIPRINT: Berisi informasi mengenai string yang akan diprint pada layar.
9. MACRO GAMEOVER\_GAM: Merupakan macro yang bertugas mengeprint "GAMEOVER" pada layar saat gameover.

Semua data yang didefinisikan di atas akan digunakan dalam program untuk mengontrol game.

* ;DATA BURUNG BIRD\_X DW 200H BIRD\_Y DW 85H BIRD\_X\_SIZE DW 16H BIRD\_Y\_SIZE DW 6H
* ;VARIABEL WAKTU TIME\_VAR DB 05H
* Variabel ini menyimpan kode waktu dalam game. TIME\_VAR disimpan dalam bentuk (05H), yang sama dengan 5 dalam satuan bilangan.
* ;FLAGS FLAG1 DW 0H FLAG2 DW 0H EXIT\_FLAG DW 0H
* Variabel-variabel ini merupakan flags atau bendera yang digunakan dalam game. FLAG1 dan FLAG2 mungkin digunakan untuk menandai suatu kondisi atau yang terjadi dalam game. EXIT\_FLAG mungkin digunakan untuk menandai kondisi dimana game harus berhenti atau keluar dari game. Semua ini disimpan dalam bentuk (0H), yang sama dengan 0 dalam satuan bilangan.
* ;STRING UNTUK DIPRINT DISPLAY\_TEXT1 DB ‘0’,’$’ DISPLAY\_TEXT2 DB ‘0’,’$’ DISPLAY\_TEXT3 DB ‘0’,’$’ DISPLAY\_TEXT4 DB ‘0’,’$’ DISPLAY\_TEXTH1 DB ‘0’,’$’ DISPLAY\_TEXTH2 DB ‘0’,’$’ DISPLAY\_TEXTH3 DB ‘0’,’$’ DISPLAY\_TEXTH4 DB ‘0’,’$’ GAMEOVER\_TEXT DB ‘GAMEOVER’,’$’
* Variabel-variabel ini merupakan string yang mungkin digunakan untuk ditampilkan atau diprint pada layar game. DISPLAY\_TEXT1, DISPLAY\_TEXT2, DISPLAY\_TEXT3, dan DISPLAY\_TEXT4 mungkin digunakan untuk menampilkan skor yang telah dicapai dalam game. DISPLAY\_TEXTH1, DISPLAY\_TEXTH2, DISPLAY\_TEXTH3, dan DISPLAY\_TEXTH4 mungkin digunakan untuk menampilkan skor tertinggi yang pernah dicapai dalam game. GAMEOVER\_TEXT mungkin digunakan untuk menampilkan pesan “GAMEOVER” saat game berakhir atau keluar dari game.



Kode di atas merupakan definisi macro assembly GAMEOVER\_GAM yang digunakan untuk mengeprint pesan “GAMEOVER” pada layar game saat kondisi gameover terjadi. Macro ini terdiri dari beberapa perintah-perintah yang akan dijalankan secara berurutan dipanggil dalam kode program.

MOV AH,02H MOV BH,00H MOV DH,01H MOV DL,01H INT 10H

Perintah-perintah di atas digunakan untuk menggerakkan cursor pada game ke lokasi tertentu yang telah ditentukan. Nilai DH dan DL digunakan untuk menentukan koordinat x dan y dari lokasi yang akan dituju, sedangkan nilai BH digunakan untuk menentukan page number (halaman) yang akan digunakan.

MOV AH,09H LEA DX,GAMEOVER\_TEXT INT 21H

Perintah-perintah di atas digunakan untuk mengeprint string yang telah disimpan pada layer GAMEOVER\_TEXT. Nilai AH diset ke 09H yang merupakan kode untuk memanggil fungsi print string pada interrupt 21H. Kemudian, perintah LEA (Load Effective Address) digunakan untuk menyimpan alamat dari string pada layer GAMEOVER\_TEXT ke register DX. Setelah itu, perintah INT 21H digunakan untuk memanggil interrupt yang bertugas mengatur fungsi-fungsi input/output (I/O).

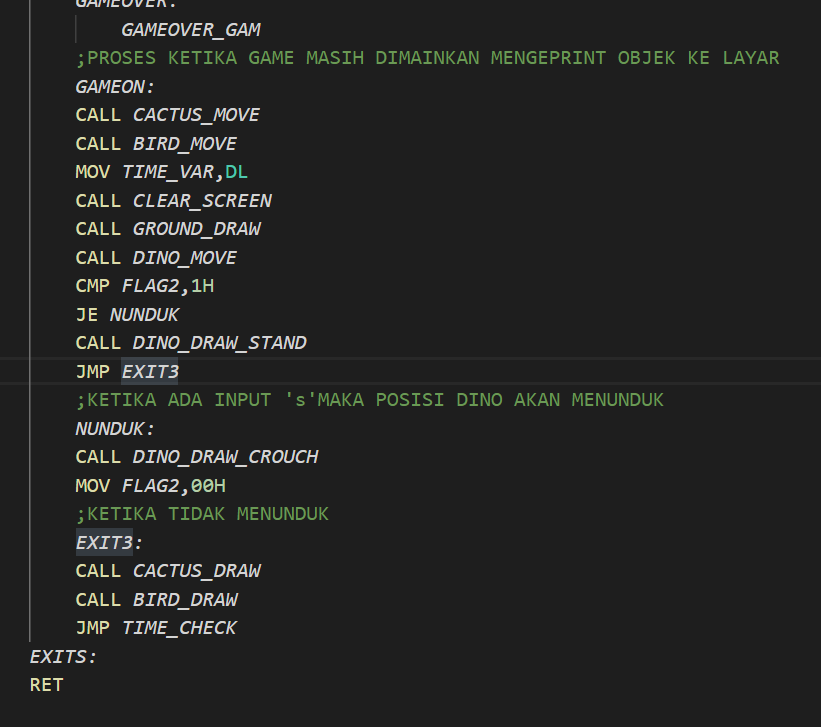
MOV AH,00H INT 16H

Perintah-perintah di atas digunakan untuk meminta input dari pengguna. Nilai AH diset ke 00H yang merupakan kode untuk memanggil fungsi baca tombol keyboard pada interrupt 16H. Setelah itu, perintah INT 16H digunakan untuk memanggil interrupt yang bertugas mengatur fungsi-fungsi input dari keyboard.

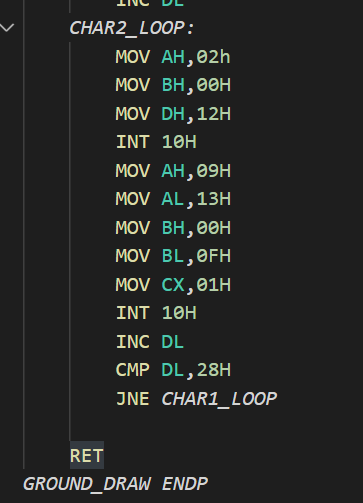
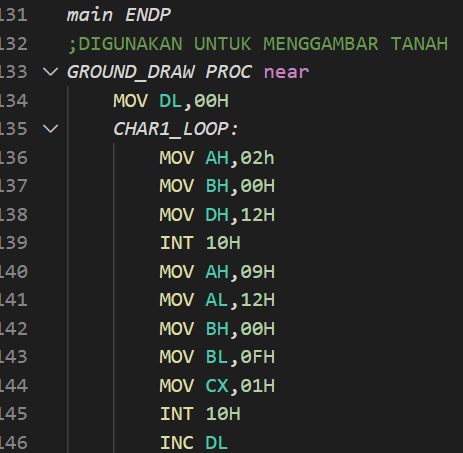
CMP AL,77h JNE EXITS

Perintah-perintah di atas digunakan untuk membandingkan nilai dari register AL dengan 77H. Jika nilai yang terdapat pada register AL tidak sama dengan 77H, maka program akan melanjutkan ke label EXITS. Jika nilai yang terdapat pada register AL sama dengan 77H, maka program akan melanjutkan ke perintah selanjutnya.

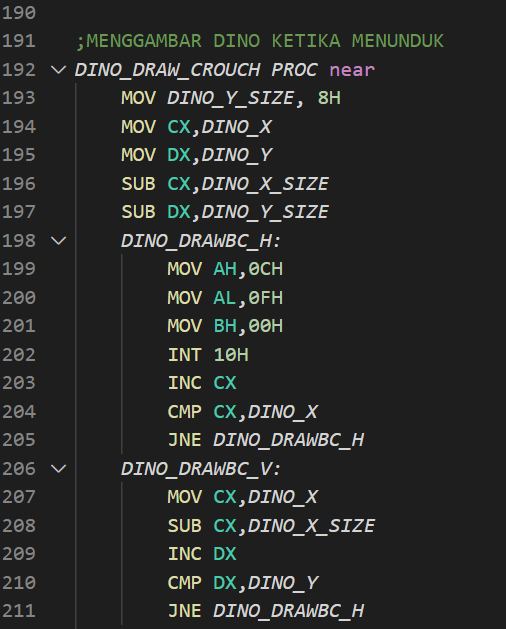
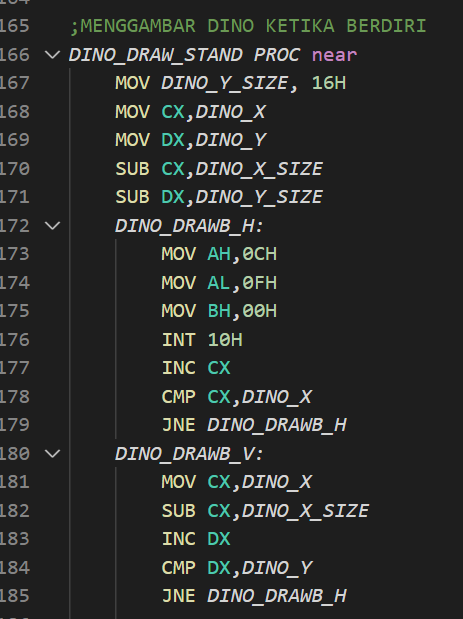


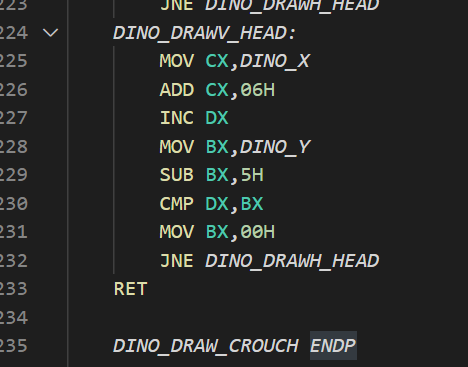
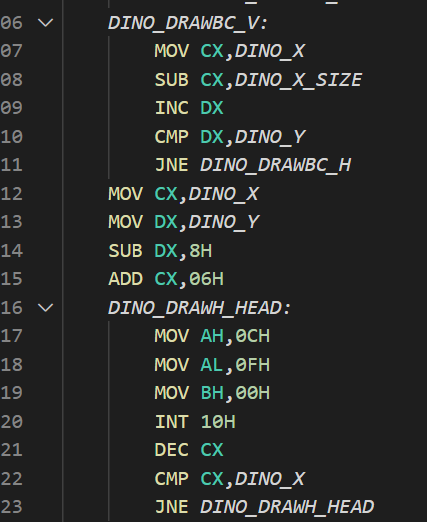


1. CALL CLEAR\_SCREEN: memanggil prosedur yang bertugas menghapus layar sehingga setiap frame akan terlihat baru.
2. TIME\_CHECK: prosedur yang bertugas mengecek waktu di layer setiap frame.
3. MOV AH,2CH: mengisi register AH dengan nilai 2CH sebagai kode untuk memanggil fungsi interrupt yang akan memberikan informasi waktu.
4. INT 21H: memanggil interrupt dengan kode 21H yang akan memberikan informasi waktu pada layar
5. CMP DL,TIME\_VAR: membandingkan nilai DL dengan TIME\_VAR untuk mengecek apakah terjadi perubahan waktu.
6. JE TIME\_CHECK: jika tidak terjadi perubahan waktu, maka kembali ke TIME\_CHECK.
7. INC SCORE\_TIME: menambah nilai SCORE\_TIME sebesar 1.
8. CMP SCORE\_TIME,5H: membandingkan nilai SCORE\_TIME dengan 5H (hexadecimal) untuk mengecek apakah terjadi perubahan waktu sebesar 20ms.
9. JNE NO\_SCORE: jika tidak terjadi perubahan waktu sebesar 20ms, maka ke NO\_SCORE.
10. MOV SCORE\_TIME,00H: mengisi SCORE\_TIME dengan 0.
11. NO\_SCORE: prosedur yang bertugas mengecek apakah terjadi tabrakan antara dino dan objek lainnya.
12. CALL COLLISION\_CHECK: memanggil prosedur yang bertugas mengecek apakah terjadi tabrakan antara dino dan objek lainnya.
13. CMP EXIT\_FLAG,00H: membandingkan nilai EXIT\_FLAG dengan 0 untuk mengecek apakah terjadi tabrakan.
14. JNE GAMEOVER: jika terjadi tabrakan, maka ke GAMEOVER.
15. JMP GAMEON: jika tidak terjadi tabrakan, maka ke GAMEON.
16. GAMEOVER: prosedur yang bertugas mengeluarkan pesan “gameover” pada layar.
17. CALL CACTUS\_MOVE: memanggil prosedur yang bertugas menggerakkan cactus dari kiri.
18. CALL BIRD\_MOVE: memanggil prosedur yang bertugas menggerakkan burung dari kiri.
19. MOV TIME\_VAR,DL: mengisi TIME\_VAR dengan nilai DL (informasi waktu yang didapat dari interrupt).
20. CALL CLEAR\_SCREEN: memanggil prosedur yang bertugas menghapus layar sehingga setiap frame akan terlihat baru.
21. CALL GROUND\_DRAW: memanggil prosedur yang bertugas mengeprint tanah ke layar.
22. CALL DINO\_MOVE: memanggil prosedur yang bertugas menggerakkan dino.
23. CMP FLAG2,1H: membandingkan nilai FLAG2 dengan 1H untuk mengecek apakah terjadi input ‘s’ (menunduk).
24. JE NUNDUK: jika terjadi input ‘s’, maka ke NUNDUK.
25. CALL DINO\_DRAW\_STAND: memanggil prosedur yang bertugas mengeprint dino yang berdiri ke layar.
26. JMP EXIT3: keluar dari prosedur.
27. NUNDUK: prosedur yang bertugas mengeprint dino yang menunduk ke bawah.
28. MOV FLAG2,00H: mengisi FLAG2 dengan 0.
29. EXIT3: keluar dari prosedur.
30. CALL CACTUS\_DRAW: memanggil prosedur yang bertugas mengeprint cactus ke layar.
31. CALL BIRD\_DRAW: memanggil prosedur yang bertugas mengeprint burung ke layar.
32. JMP TIME\_CHECK: menggerakkan ke TIME\_CHECK untuk memulai proses pada frame berikutnya.
33. EXITS: keluar dari program.

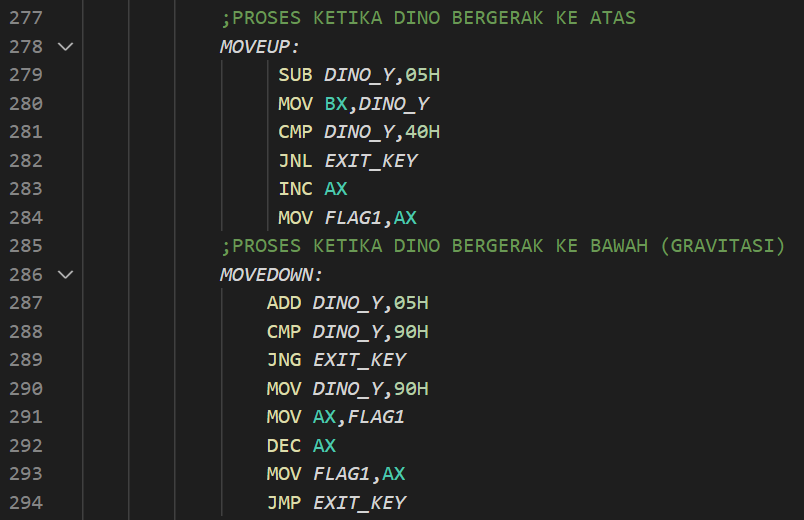
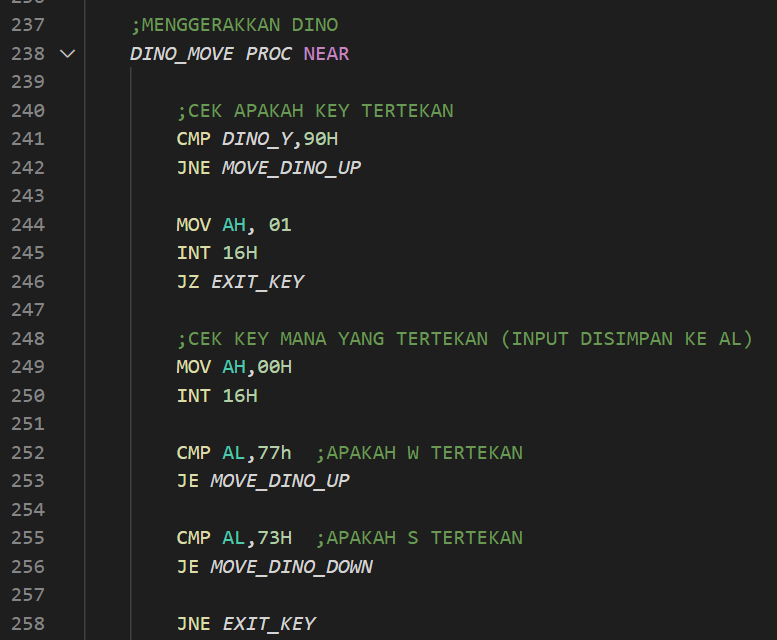


Kode Program di atas menggambar tanah dengan menggunakan bahasa assembly.

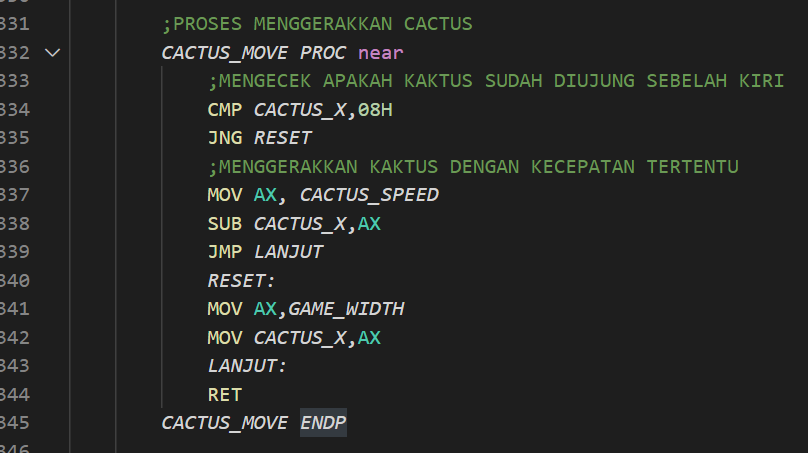
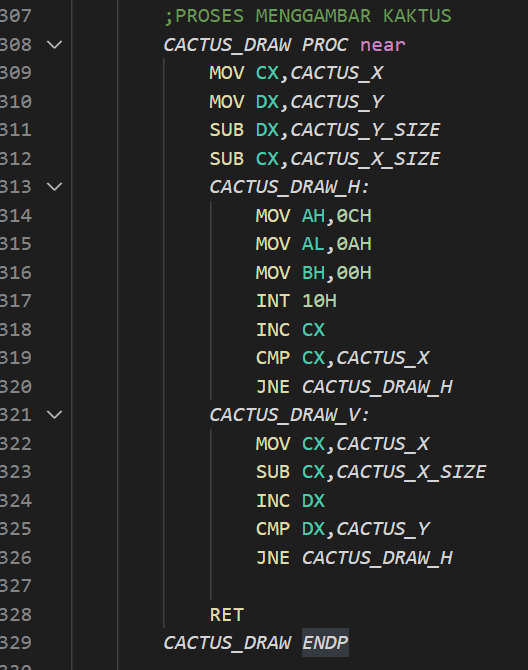




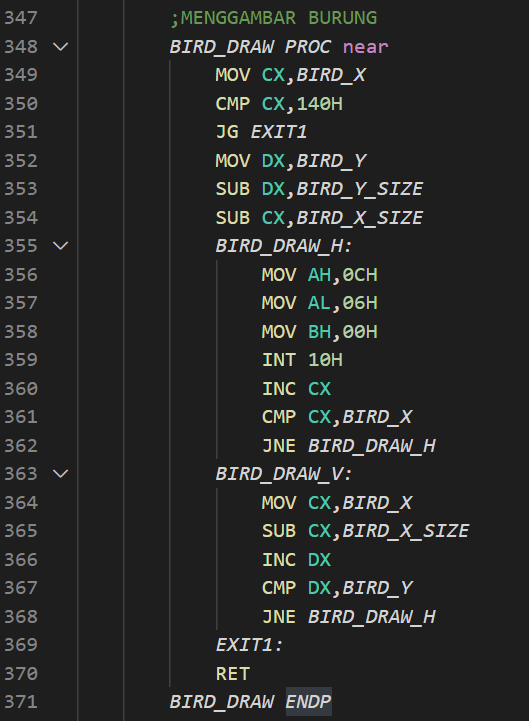
Kode program diatas digunakan untuk menggambarkan dino saat berdiri dan menunduk dengan Bahasa assembly.



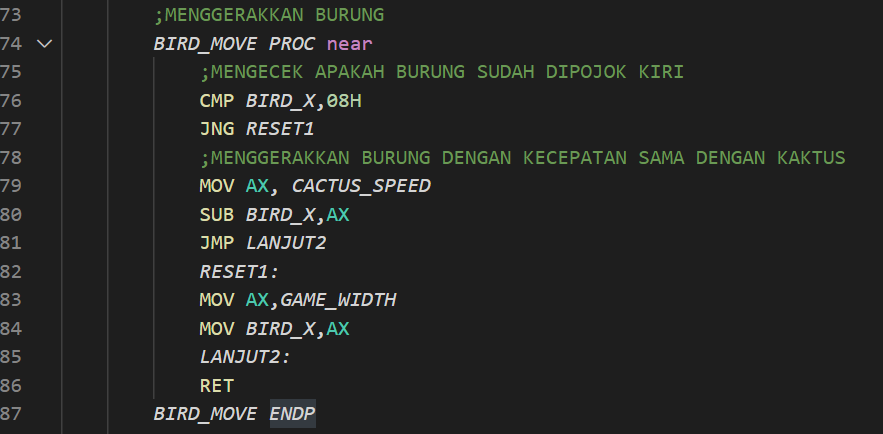
Kode program di atas digunakan untuk menggerakkan dino, digerakkan ke atas maupun ke bawah (gravitasi).



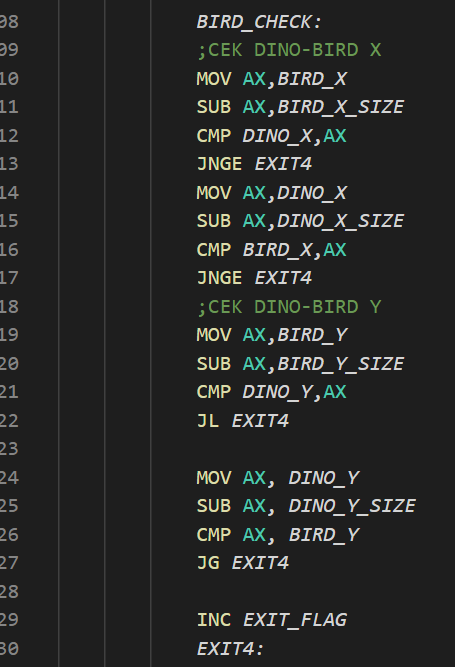
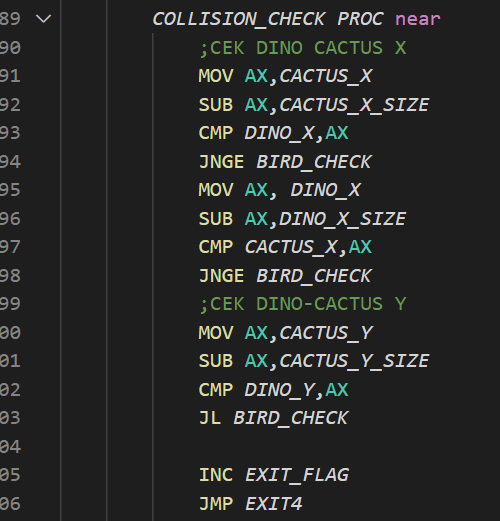
Kode program di atas digunakan untuk menggambar rintangan kaktus, kemudian digerakkan dengan kecepatan-kecepatan tertentu.



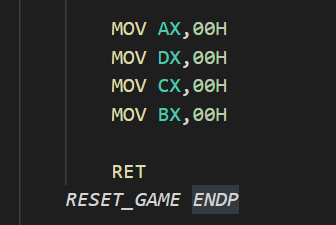
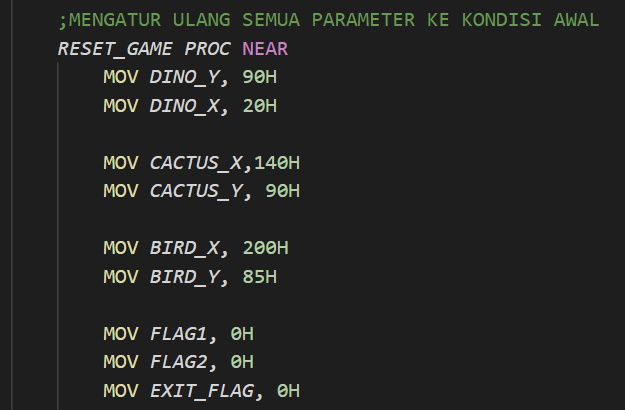
Kode diatas digunakan untuk menggambar burung.



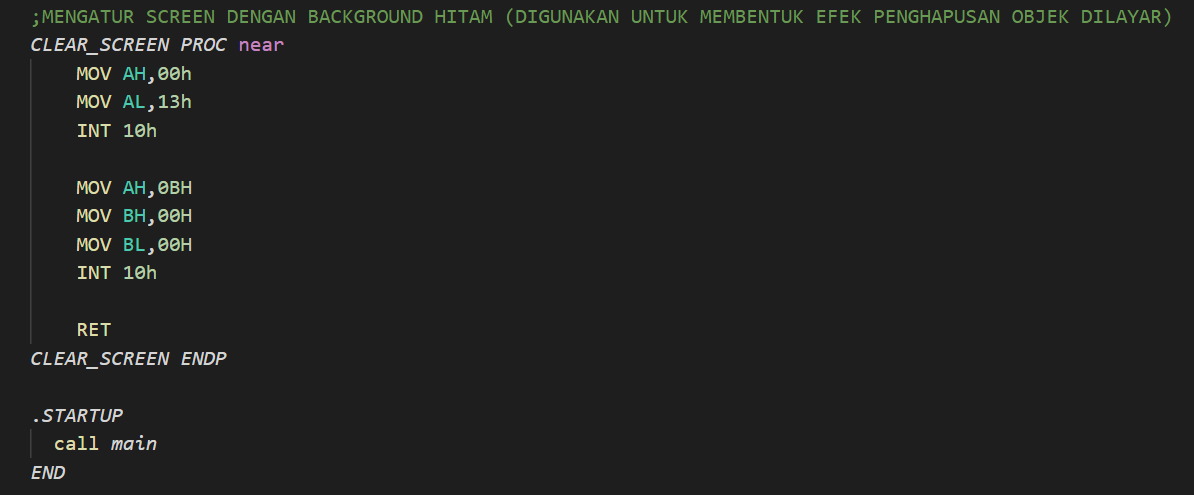
Kode tersebut digunakan untuk menggerakkan burung, apakah burung sudah dipojok kiri, dan kecepatan geraknya sama dengan kaktus.



Kode di atas bertujuan untuk mengecek terjadinya tabrakan antara dinosaurus dengan cactus atau burung di dalam game



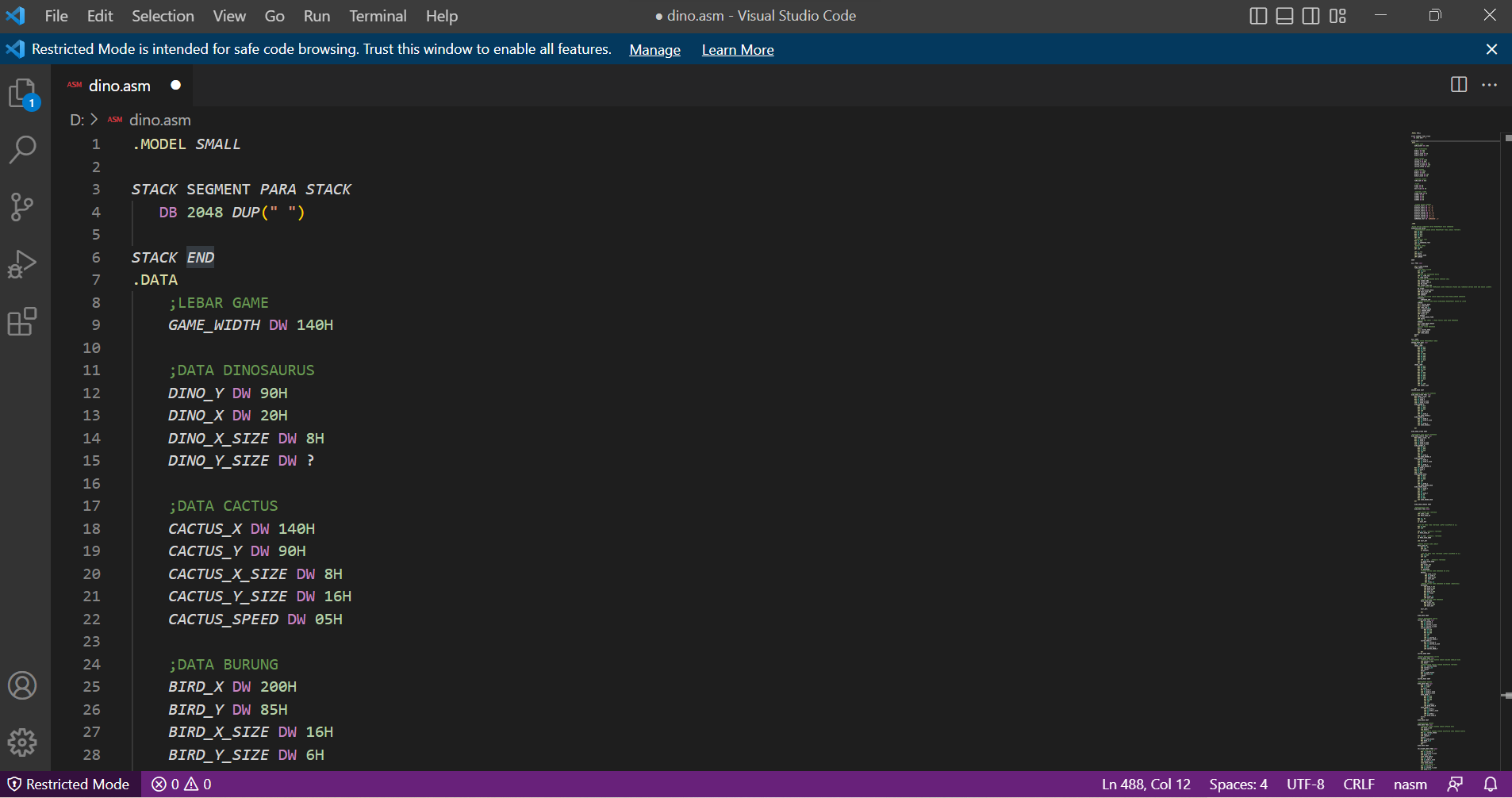
Kode di atas bertujuan untuk mengatur ulang semua parameter ke kondisi awal di dalam game. Jadi, fungsi utama dari kode di atas adalah untuk mengatur ulang semua parameter ke kondisi awal sehingga game dapat dimulai dari awal.



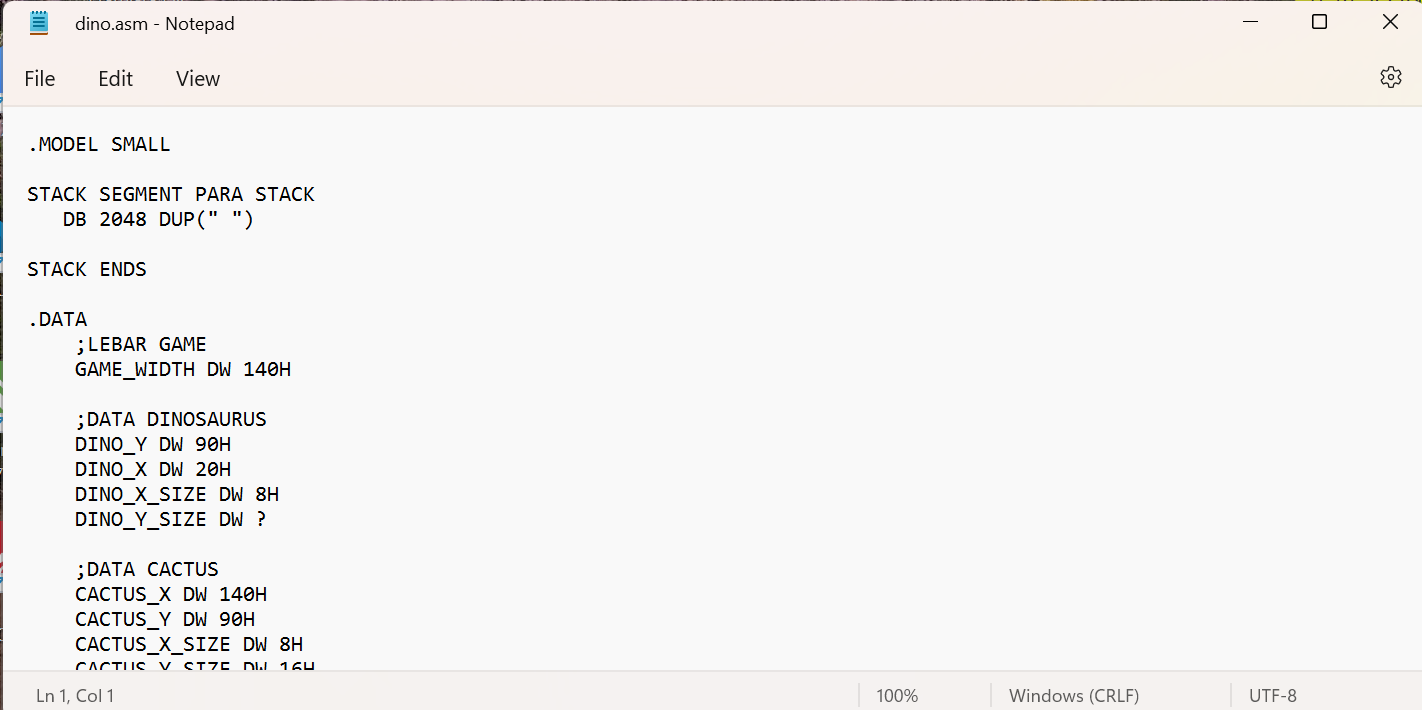
Kode di atas bertujuan untuk mengatur agar dapat dibersihkan dengan latar belakang warna hitam. Jadi, fungsi utama dari kode di atas adalah untuk mengatur mode video dan warna latar belakang agar dapat dibersihkan dengan warna hitam. Ini berguna untuk membentuk efek penghapusan objek-objek yang ada di layar sehingga dapat dibuat animasi yang lebih dinamis.

# **2.6 Pengujian Program Dino Runner Assembly**

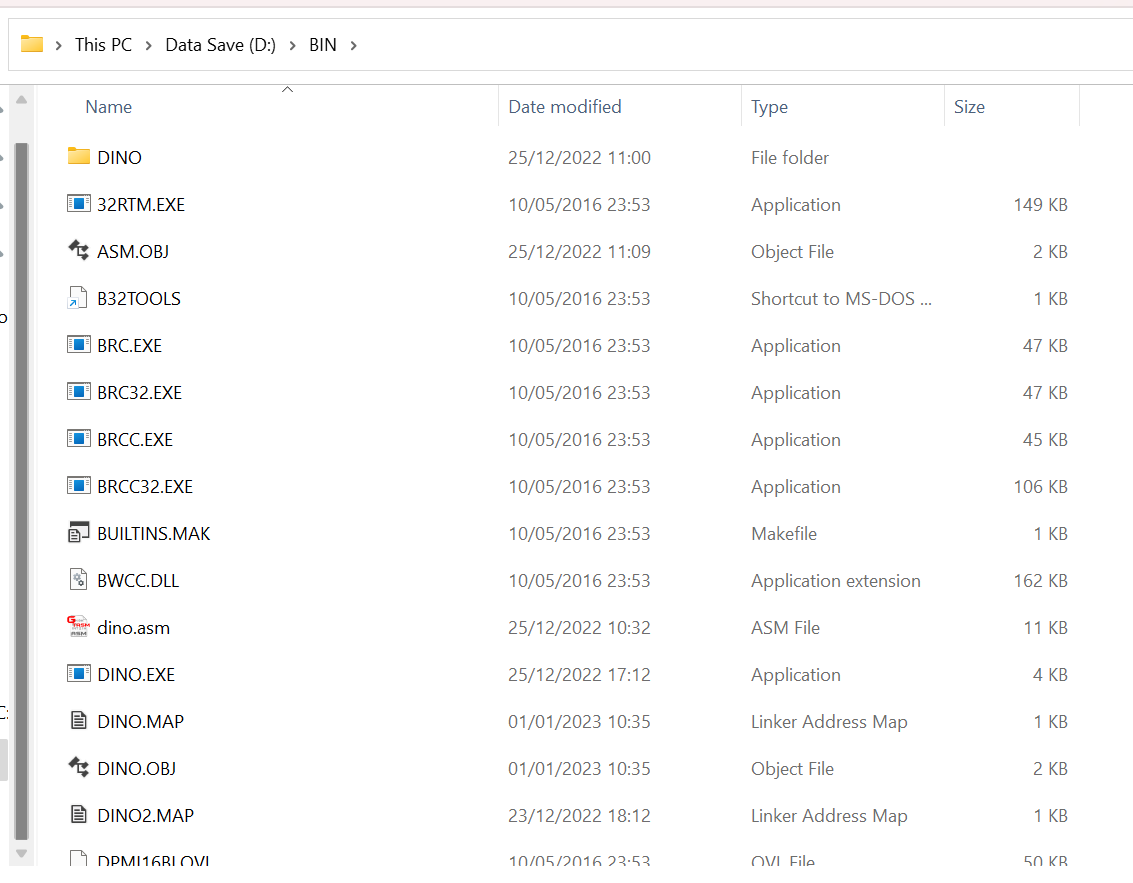
Pertama-tama kita koding program yang ingin dibuat di salah satu aplikasi untuk mengoding, disini saya menggunakan aplikasi Visual Code karena lebih mudah dan efektif.

****

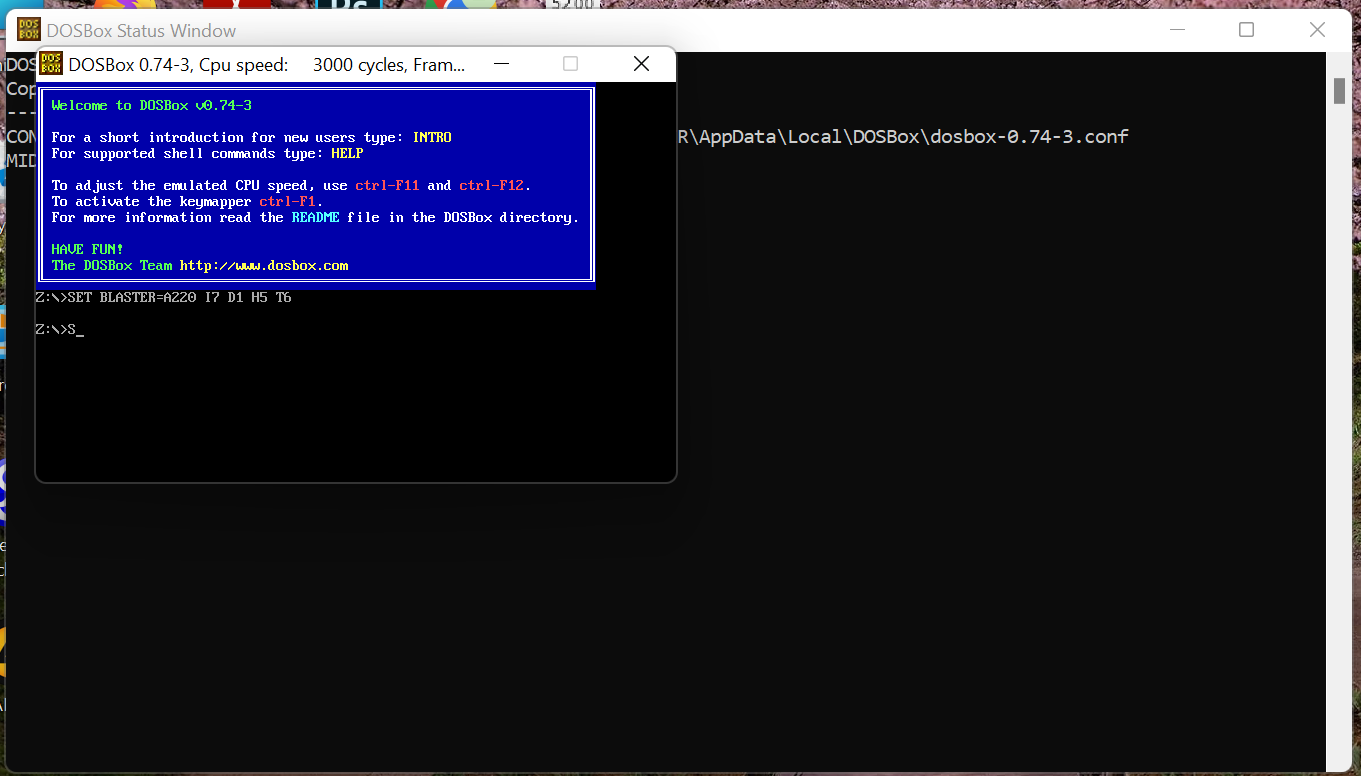
Kemudian kita buka Aplikasi notepad, kita tuliskan(paste) hasil kodingan kita tadi jika sudah, kita save dengan mengklik tombol CTRL+S, lalu kita save di dalam folder CD BIN. Disini saya namakan filenya dengan nama “dino”, karena kita menggunakan assembly maka kita buat filenya dengan file assembly dengan menambahkan “.asm” di akhir.

****

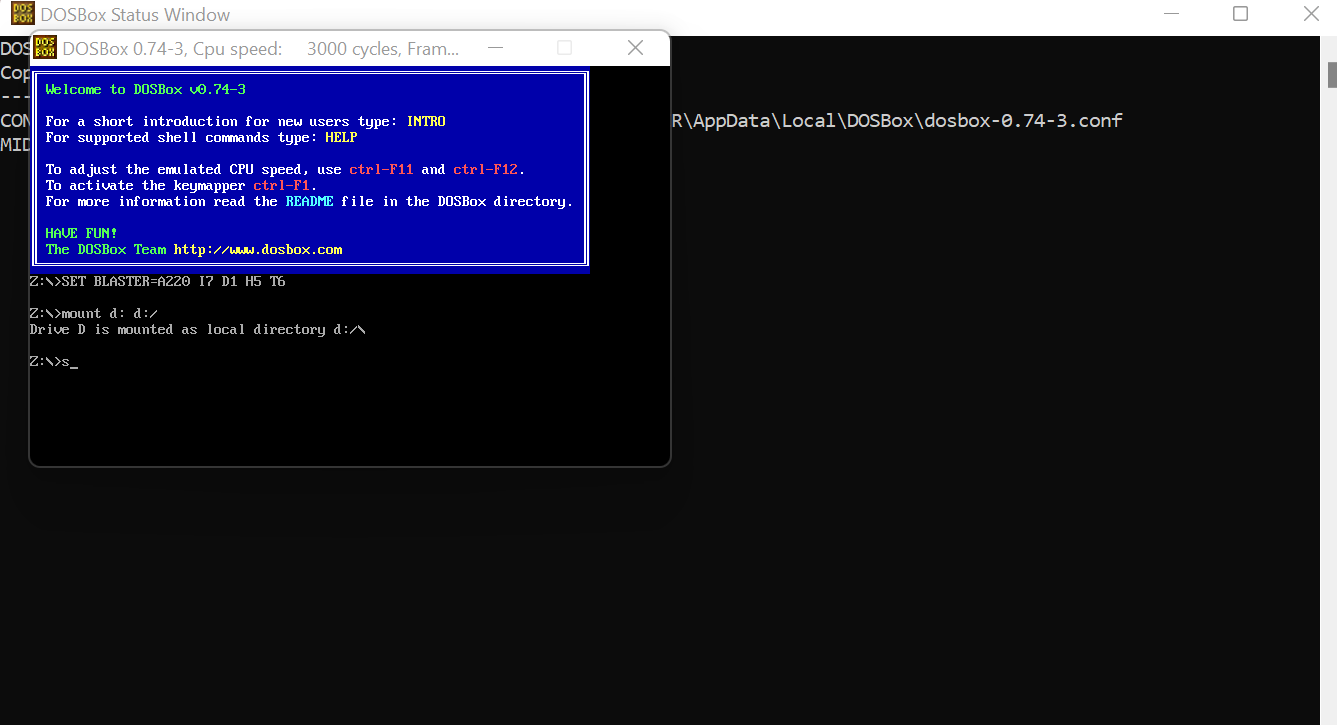
Setelah kita lik CTRL+S dari notepad tadi, kita save ke folder BIN

****

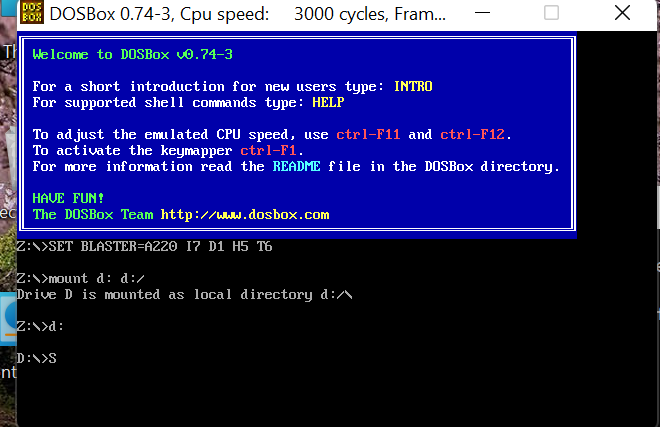
Langkah selanjutnya kita buka aplikasi DOSBOX



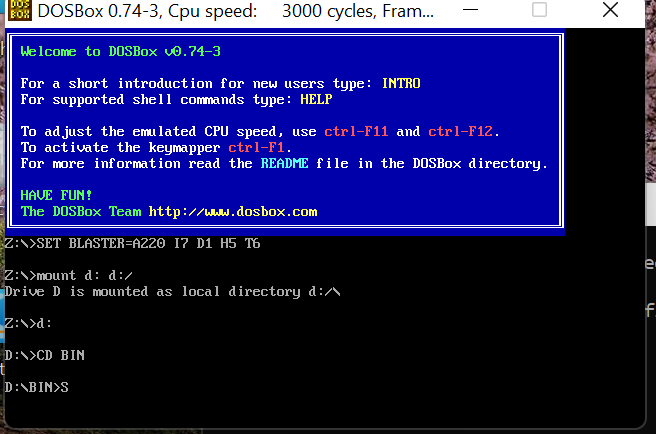
Kemudian kita tuliskan “mount d: d:/ “ pada aplikasi dosbox tersebut dan klik enter



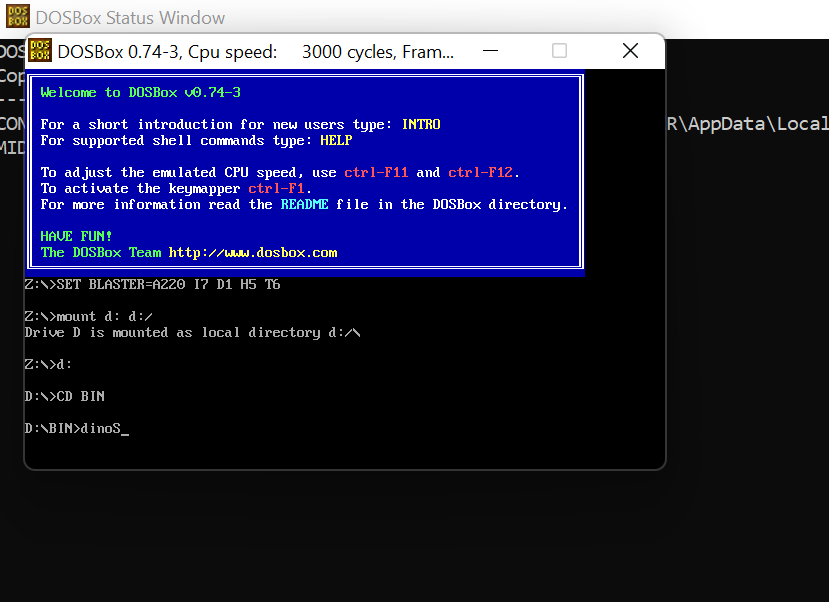
Setelah itu kita tulis “ d: “ pada dosbox dan klik enter



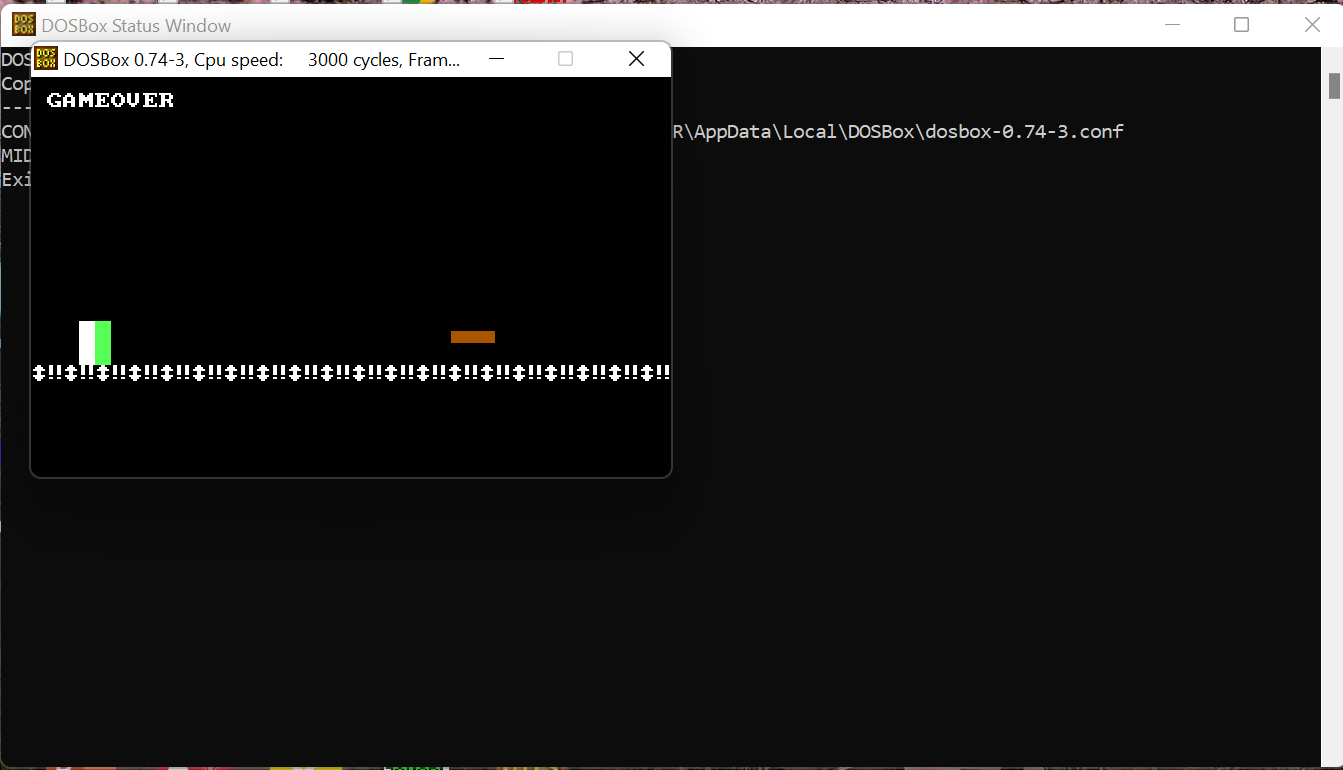
Selanjutnya kita tuliskan “CD BIN” dan klik enter



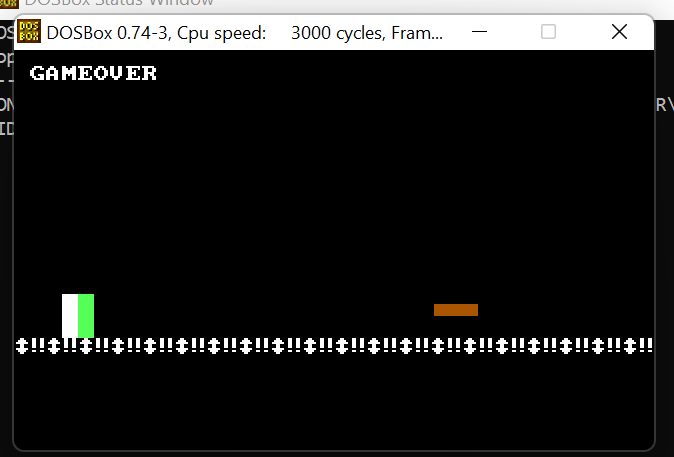
Dan kita tuliskan nama program yang dibuat tadi, karena saya tadi menuliskan filenya dengan nama “dino” maka ditulis di dosboxnya dino dan kita klik enter.



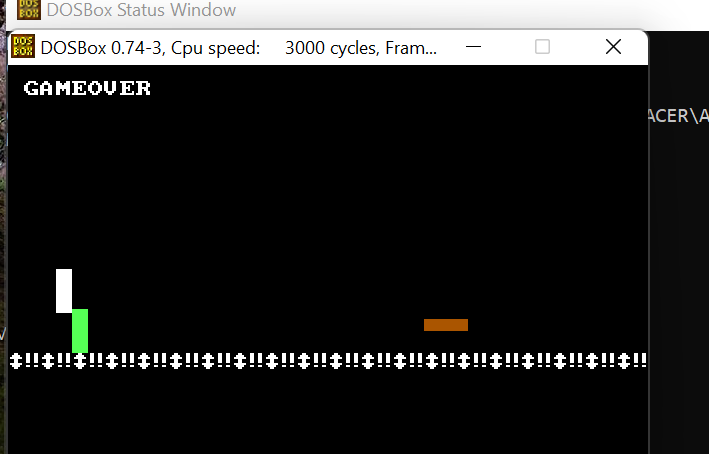
Setelah kita klik enter maka akan muncul program DINORASEM (Dino Runner Assembly).



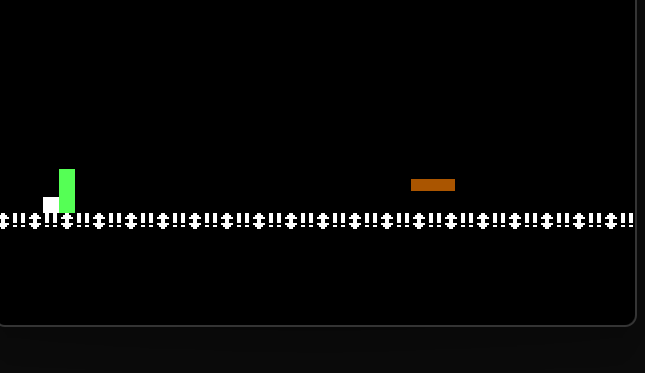
Maka Program dapat kita Jalankan tanpa error di aplikasi Dosbox tersebut



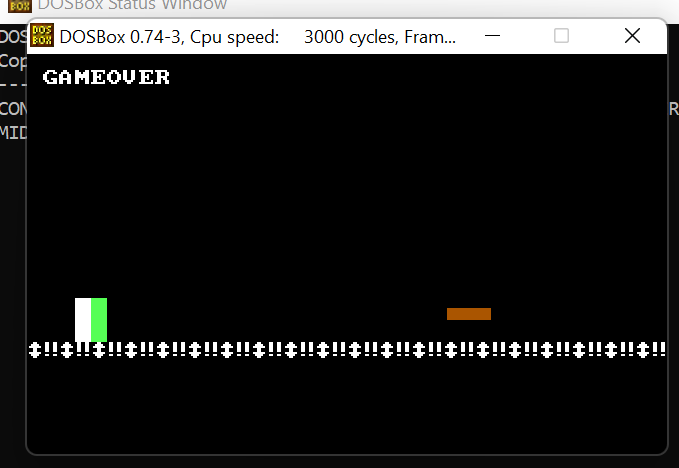
Untuk JUMP kita dapat mengklik tombo w



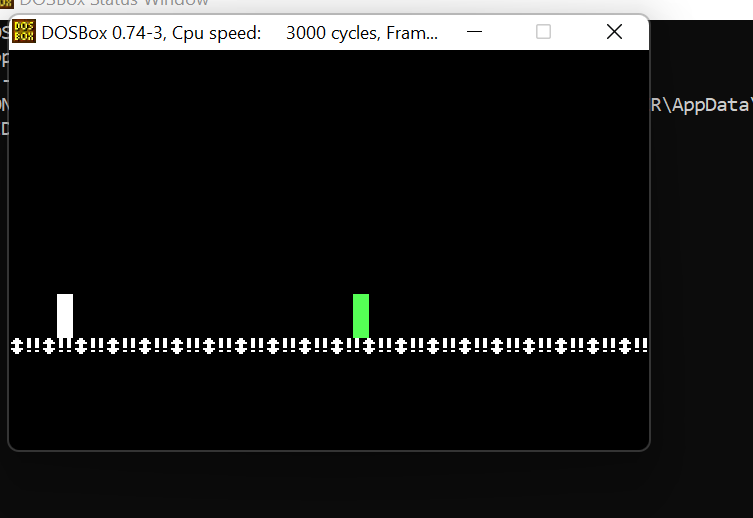
Untuk menunduk kita dapat mengklik tombol s



Apabila kita menabrak sesuatu maka akan ada tampilan GAMEOVER di kiri atas



Kemudian apabila kita ingin main Kembali, kita dapat mengklik tombol w saat ada tulisan GAMEOVER, maka program akan Kembali berjalan seperti semula.



# **BAB III**

# **PENUTUP**

Setelah melakukan proses analisis dan uji coba terhadap program aplikasi game DINORASEM(Dino Runner Assembly) yang telah saya buat, dapat disimpulkan bahwa program tersebut dapat berfungsi dengan baik sesuai dengan yang diharapkan. serta dapat menampilkan hasil akhir permainan yang sesuai dengan aturan yang berlaku. Hal ini menunjukkan bahwa program tersebut sudah dapat berjalan dengan baik dan sesuai dengan fungsi yang diinginkan.

Meskipun program aplikasi game DINORASEM(Dino Runner Assembly) yang saya buat dapat berfungsi sesuai dengan yang diharapkan, terdapat beberapa kekurangan yang perlu diperbaiki di masa yang akan datang. Salah satu kekurangan tersebut adalah tidak adanya fitur Highscore bagi pemain, sehingga pengalaman bermain tidak terasa menyenangkan bagi pemain. Selain itu, program ini juga belum memiliki fitur untuk menyimpan hasil permainan sehingga tidak dapat digunakan untuk mencatat prestasi pemain. Dengan demikian, di masa yang akan datang perlu dilakukan perbaikan-perbaikan pada program aplikasi ini agar dapat memberikan pengalaman bermain yang lebih menyenangkan dan memuaskan bagi pemain.

Selain itu, penambahan fitur karakter-karakter dinosaurus dan penyimpanan hasil permainan juga akan membuat program ini lebih berguna bagi pemain yang ingin meningkatkan keterampilannya dalam permainan Dino Runner Assembly. Fitur-fitur tersebut akan memudahkan pemain untuk melakukan evaluasi terhadap kinerjanya dan memiliki catatan yang terorganisir mengenai score yang telah diraih selama bermain. Dengan demikian, pemain dapat terus mengembangkan permainannya untuk menjadi lebih baik lagi.