Assignment Three

- Write a program
- Write a report

Overview

You want to become a professional assembly programmer.

In this assignment, you are going to implement a system for controlling the motion of a snake. **You should modify only the .asm file.** The program is called directly via the C main function. Your assembly procedures will be called while the program is being executed. Enjoy!

DO NOT CHEAT in programming. You should learn on your own.

Penalties

- If one of the following items:
  - your name, student ID or email address

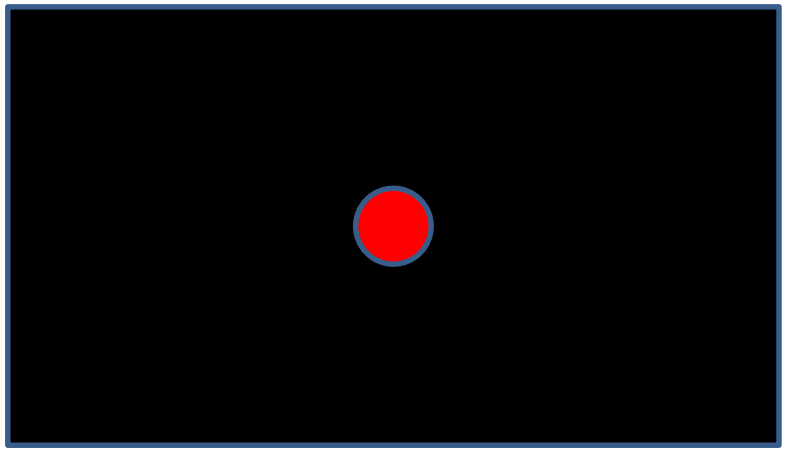
  does not appear at the top of your file, ***you will receive a score of ZERO.***
- ***If your file name is incorrect, you will receive a score of ZERO.***
- ***You MUST WORK INDEPENDENTLY for the project. If you copy / cheat in the project, the scores of all students involved are ZERO.***
- Late submission. Each day 10% deduction.

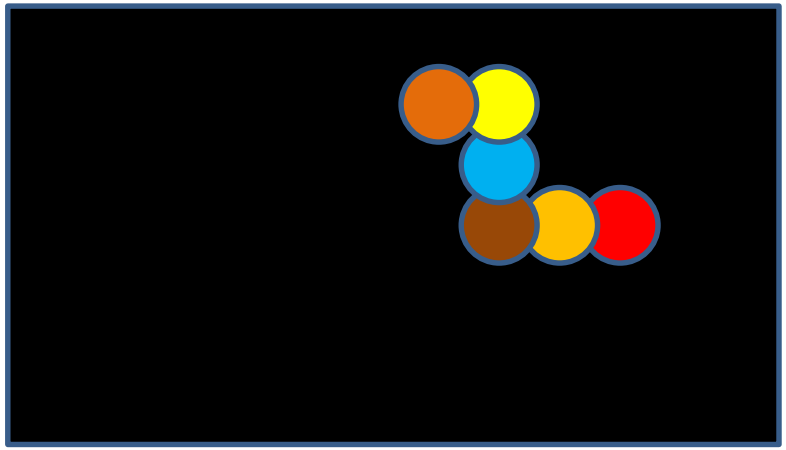
Problem Desription

- Write a program to use keys and mouse to control a snake. The snake consists of one or more than one sphere. Initially, the snake has only one sphere called *head*. The head moves around inside a canvas and the snake grows gradually.
- We also need to handle the background which is an image.

In this assignment you should understand arrays well.

Example





Structure of the program

- The program consists of C++ code and asm code. You should modify only the single .asm file. Do not change anything in C++ files.
- The work flow of the program is as follows:
  - Show a dialogue
  - Set the caption at the top bar
  - Initialize OpenGL and initialize your database (asm part)
  - Repeat until the ESC key is pressed
    - Handle key events
    - Handle mouse event
    - Update snake position
    - Render the snake and other objects (if any)
  - Show a ‘thank you’ dialogue and quit the program

Tasks

1. [2%] Show your information at the title bar of the window. See MY_INFO_AT_TOP_BAR and asm_getStudentInfoString. **MUST DO. If not, receive a score of zero.**
2. [2%] Show your information and the key usage instruction in a dialogue box. See CaptionString, MessageString and AskForInput_Initialization.

   You can insert any code in AskForInput_Initialization to initialize the database maintained by your system written in assembly language.

   See asm_HandleKey for handling key.

3. [2%] Show your information at the beginning in the console window. **MUST DO. If not, receive a score of zero.**
4. [2%] Ask the user to input the speed and life (or grow) cycle of the snake. After that the program enters into the play mode. If the user does not input anything, use the default value; speed is 100 and snake life cycle is 25. Your system should support at least 1024 spheres.
5. [2%] Before exiting the program, show a dialogue box. See asm_EndingMessage.

Programmer information and the key usage

Programmer Name:
Programmer ID:
Programmer Email Address:

‘a’: left; ‘d’: right, ‘w’: up; ‘s’: down
‘p’ : rainbow color
‘r’: random color
‘c’: clear
‘v’ : save
‘l’ : load
left mouse button: set target
spacebar : toggle grow / not grow
ESC : quit the program

Tasks

6. [10%] The snake moves in a specific direction until a control key is pressed. The initial movement direction is right.

   You should initialize the object in initGame.

   You should update the positions of the spheres in updateGame.

   Later on the OpenGL subsysem will call asm_ComputeObjPositionX and asm_ComputeObjPositionY to get the coodinates of a sphere.

   And the OpenGL subsysem will call asm_GetObjectColor to get the color of a sphere.

7. [10%] The object grows by one sphere for every *grow cycle*. The new sphere should be placed at the current position of the head of the object. Thus after the object moves for a while, there will be many spheres placed along the path of the object.

8. [5%] If the head of an object hits a boundary of the canvas, the movement direction of the object is reversed.

Tasks

9. [5%] The head of the object must be red. The colors of the other spheres are randomly generated if ‘r’ is pressed.
10. [10%] The colors of the spheres are ordered in the colors of rainbow if ‘p’ is pressed. The head’s color is red.
11. [0%] Press ‘v’ to save the positions and colors of all the spheres.
12. [5%] Press ‘c’ to clear all the spheres except for the head of the object.
13. [10%] Press ‘l’ (the lower case of ‘L’) to load the saved data of the spheres, including color and position.
14. [15%] Press the left mouse button to set a target. The head moves to the target. After the head hits the target, it should stop moving.
15. [5%] Press the space bar to toggle the growing state of the snake: can grow or cannot grow new spheres.
16. [5%] Press the ESC key to quit the program. Show the dialogue before the program is terminated. See task item 5.

Tasks

17. [5%] asm_SetImageInfo sets the image data to the address pointed by mImagePtr. The dimension is width x height and the number bytes of per pixel are provided. Make sure that mImageWidth and mImageHeight are set correctly.

18. [5%] asm_GetImageColour gets the colour (r, g, b) for each pixel at position (ix, iy).

r: red; g: green; and b: blue.

Useful Information

The maximum number of objects used in the OpenGL subsystem is defined in asm_GetNumOfObjects.

This value should not be larger than 2048 in the OpenGL subsystem.

The canvas dimension is

[canvasMinX, canvasMaxX] for x-axis

and

[canvasMinY, canvasMaxY] for y-axis.

Useful Information

asm_ComputeObjPositionX

and asm_ComputeObjPositionY return the x-coordinate and y-coordinate of an object with an object ID (identifier) objID.

asm_GetObjectColor returns an object color in (r, g, b), i.e., (red, green, blue).

About the connection between asm procedures and C/C++ functions.

In an assembly program, a procedure can be defined in several forms. We explain two different forms in this instruction.

Procedure form I:

```
asm_func01 PROC C       ; C convention
                        ; for function call
  ;function body
  mov eax, value
  ret                   ; return to the caller
asm_func01 ENDP
```

asm_function01 is the procedure name. “PROC C” means that the assembler should translate the function into a *C* calling convention (*C* is a programming language).

asm_function01 does not have any parameter. It returns a value in eax at the end. *Notice that whenever we want to return an integer, the value must be stored in the register eax before the function exits.*

Procedure form II:

```
asm_func02 PROC C,
  x: DWORD, y:DWORD, …
  ;x and y are passed by value
  ;function body
  ret          ;return to the caller
asm_func02 ENDP
```

asm_function02 has two parameters (x and y) or more.

If necessary, we can return a value in eax.
An example for passing by reference is shown as follows:

```
asm_func03 PROC C USES ebx,
x: PTR DWORD, y:DWORD, … ;x is passed by reference
;
;To assign a value to x, write two instructions:
   mov ebx, x                ; x is an address!
   mov [ebx], value          ; move a value to x by using
                             ; indirect addressing.
   ret                       ; return to the caller
asm_func02 ENDP
```

Notice that we can also use other registers instead of ebx, e.g. edi and esi.
Note ebx may be used outside. Thus, push and pop ebx (i.e., USES ebx).

Always give a good name to the value if it is a constant.

Final Note

IMPORTANT: always restore EBX, EDX, EDI and ESI of the callers as these registers are preserved by the callers in the C calling convention.

Submission

DO NOT CHEAT in programming. You should learn on your own.

1. Upload your source code to nctu E3 platform. Store source code in a folder. The folder should **contain a report and the main asm file**.

   The source code must be well aligned and documented.

   - **The asm file name must be: asm_StudentID.asm**
   - For example, if your ID is 123456789, then the file name must be asm03_123456789.asm And the folder name must be asm03_123456789

   - **At the top of the .asm file, you must fill out your name, student ID and email.**

2. **A hardcopy report must be submitted in class.**

Student Name:

Student ID:

Student email address:

Assignment Three Report

Report format

[10%] Introduction [at least 100 words]

WORD COUNT:_____ [Must be filled or zero score]

//Write down the purpose(s) of this program… (remove this line or zero score)

[10%] Structure Chart [at least 10 components]

//Draw the structure chart (remove this line or zero score)

[10%] Flow Chart

//Draw the flow chart (remove this line or zero score)

[10%] System Architecture [**at least 100 words]**

WORD COUNT:_____ [Must be filled or zero score]

//Describe the system (remove this line or zero score)

[30%] The approach [at least 300 words]

WORD COUNT:_____ [Must be filled or zero score]

//How do you implement the program (remove this line or zero score)

[20%] Discussion/Experiments [at least 200 words]

WORD COUNT:_____ [Must be filled or zero score]

//Write down what you want to share with us (remove this line or zero score)

[10%] Conclusion [at least 100 words]

WORD COUNT:_____ [Must be filled or zero score]

//Summary: write down what you have learnt…. What experiences that you want to share with others… Write down what you expect to learn in the future… (remove this line or zero score)

END

Enjoy programming ☺