ĐẠI HỌC KHOA HỌC TỰ NHIÊN – ĐẠI HỌC QUỐC GIA TP. HCM

KHOA CÔNG NGHỆ THÔNG TIN

Đồ án môn học: Kỹ thuật lập trình

BÁO CÁO ĐỒ ÁN

Họ và tên: *Nguyễn Thị Thu Hằng*
MSSV **:** 18120027
Lớp **:**18CTT1
Email **:**18120027@student.hcmus.edu.vn

Bộ môn Công Nghệ Phần Mềm
Khoa Hệ Thống Thông Tin
Đại học Khoa học tự nhiên TP HCM

MỤC LỤC

1. Đánh giá mức độ hoàn thành

- Mức độ hoàn thành 100%:
  + Đã xử lí đầu snake chạm vào thân snake
  + Đã xử lí lưu trò chơi và tải trò chơi đã lưu
  + Đã xử lí giữ nguyên độ dài snake
  + Đã xử lí khi ăn xong food ở 1 cấp
  + Đã xử lí hiệu ứng khi va chạm
  + Đã xử lí màn hình chính

2. Kịch bản trò chơi

Lúc đầu khi vào game sẽ xuất hiện một MENU để cho các bạn có thể lựa chọn bao gồm "1.PLAY GAME" , "2.LOAD GAME" ,"3.EXIT" sau đó các bạn có thể lựa chọn bằng cách bấm trực tiếp các số tưng ướng trên bàn phím.

2.1 PLAY GAME

Sau khi bạn chọn mục này trên console sẽ xuất hiện một "snake" và sẽ tự động di chuyển, người chơi sử dụng các phím "A" , "D" , "S" , "W" để điều chỉnh hướng di chuyển. Khi "snake" va chạm tường, chạm vào thân hoặc các chướng ngại vật thì chương trình thông báo yêu cầu người chơi chọn phím "Y" nếu muốn tiếp tục (chương trình sẽ reset trò chơi lại như lúc ban đầu) hoặc chọn "bất kì phím nào" nếu muốn thoát trò chơi. Khi "snake" ăn được 4 "food" thì tốc độ di chuyển nhanh hơn, nghĩa là lên 1 cấp (độ dài của snake sẽ trở về kích thước lúc đầu là 8). Khi lên cấp 3 thì dữ liệu sẽ reset lại như lúc ban đầu.

2.2 LOAD GAME

Sau khi bạn chọn mục này trên console sẽ xuất hiện một thông báo nhập tên file đã lưu các đối tượng (lưu ý file phải có sẵn) và sau đó trên console sẽ xuất hiện 1 "snake" và các đối tượng khác đã được lưu trong quá khứ sẽ tự động thực thi, người chơi sử dụng các phím "A" , "D" , "S" , "W" để điều chỉnh hướng di chuyển. Khi "snake" va chạm tường, chạm vào thân hoặc các chướng ngại vật thì chương trình thông báo yêu cầu người chơi chọn phím "Y" nếu muốn tiếp tục (chương trình sẽ reset trò chơi lại như lúc ban đầu) hoặc chọn "bất kì phím nào" nếu muốn thoát trò chơi. Khi "snake" ăn được 4 "food" thì tốc độ di chuyển nhanh hơn, nghĩa là lên 1 cấp (độ dài của snake sẽ trở về kích thước lúc đầu là 8). Khi lên cấp 3 thì dữ liệu sẽ reset lại như lúc ban đầu.

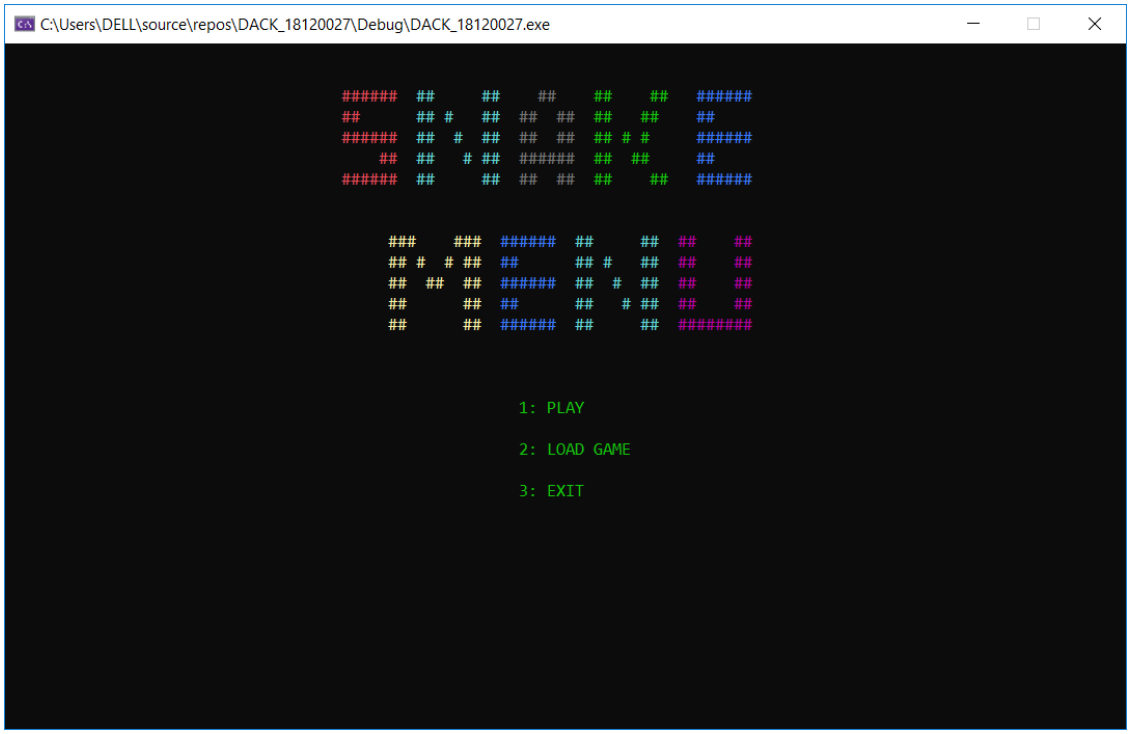
2.3 HIGH SCORE

Sau khi bạn lựa chọn mục này chương trình hiện lên một bảng xếp hạng các người chơi từ cao xuống thấp dựa trên số điểm mà các người chơi ghi được. Nhấn "P" để thoát ra menu hoặc "Esc" để thoát khỏi trò chơi.

2.4 EXIT GAME

Sau khi bạn lựa chọn mục này chương trình sẽ tự động thoát ra .

**Trong quá trình chơi các bạn có thể bấm nút "P" để dừng game tạm thời, nút "L" để có thể lưu lại game và nhấn "T" để load game.*



Menu Game Trước khi vào trò chơi

3. Yêu cầu đồ án:

3.1 Xử lý khi đầu snake chạm vào thân snake:

Trong hướng dẫn chưa xử lý việc đầu snake chạm vào thân snake. Khi điều này xảy ra ta cũng xử lý tạm dừng trò chơi và hỏi ý kiến người dùng xem có muốn tiếp tục hay không?

Trong yêu cầu này, thêm 4 hàm mới vào 4 hàm di chuyển tương ứng với 4 hướng di chuyển là

các câu lệnh if vào các hàm MoveLeft(), MoveRight(), MoveU(), MoveDown().

Ý tưởng chung của cả bốn hàm này là một vòng for. Nếu tọa độ đầu con rắn cộng thêm một bằng tọa độ của một cái thân nào đó trong con rắn thì ta cho chạy hàm Void ProcessDead();. Nếu ngược lại ta trả về hàm di chuyển.

3.2 Xử lý lưu trò chơi/ tải trò chơi đã lưu:

Sinh viên bổ sung chức năng khi người dùng nhấn phím "L" thì xuất hiện dòng lệnh yêu cầu người dùng nhập tên tập tin cần lưu lại. Tương tự khi nhấn phím "T" thì xuất hiện dòng lệnh yêu cầu người dùng nhập tên tập tin cần tải lên.

Trong yêu cầu này, em đã viết ra hai hàm:

- Void SaveGame(): Hàm này dùng để tạo một file mới với tên người dung nhập vào để lưu những dữ liệu cần thiết cho chương trình như: SIZE_SNAKE, Snake[],FOOD_INDEX, Food[], level,rock[],score,Gate[], Moving, CHAR_LOCK, … .Những dữ liệu này sẽ được in vào file vừa tạo.
- Void LoadGame(): Hàm này dùng để tìm file có tên người dùng nhập vào và đọc những dữ liệu có trong file đó lưu vào các biến toàn cục và chạy chương trình.

Cả hai hàm trên đều được cài đặt trong hàm main. Nếu ấn "T" thì chương trình sẽ chạy vào hàm LoadGame, nếu ấn "L" thì chương trình sẽ

chạy vào hàm SaveGame().

3.3 Xử lý giữ nguyên độ dài của Snake:

Trong hướng dẫn, mỗi khi lên cấp (SPEED++) thì kích thước snake được thiết lập lại từ đầu (mặc định là 8). Sinh viên xử lý sao cho độ dài của snake vẫn phải giữ nguyên khi lên cấp. Chỉ khi lên MAX_SPEED thì mới thiết lập lại từ đầu.

Trong yêu cầu này, em đã thay đổi giá trị biến toàn cục MAX_SIZE_FOOD thành 24, thay đổi vị trí phép toán SIZE_SNAKE=8; trong hàm Eat() nhét vào câu lệnh if(FOOD_INDEX==MAX_SIZE_FOOD-1),thì sẽ dùng một biến temp lưu lại độ dài rắn.

3.4 Xử lý khi ăn xong food ở một cấp:

Trong hướng dẫn, khi snake ăn hết 4 phần tử trong mảng food thì sẽ lên cấp. Tuy nhiên sinh viên hiệu chỉnh lại sao cho sau khi ăn 4 phần tử mảng food thì sẽ xuất hiện một cổng cho snake quay về. Khi snake quay về xong thì mới xử lý lên cấp.

Trong yêu cầu này, em đã thêm vào chương trình mảng cổng có 5 phần tử và các hàm sau:

- Bool IsValid(): Hàm này kiểm tra xem mỗi phần tử của mảng cổng có tọa độ trùng với mảng snake hay không ?
- Void GenerateGate(): Hàm này tạo một chiếc cổng ngẫu nhiên, cánh cổng đó nằm trong vùng snake có thể đi qua và nó không được trùng với mảng food hoặc mảng snake.
- Void DrawGate(): Hàm này dùng để vẽ cánh cổng tại tọa độ đã được định trước
- Void cleanGate(): Hàm này dùng để xóa cánh cổng đã được vẽ ra và được tất cả các tọa độ cổng về mặc định là (0,0).

Và có một câu lệnh trong hàm ThreadFunc() là nếu tọa độ cổng thứ 0 là (0,0) thì mới vẽ thức ăn, điều đó cũng dễ hiểu là khi con rắn đã đi qua cổng thì mới vẽ thức ăn ra. Và khi đầu rắn chạm vào tọa độ (x,y+1) thì lệnh SIZE_SNAKE- -sẽ giúp con rắn mất đi từng đốt tạo hiệu ứng rắn đã chui qua cổng.Đồng thời khi rắn đã chui qua cổng thành công thì level+1 tương ứng

với đã lên một cấp mới. và đồng thời các vật cản sẽ được khởi tạo và làm trò chơi thêm độ khó.

3.5 Xử lý hiệu ứng khi va chạm:

Khi snake va chạm với tường rào, thân snake hoặc cổng vào thì tạo hiệu ứng đơn giản minh họa việc va chạm.

Trong hàm xử lí chết ProcessDead() sẽ đã lồng ghép vòng lắp nhỏ giúp cho rắn có thể liên tục đối màu.

3.6 Xử lý màn hình chính:

Khi mở trò chơi, chương trình cho phép người dùng chọn "Tải lại" (phím "T") hoặc "Bắt đầu chơi" (phím bất kì). Nếu người dùng chọn phím "T" thì yêu cầu người dùng nhập tên tập tin và sau đó vào trò chơi. Ngược lại thì bắt đầu trò chơi với dữ liệu gốc mặc định. Snake màn hình chính là chuỗi mã số sinh viên, ví dụ sinh viên có mã số là 1312918 thì chuỗi snake là: 1312918, khi snake ăn mồi thì chèn kí tự đầu tiên lặp lại. Ví dụ sau khi snake ăn mồi thứ nhất thì chuỗi snake là 13129181, ăn mồi lần thứ hai sẽ là 131291813…

Trong yêu cầu này em đã tạo ra một giao diện Menu bằng hàm drawMenu() cho người sử dụng lựa chọn bắt đầu game hoặc tải lại game đã chơi. Em dùng các hàm StartGame(), LoadGame() đã viết hoặc có sẵn. Bên cạnh đó còn có hai lựa chọn khác đó là HIGH SCORE để giúp cho người chơi có thể theo dõi bảng xếp hạng và EXIT giúp người cho có thể nhanh chóng thoát khỏi game khi không muốn tiếp tục chơi nữa.

Trong yêu cầu sử lí các đốt rắn là MSSV thì em sử dụng hàm drawSnake(char sna[]) để thực hiện việc đó. Cụ thể hơn là em sẽ vẽ rắn tại các tọa độ có sẵn và kí tự in ra tương đương với kí tự trong mảng nameSnake đã được thiết lập trước. Mỗi đối rắn tương đương với kí tự nameSnake[i%8].

3.7 Các hàm bổ sung

Hàm Void Nocursortype(): Ẩn con trỏ nhấp nháy trên màn hình Console.

Hàm highCore() giúp in ra bảng xếp hạng theo thành tích của người

chơi .

Hàm GenerateRock(), drawRock(): giúp khởi tạo và vẽ vật cản tăng độ khó cho trò chơi.

Hàm drawGuideBoard(), drawInfoBoard(): hướng dẫn và cập nhật thông tin người chơi.

Và còn nhiều hàm bổ sung khác….

4. Các thư viện:

Các thư viện đã dùng:

```
#include <iostream>
#include <Windows.h>
#include <time.h>
#include <thread>
#include <conio.h>
#include <stdio.h>
#include <fstream>
#include <string.h>
```

5. Mô tả:

5.1 Màn hình chính:

Ô chính:

+ Kích thước: 70×20

+Được bao bọc xung quang bởi các kí tự "+"

+Nơi các đối tượng chính của trò chơi thực hiện nhiệm vụ của mình

Ô phụ 1:

+Kích thước 50×12

+Được bao bọc xung quanh bởi các kí tự "+"

+Được sử dụng để chứa những hướng dẫn về các phím điều khiển.

Ô phụ 2:
+Kích thước :50×8
+Được bao bọc bởi các kí tự "=" và "|"
+Được sử dụng để chứa đựng thông tin của trò chơi về cấp độ (level) và điểm số (Score).

5.2 Rắn:

Ban đầu rắn có độ dài là 8
Bao gồm các đốt là các kí tự {'1' , '8' , '1' , '2' , '0' , '0' , '2' , '7'}
Ban đầu rắn xuất phát tại tọa độ (10,5) và đi theo hướng sang phải

5.3 Thức ăn:

Thức ăn được biểu diễn bởi kí tự 'O'
Được khởi tạo trước khi vào mỗi bàn chơi và thức ăn sẽ xuất hiện lần lượt.
Sau khi ăn được 4 "food" thì lên một cấp độ mới.

5.4 Cổng:

Được biểu diễn bởi kí tự 'o'
Được khởi tạo sau kết thúc vào mỗi bàn chơi và sẽ biến mất khi rắn đi qua.
Sau khi ăn được 4 "food" thì sẽ xuất hiện một cổng báo hiệu sẽ chuyển qua bàn chơi mới.
Khi rắn chạm phải thành cổng bàn chơi sẽ kết thúc

5.5 Vật cản:

Được biểu diễn bởi kí tự '#'
Được khởi tạo khi bắt đầu vào mỗi bàn chơi (từ cấp độ 2 trở đi)
Khi rắn chạm phải bàn chơi sẽ kết thúc

5.6 Di chuyển:

Khi rắn săn được thức ăn: là khi vị trí của đầu rắn trùng với vị trí thức ăn.
Khi rắn va vào vật cản: là khi vị trí của đầu rắn trùng với vị trí của vật cản.
Rắn di chuyển sang phải: tìm vị trí hiện tại của rắn; tìm vị trí đuôi(vi trí cuối cùng của rắn) và xóa phần đuôi => vẽ lại rắn.Đối với phần tử đuôi rắn và đến sát đầu => gán vị trí thứ i=i-1.Đối với vị trí phần đầu => tọa

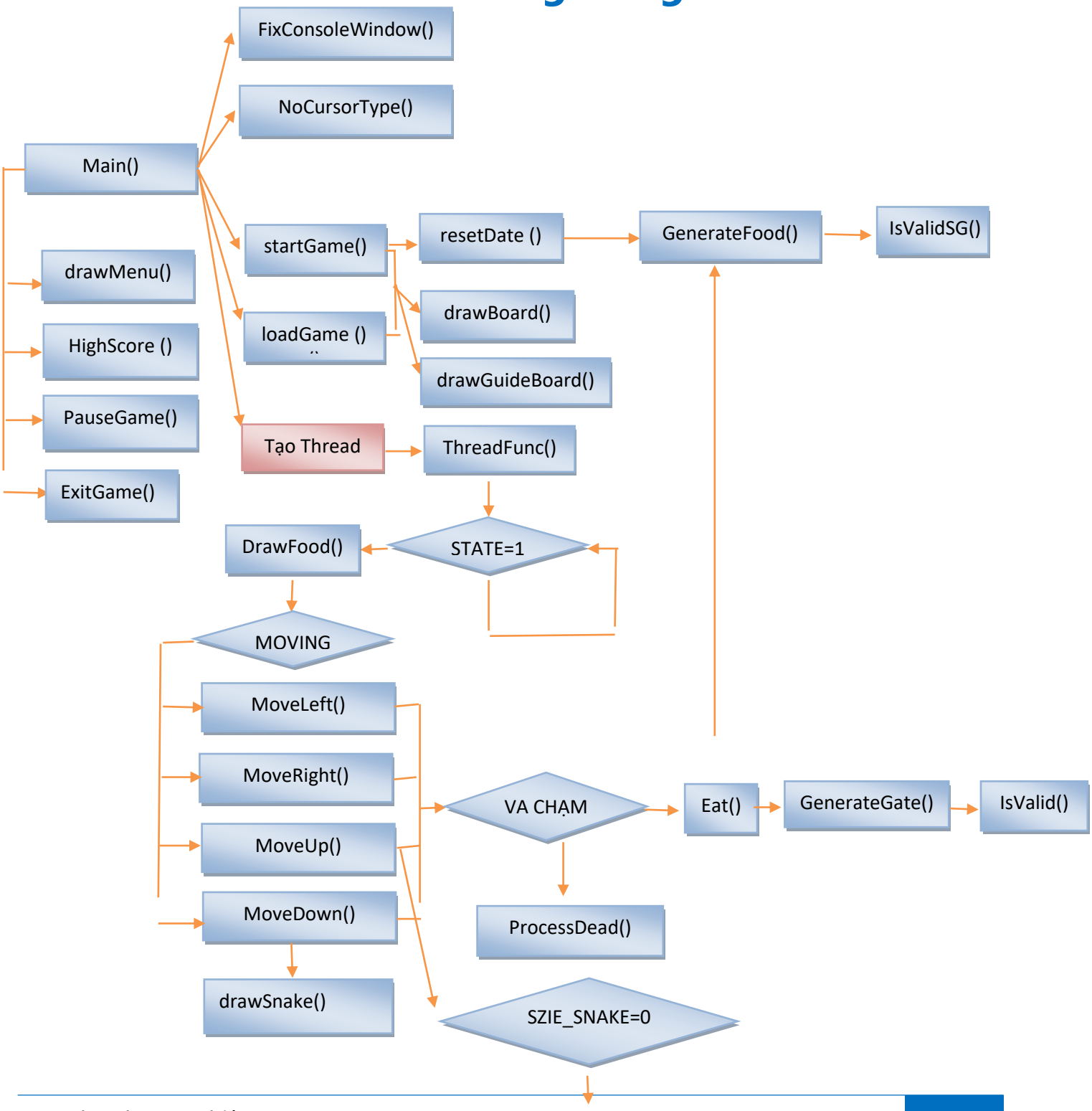
độ y+=1.
Tượng tự, di chuyển sang trái.Đối với phần đầu => tọa độ y-=1.
Tương tự di chuyển xuống dưới: Đối với phần đầu => tọa độ x+=1
Tương tự di chuyển lên trên: Đối với phần đầu => tọa độ x-=1.

5.7 Luật chơi:

Điều kiển rắn bằng các phím "A" (di chuyển sáng trái), "D" (di chuyển sang phải) , "W" (di chuyển lên trên) , "S" (di chuyển xuống dưới) sao cho rắn ăn được tất cả thức ăn xuất hiện trên console. Nếu trong quá trình di chuyển mà Rắn chạm vào thành tường bao quanh , chạm vào thân , vật cảm, thành cổng thì bàn chơi sẽ kết thúc. Tốc độ và độ dài rắn sẽ phụ thuộc vào số thức ăn mà rắn săn được trong quá trình chơi. Cứ khi ăn được 4 thức ăn thì sẽ lên một cấp, và đồng nghĩa với đó tốc độ sẽ tăng lên.
Có nhiều level khác nhau, tuy nhiên khi qua mỗi level thì tốc độ sẽ thay đổi(cho đến các cấp độ chia hết cho 3 thì tốc độ và độ dài sẽ chuyển về mặc định, chỉ có khác là số lượng vật cản sẽ tăng lên theo cấp độ).

Màn hình trò chơi cùng các đối tượng

6. Phân tích chức năng của game:



Sơ đồ gọi hàm từ main()

6.1.Cài đặt màn hình chính :

Cố định màn hình bằng hàm FixConsoleWindow() để có thể điều chỉnh dễ dàng và gọi hàm NoCursortype() để ẩn con trỏ chuột trên màn hình console.

6.2.Menu:

-Dùng bàn phím nhập yêu cầu cần thực hiện:

6.2.1 Play Game

Sau khi chọn 1 từ bàn phím,sẽ vào màn hình chính trò chơi với các thông số mặc định ban đầu thông qua hàm resetData(). Sau đó, vẽ bảng chính thông qua hàm drawBoard() và vẽ bảng phụ thứ nhất chứa các chỉ dẫn thông qua hàm drawGuideBoard(),bảng phụ thứ 2 chứ các thông tin trò chơi như điểm số và cấp độ hiện tại thông qua hàm drawInfoBoard().Sau đó tạo thread, và chạy thông qua ThreadFunc().

6.2.2 Load Game

Sau khi chọn 2 từ bàn phím, các thông số sẽ được đọc qua tên file vừa được nhập.Sau đó các bảng cũng được vẽ ra thông qua các hàm tương tự phần play game(nhưng vì sao lại gọi startGame trước khi gọi loadgame) chỉ đơn giản là vẽ bảng thôi và sau khi gọi loadGame() thì sẽ cập nhật được giá trị tọa độ mà đối tượng được lưu.Còn lại tương tự phần play game.

6.2.3 High Score

Sau khi bạn lựa chọn mục này chương trình hiện lên một bảng xếp hạng các người chơi từ cao xuống thấp dựa trên số điểm mà các người chơi ghi được nhờ hàm highScore(). Nếu muốn thoát ra thì người chơi có thể nhấn "P" để thoát ra menu hoặc "Esc" để thoát khỏi trò chơi.

6.2.3 Exit Game

Sau khi chọn 3 từ bàn phím, chương trình sẽ tự động ngừng các hoạt động nhờ hàm ExitGame()

6.3.ThreadFunc():

Trong thủ tục này ta thấy có vòng lặp vô tận, nếu STATE =1 tức là snake đang sống thì ta mới tiếp tục kiểm tra biến MOVING xử lý di chuyển, ngược lại STATE = 0 có nghĩa là rắn đã chết như vậy vòng lặp sẽ không làm gì (Màn hình khi đó sẽ giữ nguyên nội dung). Cụ thể trong vòng lặp while thì trước hết in con rắn với kí tự " " và kiểm tra nếu không xuất hiện cổng thì mới in kí tự thức ăn. Sau đó kiểm tra MOVING xem thỏa yêu cầu nào thì thực hiện(ví A: rẽ trái,...). Và cuối cùng mới gọi hàm drawSnake(nameSnake) để in con rắn đúng với tọa độ đã được cập nhật trong các hàm MoveLeft(), MoveRight(), MoveUp(), MoveDown().

6.4. Vòng lặp vô hạn trong hàm main()

Trong hàm main() có một vòng lặp vô hạn nhằm giúp chương trình chạy cho đến khi có hiệu lệnh dừng. Bao gồm một lệnh nhập kí tự từ bàn phím và các câu lệnh điều kiện.Nếu nhập "P" sẽ thực hiện hàm PauseGame() nhằm ngừng chương trình một khoảng thời gian. "T" nhằm thực hiện loadgame(). "L" thực hiện nhiệm vụ saveGame(). Và "Esc" nhằm để thoát khỏi chương trình ngay lập tức. Bên cạnh đó là các kí tự di chuyển "A", "D" , "S" , "W" giúp cho trò chơi có nhiều sự lựa chọn, linh hoạt.

7. Những khó khăn gặp phải khi thực hiện đồ án.

Một số khó khăn gặp phải

- Khó khăn thứ nhất: tổ chức dữ liệu file
- Khó khăn thứ hai: thiếu sót trong nếu bấm T thì không trở lại bàn chơi được
- Khó khăn thứ ba: nếu loadGame bằng tên không tồn tại sẽ báo lỗi.

8. Hướng phát triển:

Bên cạnh việc phát triển các game mang tính khéo léo cần đòi hỏi sự kiên nhẫn thì những thể loại game mang tính chất trí tuệ cũng trở thành những game hot trong những năm trở lại đây. Chính vì vậy mô hình game mang tính chất suy luận sẽ là đối tượng tiếp tục nghiên cứu trong thời gian tới

9. Các nguồn tài liệu tham khảo

- https://courses.fit.hcmus.edu.vn/pluginfile.php/113216/mod_resource/content/1/HDTH-DoAn1.pdf
- https://www.stdio.vn/articles/stdthread-trong-c-583
- https://daynhauhoc.com/t/an-con-tro-chuot-tren-man-hinh-console/34411
- https://www.academia.edu/37081615/H%C6%B0%E1%BB%9Bng_D%E1%BA%ABn_Vi%E1%BA%BFt_Game_TETRIS
- https://github.com/htn274/First-year-HCMUS/tree/master/FinalProject/1612880/Snake