Object-Oriented Programming

PROJECT

이　름 : 박태성

학　번 : 2015722031

학　과 : 컴퓨터정보공학부

DUE DATE : 2019 년 6 월 14 일

Contents

1. Introduction
   A. Implement a snake game using Linked List, a kind of dialog and data structure supported by MFC. The rules and limitations of the snake game are as follows.
      - A snake is placed on a 25 * 25 size map.
      - When the game runs, the snake moves to the left.
      - When playing the game, the snake's feeding will be randomly placed at different locations on the map.
      - The snake is unable to change direction from the current direction of movement to the opposite direction.
      - Whenever a snake eats its food, the length of the snake becomes longer.
      - When the snakes get out of the map or hit their bodies, the game is finished with a message box appropriate for the situation.
      - The game is over when the snake takes up the entire screen by eating all the food. You do not need to implement this feature in this project.
   B. Functions
      i. Snake.h
         1. void CreateSnake(snake** SnakeHead); // Function to create starting snake
         2. void MoveSnake(UINT SetMove, snake *SnakeHead); // Function that moves snake
         3. BOOL CrashBody(const snake *SnakeHead); // Function when snake crash to body
         4. BOOL EatFood(snake** SnakeHead, food* Food); // Function that changes when a snake eats its food
         5. BOOL CrashWall(snake* SnakeHead); // Function when snake crash to wall
         6. void CreateFood(food** Food); // Function to create first food
      ii. I_LOVE_MFCDlg.h
         1. void DrawMap(CDC *pDC); // Function to draw map
         2. void DrawSnake(CDC *pDC); // Function to draw snake
         3. void DrawFood(CDC *pDC); // Function to draw food
         4. void StartGame(); // Function to create initial state
         5. afx_msg void OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags); // public type MFCkeydown function
         6. BOOL CheckState(); // Function to check if a snake has collided or fed
         7. afx_msg void OnTimer(UINT_PTR nIDEvent); // public type MFCOnTimer function

2. Algorithm

   I_LOVE_MFC

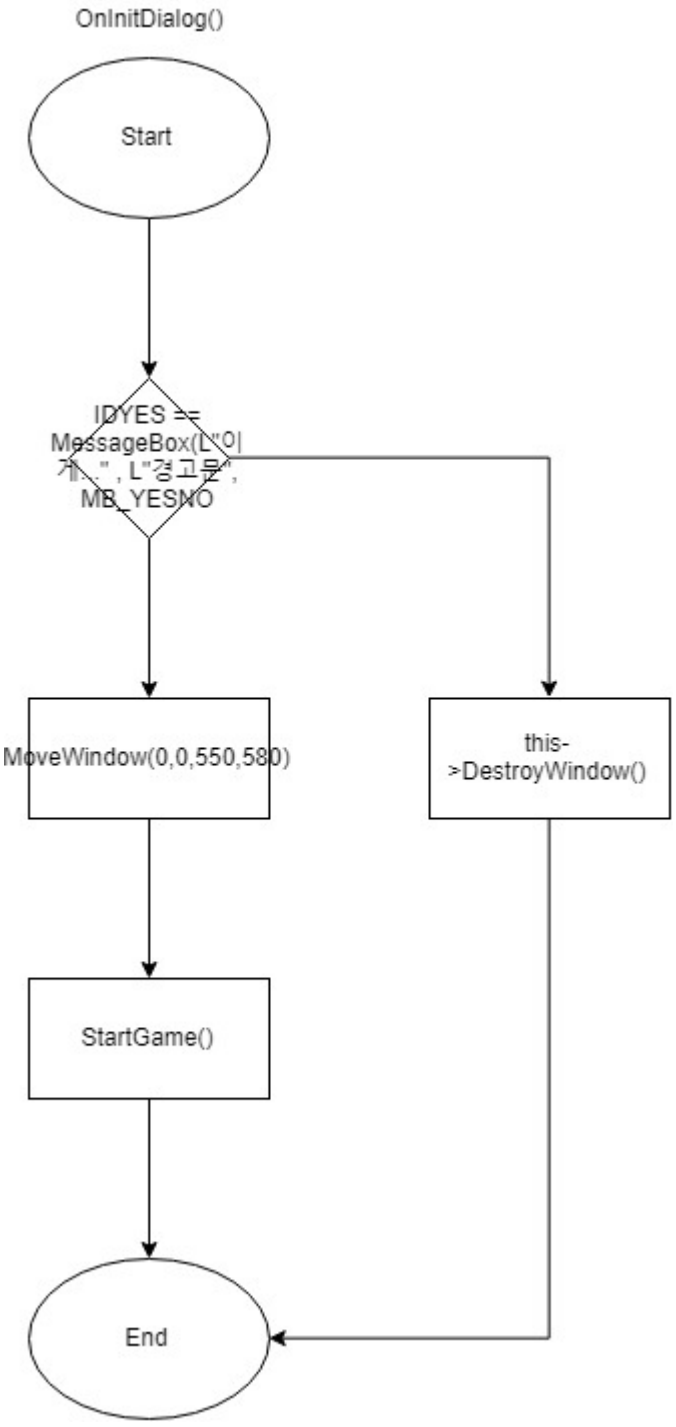
   BEGIN

   1. When the program is executed, a message box asking whether or not to proceed the game is displayed.
   2. When the player presses Yes, it prints out a window and calls StartGame (). If the player clicks NO, the game ends.
   3. If the player clicked Yes in step 2, OnTimer () and OnKeyDown () are executed sequentially, while StartGame () is executed.
   4. OnPaint () draws the status of Map, Snake, and Food in a window.
   5. Steps 2, 3, and 4 are repeated until the game completion condition of OnKeyDown () is satisfied.

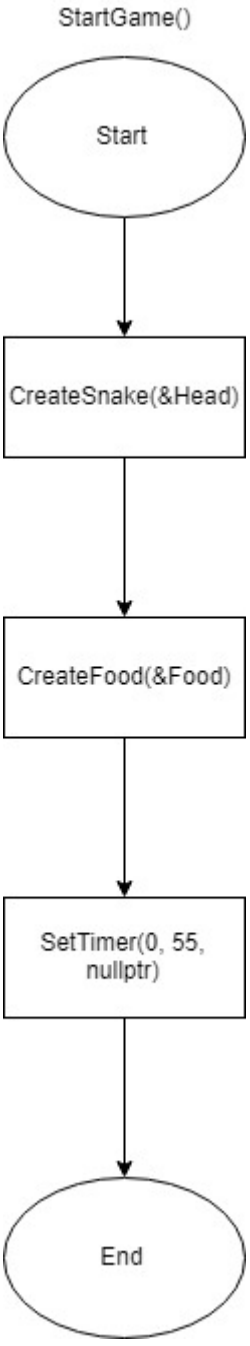
   END

3. Flowchart

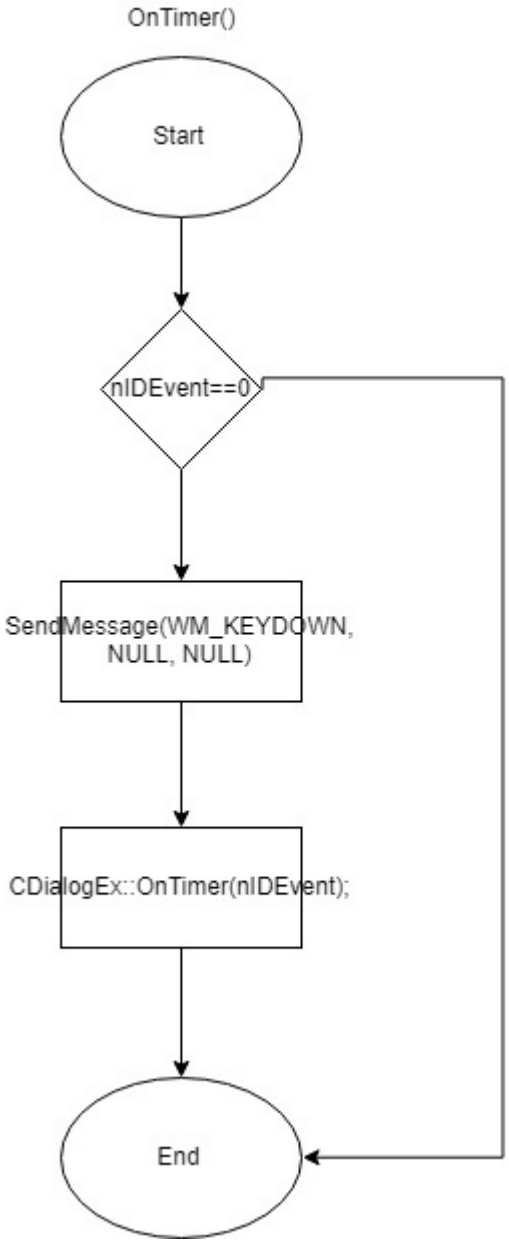
   A. OnInitDialog()


OnInitDialog()
Start
IDYES ==
MessageBox(L"이
게 ", L"경고문",
MB_YESNO
MoveWindow(0,0,550,580)
this-
>DestroyWindow()
StartGame()
End

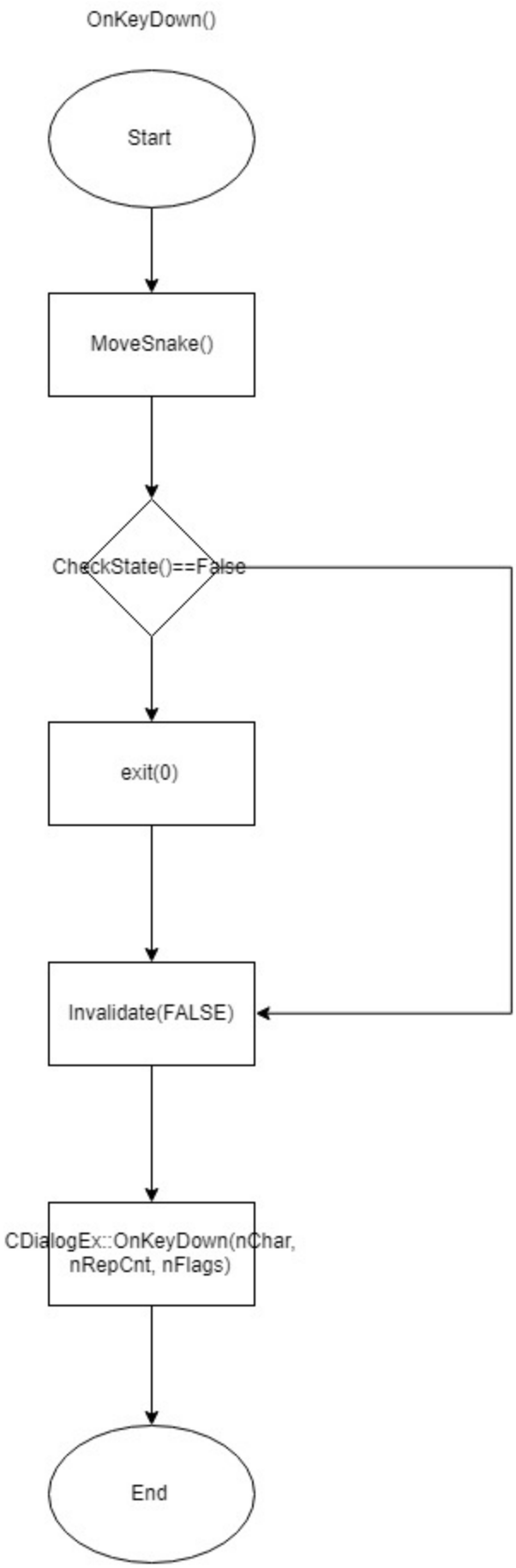
B. StartGame()


StartGame()
Start
CreateSnake(&Head)
CreateFood(&Food)
SetTimer(0, 55, nullptr)
End

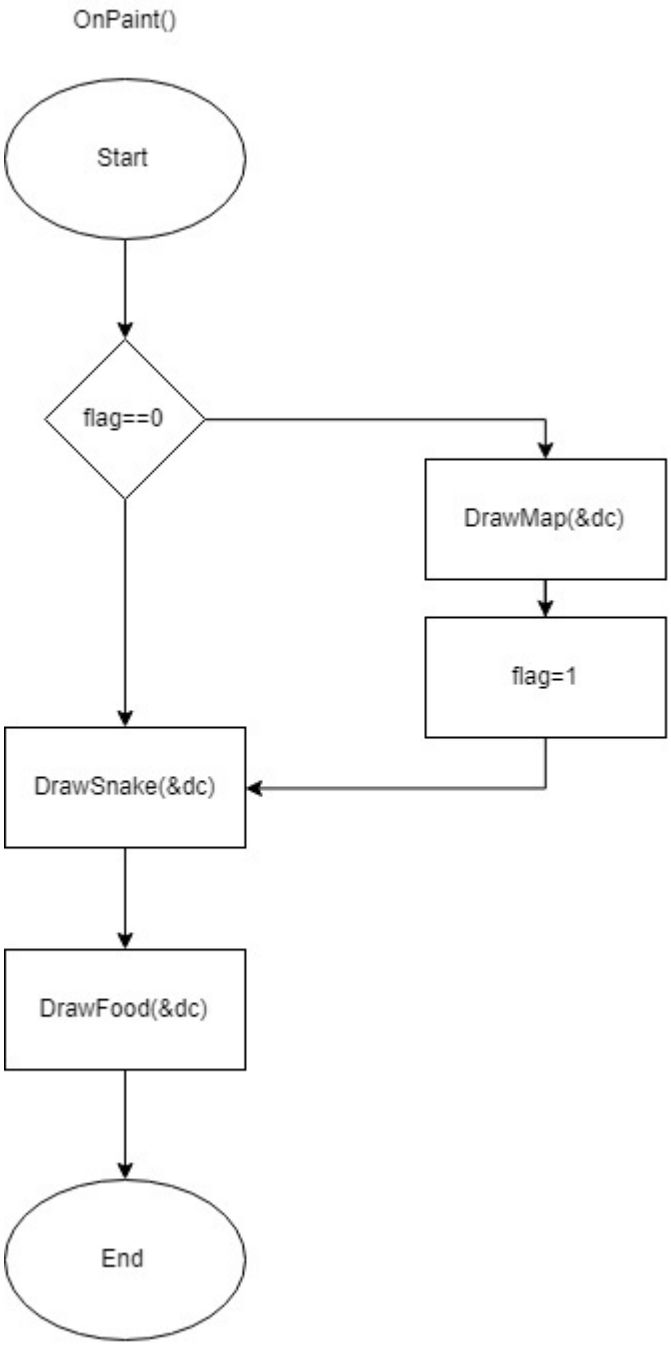
C. OnTimer()



D. OnKeyDown()

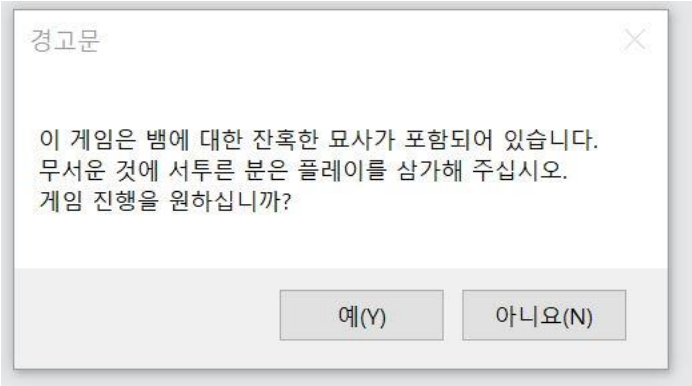

OnKeyDown()
Start
MoveSnake()
CheckState()==False
exit(0)
Invalidate(FALSE)
CDialogEx::OnKeyDown(nChar, nRepCnt, nFlags)
End

E. OnPaint()

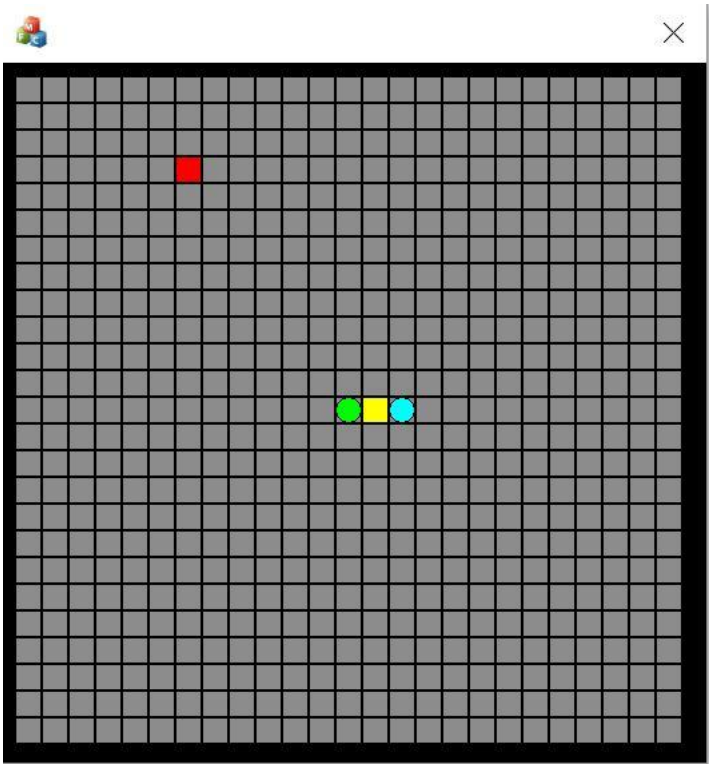


4. Result Screen

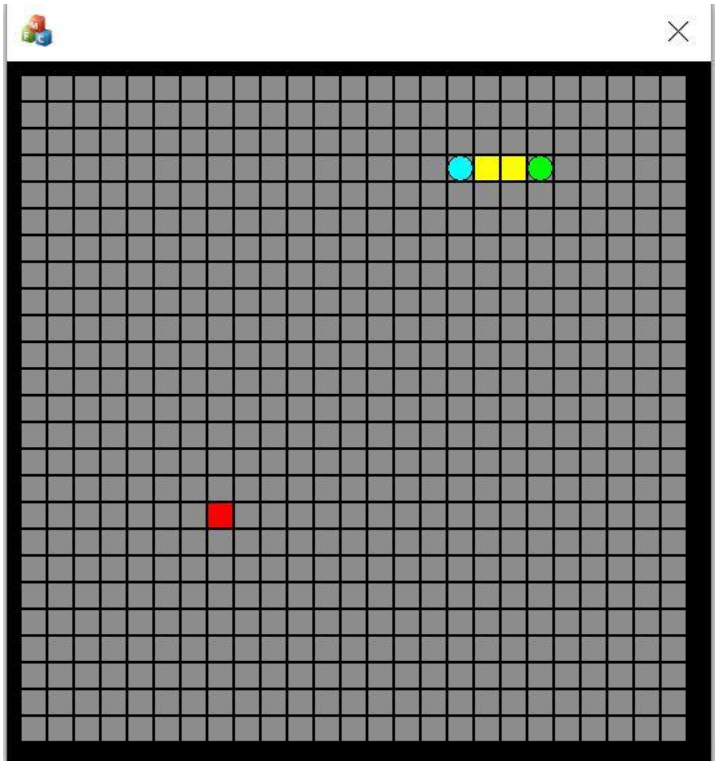
A. Message box displayed when program is executed.



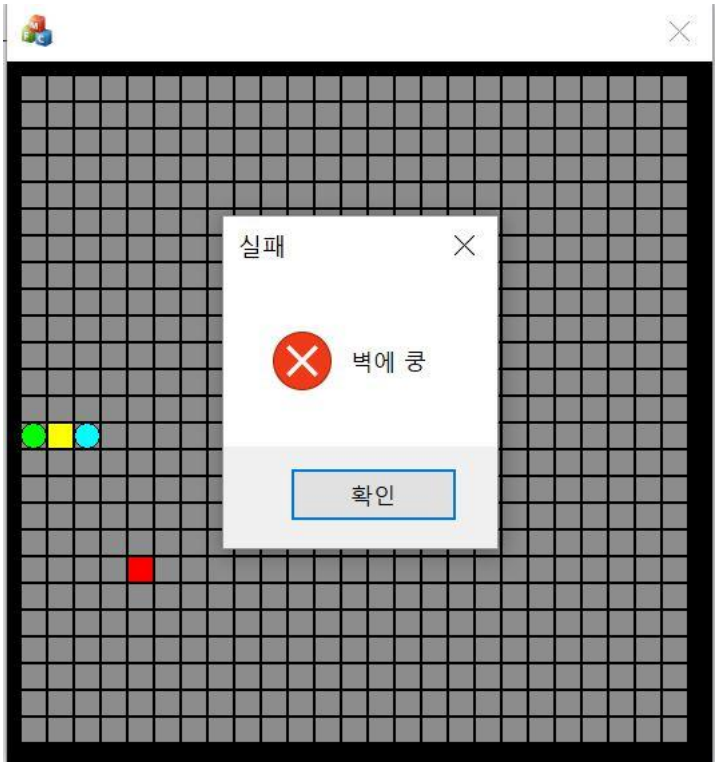
B. The initial screen displayed when the game is executed.



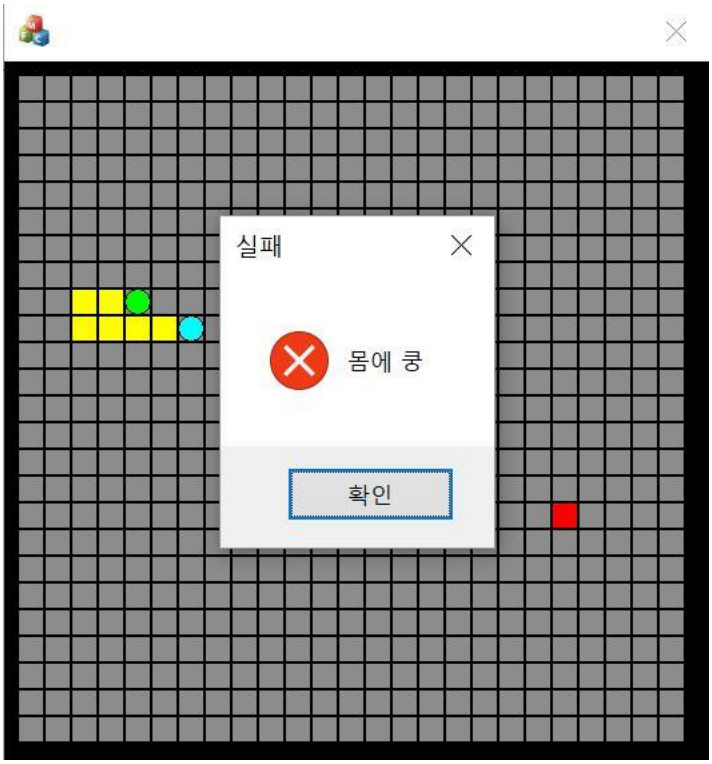
C. The situation after eating the first food.



D. When a snake's head hits a wall.



E. The snake's head hit his body.



5. Review

In Task 3-4, I created a project based on a single document. When I started the project, I created it based on the dialog box. I was able to create a dialog, edit the GUI, and resize the window with commands. In this project, since an empty window is needed, i removed the button and text contorl that is created initially. You can open a separate window to display a message box or draw a shape in the window. When drawing a shape, use the x and y coordinates of the window as arguments. I did not set the reference coordinates before writing the code, so I was having trouble drawing the shape in the coordinates I expected when writing the code. Therefore, it is a good idea for the developer to set the reference coordinates themselves and write the code. In addition, you can use functions such as OnTimer () and ONKeyDown () in the Class Wizard. You can produce various output depending on what arguments or commands are given to the above function. I controlled situations such as SetTimer (), KillTimer (), Invalidate () and OnPaint ().

Both map and snakes are reinitialized with invalidate (). So the screen flicker was severe. I decided that it would affect the play of the game, so I

only created the map once and updated the snake's position. To do this, I added a code to drawsnake () that clears the previous path of the snake.