

Snake *Returns*

Documentation

Lab Batch 33 CS101 Project

Index

1. Introduction
2. Requirements
3. Game Basics
4. Working
5. Member Variables
6. Member Functions
7. Credits

1. Introduction

Snake *Returns* brings back the era of Snakes gaming but with a twist. Now you use a mouse to control your snake through the maze to his fruit. Now you can enjoy bonus surprises. Get addicted.

2. Requirements

1. Ubuntu (preferably \geq ver. 10.04).

2. Good Processing Speed.

3. Game Basics

Snake Returns is nothing but a revamped version of the same old addicting game with new additions and features.

There are 10 levels in all. And each level has 12 fruits (plus some bonus) to offer. To complete each level the user has to eat all the 12 fruits of the level.

Snake Movement

The snake moves on its previous path until the user clicks to change the direction. When the user clicks, the snake turns in the direction of the click. The snake cannot pass through walls nor can it eat its own body. Any of the above event results in Game Over immediately. Hence the user has to manoeuvre the snake to its food.

Fruits

Every time the snake eats up the fruit, a new fruit is plotted on a new co-ordinate, which is determined using a random function. Also a different fruit is generated each time. Eating twelve fruits in a level clears the level.

Levels / Maps

The game kicks off from an easy level and the difficulty increases as the user ascends. Levels will have mazes, different theme and some twists on the higher levels.

Scoring

The scoring is distance based. Based on the distance covered by the user to reach the fruit, he will be awarded the points (maximum of 1000). The total score alongwith the highscore is always displayed on the top. The scoring for the bonus fruit is done similarly.

4. Working

Snake Returns is based on EzWindows API, which comes for C++ programming language. The following paragraphs explain how the game actually works.

As soon the game starts, first all the environment/global variables are initialised. Then all the bitmaps used in the game are loaded followed by loading the level 0 (that is the menu level).

The menu level is a standard menu with options for New Game, High Scores and Exit.

The snake has three parts: head, body and tail. All parts are different blocks represented with different bitmaps.

The Map and Maze Algorithm: Each map has two part: an actual bitmap and a two dimensional array. The array stores values corresponding to the co-ordinates which contain the walls or fruit. This array is loaded from an external file along with the level bitmap image.

Snake Movement Algorithm: The program is basically designed to control the movement of snake's head. The rest of the body part simply follows the previous body part. Also the direction of the snake's movement is controlled by a global variable. Hence whenever the snake is to be turned the globally set direction variable is changed and the snake is turned.

Bonus: The bonus appears twice in a level. One appears after the 4th fruit and one after the 8th fruit. Its stays on the screen for a given time calculated on the basis of the current distance between the fruit and the head of the snake.

Scoring Algorithm: The scoring is based on the distance. It's based on a simple formula:

$$\text{subLevel score} = \frac{\text{leastDistance (between the head the fruit)}}{\text{distanceCount (actual distance covered by the head)}} \times 1000$$

And similarly the bonus score is calculated as:

$$\text{bonusScore} = \frac{\text{leastBonusDistance}}{\text{leastDistanceCount}} \times 5000$$

Check Algorithm: After every snake movement check for snake's head co-ordinates has to be done to assert whether it has eaten the fruit or struck a wall or eaten itself. This is simply done by checking the value in map array at co-ordinates of snake's head. If that's 1 it means it has struck a wall and if it is -1 (and -2 for a bonus fruit) it has eaten a fruit! A separate check is made under an iteration to check that it's not eating its own body.

As the user eats up fruits, the length of the snake increases. This is done by increasing no. of blocks used up to build the snake's body.

5. Member Variables

`int snake[][]` : This is a 15 X 2 matrix which is used to store the co-ordinates of the snake's body.

`int map[][]` : This is a 40 X 28 matrix which is used to store the co-ordinate system of the whole map.

`int hs[]` : This is an array to store the highscores.

`int lvlReached[]` : This is a corresponding array to store the level reached corresponding to a high score.

`int level` : This represents the level of the game on which user plays.

int subLevel : *This represents the number of fruits eaten by snake and hence, determines the length of the snake.*

int Lives : *"Lives" denote the number of tries a player gets during the game (set to 3).*

int delay : *Delay is a time interval which is used to call the function "move" repeatedly after that time interval.*

int xsnake, ysnake : *"xsnake" and "ysnake" are the x-coordinate and y-coordinate of the snake respectively.*

char dir : *"dir" represents the direction of the movement of snake (initially right "R").*

char str[100] : *"str" represents the string which stores the digits of an integer. It will be used to store the scores of the player.*

int xFruit, yFruit, xBonusFruit, yBonusFruit : *"xFruit" and "yFruit" represents the x-coordinate and y-coordinate of the fruit and "xBonusFruit" and "yBonusFruit" are the x-coordinate and y-coordinate of the bonus fruit.*

long distanceCount, bonusDistanceCount, leastDistance, bonusLeastDistance : *"distanceCount" is the distance covered by the snake to eat the fruit. "bonusDistanceCount" is the distance covered by the snake to eat the bonus fruit. "leastDistance" is the minimum distance that a snake covers to eat the fruit. "bonusLeastDistance" is the minimum distance that a snake has to cover to eat the bonus fruit .*

int gameScore : *"gameScore" is the score of the player.*

int levelScore : *"levelScore" is the score of the player in a level.*

int menuMode : *"menuMode" is used to switch between various parts of the main menu.*

bool subLevelChange, bonusFruitPresence : *"subLevelChange" is to change the length of the snake when it eats the fruit. "bonusFruitPresence" is to present the bonus fruit after a certain sublevel.*

Position center : *This is to locate the centre of the window.*

BitMap head : *This is the bitmap image of the head of the snake.*

BitMap body : *This is the bitmap image of the body of the snake.*

BitMap fruit : *This is the bitmap image of the fruit.*

BitMap bonusFruit : *This is the bitmap image of the bonus fruit.*

BitMap wall : *This is the bitmap image of the wall.*

BitMap life : *This is the bitmap image of the lives left to be shown in the score pane.*

BitMap noLife : *This is the bitmap image of the lives lost to be shown in the score pane.*

BitMap background : *This the bitmap image of the background of the main menu.*

BitMap newGame : *This the bitmap image of the "new game" button.*

BitMap highScore : *This the bitmap image of the "high scores" button.*

BitMap help : *This the bitmap image of the "help" button.*

BitMap credit : *This the bitmap image of the "credits" button.*

BitMap ext : *This the bitmap image of the "exit" button.*

BitMap yes : *This the bitmap image of the "yes" button in the exit window.*

BitMap no : *This the bitmap image of the "no" button in the exit window.*

BitMap back : *This the bitmap image of the "back" button.*

BitMap instr : *This the bitmap image of the "instructions" button.*

BitMap credits : *This the bitmap image of the "credits" window.*

BitMap go1 : *This the bitmap image of the "Game Over" title.*

6. Member Functions

`int ApiMain()` : *The main function which loads images of the head, body and tail of the snake and calls for gameplay.*

`int prepLevel()` : *This function is used to plot the designed level on the map.*

`int playGame()` : *This function is used for the basic gameplay of the game.*

`int move()` : *This function is used to move the snake such that the snake's body follows its head.*

`int randomCood(int l)` : *This function is used to generate a pair of random co-ordinates in the map.*

`int loadMap()` : *This function loads the map of the level and assigns a particular value at a point where wall, fruit or snake is present.*

`int plotFruit()` : *This function plots the fruit on the random co-ordinates generated and takes care that it does not lie on the snake or a wall.*

`int mouseClicked(const Position &p)` : *This function is called when mouse click occurs. It turns the snake according to the position of the mouse click.*

`int timerClick()` : *This function is called repeatedly after a particular time interval. It is used to keep the snake moving and check for head crash at each step.*

`int check()` : *This function checks whether the snake collides with the wall and whether it eats the fruit.*

`int plotBonusFruit()` : *This function plots the bonus fruit on the random coordinates generated and takes care that it does not lie on the snake or a wall.*

`int handleBonusFruit()` : *This function is responsible for disappearing the bonus fruit and for its blinking.*

`int addScore()` : *This function is to calculate the score of the player as the game progresses.*

`int scorePane()` : *This function is to display the details of the game like the score, high score, level and lives.*

`int itoa(int)` : *This function is to store the digits of an integer into the string str.*

`int menuClick` : *This function controls the mouse clicks in the menu.*

`int exitDialogClick` : *This function controls the mouse clicks in the exit window.*

`int loadMenu` : *This function loads all the bitmaps of the menu.*

`int mainMenu` : *This function is the main function for opening of the main menu.*

`int startGame` : *This function initializes all the parameters like score, level etc and starts the game*

.

`int exitMenu` : *This function is to load the exit menu.*

`int hsDisplay` : *This function is to display the high scores.*

`int loadHS` : *This function is to load the highscores and level reached from the text file hs.hs.*

`int instruction` : *This function loads the image of the instructions page.*

`int creditLoad` : *This function loads the image of the credits page.*

`int gameOver` : *This function loads the "Game Over" image.*

`int gameOverClick` : *This function controls the mouse clicks in the game over menu.*

`int chkScore` : *This function checks the score of the user, compares it with high scores and arranges it accordingly.*

7. Credits

TEAM 1

KUMAR PALLAV**

AMAN MANGAL

AKSHAY GODARA

PALLI ASHISH

TEAM 2

PULKIT MAHESHWARI*

PRATEEK AGARWAL

KARTIK CHAUDHARY

NITESH MEENA

TEAM 3

VISHNU VARDHAN*

ASHOK KUMAR

GAUTAM SUMU

AMAN BANSAL

HIMANI JAIN

TA

KALPIT DIXIT