Snake Game

A PROJECT REPORT

Submitted in fulfillment

in

DATA STRUCTURES & ALGORITHMS

by

MITARTH JAIN
17BCE0765
SANDESH POKHARKAR
17BCE0768
PRATYAKSH YADAV
17BCE0134
SAI NIKHIL PRATHIPATI
17BCE2323
PAWAN VERMA
17BCE0270
HRUSHIKAR TEJA
17BCI0067

Under the Guidance of

PROF. NAGARAJU M.

School of Computer Science & Engineering

April, 2018

ACKNOWLEDGEMENTS

I sincerely thank Dr.G.Viswanathan - Chancellor, VIT University, for creating an opportunity to use the facilities available at VIT. I also thank Prof.Nagaraju M.- Department of Computer Science, VIT university, for giving us the opportunity to do this project. I also thank the Dean and entire department of Computer Science, School of Computer Science and Engineering, for giving us this opportunity.

TABLE OF CONTENT

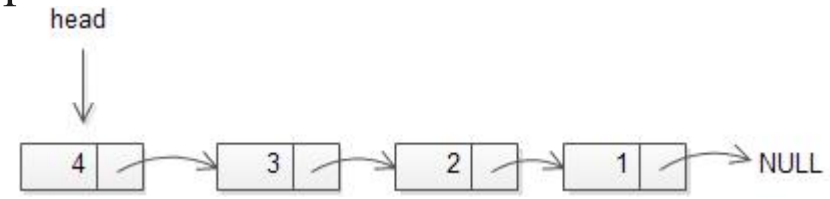
Introduction

The program uses snake in the form of a linked list. So what is a linked list?

A linked list is a data structure that consists of sequence of nodes.

Each node is composed of two fields: data field and reference field which is a pointer that points to the next node in the sequence.

Each node in the list is also called an element. The reference field that contains a pointer which points to the next node is called next pointer or next link.



A head pointer is used to track the first element in the linked list, therefore, it always points to the first element.

The linked list data structure is designed to be efficient for insertion or removal of elements from any position in the list.

A linked list is also used to implement other data structure such as stack and queue.

Abstract:

This project is based on the implementation of Linked lists in our daily life. We all have played that snake game on our mobile phones. We made the same game using Linked list data structure in C language.
Linked lists have the head and tail as their first and last input respectively, similiarly the snake has the head and tail.
So by using this concept we created a linked list of nodes which acts like a moving snake.

Functions:

Many functions have been used in this Snake game project.
Here, I will just list them below and describe the functions “gotoxy”, “GotoXY” and “delay” as they are some of the most important functions used in this project in C.

void record()
void load()
void Delay(long double)
void Move()
void Food()
void Print()
void Bend()
int Score()
void Border()
void Down()
void Left()
void Up()
void Right()
void ExitGame()

void gotoxy (int x, int y)

Void gotoxy (int x, int y):

You need to understand this function as it is one of the most important one used in Snake Game mini project in C. This function allows you to print text in any place of screen.

Void delay(long double) – This function delays the execution.

Void move():

This function commands the snake to move across the game window. Until the direction of the snake head is changed, this "move" function makes it to move in the direction that have been determined by the gamer, before he gives any other instruction. This is done when the move function call the functions 'up', 'down', 'left' and 'right' according to the choice of the gamer.

Void up():

This function uses array-like implementation for the snake to move 'up' when the gamer presses the 'up arrow key', provided initially the snake is moving in left or right direction. This can be seen after each flushing of the screen. If no other direction key is pressed, then the snake moves in the same direction as usual, i.e., up. If the snake touches the boarder, then

it starts again from the bottom of the game screen in the same column and continues to move in the same column and direction again and again until the gamer changes its direction to right or left.

Void down():

This function uses array-like implementation for the snake to move ‘down’ when the gamer presses the ‘down arrow key’, provided initially the snake is moving in left or right direction. This can be seen after each flushing of the screen. If no other direction key is pressed, then the snake moves in the same direction as usual, i.e., down. If the snake touches the boarder, then it starts again from the top of the game screen in the same column and continues to move in the same column and direction again and again until the gamer changes its direction to right or left.

Void right():

This function uses array-like implementation for the snake to move ‘right’ when the gamer presses the ‘right arrow key’, provided initially the snake is moving in up or down direction. This can be seen after each flushing of the screen. If no other direction key is pressed, then the snake moves in the same direction as usual, i.e., right.

If the snake touches the boarder, then it starts again from the left of the game screen in the same row and continues to move in the same row and direction again and again until the gamer changes its direction to up or down.

Void left():

This function uses array-like implementation for the snake to move ‘left’ when the gamer presses the ‘left arrow key’, provided initially the snake is moving in up or down direction. This can be seen after each flushing of the screen.

If no other direction key is pressed, then the snake moves in the same direction as usual, i.e., left. If the snake touches the boarder, then it starts again from the right of the game screen in the same row and continues to move in the same row and direction again and again until the gamer changes its direction to up or down.

Void Food():

In this function the food is randomly generated rand() function. Within the boundaries the food will pop up and the snake will head towards the food, as soon as the snake eats the food the length of snake gets increased by 1.

Insertion of a node

In a linked list when a data is to be inserted the tail node changes its position to tail-1, and the new node gets attached at the tail position.

Similiary here we have used Dynamic memory allocation so when the snake eats the food, first its length increase and a new node gets attached at the tail position.

Deletion of a node

In this game snake has 3 lives so whenever the snake hits the wall or touches its own node(means its head points towards any other node, the linked list will break.) its length will keep decrementing by 1.

So its check the conditions and update the linked list accordingly.

Void Life() & Void Score:

Life and score function:

This function will draw whole screen. which includes,

1.Rectangular boundary limits

2.snake

3.food

Score code:

```
int Score()
{
    int score;
    GotoXY(20,8);
    score=length-5;
    printf("SCORE : %d",(length-5));
   score=length-5;
```

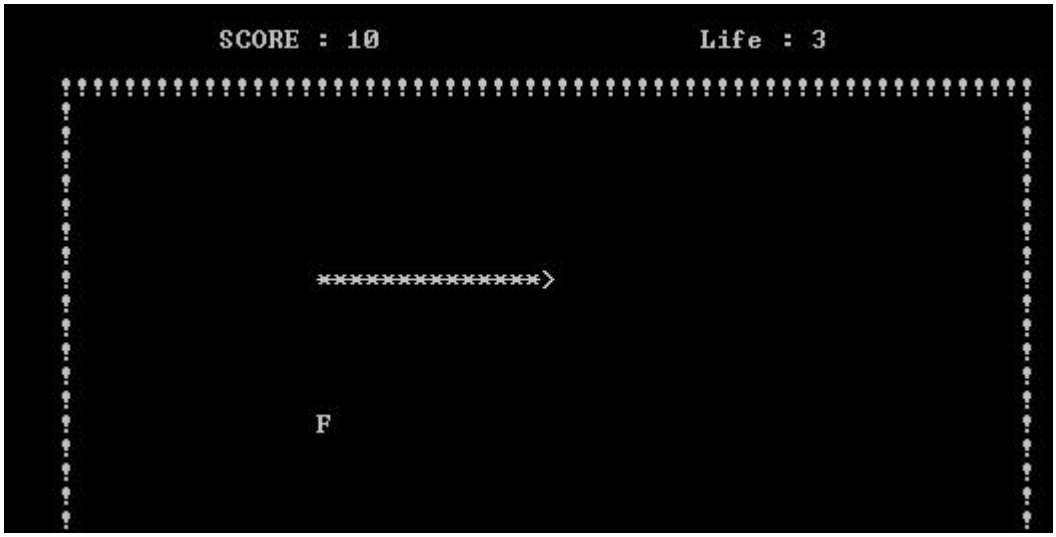
```
GotoXY(50,8);
    printf("Life : %d",life);

int Scoreonly()
{
    int score=Score();
    system("cls");
    return score;
}
```

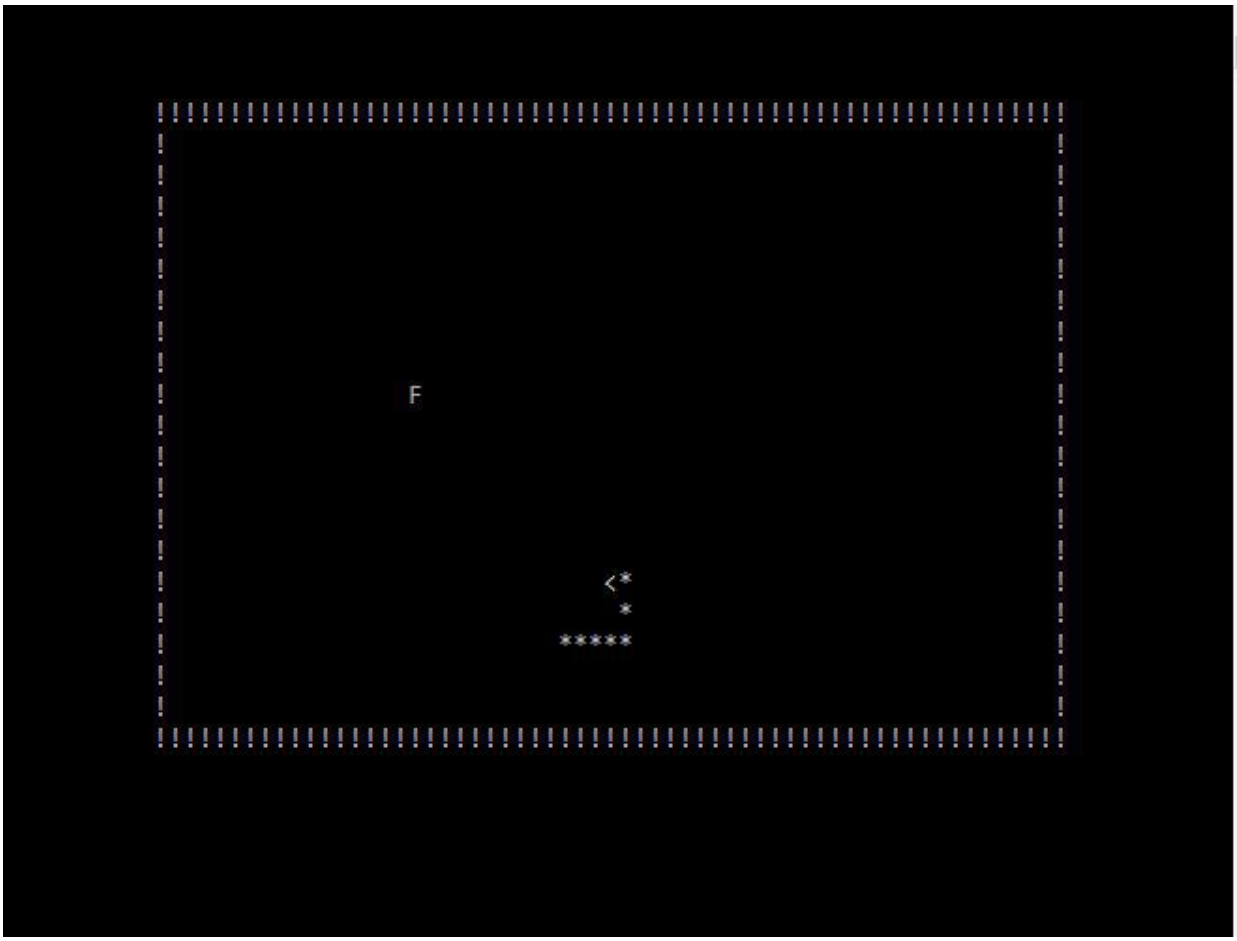
Once you hit an obstacle, that's it, game over.

The aim of the game is to collect the dots (food) and avoid the obstacles (crosses, borders, and the snake itself). As you collect food, the snake gets longer, so increasing your likelihood of crashing into yourself.
When you have collected enough food, you progress onto the next level, where your snake gets longer, and the amount of food to collect to progress through the level gets larger.
The growth of snake is depends on The food it takes.

OUTPUT

```
        Game instructions:

-> Use arrow keys to move the snake.

-> You will be provided foods at the several coordinates of the screen which you have to
eat. Everytime you eat a food the length of the snake will be increased by 1 element and
thus the score.

-> Here you are provided with three lives. Your life will decrease as you hit the wall or
 snake's body.

-> YOu can pause the game in its middle by pressing any key. To continue the paused game
press any other key once again

-> If you want to exit press esc.


Press any key to play game...
```




F
<*
*
*****

```
Player Name :Sandesh
Played Date:Sun Apr 01 14:07:39 2018
Score:8
--------------------------------------------------

Process returned 0 (0x0)   execution time : 157.674 s
Press any key to continue.
```

```
All lives completed
Better Luck Next Time!!!
Press any key to quit the game
```