

Name: MD: Aynul Islam

Chinese Name: 叶子

Student Id: 4420190030

Subject : Data Structure

# Snake Game in C

Contents

[Snake Game in C 1](#_Toc60020793)

[Step 1: Learning Some Basic Syntax for Devolping Gui 2](#_Toc60020794)

[Step 3: Selecting the Level of Game 3](#_Toc60020795)

[Step 4: Algorithm 3](#_Toc60020796)

[Step 5: Algorithm for Increasing Size of Snake and Increasing Score 3](#_Toc60020797)

[Step 6: Game Over Condition: 4](#_Toc60020798)

[Step 7: Quit Page 4](#_Toc60020799)

[concept of Linked list - Graphics in C 4](#_Toc60020800)

[Structures : 4](#_Toc60020801)

[*Functions* **:** 5](#_Toc60020802)

[ "draw" : 5](#_Toc60020803)

[ "gameover": 6](#_Toc60020804)

[Program Screenshot: 8](#_Toc60020805)

[MY GitHub account: 8](#_Toc60020806)

[Reference: 8](#_Toc60020807)

Snake game was popular in old mobile phones which can be very easily devolped using c program. To build this project we require basic understanding of c syntax.

Example: for loop, while loop, etc.

With building this type of game project our programming skill will improve to great extend.

## Step 1: Learning Some Basic Syntax for Devolping Gui

We can add colour and you can print where you want to write

You need to learn basic syntax like:

1)gotoxy(x,y)

2)textcolour()

3)textbackground()

4)kbhit()

using above select proper background and font.

welcome the user by showing game title

## Step 3: Selecting the Level of Game

After displaying the game title.

Do clear screen()

show the levels and ask user to select the level

save the level selected by user in variable and use it to increase the difficulty level of user.

## Step 4: Algorithm

random function:

In c program there is a random function which is used to place the food at any point on the screen.

Thus, use this function to place food at any point on the screen

Moving snake in the screen:

As you can see snake is printed by printing ''\*'' on the screen on the screen. Thus, we need to print the one character at the beginning and erasing one character at the end.

Changing Direction:

The direction of snake can be changed using kbhit() function .

when you press the character accordingly it will change the direction of the snake.

## Step 5: Algorithm for Increasing Size of Snake and Increasing Score

Increasing size of snake:

When snake moves forward it's size increases and when the coordinate of snake match with food coordinate then the size of snake is increased.

Increasing score:

when snanke get's the food the score of the user is also increased

From above image it can be clear that user score and size of snake is increased.

## Step 6: Game Over Condition:

When snake touch the boundary of the screen. The game over will come.

This can be done by comparing the coordinate of boundary with the snake co-ordinate.

## Step 7: Quit Page

Last step ask user if he wants to continue playing or not.

### concept of Linked list - Graphics in C

### Structures :

|  |
| --- |
| **struct** loc  {  **int** x,y;  };  **struct** snake  {  **struct** loc sloc;  **struct** snake \*link;  **char** dir;  };  **struct** game\_data  {  **int** score;  **int** no\_food;  };  **struct** game\_data gd={**0**,**0**};  **struct** limit  {  **int** lx1,ly1,lx2,ly2;  };  **struct** limit l={**96**,**96**,**404**,**404**};  **struct** food  {  **struct** loc floc;  }; |

* Structure "loc" have 2 integer variables to save x and y coordinates.
* Structure "snake" is basically a linked list , new node of this structure will be created when snake will increase in length.
* "game\_data" will store score and number of food consumed by snake
* "limit" defines the boundary of rectangle in which snake can move. crossing this limits will result in game over.
* "food" contains coordinates of food to be displayed

### Functions **:**

###  "draw" :

This function will draw whole screen. which includes,

1. Rectangular Boundary limits
2. snake
3. food

|  |
| --- |
| 1. draw(**struct** snake \*head,**struct** food \*f) 2. { 3. **struct** snake \*temp; 4. temp=head; 5. rectangle(**96**,**96**,**404**,**404**); 6. rectangle(**98**,**98**,**402**,**402**); 7. rectangle(**100**,**100**,**400**,**400**); 8. setfillstyle(**9**,**13**); 9. bar(temp->sloc.x-**6**,temp->sloc.y-**6**,temp->sloc.x+**6**,temp->sloc.y+**6**); 10. temp=temp->link; 11. setfillstyle(**9**,**2**); 12. **while**(temp->link!=NULL) 13. { 14. bar(temp->sloc.x-**5**,temp->sloc.y-**5**,temp->sloc.x+**5**,temp->sloc.y+**5**); 15. temp=temp->link; 16. } 17. bar(temp->sloc.x-**5**,temp->sloc.y-**5**,temp->sloc.x+**5**,temp->sloc.y+**5**); 18. circle(f->floc.x-**2**,f->floc.y-**2**,**5**); 19. circle(f->floc.x+**2**,f->floc.y+**2**,**5**); 20. circle(f->floc.x-**2**,f->floc.y+**2**,**5**); 21. circle(f->floc.x+**2**,f->floc.y-**2**,**5**); 22. delay(**20**); 23. **while**(!kbhit()){**goto** e;} 24. key=getche(); 25. **e:** 26. cleardevice(); 27. } |

###  "gameover":

This function will draw screen when game is over, it includes score and number of food consumed.

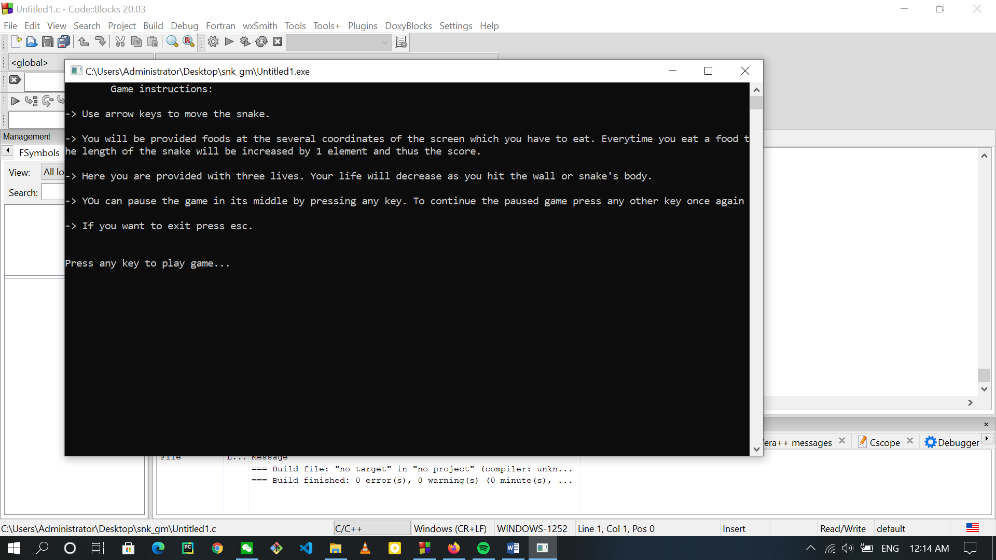
|  |
| --- |
| gameover(**void**)  {  cleardevice();  outtextxy(**100**,**100**,"----------------- GAME OVER ----------------------");  printf("**\n\n\n\n\n\n\n\n**");  printf("**\t\t**# Score - %d",gd.score);  printf("**\n\t\t**# No of food - %d", gd.no\_food);  **s1:**  sound(**300**);delay(**300**);sound(**450**);delay(**150**);sound(**500**);delay(**150**);  sound(**300**);delay(**200**);sound(**450**);delay(**100**);sound(**450**);delay(**200**);  **while**(!kbhit()){**goto** s1;}  nosound();  exit(**0**);  } |

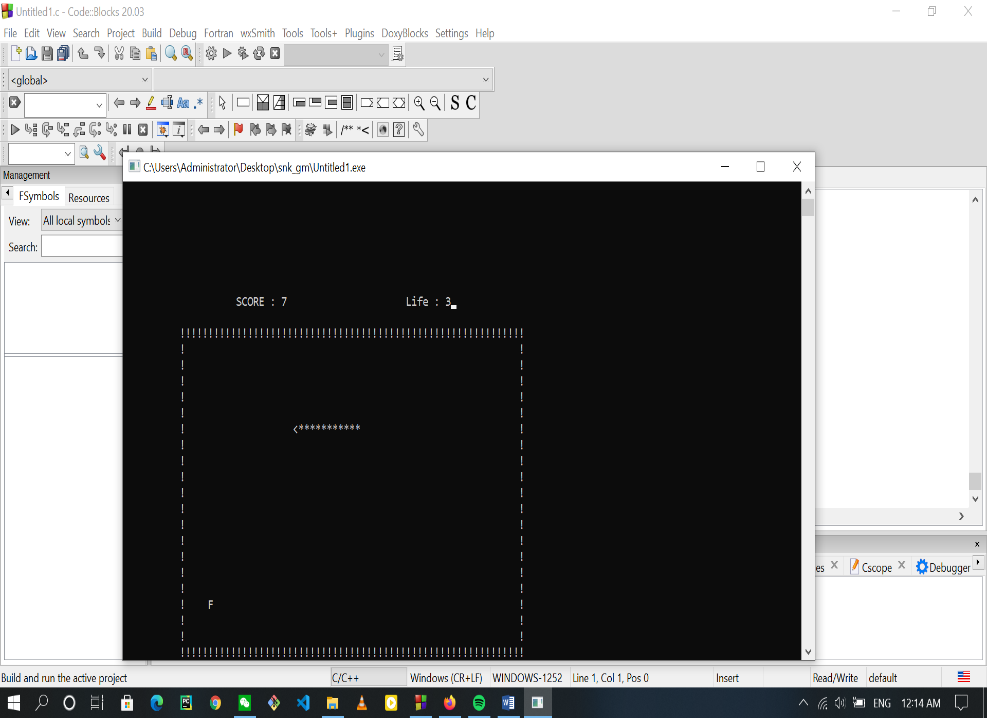
#### "game" :

* This is main logic of our snake game. It does following tasks in sequence within a while loop.  
  (while loop checks whether 'p' (for pause) is pressed or not.)
  1. checking condition whether snake hits boundary. if it hits boundary then call "gameover" function
  2. check whether snake consumes food or not.  
     if it consumes food then generate new food at new location, increase score and food count.  
     and create new node of snake structure to increase the length of snake.
  3. check whether any key is pressed to change direction of snake. according to that set the direction of snakes head.
  4. update direction and location of all node of snake body.

|  |
| --- |
| * **void** **game**(**struct** snake \*head,**struct** food \*f) * { * **struct** snake \*temp,pre,nxt; * temp=head; * **while**(key!='p') * { * **if**(head->sloc.x==l.lx1||head->sloc.x==l.ly2||head->sloc.y==l.ly1||head->sloc.y==l.ly2) * {gameover();} * **if**(head->sloc.x>=f->floc.x-**5**&&head->sloc.x<=f->floc.x+**5**&&head->sloc.y>=f->floc.y-**5**&&head->sloc.y<=f->floc.y+**5**) * { * temp=head; * sound(**420**); * f->floc.x=**150**+random(**245**); * f->floc.y=**150**+random(**245**); * gd.score+=**100**; * gd.no\_food+=**1**; * n=n+**1**; * **while**(temp->link!=NULL){temp=temp->link;} * temp->link=(**struct** snake \*)malloc(**sizeof**(**struct** snake)); * temp->link->link=NULL; * temp->link->sloc.x= temp->sloc.x; * temp->link->sloc.y= temp->sloc.y; * temp->link->dir=temp->dir; * n=**0**; * } * **switch**(key) //this key varible is set in draw() function * { * **case** 'a': **if**(head->dir!='d'){head->dir='a'; head->sloc.x-=**2**; } **else** {key=head->dir;} **break**; * **case** 'w': **if**(head->dir!='s'){head->dir='w'; head->sloc.y-=**2**; } **else** {key=head->dir;} **break**; * **case** 'd': **if**(head->dir!='a'){head->dir='d'; head->sloc.x+=**2**; } **else** {key=head->dir;} **break**; * **case** 's': **if**(head->dir!='w'){head->dir='s'; head->sloc.y+=**2**; } **else** {key=head->dir;} **break**; * } * draw(head,f); * nosound(); * temp=head; * pre=\*temp; * **while**(temp->link!=NULL) * { * nxt.sloc.x=temp->link->sloc.x; * nxt.sloc.y=temp->link->sloc.y; * nxt.dir=temp->link->dir; * temp->link->sloc.x=pre.sloc.x; * temp->link->sloc.y=pre.sloc.y; * temp->link->dir=pre.dir; * temp=temp->link; * pre=nxt; * } * } * } |

## Program Screenshot:





## MY GitHub account:

For code and all documentation please visit my GitHub pages. The link is [here](https://github.com/Pollob001).

For My Snake Game Source Code please visit [here](https://github.com/Pollob001/Snake_Game_in_c).

## Reference:

For data structure click this link…..[Data Structure](:%20https:/lti-adx.adelaide.edu.au/data/snake/)

For understanding more information …….[For more update](https://stackoverflow.com/questions/1945345/what-data-structure-should-i-use-for-a-snake-game)

For source code highlight….click [here](http://hilite.me/).