Mini Project Report

on

Snake Game

Submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY

DEGREE

Session 2021-22

In

Information Technology

By

Azam Ali

2000320130051

Coordinated By: Mr. Jitendra Chauhan
(Assistant Professor)

DEPARTMENT OF INFORMATION TECHNOLOGY
ABES ENGINEERING COLLEGE, GHAZIABAD

AFFILIATED TO
DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY, U.P., LUCKNOW
(Formerly UPTU)

Student's Declaration

I hereby declare that the work being presented in this report entitled "Snake Game" is an authentic record of my own work carried out under the supervision of **Mr. Jitendra Chauhan, Assistant Professor, Information Technology.**

The matter embodied in this report has not been submitted by me for the award of any other degree.

Date:

Name: AZAM ALI

Roll No : 2000320130051

Department: Information Technology

This is to certify that the above statement made by the candidate(s) is correct to the best of my knowledge.

Signature of HOD :	**Signature of Coordinator :**
Prof. Amit Sinha)	**(Name: Mr. Jitendra Chauhan)**
(Information Technology)	**(Assistant Professor)**
Date: ……..	**(Information Technology)**

ACKNOWLEDGEMENT

Presentation inspiration and motivation have always played a key role in the success of any venture.

I express my sincere thanks to my project coordinator

Mr. Jitendra Chauhan to encourage me to the highest peak and to provide me this opportunity to prepare this project. I extend my heartly thanks for giving me the proper guidance even in this time, when everything is continuing on just online platforms.

I am highly indebted to **Mr. Jitendra Chauhan** for the constant supervision, for providing all the necessary information and support in completing the project.

Finally, I am sincerely thankful to all those people who are directly or indirectly involved in the successful completion of this project work.

Signature of student

Name: Azam Ali

Roll No: 2000320130051

TABLE OF CONTENTS

INTRODUCTION

Snake Game is a video game genre where the player maneuvers a growing line that becomes a primary obstacle to itself. The concept originated in the 1976 two-player arcade game Blockade from Gremlin Industries, and the ease of implementation has led to hundreds of versions (some of which have the word snake or worm in the title) for many platforms. 1982's Tron arcade game, based on the film, includes snake gameplay for the single-player Light Cycles segment. After a variant was preloaded on Nokia mobile phones in 1998, there was a resurgence of interest in snake games as it found a larger audience. There are several hundred snake games for iOS alone.

The game called "Snake" or "**Snake Game**" typically involve the player controlling a line or snake, there is no official version of the game, so gameplay varies. The most common version of the game involves the snake or line eating items which make it longer, with the objective being to avoid running into a border or the snake itself for as long as possible.

The player loses when the snake either runs into a border or its own body. Because of this, the game becomes more difficult as it goes on, due to the growth of the snak**e**

PPROJECT OBJECTIVE

Objective of this project is as follows: -

This game aims to change the way people think of traditional snake game. It will offer the

experience of commercial multilayer games to the player retaining the simplicity of **traditional snake game**

> Create a snake game that will have all the functionality of traditional

snake games.

> Introduce multilayer functionality in the game that will allow several players to play a game simultaneously. It should be able to give the experience of a real time multiplayer game to the players.

> Introduce computer controlled intelligent opponent (unique feature of this game) to make the game more challenging and interesting. The movement and action of these intelligent opponents will be controlled by computer whose aim will be to eat the food before human players capture it

PROJECT METHODOLOGY

Snake is a multiplayer version of traditional snake game (popular among cell phone gamers) with computer controlled intelligent opponents that challenges the human players.The player who hosts the game server is called "local player". The Snake Client Application and Game Server run in separate execution domains. The Snake Client App. for local player communicates with the game server through network layer, just like other remote players, as shown in illustration 1

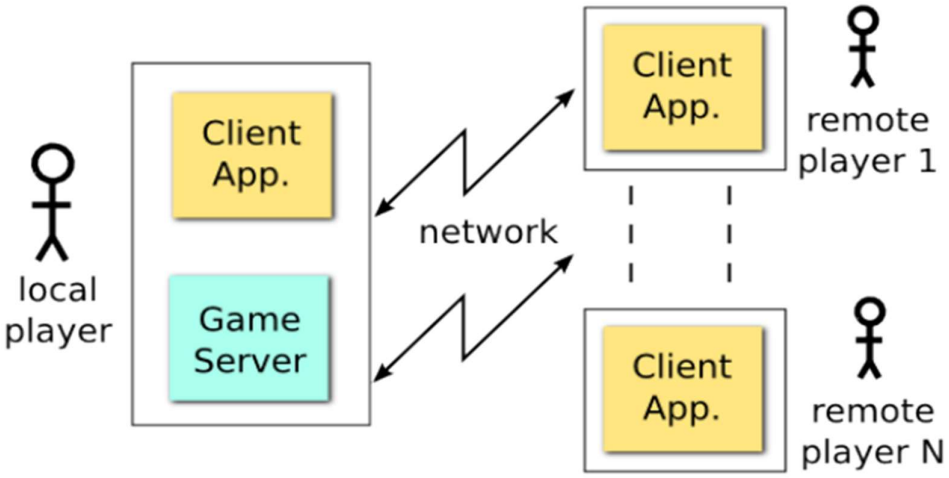


illustration 1

Software requirements

- Compiler: CodeBlock
- Operating System: Windows xp, 7, 8 or 10 …
- RAM: 512 MB to 2 GB.
- Processor: Dual Core to Core i7.
- Hard Disk Usage: 15 to 25MB.

Hardware requirements

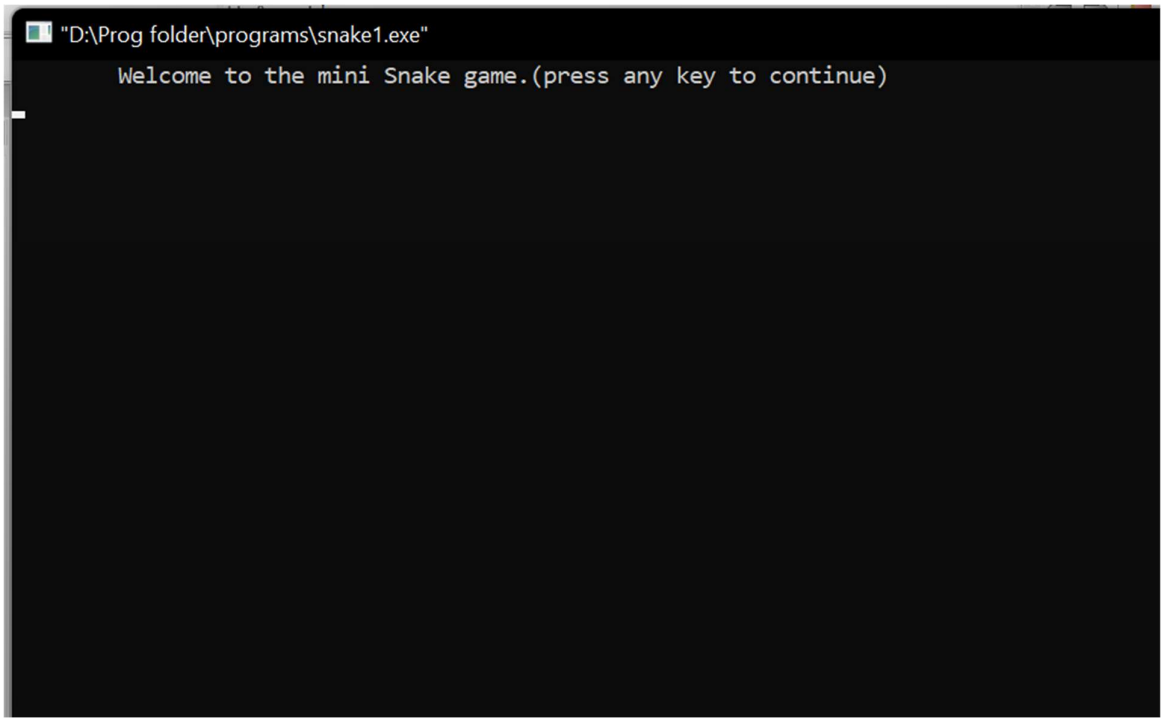
- High resolution screen.
- Mouse.
- Keyboard.

DETAILS OF THE PROJECT

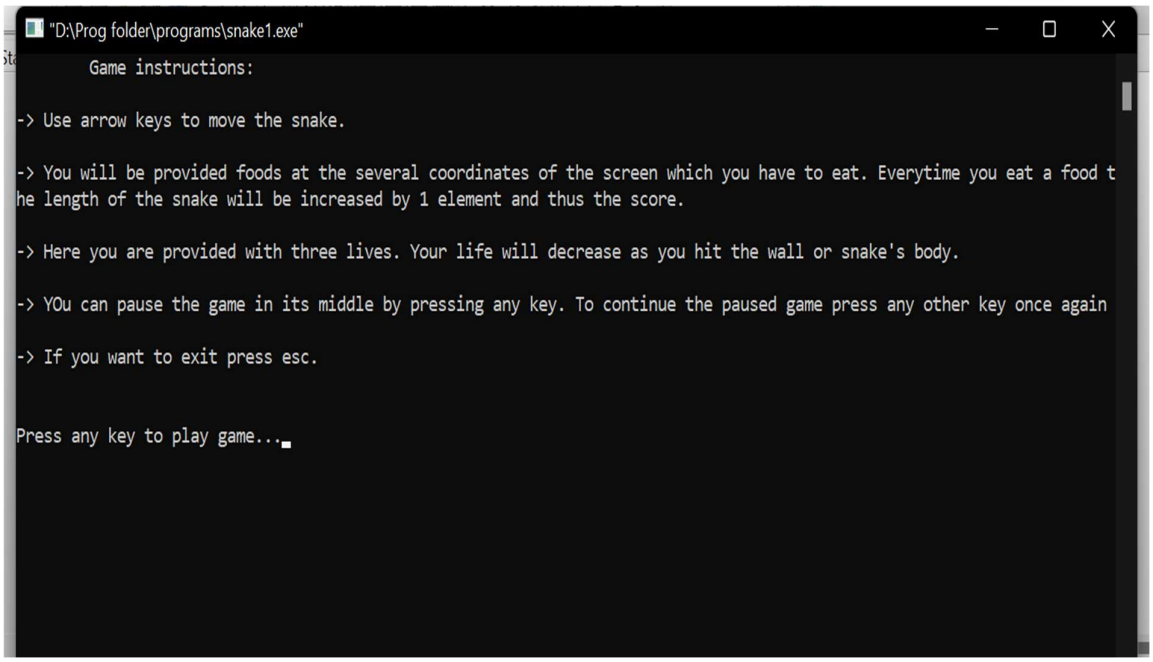
OUTPUT SCREEN

A. Quantitative Analytical explanation of the theoretical modeling

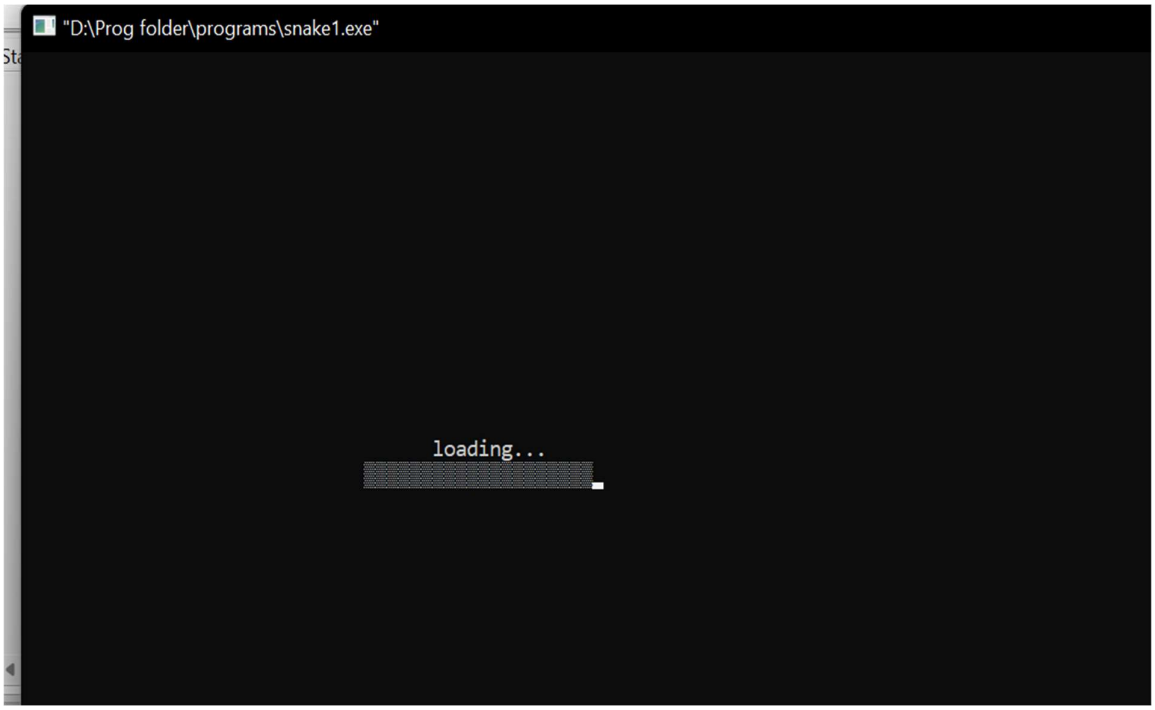
Login Page



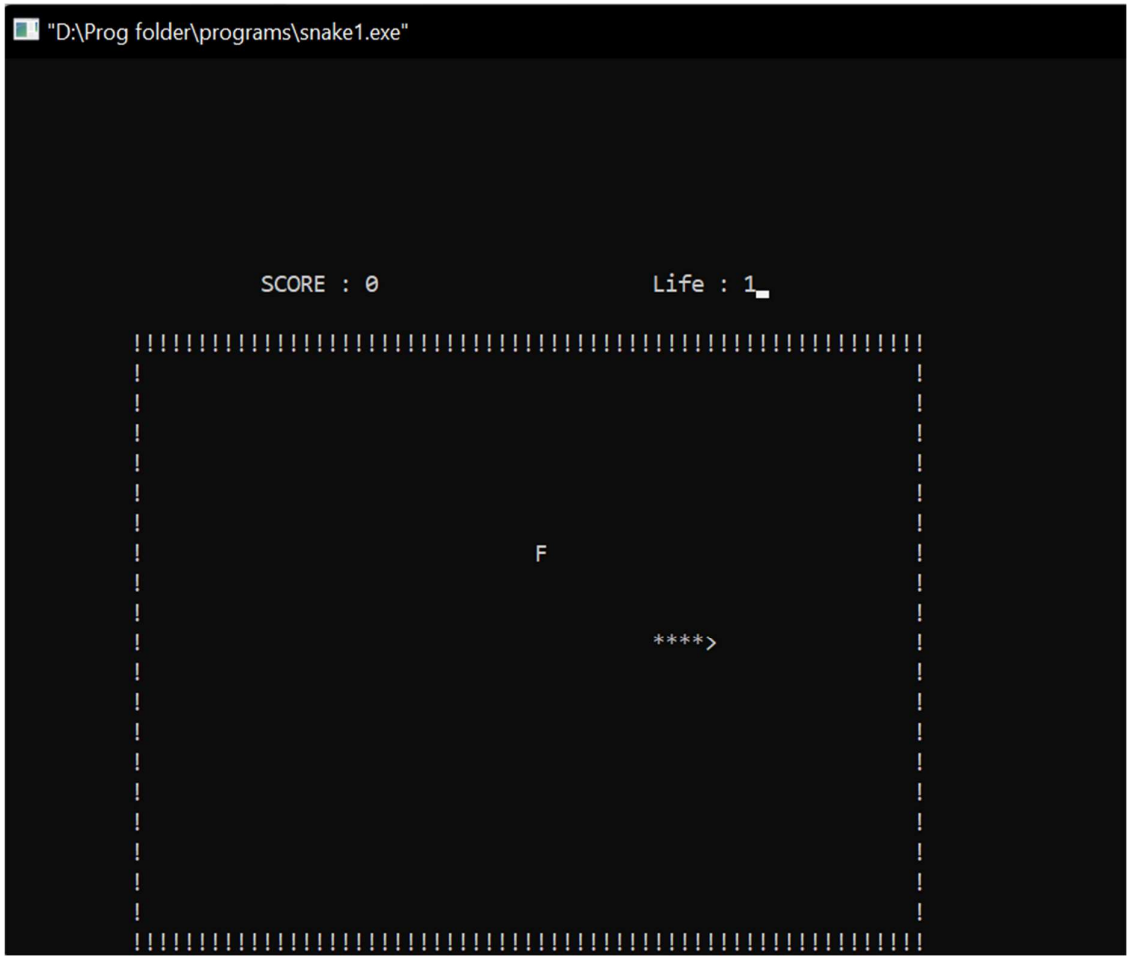
Game instruction:



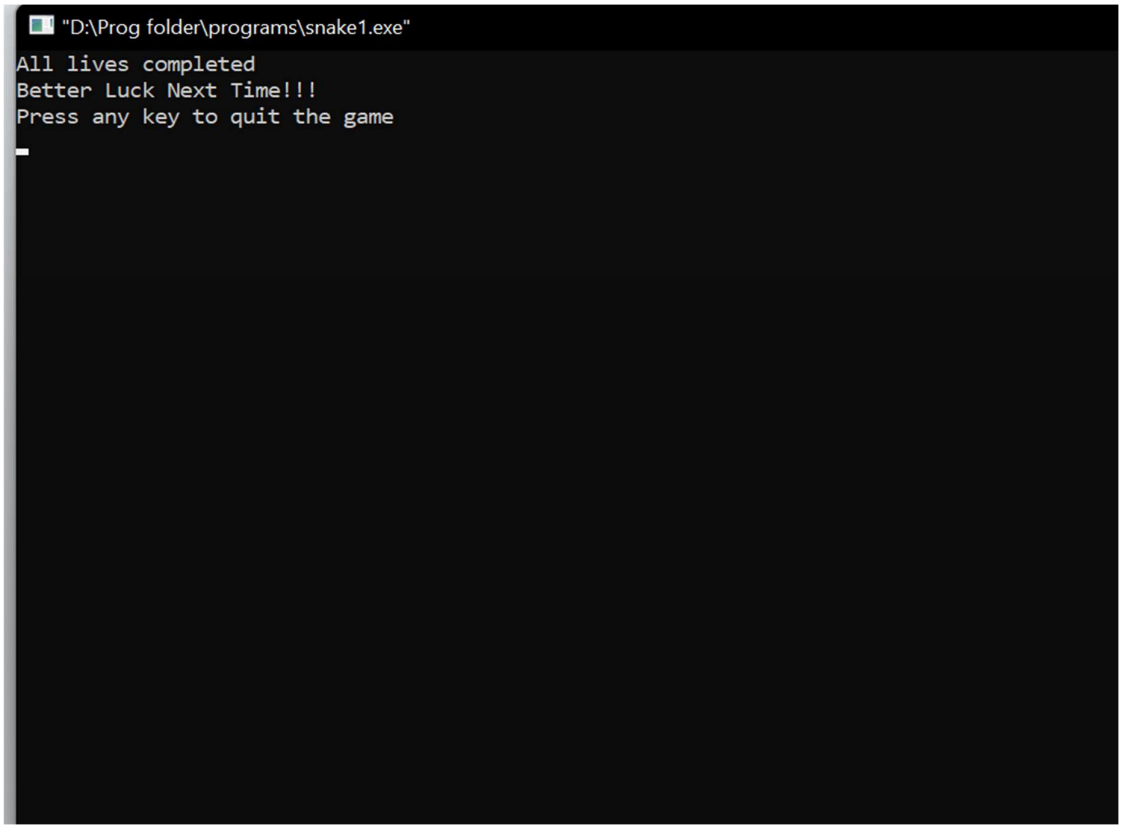
Loading page:



Welcome to Game Page:



Exit Page/All level competed:



CONCLUSION & FUTURE DEVELOPMENT

Conclusion:

The following conclusion can be drawn from my application:

>We were successful in creating a multiplayer version of traditional snake game. The computer controlled intelligent opponents have been successfully tested in the game is a unique feature of Snake.

>We learned several project management techniques used by professionals to develop large scale project. The experience of working in team and integration of modules developed independently, with just requirement specifications, is a very important achievement for the Snake team.

Future Development:

>Port Snake to cell phone platform and One Laptop Per Child – OLPC (which uses Sugar Desktop environment). The presence of several connectivity options(Bluetooth, WIFI, GPRS, CDMA) in cell phones makes it a very attractive platform for a multiplayer game like Snake. Local WIFI network formed by kids using OLPC laptops can be used as a platform for Snake's deployment.

>As Snake game server communicates with remote playing using a well defined and very simple protocol (Refer to ANNEX- D), Snake clients programmed in other programming platform like Flash, Python, etc can be developed

Appendix-1

Coding

Importing various libraries so as to connect with **C language** and check for login credentials.

```c
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <conio.h>
#include<time.h>
#include<ctype.h>
#include <time.h>
#include <windows.h>
#include <process.h>

#define UP 72
#define DOWN 80
#define LEFT 75
#define RIGHT 77

int length;
int bend_no;
int len;
char key;
void record();
void load();
int life;
void Delay(long double);
void Move();
void Food();
```

```c
void Food();
int Score();
void Print();
void gotoxy(int x, int y);
void GotoXY(int x,int y);
void Bend();
void Boarder();
void Down();
void Left();
void Up();
void Right();
void ExitGame();
int Scoreonly();
```

Coding

Start here | snake1.c

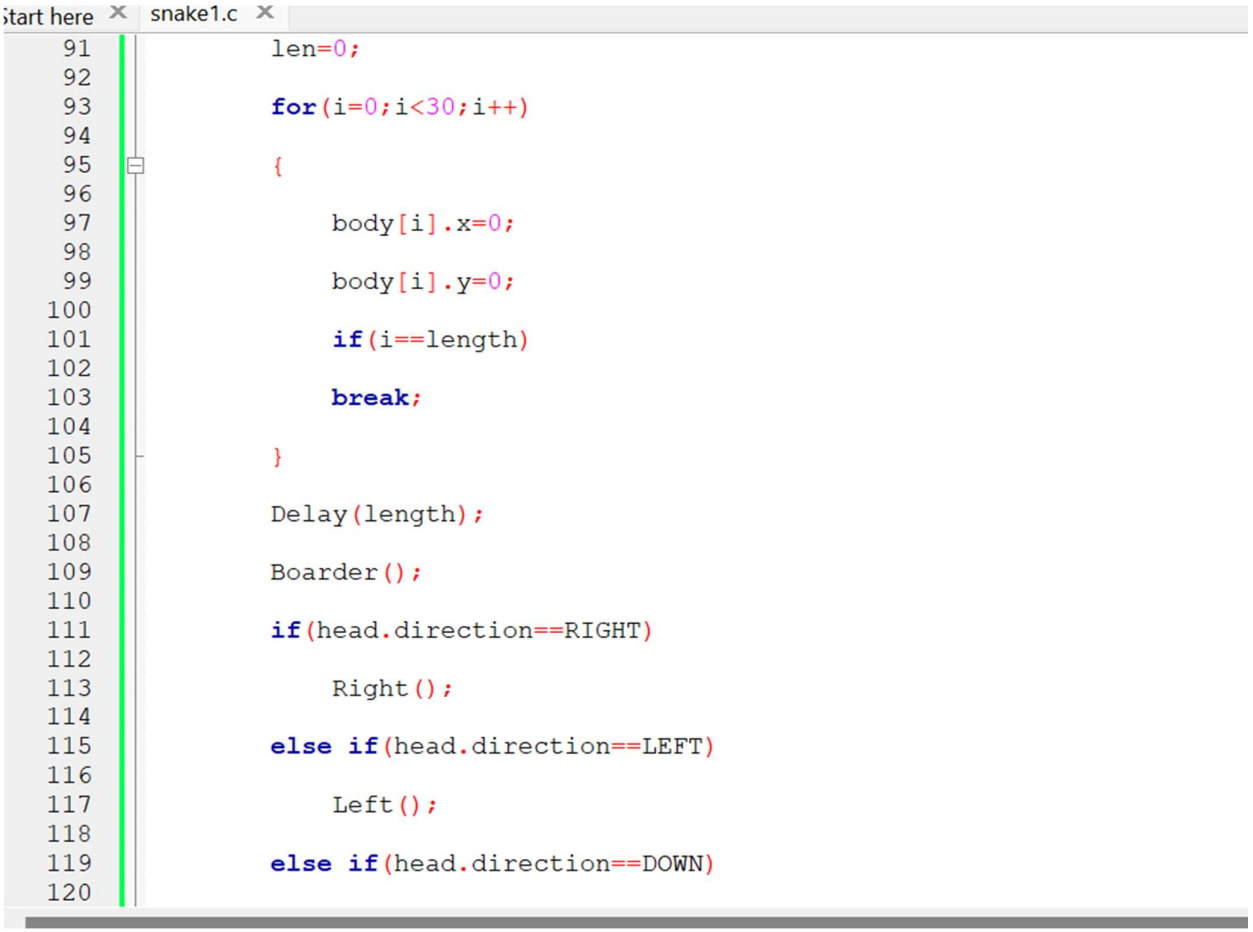
```c
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <conio.h>
#include<time.h>
#include<ctype.h>
#include <time.h>
#include <windows.h>
#include <process.h>

#define UP 72
#define DOWN 80
#define LEFT 75
#define RIGHT 77

int length;
int bend_no;
int len;
char key;
void record();
void load();
int life;
void Delay(long double);
void Move();
void Food();
int Score();
void Print();
void gotoxy(int x, int y);
void GotoXY(int x,int y);
void Bend();
```

Start here | snake1.c

```
31  void Boarder();
32  void Down();
33  void Left();
34  void Up();
35  void Right();
36  void ExitGame();
37  int Scoreonly();
38
39  struct coordinate{
40      int x;
41      int y;
42      int direction;
43  };
44
45  typedef struct coordinate coordinate;
46
47  coordinate head, bend[500],food,body[30];
48
49  int main()
50  {
51
52      char key;
53
54      Print();
55
56      system("cls");
57
58      load();
59
60      length=5;
```

Start here | snake1.c

```
61
62      head.x=25;
63
64      head.y=20;
65
66      head.direction=RIGHT;
67
68      Boarder();
69
70      Food(); //to generate food coordinates initially
71
72      life=3; //number of extra lives
73
74      bend[0]=head;
75
76      Move();   //initialing initial bend coordinate
77
78      return 0;
79
80  }
81
82  void Move()
83  {
84      int a,i;
85
86      do{
87
88          Food();
89          fflush(stdin);
90
```


start here
snake1.c
91 len=0;
92
93 for(i=0;i<30;i++)
94
95 {
96
97 body[i].x=0;
98
99 body[i].y=0;
100
101 if(i==length)
102
103 break;
104
105 }
106
107 Delay(length);
108
109 Boarder();
110
111 if(head.direction==RIGHT)
112
113 Right();
114
115 else if(head.direction==LEFT)
116
117 Left();
118
119 else if(head.direction==DOWN)
120

Start here X *snake1.c X

```
121             Down();
122
123         else if(head.direction==UP)
124
125             Up();
126
127         ExitGame();
128
129     }while(!kbhit());
130
131     a=getch();
132
133     if(a==27)
134
135     {
136
137         system("cls");
138
139         exit(0);
140
141     }
142     key=getch();
143
144     if((key==RIGHT&&head.direction!=LEFT&&head.direction!=RIGHT)||(key==LEFT&&head.direction!=RIGHT&&head.direction!=LEFT)
145        ||(key==UP&&head.direction!=DOWN&&head.direction!=UP)||(key==DOWN&&head.direction!=UP&&head.direction!=DOWN))
146
147     {
148
149         bend_no++;
150
```

Start here | *snake1.c

```
151             bend[bend_no]=head;
152
153             head.direction=key;
154
155             if(key==UP)
156
157                 head.y--;
158
159             if(key==DOWN)
160
161                 head.y++;
162
163             if(key==RIGHT)
164
165                 head.x++;
166
167             if(key==LEFT)
168
169                 head.x--;
170
171             Move();
172
173         }
174
175         else if(key==27)
176
177         {
178
179             system("cls");
180
```

Start here | *snake1.c

```
181             exit(0);
182
183         }
184
185         else
186
187         {
188
189             printf("\a");
190
191             Move();
192
193         }
194     }
195
196     void gotoxy(int x, int y)
197     {
198
199      COORD coord;
200
201      coord.X = x;
202
203      coord.Y = y;
204
205      SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coord);
206
207     }
208     void GotoXY(int x, int y)
209     {
210         HANDLE a;
```

```
211         COORD b;
212         fflush(stdout);
213         b.X = x;
214         b.Y = y;
215         a = GetStdHandle(STD_OUTPUT_HANDLE);
216         SetConsoleCursorPosition(a,b);
217     }
218 void load(){
219         int row,col,r,c,q;
220         gotoxy(36,14);
221         printf("loading...");
222         gotoxy(30,15);
223         for(r=1;r<=20;r++){
224         for(q=0;q<=100000000;q++);//to display the character slowly
225         prir int load::q ;}
226         getch();
227 }
228  void Down()
229 {
230         int i;
231         for(i=0;i<=(head.y-bend[bend_no].y)&&len<length;i++)
232         {
233             GotoXY(head.x,head.y-i);
234             {
235                 if(len==0)
236                     printf("v");
237                 else
238                     printf("*");
239             }
240             body[len].x=head.x;
```

Start here X *snake1.c X

```
241         body[len].y=head.y-i;
242         len++;
243     }S
244     Bend();
245     if(!kbhit())
246         head.y++;
247 }
248 void Delay(long double k)
249 {
250     Score();
251     long double i;
252     for(i=0;i<=(10000000);i++);
253 }
254 void ExitGame()
255 {
256     int i,check=0;
257     for(i=4;i<length;i++)   //starts with 4 because it needs minimum 4 element to touch its own body
258     {
259         if(body[0].x==body[i].x&&body[0].y==body[i].y)
260         {
261             check++;    //check's value increases as the coordinates of head is equal to any other body coordinate
262         }
263         if(i==length||check!=0)
264             break;
265     }
266     if(head.x<=10||head.x>=70||head.y<=10||head.y>=30||check!=0)
267     {
268         life--;
269         if(life>=0)
270         {
```

Start here X *snake1.c X

```
271             head.x=25;
272             head.y=20;
273             bend_no=0;
274             head.direction=RIGHT;
275             Move();
276         }
277         else
278         {
279             system("cls");
280             printf("All lives completed\nBetter Luck Next Time!!!\nPress any key to quit the game\n");
281             record();
282             exit(0);
283         }
284     }
285 }
286 void Food()
287 {
288     if(head.x==food.x&&head.y==food.y)
289     {
290         length++;
291         time_t a;
292         a=time(0);
293         srand(a);
294         food.x=rand()%70;
295         if(food.x<=10)
296             food.x+=11;
297         food.y=rand()%30;
298         if(food.y<=10)
299
300             food.y+=11;
```

Start here X *snake1.c X

```
301     }
302     else if(food.x==0)/*to create food for the first time coz global variable are initialized with 0*/
303     {
304         food.x=rand()%70;
305         if(food.x<=10)
306             food.x+=11;
307         food.y=rand()%30;
308         if(food.y<=10)
309             food.y+=11;
310     }
311 }
312 void Left()
313 {
314     int i;
315     for(i=0;i<=(bend[bend_no].x-head.x)&&len<length;i++)
316     {
317         GotoXY((head.x+i),head.y);
318        {
319                 if(len==0)
320                     printf("<");
321                 else
322                     printf("*");
323         }
324         body[len].x=head.x+i;
325         body[len].y=head.y;
326         len++;
327     }
328     Bend();
329     if(!kbhit())
330         head.x--;
```

Start here | *snake1.c

```
331
332 }
333 void Right()
334 {
335     int i;
336     for(i=0;i<=(head.x-bend[bend_no].x)&&len<length;i++)
337     {
338         //GotoXY((head.x-i),head.y);
339         body[len].x=head.x-i;
340         body[len].y=head.y;
341         GotoXY(body[len].x,body[len].y);
342         {
343             if(len==0)
344                 printf(">");
345             else
346                 printf("*");
347         }
348         /*body[len].x=head.x-i;
349         body[len].y=head.y;*/
350         len++;
351     }
352     Bend();
353     if(!kbhit())
354         head.x++;
355 }
356 void Bend()
357 {
358     int i,j,diff;
359     for(i=bend_no;i>=0&&len<length;i--)
360     {
```

Start here | *snake1.c

```
361             if(bend[i].x==bend[i-1].x)
362             {
363                 diff=bend[i].y-bend[i-1].y;
364                 if(diff<0)
365                     for(j=1;j<=(-diff);j++)
366                     {
367                         body[len].x=bend[i].x;
368                         body[len].y=bend[i].y+j;
369                         GotoXY(body[len].x,body[len].y);
370                         printf("*");
371                         len++;
372                         if(len==length)
373                             break;
374                     }
375                 else if(diff>0)
376                     for(j=1;j<=diff;j++)
377                     {
378                         /*GotoXY(bend[i].x,(bend[i].y-j));
379                         printf("*");*/
380                         body[len].x=bend[i].x;
381                         body[len].y=bend[i].y-j;
382                         GotoXY(body[len].x,body[len].y);
383                         printf("*");
384                         len++;
385                         if(len==length)
386                             break;
387                     }
388             }
389         else if(bend[i].y==bend[i-1].y)
390         {
```

Start here X *snake1.c X

```
391             diff=bend[i].x-bend[i-1].x;
392             if(diff<0)
393                 for(j=1;j<=(-diff)&&len<length;j++)
394                 {
395                     /*GotoXY((bend[i].x+j),bend[i].y);
396                     printf("*");*/
397                     body[len].x=bend[i].x+j;
398                     body[len].y=bend[i].y;
399                     GotoXY(body[len].x,body[len].y);
400                         printf("*");
401                     len++;
402                     if(len==length)
403                             break;
404                 }
405             else if(diff>0)
406                 for(j=1;j<=diff&&len<length;j++)
407                 {
408                     /*GotoXY((bend[i].x-j),bend[i].y);
409                     printf("*");*/
410                     body[len].x=bend[i].x-j;
411                     body[len].y=bend[i].y;
412                     GotoXY(body[len].x,body[len].y);
413                         printf("*");
414                     len++;
415                     if(len==length)
416                         break;
417                 }
418         }
419     }
420 }
```

```
void Boarder()
{
    system("cls");
    int i;
    GotoXY(food.x,food.y);   /*displaying food*/
        printf("F");
    for(i=10;i<71;i++)
    {
        GotoXY(i,10);
            printf("!");
        GotoXY(i,30);
            printf("!");
    }
    for(i=10;i<31;i++)
    {
        GotoXY(10,i);
            printf("!");
        GotoXY(70,i);
        printf("!");
    }
}
void Print()
{
    //GotoXY(10,12);
    printf("\tWelcome to the mini Snake game.(press any key to continue)\n");
   getch();
    system("cls");
    printf("\tGame instructions:\n");
    printf("\n-> Use arrow keys to move the snake.\n\n-> You will be provided foods at the several coordinates of the screen
    printf("\n\nPress any key to play game...");
```

```
451     if(getch()==27)
452     exit(0);
453 }
454 void record(){
455     char plname[20],nplname[20],cha,c;
456     int i,j,px;
457     FILE *info;
458     info=fopen("record.txt","a+");
459     getch();
460     system("cls");
461     printf("Enter your name\n");
462     scanf("%[^\n]",plname);
463     //************************
464     for(j=0;plname[j]!='\0';j++){ //to convert the first letter after space to capital
465     nplname[0]=toupper(plname[0]);
466     if(plname[j-1]==' '){
467     nplname[j]=toupper(plname[j]);
468     nplname[j-1]=plname[j-1];}
469     else nplname[j]=plname[j];
470     }
471     nplname[j]='\0';
472     //*****************************
473     //sdfprintf(info,"\t\t\tPlayers List\n");
474     fprintf(info,"Player Name :%s\n",nplname);
475      //for date and time
476
477     time_t mytime;
478    mytime = time(NULL);
479    fprintf(info,"Played Date:%s",ctime(&mytime));
480       //*************************
```

Start here X *snake1.c X

```
481         fprintf(info,"Score:%d\n",px=Scoreonly());//call score to display score
482         //fprintf(info,"\nLevel:%d\n",10);//call level to display level
483       for(i=0;i<=50;i++)
484       fprintf(info,"%c",'_');
485       fprintf(info,"\n");
486       fclose(info);
487       printf("wanna see past records press 'y'\n");
488       cha=getch();
489       system("cls");
490       if(cha=='y'){
491       info=fopen("record.txt","r");
492       do{
493           putchar(c=getc(info));
494           }while(c!=EOF);}
495         fclose(info);
496   }
497   int Score()
498   {
499       int score;
500       GotoXY(20,8);
501       score=length-5;
502       printf("SCORE : %d",(length-5));
503       score=length-5;
504       GotoXY(50,8);
505       printf("Life : %d",life);
506       return score;
507   }
508   int Scoreonly()
509   {
510   int score=Score();
```

```
511  system("cls");
512  return score;
513  }
514  void Up()
515  {
516      int i;
517      for(i=0;i<=(bend[bend_no].y-head.y)&&len<length;i++)
518      {
519          GotoXY(head.x,head.y+i);
520          {
521              if(len==0)
522                  printf("^");
523              else
524                  printf("*");
525          }
526          body[len].x=head.x;
527          body[len].y=head.y+i;
528          len++;
529      }
530      Bend();
531      if(!kbhit())
532          head.y--;
533  }
534
```

CERTIFICATION

- I had completed my course in “C language” technology from “Coursera”.
- The name of my course is “C For Everyone: Programing Fundamentals”