

Project Report

On

THE SNAKE GAME

Punjab University, Chandigarh

Master of Computer Applications (M.C.A)

(Session 2021-2022)



Under The Supervision of:

Ms. M Syamala Devi

Professor

DCSA , PU

Submitted by :

Mohit

M.C.A. 2nd Semester

Rollno.: 24

Department Of Computer Science & Applications

CERTIFICATE

This is to certify that Mr. Mohit , Class Roll No. 24 a bonafide student of M.C.A. 2nd Sem being run by DCSA , PU, Chandigarh of batch 2021-2022 has completed the project entitled “ The Snake Game ” under my supervision & Guidance. It is further certified that the work done in this project is a result of candidate's own efforts.

Date: 06-06-2022

Ms. M Syamala Devi
Professor
DCSA, PU

ACKNOWLEDGMENT

“it is not possible to prepare report without the assistance & encouragement of the other people. This one is certainly no exception”

On the very outset of this report. I would like to extend my sincere & heartfelt Obligation towards all the personages who have helped me in this endeavour. Without their active guidance , help , cooperation & encouragement. I would not have made headway in the project.

I am grateful to Respected Madam, **Prof. M Syamala Devi**, Department of Computer Science and Applications, Panjab University, Chandigarh. for the guidance, inspiration and constructive suggestions that helped me in the preparation of this project. I could not have built such a challenging project without your help.

I extend my gratitude to Madam, **DR. Rohini Sharma**, Chairperson, Department of Computer Science and Applications, Panjab University, Chandigarh. for providing me with excellent tutors and a good and enhanced infrastructure and awesome environment that laid potentially strong foundation for my professional life.

I also extend my gratitude to Sir **Prof Raj Kumar** , Vice-Chancellor, Panjab University, Chandigarh. for providing me with all the facilities in DCSA, PU including college tutors staff, excellent infrastructure and awesome environment that laid potentially strong foundation for my professional life. I am also thankful to my colleagues/friends and others who have helped me in successful completion of the project.

Table of Contents

Sr. no	Contents	Page No.
1	Basic Information (SNAKE GAME)	5
2	INTRODUCTION	6
3	Need of the Project	7
4	Goals and Objective of the Project	7
5	Project Design: Methodology	8
6	Flow Chart of the Project	9
7	Feature of the Project	10-12
8	Implementation	13
9	Coding	14-29
10	Testing and Validation	30-31
11	Screenshot of the Project	31-35
12	Conclusion	36
13	Future Scope of the Project	36
14	Bibliography	37

The Snake Game

Snake 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.

The Snake design dates back to the arcade game Blockade, developed and published by Gremlin in 1976. It was cloned as Bigfoot Bonkers the same year. In 1977, Atari, Inc. released two Blockade-inspired titles: the arcade game Dominos and Atari VCS game Surround. Surround was one of the nine Atari VCS launch titles in the US and was sold by Sears under the name Chase. That same year, a similar game was launched for the Bally Astro Cade as Checkmate.

INTRODUCTION

The following is an example game written in C based on the game called 'snake' which has been around since the earliest days of home computing and has re-emerged in recent years on mobile phones. It isn't the world's greatest game, but it does give you an idea of what you can achieve with a relatively simple C program, and perhaps the basis by which to extend the principles and create more interesting games of your own.

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. Nokia has installed the "Snake Game" on many of its phones. The game is also available on several websites online.

Need Of Project

- will provide a fun and social form of entertainment.
- encourage teamwork and cooperation when played with others.
- make kids feel comfortable with technology.
- increase children's self-confidence and self-esteem as they master games.

Goals and Objectives of the Project

- This Project in C language of Snake Game is a simple console application with very simple graphics. In this project, you can play the popular "Snake Game" just like you played it elsewhere. You have to use the w, s, d or a keys to move the snake.
- Foods are provided at the several co-ordinates of the screen for the snake to eat. Every time the snake eats the food, its length will be increased by one element along with the score.
- It isn't the world's greatest game, but it does give you an idea of what you can achieve with a relatively simple C program, and perhaps the basis by which to extend the principles and create more interesting games of your own.

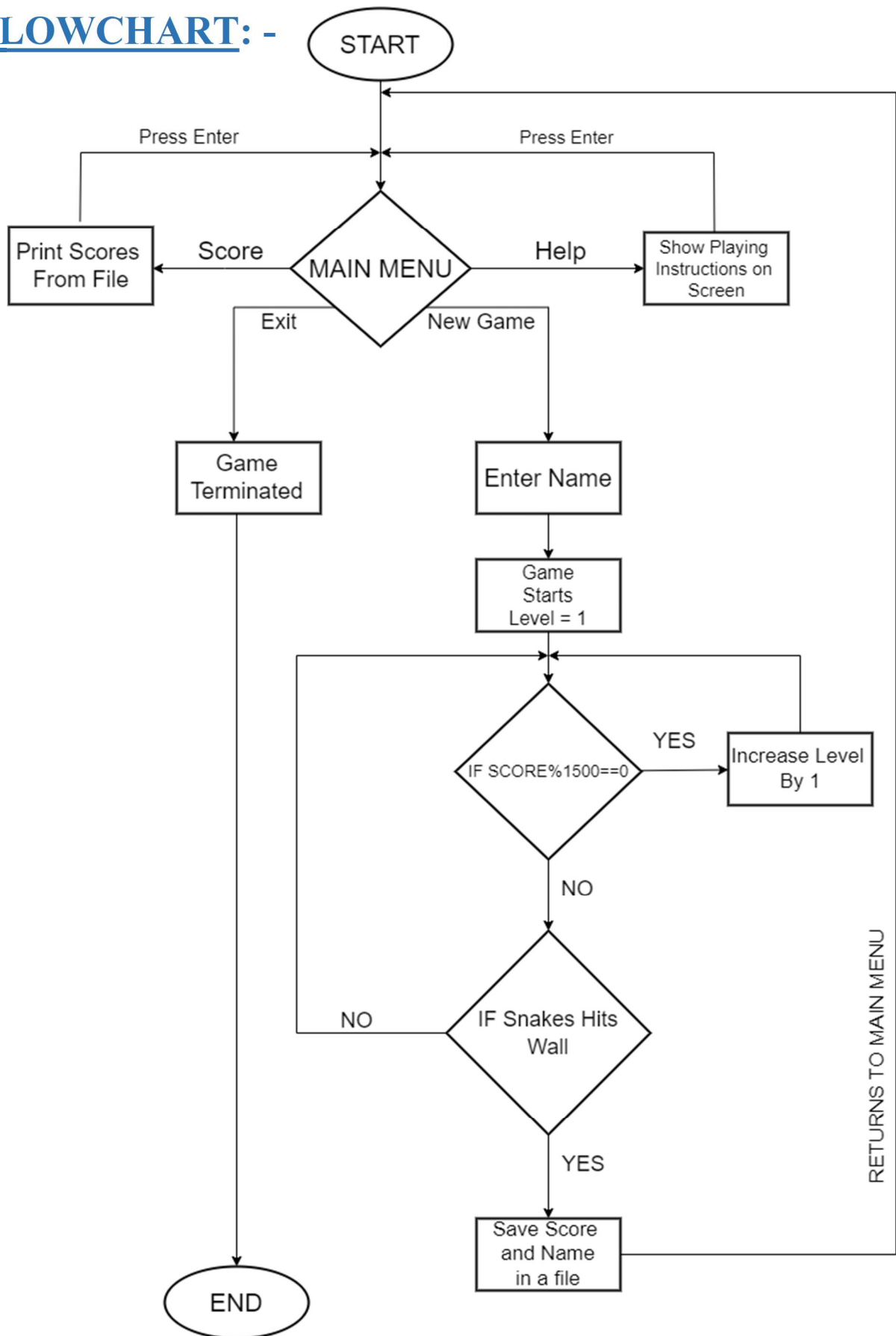
PROJECT DESIGN

Project design is an early phase of the project lifecycle where ideas, processes, resources, and deliverables are planned out. A project design comes before a project plan as it's a broad overview whereas a project plan includes more detailed information.

METHODOLOGY

This snake game is just similar to games which are found under mobile games section. It uses the concept of graphics to display the menu items and all objects on a single screen. There will be a main menu at starting, here user will select a option. If he selects Help option, it will display instructions on screen, if he selects Score option, it will display score on the screen, if he selects Exit option, the game will terminate. To play game user need to Select new game option. After selecting new game option, System asks for name of user, which he will enter. The right section will display the total score of the player, game level, player name, etc. Initial position of snake is in the middle of board. The snake object will always appear in combined white color squares. The Arrow keys have been set within the program to move the snake to right, left, top and bottom respectively.

FLOWCHART: -



FEATURES OF PROJECT

COMPUTER GRAPHICS:

- Computer Graphics is one of the most powerful and interesting aspect of computers.
- There are many things we can do in graphics apart from drawing figures of various shapes.
- All video games, animation, multimedia predominantly works using computer graphics.

Graphics in C:

- There is a large number of functions in C which are used for putting pixel on a graphic screen to form lines, shapes and patterns.
- The Default output mode of C language programs is "Text" mode.
- We have to switch to "Graphic" mode before drawing any graphical shape like line, rectangle, circle etc.
- First of all, we must include the "graphics.h" header file in our source program.
- GCC compiler doesn't provides inbuilt facility to run graphics.h library. So, you are not able to run graphics in C language.
- The initgraph function is used to switch the output from text mode to graphics mode.
- The initgraph function takes three arguments

```
initgraph(int *graphdriver, int *graphmode, char *pathtodriver);
```
- Graphic mode must be close at end using closegraph() function.

FILE HANDLING:

In this snake project, I am currently using one file to store name and score of the user who played game.

When a user starts the program, it will give four options to user: new game, help, score, exit.

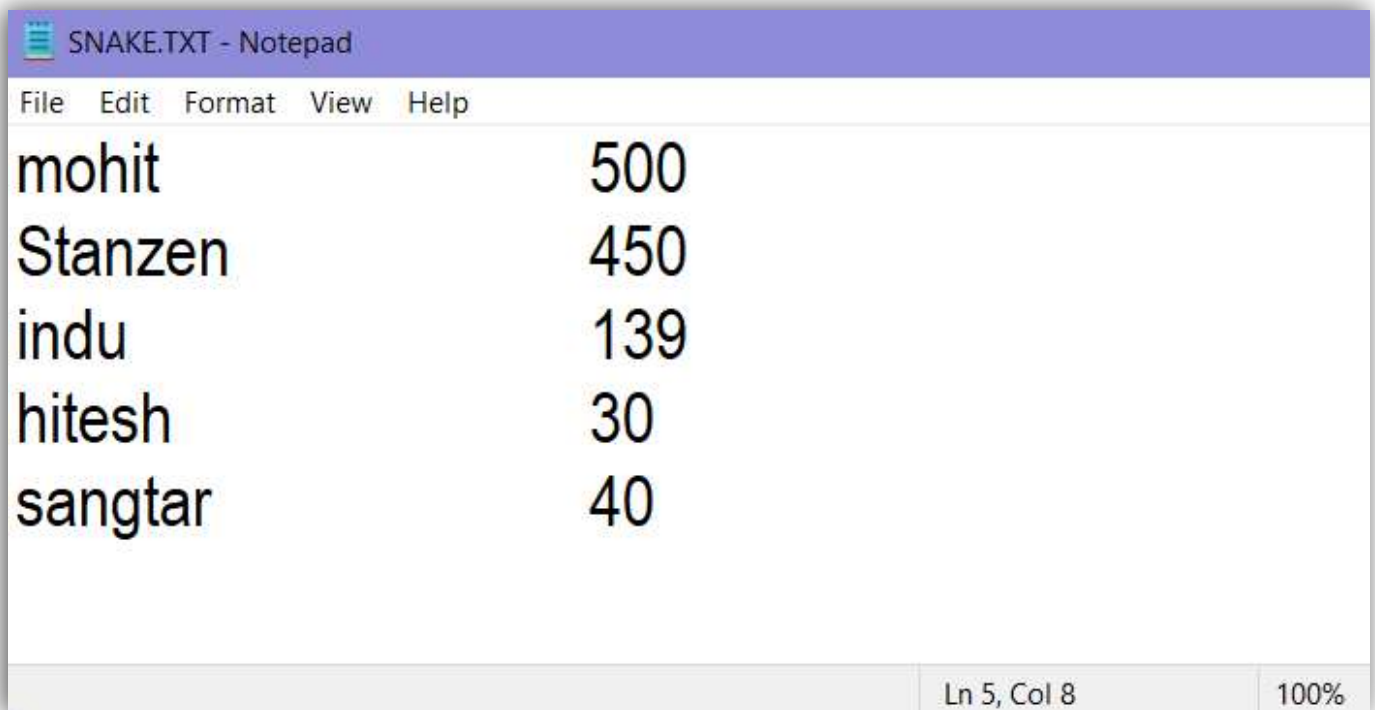
After selecting new game, system will ask for Name of the person who is playing the game.

After entering name, the game will start.

After completing the game, the system will append name and marks to the file.

We can also see our scores from score option in main menu.

Here is an example of how data is stored in the file:



The screenshot shows a Notepad window with the title 'SNAKE.TXT - Notepad'. The menu bar includes 'File', 'Edit', 'Format', 'View', and 'Help'. The text content is as follows:

mohit	500
Stanzen	450
indu	139
hitesh	30
sangtar	40

The status bar at the bottom indicates 'Ln 5, Col 8' and '100%' zoom.

DATA STRUCTURES USED IN PROJECT: -

1. Array: -

An array is a collection of items stored at contiguous memory locations. The idea is to store multiple items of the same type together. This makes it easier to calculate the position of each element by simply adding an offset to a base value, i.e., the memory location of the first element of the array (generally denoted by the name of the array). The base value is index 0 and the difference between the two indexes is the offset.

2. Linked List: -

A linked list is a sequence of data structures, which are connected together via links. Linked List is a sequence of links which contains items. Each link contains a connection to another link. Linked list is the second most-used data structure after array. It uses dynamic memory allocation. Insertion and deletion operation are easier as compared to array.

3. File Handling: -

File handling refers to the method of storing data in the C program in the form of an output or input that might have been generated while running a C program in a data file, i.e., a binary file or a text file for future analysis and reference in that very program.

A file refers to a source in which a program stores the information/data in the form of bytes of sequence on a disk (permanently).

IMPLEMENTATION

Input/Output Device Requirement:

Input Device Require

- Keyboard

Output Device Require

- Monitor

Hardware and Software Requirement:

Hardware Required

- Processor : Pentium IV or Above
- RAM : 512 or Above
- Hard disk : 20Gb or Above

Software Required

- Operating system :- Linux , Ubuntu , Mac , Windows XP , 7 , 8 , 8.1 , 10,11
- IDE : Turbo C++

Note: It will not work properly with latest compilers because latest compiler doesn't support graphics.

Details :

C Filename: - CProject.c

TXT Filename: - SNAKE.TXT

EXE Filename: - (NONE)

(Turbo c++ doesn't make .exe file on compilation of code, and this code will not properly work with latest compilers like Dev c++/Code blocks/Visual code because of Graphics.)

CODE : -

```
#include<graphics.h>
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<dos.h>

char *sname[100],key;    //sname is used for name of player,key is used to read
                        //key entered

int score1,level=1,speed=60,maxx,maxy,n=0;
struct loc    //structure for location
{
    int x,y;
};
```

```
struct snake //structure for snake length and direction
{
    struct loc sloc;
    struct snake *link;
    char dir;
};
```

```
struct game_data //structure for score and food
{
    int score;
    int no_food;
};
struct game_data gd={0,0};
```

```
struct limit //structure to limit playable area
{
    int lx1,ly1,lx2,ly2;
};
struct limit l={50,50,600,400};
```

```

struct food          //structure to generate food on screen
{
    struct loc flocc;
    int number;
};

void draw_board() //Basic UI function for main menu
{
    maxx=getmaxx(); //to get max value of x coordinate
    maxy=getmaxy(); //to get max value of y coordinate
    setcolor(WHITE);
    rectangle(0,0,maxx,maxy); //to draw rectangle on screen
    rectangle(8,8,maxx-8,maxy-8);
    rectangle(maxx-130,38,maxx-133,maxy-8);
    rectangle(maxx-130,70,maxx-9,73);
    setfillstyle(9,2); //forming fill style
    floodfill(6,6,WHITE); //to fill a specific area
    rectangle(8,35,maxx-8,38);
    setfillstyle(SOLID_FILL,12);
    bar(9,9,maxx-9,34);
    settextstyle(0,0,2);
    setcolor(9);
    outtextxy(maxx/2-150,16,"S N A K E  G A M E");
    setcolor(YELLOW);
    settextstyle(2,0,4);
    settextstyle(2,0,4);
    setcolor(3);

```



```

outtextxy(maxx-120,40, "Panjab University");
outtextxy(maxx-100,50, "Chandigarh");
settextstyle(2,0,5);
outtextxy(maxx-115,310, "Developed By");
setcolor(2);
settextstyle(2,0,4);
outtextxy(maxx-90, 330, "MOHIT");
setcolor(10);
outtextxy(maxx-105, 350, "MCA 2nd Sem");
outtextxy(maxx-100, 370, "Guided By");
outtextxy(maxx-120, 390, "Mrs. Shyamla Devi");
}

```

```

//Funtion to move snake around the area
void game1(struct snake *head,struct food *f)
{
int ch;
struct snake *temp,pre,nxt;
temp=head;
speed=60;
level=1;
gd.no_food=0;
gd.score=0;
while(key!='p')
{

```

```

if(head->sloc.x==l.lx1||head->sloc.x==l.lx2||head->sloc.y==l.ly1||head-
>sloc.y==l.ly2)
{
loss_game(gd.score);
}

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;
f->floc.x=60+random(430);
f->floc.y=60+random(330);
gd.score+=100;    //increment in score
gd.no_food+=1;    //increment in food
n=n+1;
//if(gd.no_food%15==0)
if(gd.score%1500==0)
{
if(speed>10)
{
speed=speed-10;
delay(2);
settextstyle(1,0,6);
outtextxy(180,200,"NEXT LEVEL");
settextstyle(1,0,2);
outtextxy(180,260,"Press any key to continue");
delay(2);
getch();

```

```

}
}
if(gd.no_food%15==0)
{
level=level+1;    //increment in level
}
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;
}
settextstyle(2,0,4);
outtextxy(maxx-200,17, "Your Score:");
gotoxy(66,2);
printf("%d",gd.score);
outtextxy(maxx-300,17,"Speed :");
gotoxy(50,2);
printf("%d",70-speed);
outtextxy(maxx-400,17,"Level :");
gotoxy(38,2);
printf("%d",level);
outtextxy(maxx-550,17,"Food Collected :");
gotoxy(26,2);
printf("%d",gd.no_food);

```

```

settextstyle(3,0,4);
outtextxy(maxx-510,maxy-65,"THE SNAKE GAME USING LL");
switch(key)
{
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);
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;
}
}
}

```

//function to draw snake and other info during gameplay

```

draw(struct snake *head,struct food *f)
{
struct snake *temp;
temp=head;
rectangle(50,50,600,400);
rectangle(45,45,605,405);
setfillstyle(9,13);
bar(temp->sloc.x-6,temp->sloc.y-6,temp->sloc.x+6,temp->sloc.y+6);
temp=temp->link;
setfillstyle(9,2);
while(temp->link!=NULL)
{
bar(temp->sloc.x-5,temp->sloc.y-5,temp->sloc.x+5,temp->sloc.y+5);
temp=temp->link;
}
bar(temp->sloc.x-5,temp->sloc.y-5,temp->sloc.x+5,temp->sloc.y+5);

```

```

circle(f->floc.x-2,f->floc.y-2,5);
circle(f->floc.x+2,f->floc.y+2,5);
circle(f->floc.x-2,f->floc.y+2,5);
circle(f->floc.x+2,f->floc.y-2,5);
delay(speed);
while(!kbhit()){goto e;}
key=getche();
e:
cleardevice();
return 0;
}

void help() //function to display instructions
{
char *str[]={ "1:-Make Snake Long By Eating Food.",
"2:-Move Snake By Using These Keys:-",
" * A :- Move Left",
" * D :- Move Right",
" * W :- Move Up",
" * S :- Move Down",
"NOTE 1 :- Do not touch snake to boundries.",
"If you do this then result is GAME OVER.",
"NOTE 2 :- Speed will not go above 60",
"Have Fun Playing Snake Game"};
int y=120,i;
setcolor(WHITE);
settextstyle(3,0,4);

```

```

rectangle(50,40,590,100);
outtextxy(230,50,"Instructions");
setfillstyle(SOLID_FILL,RED);
bar3d(50,100,590,430,0,5);
settextstyle(0,0,1);
setcolor(YELLOW);
for(i=0;i<10;i++)
{
outtextxy(70,y,str[i]);
delay(700);
y=y+25;
}
getch();
cleardevice();
fflush(stdin);
main();
}

```

```

//function to save score in a file
void sethigh(int s,char *sname)
{
FILE *fp;
fp=fopen("snake.txt","a");
if(!fp)
printf("ERROR");
fprintf(fp,"\n%s \t\t %d",sname,s);
fclose(fp);
}

```



```

}
//function to display scores from file
void showhigh()
{
char buffer[100];
int y=8;
FILE *fp1;
setcolor(WHITE);
settextstyle(3,0,4);
rectangle(50,40,590,100);
outtextxy(230,50,"SCORES");
setfillstyle(SOLID_FILL,RED);
bar3d(50,100,590,430,0,5);
settextstyle(0,0,1);
setcolor(YELLOW);
fp1=fopen("snake.txt","r");
while(!feof(fp1))
{
if(feof(fp1))
break;
else
{
fgets(buffer,100,fp1);
gotoxy(28,y);
y=y+1;
printf("%s",buffer); } }
fclose(fp1);

```

```

getch();
cleardevice();
main();
}

//function to get player name
//to call other funtion after entering name
void game(void)
{
    struct snake *s;
    struct food f;
    cleardevice();
    outtextxy(140,100,"ENTER PLAYER NAME");
    draw_board();
    gotoxy(28,11);
    setcolor(YELLOW);
    rectangle(140,140,370,200);
    scanf("%s",&sname);
    cleardevice();
    s=(struct snake *)malloc(sizeof(struct snake));
    s->dir='w';
    s->link=NULL;
    s->sloc.x= s->sloc.y=300;
    f.floc.x=150+random(250);
    f.floc.y=150+random(250);
    draw_board();
    game1(s,&f);
}

```

```

}

//function to display game over on screen
//and call another function to save score in file
int loss_game(int score)
{
    settextstyle(1,0,6);
    setcolor(RED);
    outtextxy(180,200,"GAME OVER");
    settextstyle(1,0,2);
    outtextxy(180,260,"Press any key to continue");
    sethigh(score,sname);
    getch();
    cleardevice();
    main();
    return 0;
}

//main funtion, it contain main menu screen
//New Game,Help,Score,Exit
int main()
{

    int gd=DETECT,gm;
    int cursor_y = 130;
    char ch;
    int x;

```

```

initgraph(&gd,&gm,"c:\\tc\\bgi");
maxx=getmaxx();
maxy=getmaxy();
while(1)
{
draw_board();
settextstyle(8,0,6);
setcolor(YELLOW);
settextstyle(3,0,3);
rectangle(160,125,390,163);
outtextxy(200,130,"New Game");
rectangle(160,167,390,201);
outtextxy(200,170,"Help");
rectangle(160,205,390,247);
outtextxy(200,210,"Score");
rectangle(160,243,390,287);
outtextxy(200,250,"Exit");
setcolor(RED);
outtextxy(170,cursor_y,"->");
setcolor(WHITE);
settextstyle(3,0,3);
outtextxy(90,350,"<<<WELCOME TO SNAKE GAME>>>");
x=0;
if(x==0){
ch = getch();
cleardevice();
if(ch == 13)

```

```

{
if(cursor_y==250) exit(0);
else if(cursor_y==130) game();
else if(cursor_y==170) help();
else if(cursor_y==210) showhigh();
}
}
//setcolor(WHITE);
if(ch==80) {cursor_y+=40;
if(cursor_y > 250) {cursor_y=130;}
}
if(ch==72) {cursor_y-=40;
if(cursor_y < 130) {cursor_y=250;}
}
}
}
}

```

TESTING AND VALIDATION

After the preparation of my project, I used it with the help of hypothetical data. As the requirement was satisfied with these data, I will try to implemented the project on the original data. The system though developed carefully, whenever it is taken in actual use may generate errors. So the main purpose of testing is to remove such type of runtime error and correct them. The scope of the system test includes both manual operation and computerized operations. We performed various system tests such as program test, string test, and system test.

Program test:

These are designed to test the logic of program. Under this testing, the individual forms were considered as a program and verification was done by entering hypothetical as well as original data.

System test:

These are used to test all programs which together constitute the system consisted of various forms. All forms are liked with each other perfectly and make our system a perfect one.

TESTING AND BUGS: -

Well, there wasn't any major issues during Testing, just sometime bugs appear, which I fixed with time.

There was a bug which does not let user enter name, to fix that I need to modify the whole code.

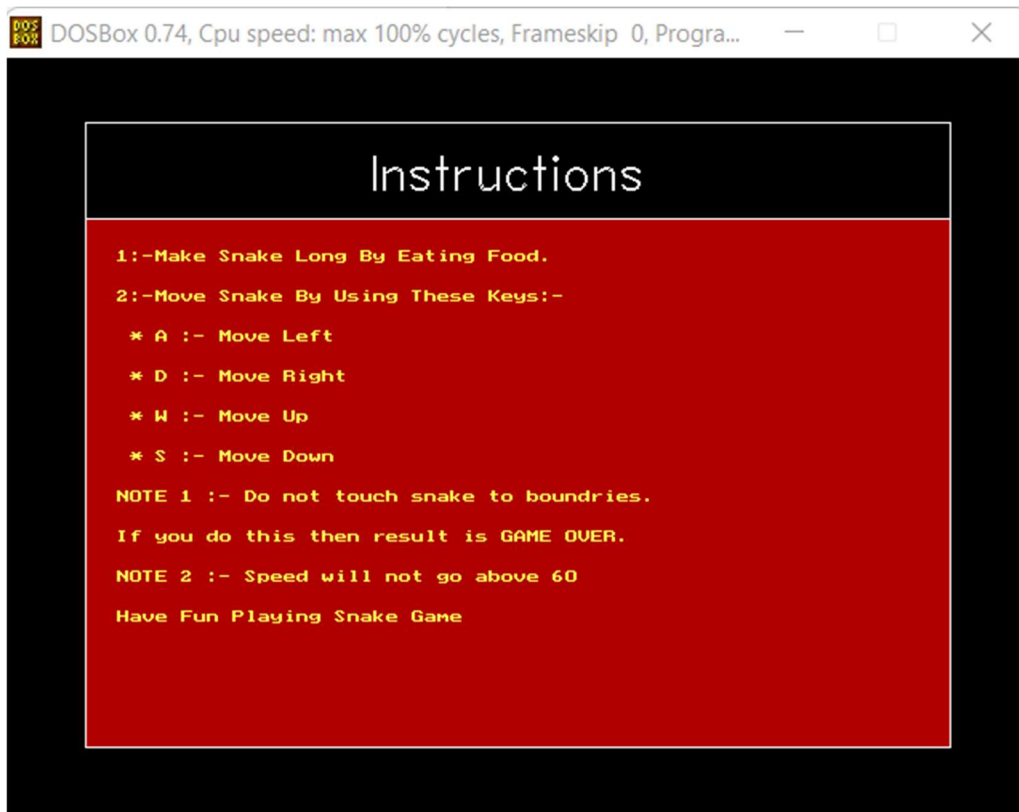
There was a bug with Linked list in which snake wasn't getting aligned, I took me well enough time to fix that.

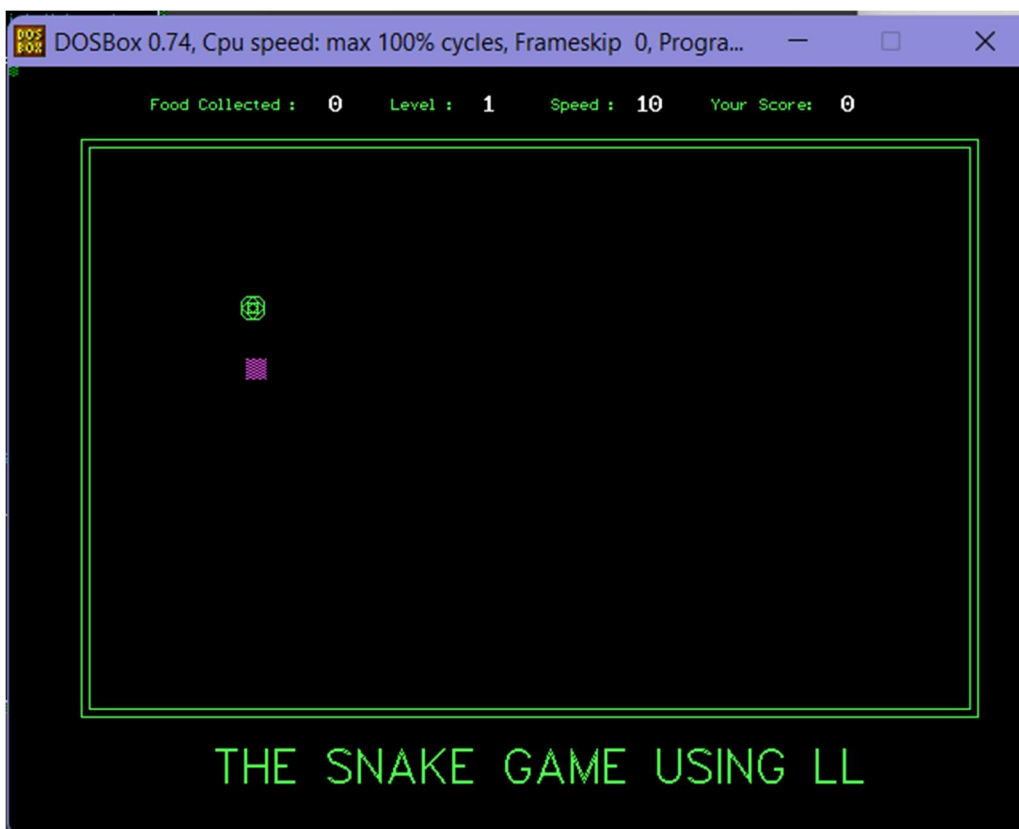
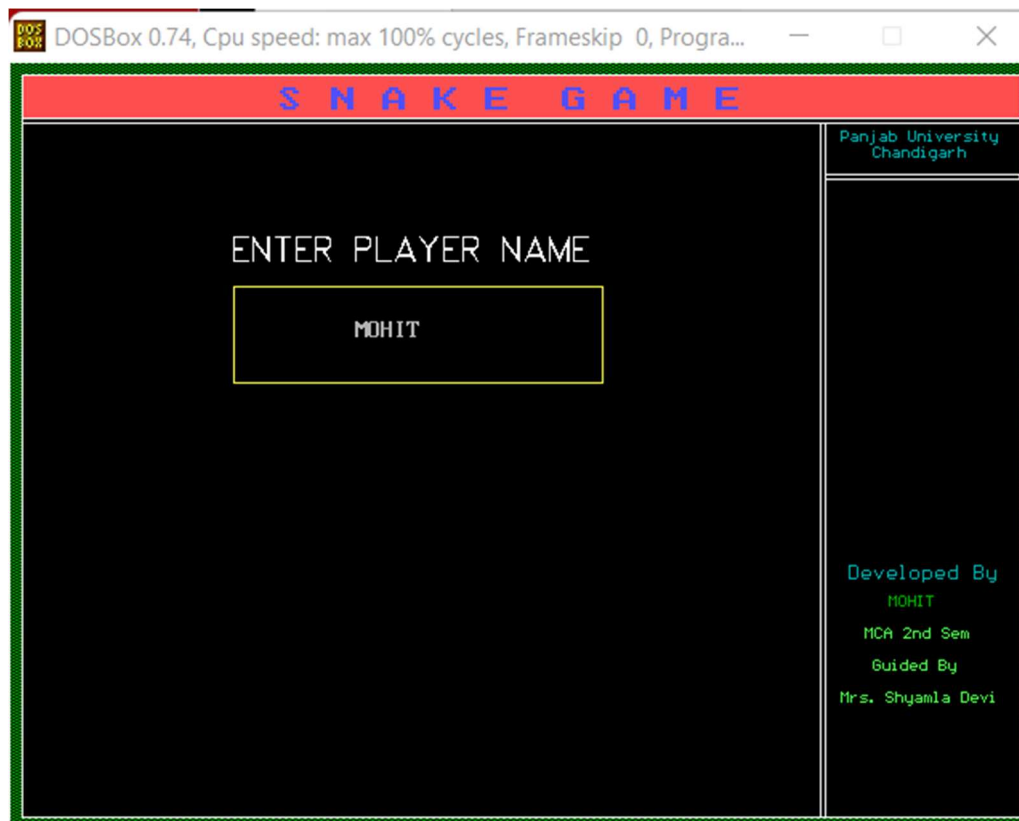
There were also some UI glitches, which I fixed with time.

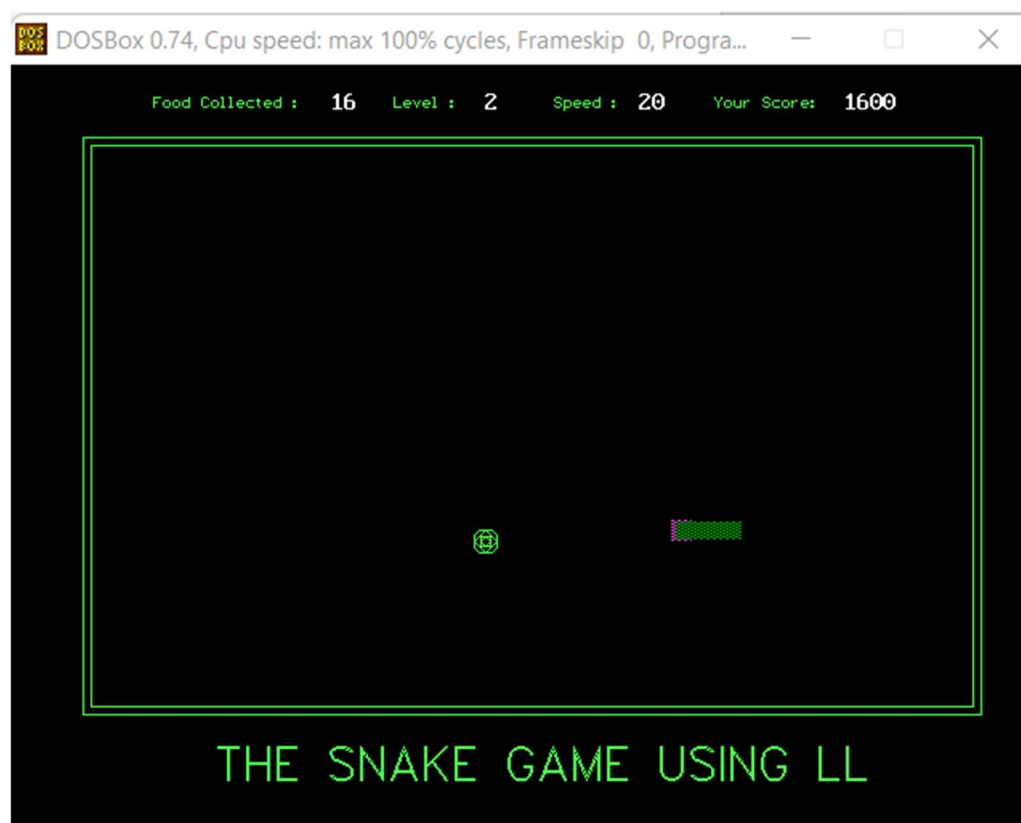
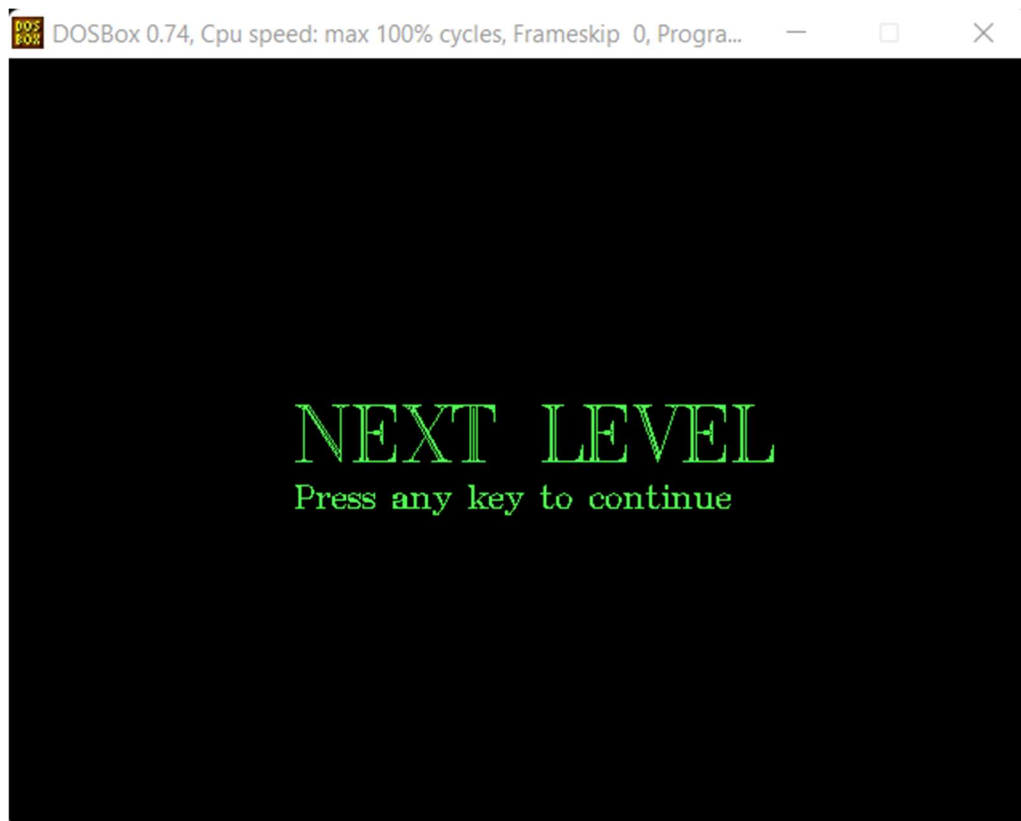
Now it is working properly.

SCREENSHOT OF PROJECT:











CONCLUSION

Conclusion of this Report is that Snake Game is one of the major Retro Games which we can play on mobiles, online, etc.

With the help of this project, I get to know about how much fun and entertaining game it is.

This project increased my knowledge about C Graphics.

I also get proper knowledge about some data structure concepts like, Arrays, Linked List, etc.

This Project was quite complex during snake designing phase using linked list.

C Graphics are quite old, and not as interactive as latest graphics Technology.

I felt very happy after completing this Project under given time.

Further Scope of The Project

This project will be able to implement in future after making some changes and modifications as we make our project at a very low level. So, the modifications that can be done in our project are:

- It can be made with better graphics.
- We can add more options like Top scores and Player Profile
- We can add multiplayer option.

BIBLIOGRAPHY

- <https://www.youtube.com/>
- <https://www.geeksforgeeks.com/>
- <https://www.tutorialspoint.com/>
- <https://www.stackoverflow.com/>
- Programming with C by E Balagurusamy
- Fundamental of Data Structure by Anshuman Sharma

THANK
YOU