VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA BELAGAVI-590018

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (DATA SCIENCE)

AMC ENGINEERING COLLEGE

(NBA Accredited, Approved by AICTE, New Delhi & Affiliated to VTU, Belagavi)

18th K.M, Bannerghatta Main Road, Bangalore-560083

2023-2024

REPORT ON

DSA PROJECT - THE SNAKE GAME

Submitted by:

MOHAMMED AMMAR BIN ZAMEER	1AM22CD057
MOHAMMED FARAAZ AHMED	1AM22CD058
MOHAN RAVI BARAGI	1AM22CD059
MANAN	1AM22CD049
DHIKSHA C G	1AM22CD023

Under the guidance of:

Mr Vinod K
Assistant Professor
Dept Of CSE (Data Science)
AMCEC

CONTENTS

Sl.No	Description
1	Introduction
2	Source Code
3	Output
4	Conclusion

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.

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.You get scored according to the length of the snake and the number of 'x' obstacles on the screen.The speed increases every 5 levels. You get a bonus when you complete the level of 1000, increasing by 1000 each level (e.g. complete level 5, you get a 5000 bonus).There is no concept of lives. Once you hit an obstacle, that's it, game over.

SOURCE CODE

```
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
/* prototypes */
void draw_line(int col, int row);
void show_score();
void add_segment();
void setup_level();
/* constants */
const int maxrow=15, maxcol=77;
const int snake_start_col=33,snake_start_row=7;
const char up_key='a', down_key='z', left_key='o', right_key='p';
const int pause_length=500000;
/* global variables */
int score, snake_length, speed, obstacles, level, firstpress, high_score=0;
char screen_grid[maxrow][maxcol];
char direction = right_key;
struct snake_segment {
int row,col;
} snake[100];
void main()
{
/* Variable declarations within main() only */
char keypress;
do/* restart game loop */
{
obstacles=4; level=1; score=0; speed=14;
randomize(); /* Ensure random seed initiated */
setup_level();
/* main loop */
do
{
for (int i=0;i<(speed*pause_length);i++) int j=1+i; /*pause*/
/* If key has been hit, then check it is a direction key - if so,change
direction */
if (kbhit())
{
keypress=(char)getch();
```

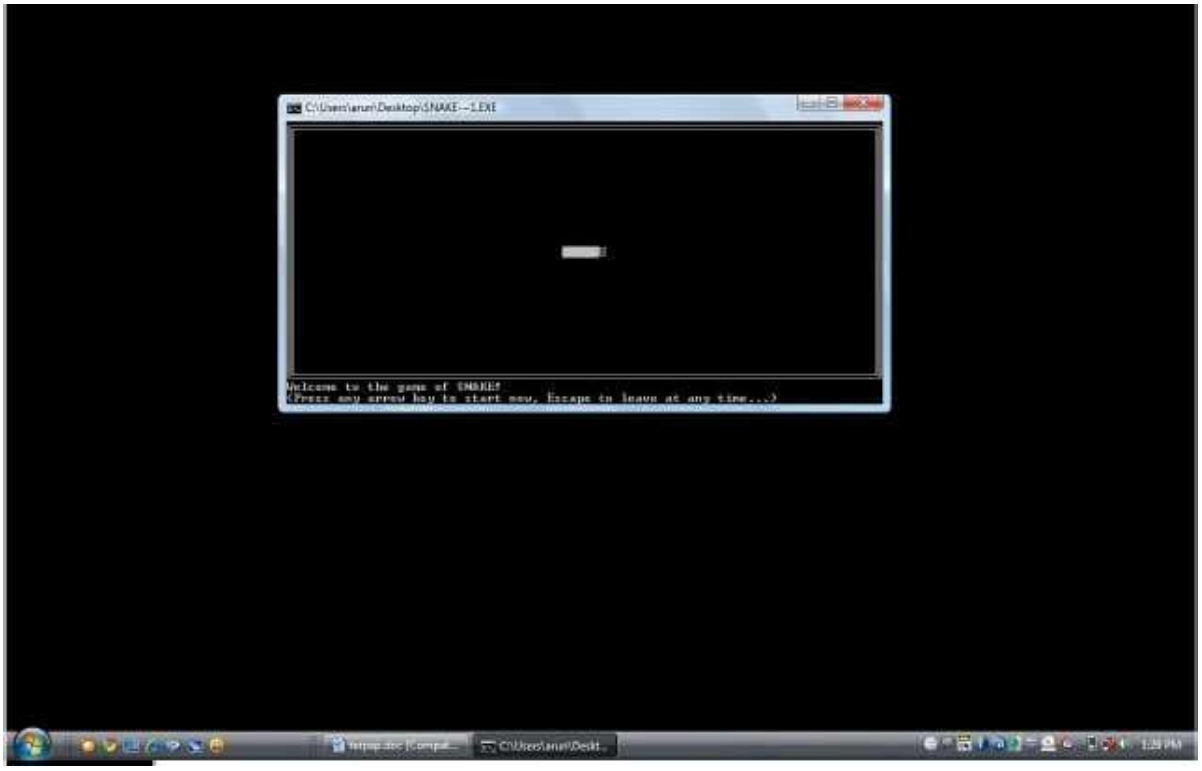
```
if((keypress==right_key)||(keypress==left_key)||
(keypress==up_key)||(keypress==down_key))
direction = keypress;
}
/* Add a segment to the end of the snake */
add_segment();
/* Blank last segment of snake */
gotoxy(snake[0].col,snake[0].row);
cprintf(" ");
/* ... and remove it from the array */
for (int i=1;i<=snake_length;i++)
snake[i-1]=snake[i];
/* Display snake in yellow */
textcolor(YELLOW);
for (int i=0;i<=snake_length;i++)
{
gotoxy(snake[i].col,snake[i].row);
cprintf("O");
}
/* keeps cursor flashing in one place instead of following snake */
gotoxy(1,1);
/* If first press on each level, pause until a key is pressed */
if (firstpress) { while(!kbhit()); firstpress = 0; }
/* Collision detection - walls (bad!) */
if ((snake[snake_length-1].row>maxrow+1)||(snake[snake_length-
1].row<=1)||
(snake[snake_length-1].col>maxcol+1)||(snake[snake_length-1].col<=1)||
/* Collision detection - obstacles (bad!) */
(screen_grid[snake[snake_length-1].row-2][snake[snake_length-1].col-
2]=='x'))
keypress='x'; /* i.e. exit loop - game over */
/* Collision detection - snake (bad!) */
for (int i=0;i<snake_length-1;i++)
if ( (snake[snake_length-1].row)==(snake[i].row) &&
(snake[snake_length-1].col)==(snake[i].col))
{
keypress='x'; /* i.e. exit loop - game over */
break; /* no need to check any more segments */
}
/* Collision detection - food (good!) */
if (screen_grid[snake[snake_length-1].row-2][snake[snake_length-1].col-
2]=='.')
{
```

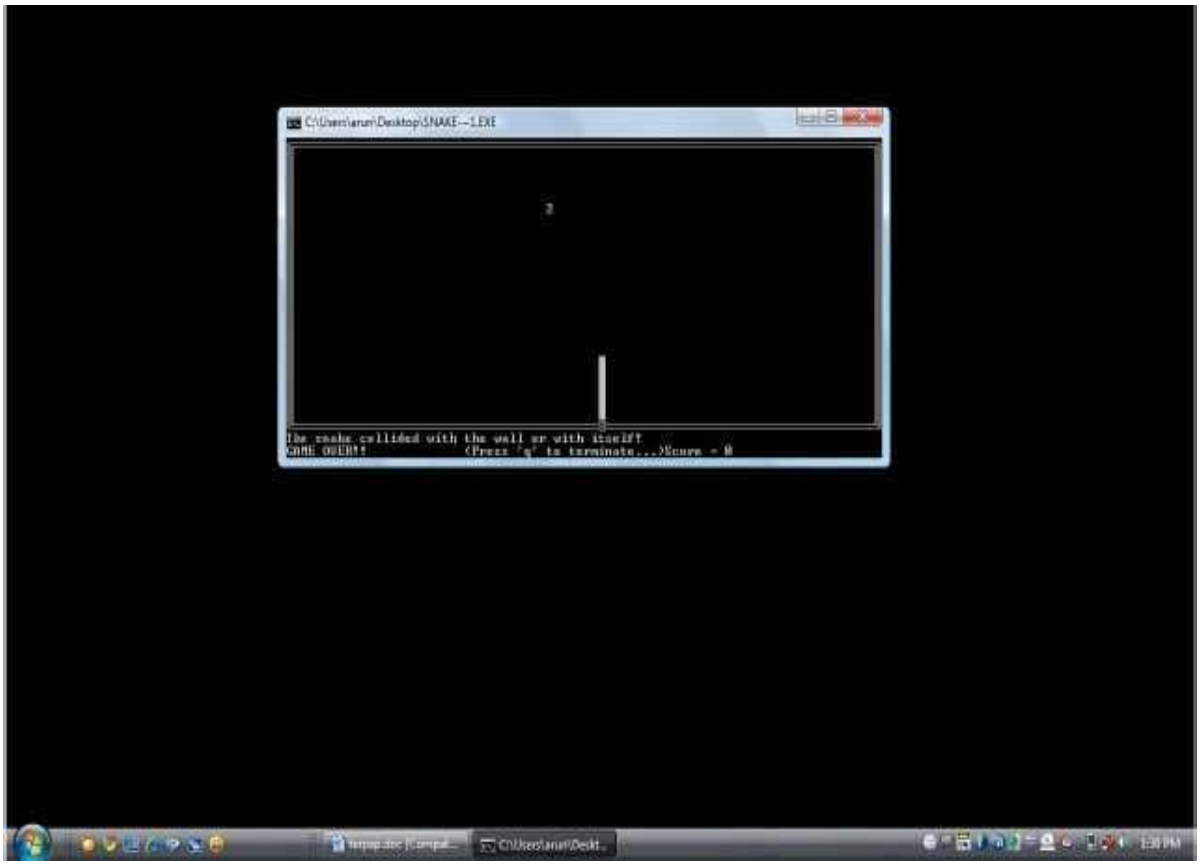
```
/* increase score and length of snake */
score+=snake_length*obstacles; show_score(); snake_length++;
add_segment();
/* if length of snake reaches certain size, onto next level */
if (snake_length==(level+3)*2)
{
score+=level*1000; obstacles+=2; level++; /* add to obstacles */
if ((level%5==0)&&(speed>1)) speed--; /* increase speed every 5 levels
*/
setup_level(); /* display next level */
}
}
} while (keypress!='x');
/* game over message */
if (score > high_score) high_score = score;
show_score();
gotoxy(30,6);
textcolor(LIGHTRED);
cprintf("G A M E O V E R");gotoxy(30,9); textcolor(YELLOW);
cprintf("Another Gamey/n)? ");
do keypress=getch(); while((keypress!='y')&&(keypress!='n'));
} while (keypress=='y');
void setup_level()
{
/* variables local to setup_level() */
int row,col;
/* Set up global variables for new level */
snake_length=level+4; direction = right_key;firstpress = 1;
/* Fill grid with blanks */
for(row=0;row<maxrow;row++)
for(col=0;col<maxcol;col++)
screen_grid[row][col]= ' ';
/* Fill grid with Xs and food */
for(int i=0;i<obstacles*2;i++)
{
row= rand()%maxrow;
col= rand()%maxcol;
if(i<obstacles)
screen_grid[row][col]='x';
else
screen_grid[row][col]='.';
}
/* Create snake array of length snake_length */
```

```
for(int i=0;i<snake_length;i++)
{
snake[i].row=snake_start_row;snake[i].col=snake_start_col+i;
}
/* Draw playing board */
draw_line(1,1);
for(row=0;row<maxrow;row++)
{
gotoxy(1,row+2);
textcolor(LIGHTBLUE); cprintf("|");
textcolor(WHITE);
for(col=0;col<maxcol;col++)
cprintf("%c",screen_grid[row][col]);
textcolor(LIGHTBLUE);
cprintf("|");
}
draw_line(1,maxrow+2);
show_score();
gotoxy(2,maxrow+5);
textcolor(LIGHTRED);
cprintf("~~ SNAKE GAME~~ Left: %c, Right: %c, Up: %c, Down: %c,
Exit: x. Anykey to start.",left_key,right_key,up_key,down_key);
}
void draw_line(int col, int row)
{
gotoxy(col,row); textcolor(LIGHTBLUE);
for (int col=0;col<maxcol+2;col++)
cprintf("=");
}
void show_score()
{
textcolor(LIGHTCYAN);
gotoxy(2,maxrow+3);
cprintf("Level: %05d",level);
gotoxy(40,maxrow+3);
textcolor(LIGHTGREEN);
cprintf("Score: %05d",score);
gotoxy(60,maxrow+3);
textcolor(LIGHTMAGENTA);
cprintf("High Score: %05d",high_score);
}
void add_segment()
{
```

```
switch(direction)
{
case(right_key): snake[snake_length].row=snake[snake_length-
1].row;snake[snake_length].col=snake[snake_length-1].col+1;
break;
case(left_key) : snake[snake_length].row=snake[snake_length-
1].row;snake[snake_length].col=snake[snake_length-1].col-1;
break;
case(up_key) : snake[snake_length].row=snake[snake_length-1].row-
1;snake[snake_length].col=snake[snake_length-1].col;
break;
case(down_key) : snake[snake_length].row=snake[snake_length-
1].row+1;snake[sn snake_length].col=snake[snake_length-1].col;
}
}
```

OUTPUT


C:\Users\arun\Desktop\SNAKE--1.EXE
Welcome to the game of SNAKE!
(Press any arrow key to start now, Escape to leave at any time...)


C:\Users\arun\Desktop\SNAKE--1.EXE
The snake collided with the wall or with itself!
GAME OVER!!
(Press 'q' to terminate...)Score = 0

CONCLUSION

In this project, the implementation of the infamous “Snake Game” is described. It discusses the steps to be taken while programming this game. The language used to write the game is C programming. The result has been shown with the help of pictures. This Snake Game has its own features and is interesting to play. Today games are rarely made using programming languages but the result is effective and helps one to learn more in depth about the programming language.