

**PSP [20ES104] COURSE PROJECT REPORT**

**On**

**Snake Game**

Developed By:

H.T.NO STUDENT NAME

2203A51541 B. ANJANKUMAR

**2205A21020** K. RAJU

2205A21028 O. RAJKUMAR

**2205A21033** T. SRI HARI

Under the Guidance of

Mr. Riyaz Mohammed

Assistant Professor

Submitted to

Department Computer Science and Artificial Intelligence SR University

Ananthasagar(V), Hasanparthy(M), Hanamkonda(Dist.) – 506371

[www.sru.edu.in](http://www.sru.edu.in/)

**June 2023**

**Department of Computer Science and Artificial Intelligence**

**CERTIFICATE**

This is to certify that the PSP course project report entitled **“SNAKE GAME”** is a record of bonafide work carried out by the student(s) B. ANJANKUMAR, K. RAJU, O. RAJ KUMAR & T. SRI HARI bearing roll number(s) 2203A51541, 2205A21020, 2205A21028 & 2205A21033 of Computer Science and Artificial Intelligence department during the academic year 2022-23.

**Supervisor**

(Riyaz Mohammed)

**INDEX**

**Sl. No Title Page No.**

1. Problem statement 4
2. Module-wise description 5
3. Knowledge required to develop the project 7
4. Source code (.c file code followed by .h file code) 8
5. Result 25

# PROBLEM STATEMENT:

**Develop a c program to display a snake game which allows user to play the game to move up,down,right,left.**

***Provide the functionality for below mentioned:***

**1.Increase the snake size by food**

**2.Bends according to instructions**

**3.Display the score board**

**4.Pause/play the game**

**5.stop the game**

**6.Moves in 4 directions**

**7.Border co-ordinates of the screen**

**8.Having three life's**

**9.To see past records press y**

**10.To exit press esc.**

# MODULES:

**1. Game Mechanics:**

**- The game should have a playing field represented by a grid, typically rectangular in shape.**

**- The player controls a snake that moves around the grid.**

**- The snake can move in four directions: up, down, left, and right.**

**- The snake grows in length when it consumes food.**

**- The game ends if the snake collides with the game boundaries or itself.**

**2. User Interface:**

**- The game should have a graphical user interface (GUI) to display the game elements.**

**- The GUI should show the current state of the game, including the snake, food, and score.**

**- It should provide controls for the player to navigate the snake (e.g., arrow keys, WASD).**

**3. Game Logic:**

**- The game should generate food at random positions on the grid.**

**- When the snake consumes food, its length should increase, and the score should be updated.**

**- The snake's movement should be continuous, and it should follow the player's input.**

**- The game should track the score and display it to the player.**

**- The game should have a high score feature that records and displays the best score achieved**

**4. Game Over and Restart:**

**- When the game ends (snake collides with boundaries or itself), it should display a game-over message.**

**- It should provide an option to restart the game and reset the score.**

**5. Optional Enhancements:**

**- Implement different levels of difficulty (e.g., increased speed, obstacles).**

**- Add power-ups or special food items that provide additional gameplay features.**

**- Include sound effects or background music to enhance the gaming experience.**

**- Create a leaderboard to track the top scores of different players.**

**KNOWLEDGE REQUIRED TO DEVELOP THIS APPLICATION**

* + Control Statements (if, if-else, switch)
  + Loop Statements (while/do while, for)
  + Arrays (1D/2D-arrays)
  + Increment and decrement
  + Structure (structures and nested structures)
  + Strings (Strings and Table of strings)
  + Functions (Any type of user defined functions)
  + Basic system function

**SOURCE CODE [.C FILE]:**

#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();

void Boarder();

void Down();

void Left();

void Up();

void Right();

void ExitGame();

int Scoreonly();

struct coordinate

{

int x;

int y;

int direction;

};

typedef struct coordinate coordinate;

coordinate head, bend[500],food,body[30];

int main()

{

char key;

Print();

system("cls");

load();

length=5;

head.x=25;

head.y=20;

head.direction=RIGHT;

Boarder();

Food(); //to generate food coordinates initially

life=3; //number of extra lives

bend[0]=head;

Move(); //initialing initial bend coordinate

return 0;

}

void Move()

{

int a,i;

do

{

Food();

fflush(stdin);

len=0;

for(i=0; i<30; i++)

{

body[i].x=0;

body[i].y=0;

if(i==length)

break;

}

Delay(length);

Boarder();

if(head.direction==RIGHT)

Right();

else if(head.direction==LEFT)

Left();

else if(head.direction==DOWN)

Down();

else if(head.direction==UP)

Up();

ExitGame();

}

while(!kbhit());

a=getch();

if(a==27)

{

system("cls");

exit(0);

}

key=getch();

if((key==RIGHT&&head.direction!=LEFT&&head.direction!=RIGHT)||(key==LEFT&&head.direction!=RIGHT&&head.direction!=LEFT)||(key==UP&&head.direction!=DOWN&&head.direction!=UP)||(key==DOWN&&head.direction!=UP&&head.direction!=DOWN))

{

bend\_no++;

bend[bend\_no]=head;

head.direction=key;

if(key==UP)

head.y--;

if(key==DOWN)

head.y++;

if(key==RIGHT)

head.x++;

if(key==LEFT)

head.x--;

Move();

}

else if(key==27)

{

system("cls");

exit(0);

}

else

{

printf("\a");

Move();

}

}

void gotoxy(int x, int y)

{

COORD coord;

coord.X = x;

coord.Y = y;

SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), coord);

}

void GotoXY(int x, int y)

{

HANDLE a;

COORD b;

fflush(stdout);

b.X = x;

b.Y = y;

a = GetStdHandle(STD\_OUTPUT\_HANDLE);

SetConsoleCursorPosition(a,b);

}

void load()

{

int row,col,r,c,q;

gotoxy(36,14);

printf("loading...");

gotoxy(30,15);

for(r=1; r<=20; r++)

{

for(q=0; q<=100000000; q++); //to display the character slowly

printf("%c",177);

}

getch();

}

void Down()

{

int i;

for(i=0; i<=(head.y-bend[bend\_no].y)&&len<length; i++)

{

GotoXY(head.x,head.y-i);

{

if(len==0)

printf("v");

else

printf("\*");

}

body[len].x=head.x;

body[len].y=head.y-i;

len++;

}

Bend();

if(!kbhit())

head.y++;

}

void Delay(long double k)

{

Score();

long double i;

for(i=0; i<=(10000000); i++);

}

void ExitGame()

{

int i,check=0;

for(i=4; i<length; i++) //starts with 4 because it needs minimum 4 element to touch its own body

{

if(body[0].x==body[i].x&&body[0].y==body[i].y)

{

check++; //check's value increases as the coordinates of head is equal to any other body coordinate

}

if(i==length||check!=0)

break;

}

if(head.x<=10||head.x>=70||head.y<=10||head.y>=30||check!=0)

{

life--;

if(life>=0)

{

head.x=25;

head.y=20;

bend\_no=0;

head.direction=RIGHT;

Move();

}

else

{

system("cls");

printf("All lives completed\nBetter Luck Next Time!!!\nPress any key to quit the game\n");

record();

exit(0);

}

}

}

void Food()

{

if(head.x==food.x&&head.y==food.y)

{

length++;

time\_t a;

a=time(0);

srand(a);

food.x=rand()%70;

if(food.x<=10)

food.x+=11;

food.y=rand()%30;

if(food.y<=10)

food.y+=11;

}

else if(food.x==0)/\*to create food for the first time coz global variable are initialized with 0\*/

{

food.x=rand()%70;

if(food.x<=10)

food.x+=11;

food.y=rand()%30;

if(food.y<=10)

food.y+=11;

}

}

void Left()

{

int i;

for(i=0; i<=(bend[bend\_no].x-head.x)&&len<length; i++)

{

GotoXY((head.x+i),head.y);

{

if(len==0)

printf("<");

else

printf("\*");

}

body[len].x=head.x+i;

body[len].y=head.y;

len++;

}

Bend();

if(!kbhit())

head.x--;

}

void Right()

{

int i;

for(i=0; i<=(head.x-bend[bend\_no].x)&&len<length; i++)

{

//GotoXY((head.x-i),head.y);

body[len].x=head.x-i;

body[len].y=head.y;

GotoXY(body[len].x,body[len].y);

{

if(len==0)

printf(">");

else

printf("\*");

}

/\*body[len].x=head.x-i;

body[len].y=head.y;\*/

len++;

}

Bend();

if(!kbhit())

head.x++;

}

void Bend()

{

int i,j,diff;

for(i=bend\_no; i>=0&&len<length; i--)

{

if(bend[i].x==bend[i-1].x)

{

diff=bend[i].y-bend[i-1].y;

if(diff<0)

for(j=1; j<=(-diff); j++)

{

body[len].x=bend[i].x;

body[len].y=bend[i].y+j;

GotoXY(body[len].x,body[len].y);

printf("\*");

len++;

if(len==length)

break;

}

else if(diff>0)

for(j=1; j<=diff; j++)

{

/\*GotoXY(bend[i].x,(bend[i].y-j));

printf("\*");\*/

body[len].x=bend[i].x;

body[len].y=bend[i].y-j;

GotoXY(body[len].x,body[len].y);

printf("\*");

len++;

if(len==length)

break;

}

}

else if(bend[i].y==bend[i-1].y)

{

diff=bend[i].x-bend[i-1].x;

if(diff<0)

for(j=1; j<=(-diff)&&len<length; j++)

{

/\*GotoXY((bend[i].x+j),bend[i].y);

printf("\*");\*/

body[len].x=bend[i].x+j;

body[len].y=bend[i].y;

GotoXY(body[len].x,body[len].y);

printf("\*");

len++;

if(len==length)

break;

}

else if(diff>0)

for(j=1; j<=diff&&len<length; j++)

{

/\*GotoXY((bend[i].x-j),bend[i].y);

printf("\*");\*/

body[len].x=bend[i].x-j;

body[len].y=bend[i].y;

GotoXY(body[len].x,body[len].y);

printf("\*");

len++;

if(len==length)

break;

}

}

}

}

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 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.\n\n-> Here you are provided with three lives. Your life will decrease as you hit the wall or snake's body.\n\n-> YOu can pause the game in its middle by pressing any key. To continue the paused game press any other key once again\n\n-> If you want to exit press esc. \n");

printf("\n\nPress any key to play game...");

if(getch()==27)

exit(0);

}

void record()

{

char plname[20],nplname[20],cha,c;

int i,j,px;

FILE \*info;

info=fopen("record.txt","a+");

getch();

system("cls");

printf("Enter your name\n");

scanf("%[^\n]",plname);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

for(j=0; plname[j]!='\0'; j++) //to convert the first letter after space to capital

{

nplname[0]=toupper(plname[0]);

if(plname[j-1]==' ')

{

nplname[j]=toupper(plname[j]);

nplname[j-1]=plname[j-1];

}

else nplname[j]=plname[j];

}

nplname[j]='\0';

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//sdfprintf(info,"\t\t\tPlayers List\n");

fprintf(info,"Player Name :%s\n",nplname);

//for date and time

time\_t mytime;

mytime = time(NULL);

fprintf(info,"Played Date:%s",ctime(&mytime));

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

fprintf(info,"Score:%d\n",px=Scoreonly());//call score to display score

//fprintf(info,"\nLevel:%d\n",10);//call level to display level

for(i=0; i<=50; i++)

fprintf(info,"%c",'\_');

fprintf(info,"\n");

fclose(info);

printf("Wanna see past records press 'y'\n");

cha=getch();

system("cls");

if(cha=='y')

{

info=fopen("record.txt","r");

do

{

putchar(c=getc(info));

}

while(c!=EOF);

}

fclose(info);

}

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);

return score;

}

int Scoreonly()

{

int score=Score();

system("cls");

return score;

}

void Up()

{

int i;

for(i=0; i<=(bend[bend\_no].y-head.y)&&len<length; i++)

{

GotoXY(head.x,head.y+i);

{

if(len==0)

printf("^");

else

printf("\*");

}

body[len].x=head.x;

body[len].y=head.y+i;

len++;

}

Bend();

if(!kbhit())

head.y--;

}

SOURCE CODE [HEADER FILE]:

#ifndef SNAKE\_H

#define SNAKE\_H

// Constants

#define WIDTH 20

#define HEIGHT 20

// Structures

typedef struct {

int x;

int y;

} Position;

typedef struct {

int score;

int gameOver;

} GameStatus;

typedef struct {

Position position;

struct SnakeSegment\* next;

} SnakeSegment;

typedef struct {

SnakeSegment\* head;

SnakeSegment\* tail;

Position food;

GameStatus status;

} GameState;

// Function declarations

void initializeGame(GameState\* gameState);

void drawGame(const GameState\* gameState);

void handleInput(GameState\* gameState, char input);

void updateGame(GameState\* gameState);

void gameOver(GameState\* gameState);

void cleanUpGame(GameState\* gameState);

#endif

**SOURCE CODE [HEADER FILE]:**

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

#include <conio.h>

#include<time.h>

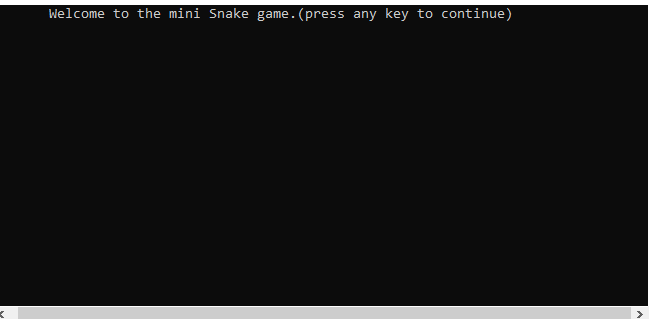
#include<ctype.h>

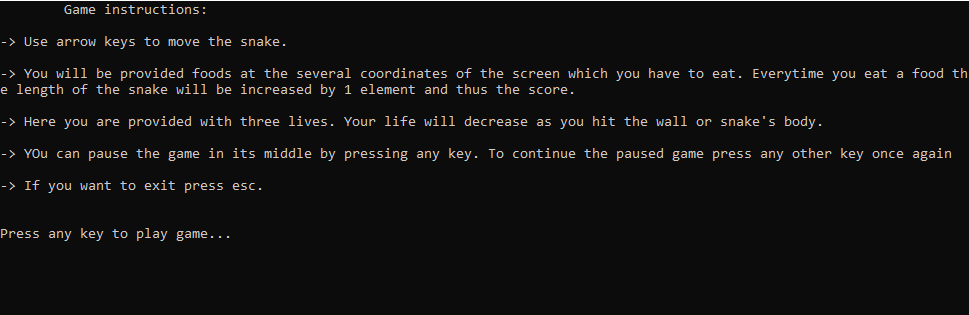
#include <time.h>

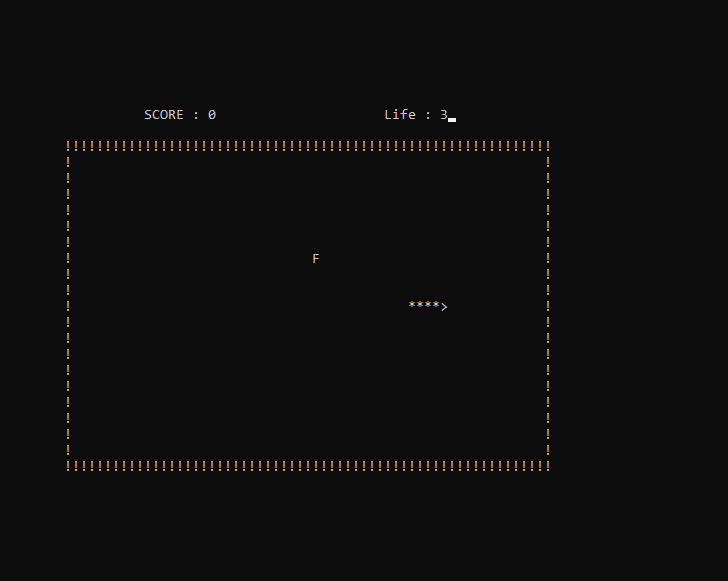
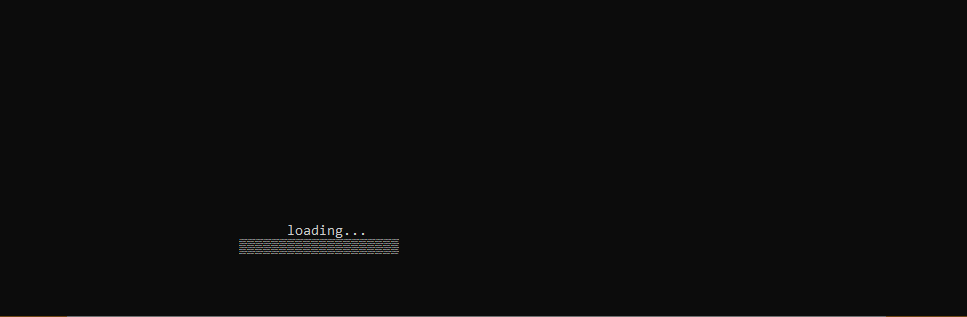
#include <windows.h>

#include <process.h>

**RESULTS:**







THE END

THANK YOU