

Batch – 2019-2023

Class- CS-D

Subject- Object Oriented Programming

Semester – 3rd

Project name – **Snake Game In OOPS**

Submitted to – Submitted by –

Mr.Gaurav Sharma EN19CS301223 NIHAR KUSHWAH

**Introduction:**

The history of the Snake game goes back to the 1970's. However, it was the 1980's when the game took on the look that we will be using. It was sold under numerous names and many platforms but probably gained widespread recognition when it was shipped as standard on Nokia mobile phones in the late 1990's.

The history of the Snake game goes back to the 1970's. However, it was the 1980's when the game took on the look that we will be using. It was sold under numerous names and many platforms but probably gained widespread recognition when it was shipped as standard on Nokia mobile phones in the late 1990's.

The game involves controlling a single block or snakehead by turning only left or right, up and down by ninety degrees until you manage to eat an fruit. When you get the fruit, the Snake grows an extra block or body segment.

If, or rather when, the snake bumps into the edge of the screen or accidentally eats himself the game is over. The more fruits the snake eats the higher the score.

The project started as a hobby project to explore our C/C++ skills and we used it as my project .

This is the classic version of the most popular mobile and computer game named “SNAKE”. The main objective of this game is to feed an increasing length of a snake with food particles which are found at random positions, picking up bonus fruits that occur at regular intervals.

Features Of Game:

1 A very user-friendly game.

2 Simple and easy game to understand.

3 Explanations provided in the game as and when required.

4 Presentations of score card in a precise tabular form.

**Description Of Game:**

Header files:

1. **<**iostream>  iostream

stands for standard input-output stream. This header file contains definitions to objects like cin, cout, cerr etc. iomanip: iomanip stands for input output manipulators. The methods declared in this files are used for manipulating streams. This file contains definitions of setw, setprecision, etc

1. <windows.h>

windows.h is a Windows-specific header file for the C and C++ programming languages which contains declarations for all of the functions in the Windows API, all the common macros used by Windows programmers, and all the data types used by the various functions and subsystems.

1. <conio.h>

**conio.h** is a [C](https://en.wikipedia.org/wiki/C_(programming_language)) header file used mostly by [MS-DOS](https://en.wikipedia.org/wiki/MS-DOS) compilers to provide console [input/output](https://en.wikipedia.org/wiki/C_file_input/output).[[1]](https://en.wikipedia.org/wiki/Conio.h#cite_note-1) It is not part of the [C standard library](https://en.wikipedia.org/wiki/C_standard_library) or [ISO C](https://en.wikipedia.org/wiki/ISO_C), nor is it defined by [POSIX](https://en.wikipedia.org/wiki/POSIX).

Concepts Of Object Oriented Programming Used in our game:

* Multilevel Inheritance:

Definition**-** programming, not only you can derive a class from the base class but you can also derive a class from the derived class. This form of inheritance is known as multilevel inheritance.

Role in Our Program –

We have made three classes class snake 1, class snake 2, class snake 3 . where class snake 1 is inherited publicly by snake class 2 and class snake 2 is inherited publicly by class snake 3 and Similarly class snake 3 .

And then in main function we made object snake3.obj and then call our all functions.

* Member Function Used –

we have used four member function to build our game.

1. Setup ()

we made setup function to design the sketch of our snake game. Initially we fixed the position of our snake and randomly fixed our fruit using rand() function

1. Draw()

in this **function** we made the map or layout of our game between which our snake will move and generate head of our snake with O and fruits by F .

1. Input()

In this function we get direction from the user though keyboard for movement of snake .

we use \_kbhit function to get input from keyboard and store it in \_getch function and then we use switch function to move our snake in different directions.

1. Logic()

we create this function to tail to follow the head of the snake and we use switch case to move in direction.

1. Main ()

in main function we create object obj1 with class snake3 with the help of this object we call other function using multilevel inheritance .

1. Sleep()

for using this function we use windows.h header file . by using this function we delay the speed of game .

When the player taps the screen, the game begins, and he must guide the tiny snake to get the first apple. And

we used “F” to represent fruit which is to be eaten by the snake which finally increases the tail of the snake... Whose coordinates are represented by (fruitX, fruitY).

We have used “ O ” to represent the snake head whose coordinate in 2D plan in represented by (X ,Y) . And we use “#” to fix the map of our snake to move between. And “\*” for the tail of snake. And whenever snake eat the fruit the score get increase by +10 points and the length of tail of snake increase respectively. And we used switch case to take the instruction from the user.

**Code:**

#include <iostream>

#include <windows.h>

#include <conio.h>

#define ARROW\_UP 72

#define ARROW\_DOWN 80

#define ARROW\_LEFT 75

#define ARROW\_RIGHT 77

using namespace std;

bool gameOver;

const int width = 20;

const int height = 20;

int x, y, fruitX, fruitY, score;

int tailX[100], tailY[100];

int nTail;

enum eDirecton { STOP = 0, LEFT, RIGHT, UP, DOWN};

eDirecton dir;

class snake1 {

public:

void Setup()

{

gameOver = false;

dir = STOP;

x = width / 2;

y = height / 2;

fruitX = rand() % width;

fruitY = rand() % height;

score = 0;

}};

class snake2:public snake1 {

public:

void Draw()

{

system("cls");

for (int i = 0; i < width+2; i++)

cout << "#";

cout << endl;

for (int i = 0; i < height; i++)

{

for (int j = 0; j < width; j++)

{

if (j == 0)

cout << "#";

if (i == y && j == x)

cout << "O";

else if (i == fruitY && j == fruitX)

cout << "F";

else

{

bool print = false;

for (int k = 0; k < nTail; k++)

{

if (tailX[k] == j && tailY[k] == i)

{

cout << "\*";

print = true;

}

}

if (!print)

cout << " ";

}

if (j == width - 1)

cout << "#";

}

cout << endl;

}

for (int i = 0; i < width+2; i++)

cout << "#";

cout << endl;

cout << "Score:" << score << endl;

}};

class snake3 : public snake2

{

public:

void Input()

{

if (\_kbhit())

{

switch (\_getch())

{

case 75 :

dir = LEFT;

break;

case 77:

dir = RIGHT;

break;

case 72:

dir = UP;

break;

case 80:

dir = DOWN;

break;

case 'x':

gameOver = true;

break;

}

}

}

void Logic()

{

int prevX = tailX[0];

int prevY = tailY[0];

int prev2X, prev2Y;

tailX[0] = x;

tailY[0] = y;

for (int i = 1; i < nTail; i++)

{

prev2X = tailX[i];

prev2Y = tailY[i];

tailX[i] = prevX;

tailY[i] = prevY;

prevX = prev2X;

prevY = prev2Y;

}

switch (dir)

{

case LEFT:

x--;

break;

case RIGHT:

x++;

break;

case UP:

y--;

break;

case DOWN:

y++;

break;

default:

break;

}

if (x >= width) x = 0; else if (x < 0) x = width - 1;

if (y >= height) y = 0; else if (y < 0) y = height - 1;

for (int i = 0; i < nTail; i++)

if (tailX[i] == x && tailY[i] == y)

gameOver = true;

if (x == fruitX && y == fruitY)

{

score += 10;

fruitX = rand() % width;

fruitY = rand() % height;

nTail++;

}

}};

int main()

{

snake3 obj1;

obj1.Setup();

while (!gameOver)

{

obj1.Draw();

obj1.Input();

obj1.Logic();

Sleep(200);

}

return 0;

}

**Output:**



