



INSTITUTE OF SPACE TECHNOLOGY, ISLAMABAD

BSCS 01 B

**Programming Fundamentals Lab Project – Snake Game
(Report)**

Course Code: 100103

**Abeer Ilyas - 200901055
Fasih Muhammad Virk - 20091012
Amna Raheel - 200901004
Malik Abdul Hadi - 200901053
Shiza Batool – 200901041**

Batch: 2024

Project Supervisor: Ms Sanya Zafar

2020 – 21

Word Count: 1492

Snake Game

Objective:

Gaming industry is a multibillion-dollar industry and has maligned the youth to itself. Being fresh CS students and having interest in gaming, we built a snake game using the practical knowledge of C++ GAINED IN Programming Fundamentals Lab classes.

Following is the detail of the components of our code:

Libraries:

We used iostream for regular input-output functions, windows.h for mode(), setup(), draw(), input(), logic() functions and conio.h is necessarily for _kbhit(), _getch() functions.

Variable:

- **gameover** to keep a check on game.
- **width** and **height** are constants which are used to set the size of the walls.
- **choice** for choosing among menu options.
- **mood** for choosing the mode.
- **x** and **y** for snake's head position.
- **fruitx** and **fruity** for positioning of fruit.
- **score** for showing the score.
- **tailx** and **taily** for positioning of tail.
- **ntail** for the length of tail.
- **eDirection** for control.

Mood Function:

This function is for the wall mode off the game in which when the head of the snake collides with the wall, it is game over.

Line of code: 15

```

15 int mode(int mood)
16 {
17     switch (mood) {
18     case 1:
19         //WHEN HEAD COLLIDES WITH THE WALL PASS TO THE OTHER SIDE AND COME OUT FROM THE PARALLEL WALL
20         if (x >= width)x = 0; else if (x < 0)x = width - 1;
21         if (y >= height)y = 0; else if (y < 0)y = height - 1;
22         break;
23     case 2:
24         //WHEN THE HEAD COLLIDES WITH THE WALL ITS GAME OVER
25         if (x > width || x<0 || y>height || y < 0)
26             gameover = true;
27     }
28     return mood;
29 }

```

Setup Function:

Here, we initialized the gameover to false, dir to stop. Though we can generate the snake anywhere, but we have generated it in the middle. And after generating the snake, we then generated the food at any random point but within the height and width of the wall.

Line of code: 38

```

38 void setup()
39 {
40     gameover = false;
41     //DEFINING VALUE OF X AND Y
42     x = width / 2;
43     y = height / 2;
44     srand(time(0));
45     //GENERATION RANDOM FRUIT POSITION
46     fruitx = rand() % width;
47     fruity = rand() % height;
48     score = 0;
49 }

```



Draw Function:

Here we just build up the wall boundary. Display the snake from head to tail and the fruit. The walls are shown by '#' character, the snake's body is shown by 'o' characters and the fruit by '*'.

Line of the code: 52

```

51 //FUNCTION TO DRAW THE WALLS IN THE GAME AND HEAD OF THE SNAKE AND THE OTHER BODY OF SNAKE
52 void draw()
53 {
54     system("cls");//FOR CLEARING THE SCREEN
55     //using loops for displaying everthing on screen from walls to the head and tail if the snake
56     for (int i = 0; i < width + 2; i++)
57         cout << "#";
58     cout << endl;
59
60     for (int i = 0; i < height; i++)
61     {
62         for (int j = 0; j < width; j++)
63         {
64             if (j == 0)
65                 cout << "#";
66             if (i == y && j == x)
67                 cout << "O";
68             else if (i == fruity && j == fruitx)
69                 cout << "*";
70             else
71             {
72                 bool print = false;
73                 for (int k = 0; k < ntaily; k++)
74                 {
75                     if (tailx[k] == j && taily[k] == i)
76                     {
77                         cout << "o";
78                         print = true;
79                     }
80                 }
81                 if (!print)
82                     cout << " ";
83             }
84             if (j == width - 1)
85                 cout << "#";
86         }
87         cout << endl;
88     }
89
90     for (int i = 0; i < width + 2; i++)
91         cout << "#";
92     cout << endl;
93     cout << "Score = " << score << endl;
94 }
95

```

Input Function:

In the input function, we used switch statements and when the `_kbhit()` function occurs we just maintained the switch cases (w, a, s, d) and change the direction respectively. The x key is for closing the game.

Line of the code: 97

```
97 void input()
98 {
99     //for controls
100     if (_kbhit())
101     {
102         switch (_getch())
103         {
104             case 'a':
105                 dir = LEFT;
106                 break;
107             case 'w':
108                 dir = UP;
109                 break;
110             case 'd':
111                 dir = RIGHT;
112                 break;
113             case 's':
114                 dir = DOWN;
115                 break;
116             case 'x':
117                 gameover = true;
118         }
119     }
120 }
121
122 }
```

KEYBOARD							
		w					
	a	s	d				
		x					

~Controls~

s = "down"
a = "left"
d = "right"
x = "EXIT"

| w = "up" |

Logic Function:

In the logic function, we first initialized the tail. And after that, we switched the position of the snake's body with its previous position. And after that, the program simply needs to implement the body according to the keyboard hit. Next, as this follows the concept of an open maze, so when it disappears at one side, it appears from the other side, so we kept in mind that once it touches the right wall it appears from the other left wall and vice-versa and same follows for the up and down walls. Next, the head touches the body, the game crashes. For the scoring system we added 10 points when the head touches the food (their position becomes the same). And every touch increases the score.

Line of code: 124

```

123 //FUNCTION TO APPLY LOGIC
124 void logic()
125 {
126
127     int prevx = tailx[0];
128     int prevy = taily[0];
129     int prev2x, prev2y;
130     tailx[0] = x;
131     taily[0] = y;
132     //LOOP FOR TAIL TO FOLLOW THE HEAD
133     for (int i = 1; i < ntail; i++)
134     {
135         prev2x = tailx[i];
136         prev2y = taily[i];
137         tailx[i] = prevx;
138         taily[i] = prevy;
139         prevx = prev2x;
140         prevy = prev2y;
141     }
142     //FOR CHANGING THE DIRECTION OF THE SNAKE HEAD
143     switch (dir)
144     {
145     case RIGHT:
146         x++;
147         break;
148     case LEFT:
149         x--;
150         break;
151     case UP:
152         y--;
153         break;
154     case DOWN:
155         y++;
156         break;
157     default:
158         break;
159     }
160     //FOR MODE
161     mode(mood);
162     for (int i = 0; i < ntail; i++)
163         if (tailx[i] == x && taily[i] == y)
164             gameover = true;
165     //FOR INCREMENTING THE SCORE AND GENERATING NEW FRUIT POSITION
166     if (x == fruitx && y == fruity)
167     {
168         score += 10;
169         srand(time(0));
170         fruitx = rand() % width;
171         fruity = rand() % height;
172         ntail++;
173     }
174 }

```

[illegible]

Main Function:

The main boy calls all the function and has the instructions to show the main menu and give a response according to the user input.

Line of code: 176

Input:

The user uses the prescribed keyboard keys to play the game:

- Keys:

[W] = Up

[A] = Left

[S] = Down

[D] = Right

Output:

Using the keys, the user can toggle through the menu and navigate the snake to get food.

Program:

```
#include <iostream>
#include <conio.h> //FOR INPUT OUTPUT PURPOSE
#include <windows.h> //FOR FUNCTIONS IN WINDOWS API
using namespace std;
//GLOBALLY DECLARE VARIBALE
char again;
bool gameover;
const int width = 20, height = 20;
int goback, choice, mood, x, y, fruitx, fruity, score;
int tailx[100], taily[100];
int ntail;
enum eDirection { STOP = 0, RIGHT, LEFT, DOWN, UP };
eDirection dir;
//FUNCTION TO CHOOSE A MODE
int mode(int mood)
{

    switch (mood) {
    case 1:

        //WHEN HEAD COLLIDES WITH THE WALL PASS TO THE OTHER SIDE AND COME OUT FROM
        THE PARALLEL WALL
```

```

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

        break;
    case 2:

        //WHEN THE HEAD COLLIDES WITH THE WALL ITS GAME OVER

        if (x > width || x<0 || y>height || y < 0)
            gameover = true;
    }
    return mood;
}
//FUNCTION TO GENERATE THE FRUIT
void setup()
{
    gameover = false;
    //DEFINING VALUE OF X AND Y
    x = width / 2;
    y = height / 2;
    srand(time(0));
    //GENERATION RANDOM FRUIT POSITION
    fruitx = rand() % width;
    fruity = rand() % height;
    score = 0;
}
//FUCTION TO DRAW THE WALLS IN THE GAME AND HEAD OF THE SNAKE AND THE OTHER BODY OF SNAKE
void draw()
{
    system("cls");//FOR CLEARING THE SCREEN
    //using loops for displaying everything on screen from walls to the head and tail
    if the snake
    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 << "*";
            else
            {
                bool print = false;
                for (int k = 0; k < ntaily; k++)
                {
                    if (tailx[k] == j && taily[k] == i)
                    {

```



```

        cout << "o";
        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;
}
//FUNCTION TO TAKE INPUT CONTROL FROM THE USER
void input()
{
    //for controls
    if (_kbhit())
    {
        switch (_getch())
        {
            case 'a':
                dir = LEFT;
                break;
            case 'w':
                dir = UP;
                break;
            case 'd':
                dir = RIGHT;
                break;
            case 's':
                dir = DOWN;
                break;
            case 'x':
                gameover = true;
        }
    }
}

}
//FUNCTION TO APPLY LOGIC
void logic()
{
    int prevx = tailx[0];
    int prevy = taily[0];
    int prev2x, prev2y;
    tailx[0] = x;
    taily[0] = y;

```

[illegible]

```
"\t\t | \n"
"\t\t |          2.   MODE      | \n"
"\t\t |          \n"
"\t\t |          3.   EXIT        | \n"
"\t\t |          \n"
"\t\t |          4.   CONTROLS    | \n"
"\t\t | _____| \n\n\n"
"\n";
```

```

cout << "\t\t\tENTER ->>";    cin >> choice;
// FOR SELECTING A MODE
if (choice == 2)
{

    cout << "\t\t\tCHOOSE A MODE FOR THE GAME\n"
         << "\t\t\tAVAILABLE MODE ARE TWO\n"
         << "\t\t\tPRESS 1 FOR NO WALLS\n"
         << "\t\t\tPRESS 2 FOR WALLS\n";
    cin >> mood;
    mode(mood);
    setup();
    while (!gameover)
    {
        draw();
        input();
        logic();
        Sleep(10);
    }
    cout << "\t\t\tGame End" << endl;
    cout << "\t\t\tYour Total Score is = " << score << endl;
}

//FOR DEFAULT MODE AND START GAME GAME
else if (choice == 1)
{
    mood = 1;
    mode(mood);
    setup();
    while (!gameover)
    {
        draw();
        input();
        logic();
        Sleep(10);
    }
    cout << "Game End" << endl;
    cout << "Your Total Score is = " << score << endl;
}

else if (choice == 3)
{
    score = 0;
    cout << "\t\t\tGame End" << endl;
}
}

```

```

        cout << "\tYour Total Score is = " << score << endl;
        exit;
    }
    else if (choice == 4)
    {
        //SHOWING CONTROLS ON KEYBOARD
        cout << endl;
        cout << "\t\t\t\t |                KEYBOARD                | \n"
            "\t\t\t\t |-----| \n"
            "\t\t\t\t | | | | | | | | | | \n"
            "\t\t\t\t |-----| \n"
            "\t\t\t\t | | | w | | | | | | | \n"
            "\t\t\t\t |-----| \n"
            "\t\t\t\t | | a | s | d | | | | | | \n"
            "\t\t\t\t |-----| \n"
            "\t\t\t\t | | | x | | | | | | | \n"
            "\t\t\t\t |-----| \n"
            "\t\t\t\t |                |                | \n"
            "\t\t\t\t |-----| \n"
            "\t\t\t\t |                |                | \n";
        cout << "\t\t\t\t\t~Controls~";
        cout << "\t\t\t\t\t |w = \"up\"      | \n"
            "\t\t\t\t\t |s = \"down\"    | \n"
            "\t\t\t\t\t |a = \"left\"    | \n"
            "\t\t\t\t\t |d = \"right\"   | \n"
            "\t\t\t\t\t |x = \"EXIT\"   | \n";
        cout << endl;
        cout << "\t\t\t\t\tPRESS 1 to GoBack ->>"; cin >> goback;
        cout << endl;
        if (goback == 1)
        {
            goto back;
        }
    }
}

```