**Pf project   
final submission**

#include <iostream>

#include <iomanip>

#include <ctime>

#include <string>

#include <conio.h>

#include <windows.h>

using namespace std;

char movement;

string playerName; // To store player name

int user = 0;

int gameMenu()

{

cout << "WELCOME TO THE GAME" << endl;

cout << "Enter your name: ";

getline(cin, playerName);

cout << "Press 1: To Start Game" << endl;

cout << "Press 0 To exit" << endl;

cin >> user;

if (user == 1)

{

cout << "Hello, " << playerName << "! The game is starting" << endl;

}

else if (user == 0)

{

return 1;

}

}

void displayInstructions()

{

cout << "1. Press 'a' = left" << endl;

cout << "2. Press 'd' = right" << endl;

cout << "3. Press 'w' = top" << endl;

cout << "4. Press 's' = bottom" << endl;

cout << "You will be given 3 lives" << endl;

cout << "THE LIFE WILL REDUCE IF YOU HIT THE WALL OR SNAKE BODY " << '\n';

system("Color 3");

Sleep(2000);

}

bool gameover;

int prevX = 9;

int prevY = 10;

int x = 10;

int y = 10;

int score = 0;

int foodx = 0;

int foody = 0;

const int qq = 14;

const int ww = 14;

int tailx[100];

int taily[100];

int ntail;

int lives = 2; // Number of lives

void setup()

{

gameover = false;

// srand is used in order to generate the random food.

srand(time(0));

foodx = rand() % qq + 1;

foody = rand() % ww + 1;

score = 0;

ntail = 0;

}

int highestscore = 20;

int sscore;

void drawgameBoard()

{

system("cls");

// cls is used here in order to clear the screen after the output

for (int i = 0; i < 20; i++)

{

for (int j = 0; j < 20; j++)

{

// '\*' represents the walls

if (i == 0 || i == 19 || j == 0 || j == 19)

{

cout << '\*';

}

//'0' represents the snake head

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

{

cout << '0';

}

// 'F' represents the food

else if (i == foodx && j == foody)

{

cout << '$';

}

// ' ' represents the empty space

else

{

bool printtail = false;

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

{

if (tailx[k] == i && taily[k] == j)

{

cout << 'o';

printtail = true;

}

}

if (!printtail)

{

cout << " ";

}

}

}

cout << endl;

}

cout << "score : " << score << '\n';

cout << "The remaining lives are : " << lives << '\n';

}

enum eDirection

{

STOP,

LEFT,

RIGHT,

UP,

DOWN

}; // enum is used as the user input the value than it will work.

eDirection dir;

void getinput()

{

if (\_kbhit())

{

switch (\_getch())

{

case 'a':

dir = LEFT;

break;

case 'd':

dir = RIGHT;

break;

case 'w':

dir = UP;

break;

case 's':

dir = DOWN;

break;

default:

break;

}

}

}

void updateSnake()

{

prevX = x;

prevY = y;

switch (dir)

{

case LEFT:

y--;

break;

case RIGHT:

y++;

break;

case UP:

x--;

break;

case DOWN:

x++;

break;

default:

break;

}

if (x < 0 || x == 19 || y < 0 || y == 19)

{

lives--; // Reduce life when hitting walls

if (lives <= -1)

{

gameover = true;

}

else

{

cout << "Lost a life! Lives remaining: " << lives << endl;

}

}

// By this, we are generating random food again as the position of food will change automatically.

if (x == foodx && y == foody)

{

const int qq = 14;

const int ww = 14;

score += 10;

foodx = rand() % qq + 1;

foody = rand() % ww + 1;

ntail++;

}

// Move the tail of the snake

for (int i = ntail - 1; i > 0; i--)

{

tailx[i] = tailx[i - 1];

taily[i] = taily[i - 1];

}

// Update the tail's position with the new head position

tailx[0] = prevX;

taily[0] = prevY;

// Check for collision with the tail

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

{

if (x == tailx[i] && y == taily[i])

{

lives--;

}

}

}

int main()

{

gameMenu();

setup();

displayInstructions();

while (!gameover)

{

if (lives <= -1)

{

break;

}

Sleep(100);

drawgameBoard();

getinput();

updateSnake();

}

if (score > highestscore)

{

cout << "Good play " << playerName << '\n';

cout << "You have made the highest score: " << score << '\n';

score = highestscore;

sscore = score;

cout << "Previous highest score was: " << sscore << '\n';

system("Color 4");

}

else

{

cout << "Good play " << playerName << '\n';

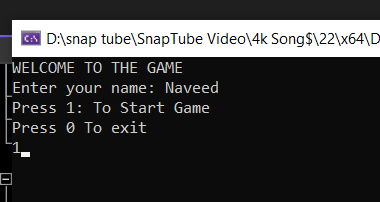
cout << "Your score was: " << score << '\n';

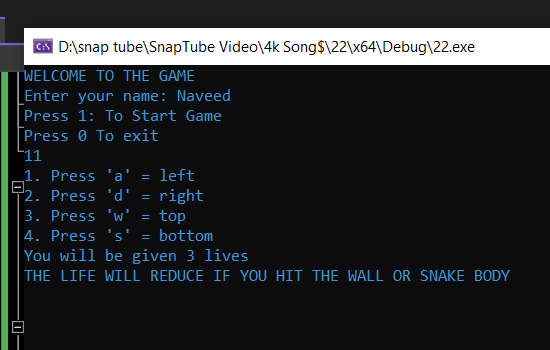
cout << "But you didn't score highest " << '\n';

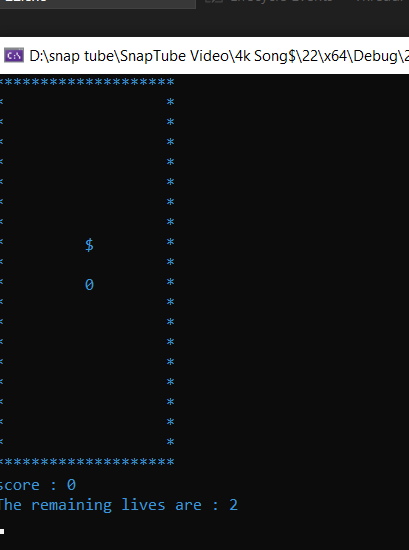
}

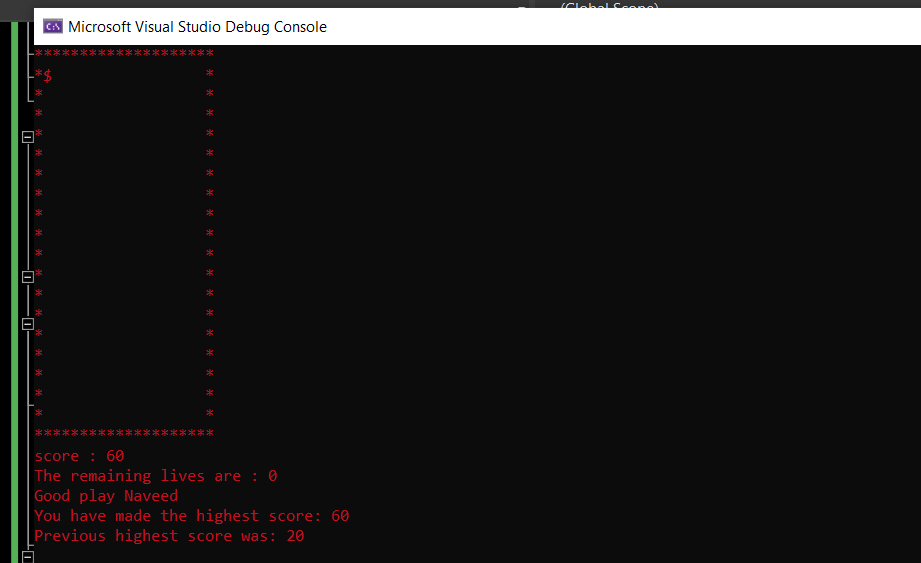
return 0;

}

****

****

****

****