به نام خدا

پروژه برنامه سازی پیشرفته

بابک محمودی

۴۰۱۱۲۳۵۸۰۳۹

متن سوال

Snake Game

روش حل

- اتبدا برای مار و غذای آن هرکدام یک فایل هدر میزنیم که در آن کلاس های مربوطه را ایجاد میکنیم همراه با اعضای public , privateو در فایل های cppهرکدام کد مربوطه را مینویسیم برای فایل snakeمنطق برخورد مار با دیواره ها یا برخورد با خودش و ذخیره امتیاز و اندازه طول مار در فایل را میکنیم و همچمین حرکت مار به اطراف. در فایل food مکان غذا آن را مشخص میکنیم.
- در فایل اصلی ابتدا منو آن را طراحی میکنیم و بعد دیواره های آن را مشخص میکنیم و همچنین سر و بدنه مار و همچنین غذا.
- اگر کاربر خواست که بازی قبل را ادامه دهد کلید b را میزند و امتیاز و طول مار از فایل خوانده میشود و به بازی ادامه میدهد.

Snake.h

```h
#ifndef SNAKE_H
#define SNAKE_H

#include <windows.h>
#include <vector>

#define WIDTH 50
#define HEIGHT 25

using namespace std;

class Snake // difine snake class
{
    private:
        COORD pos;
        int vel;
        char dir;
        int len;
        vector<COORD> body;

    public:
        Snake(COORD pos, int vel , int len ); // snake constroctor

        void grow();
        void move_snake();
        void direction(char dir);

        vector<COORD> get_body();

        bool collided();
        bool eaten(COORD food);

        COORD get_pos();

        void save(int score);

};

#endif // SNAKE_H

```

Food.h

```
#ifndef FOOD_H
#define FOOD_H

#include <windows.h>
#include <cstdio>

#define WIDTH 50
#define HEIGHT 25

class Food // difine food class
{
    private:
        COORD pos;

    public:
        void gen_food();

        COORD get_pos();
};

#endif // FOOD_H

```

Snake.cpp

```cpp
#include "Snake.h"
#include <fstream>
#include <iostream>

Snake::Snake(COORD pos, int vel, int len ) //constroctor
{
    this->pos = pos;
    this->vel = vel;

    dir = 'n';
    this->len = len;

    body.push_back(pos);
}

void Snake::direction(char dir) { this->dir = dir; }
void Snake::grow() { len++; }
COORD Snake::get_pos() { return pos; }

vector<COORD> Snake::get_body() { return body; }

void Snake::move_snake() // difaine move
{
    switch(dir)
    {
        case 'u': pos.Y -= vel; break;
        case 'd': pos.Y += vel; break;
        case 'l': pos.X -= vel; break;
        case 'r': pos.X += vel; break;
    }

    body.push_back(pos);
    if(body.size() > len) body.erase(body.begin());
}

bool Snake::collided() // lose snake with collided to wall or owen
{
    if(pos.X < 1 || pos.X > WIDTH-2 || pos.Y < 1 || pos.Y > HEIGHT-2) return true;

    for(int i = 0; i < len-1; i++)
    {
        if(pos.X == body[i].X && pos.Y == body[i].Y) return true;
    }
    return false;
}

bool Snake::eaten(COORD food) // eat food
{
    if(pos.X == food.X && pos.Y == food.Y) return true;
    return false;
}

void Snake::save( int score ){ // save score player on file

            ofstream outFile("player.txt"); // craet player file
            if (outFile.is_open())
            {
                outFile << score << endl; // cout score in file
                outFile << vel << endl;
                outFile << len << endl;
                outFile.close();
                cout << endl << "Player data has been saved to file." << endl;
            }
            else
            {
                cerr << "Unable to open file for writing." << endl;
            }
}
```

Please install clang or check configuration `clang.executable`

Food.cpp

```cpp
#include "Food.h"

void Food::gen_food() // randoom food
{
    pos.X = (rand() % WIDTH - 3) + 1;
    pos.Y = (rand() % HEIGHT - 3) + 1;
}

COORD Food::get_pos() { return pos; }

```

Main.cpp

main.cpp > main()

```cpp
#include <iostream>
#include <conio.h>
#include <ctime>
#include <fstream>

#include "Snake.h"
#include "Food.h"

#define WIDTH 50
#define HEIGHT 25

using namespace std;

Snake snake({WIDTH / 2, HEIGHT / 2}, 1 ,1);
Food food;

int score;

void gotoxy(int x, int y)
{ // to print in any place you want
    COORD coord;
    coord.X = x;
    coord.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coord);
}

void board()
{
    COORD snake_pos = snake.get_pos();
    COORD food_pos = food.get_pos();

    vector<COORD> snake_body = snake.get_body();

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

    for (int i = 0; i < HEIGHT; i++) // wall game
    {
        cout << "\t\t#";
        for (int j = 0; j < WIDTH-2 ; j++)
        {
            if (i == 0 || i == HEIGHT - 1)
                cout << '#';

            else if (i == snake_pos.Y && j + 1 == snake_pos.X) // symbol head snake
                cout << 'O';
            else if (i == food_pos.Y && j + 1 == food_pos.X) // symbol eat
                cout << 'O';
            else if (i == food_pos.Y && j + 1 == food_pos.X) // symbol eat
                cout << '*';

            else
            {
                bool isBodyPart = false;
                for (int k = 0; k < snake_body.size() - 1; k++)
                {
                    if (i == snake_body[k].Y && j + 1 == snake_body[k].X) // symbol body snak
                    {
                        cout << 'o';
                        isBodyPart = true;
                        break;
                    }
                }

                if (!isBodyPart)
                    cout << ' ';
            }
        }
        cout << "#\n";
    }
}

int main()
{
    gotoxy(0, 2); // menu Snake Game
    cout << "*                *";
    gotoxy(0, 3);
    cout << "*    Snake Game   *";
    gotoxy(0, 4);
    cout << "*                *";
    gotoxy(0, 4);
    cout << "*                *";
    gotoxy(0, 5);
    cout << "*                *";
    gotoxy(0, 6);
    cout << "*                *";
    gotoxy(0, 7);
    cout << "*                *";
    gotoxy(0, 8);
    cout << "*                *";
    gotoxy(0, 9);
    cout << "*                *";
    gotoxy(0, 10);
```

Main.cpp

```
89      cout << "*                  *";
90      gotoxy(0, 10);
91      cout << "*                  *";
92      gotoxy(0, 11);
93      cout << "*                  *";
94      gotoxy(0, 12);
95      cout << "*    press   b     *";
96      gotoxy(0, 13);
97      cout << "*  Go Last Game    *";
98      gotoxy(0, 14);
99      cout << "*                  *";
100     gotoxy(0, 15);
101     cout << "*  Pess 'Space'    *";
102     gotoxy(0, 16);
103     cout << "*    To Start      *";
104     gotoxy(0, 17);
105     cout << "*                  *";
106
107     int select = _getch();
108
109     if (select == 32)
110     {
111         score = 0;
112         srand(time(NULL));
113
114         food.gen_food();
115
116         bool game_over = false;
117
118         while (!game_over)
119         {
120             board();
121
122             if (kbhit())
123             {
124                 switch (getch()) // Kontrol move snake
125                 {
126                 case 'w':
127                     snake.direction('u');
128                     break;
129                 case 'a':
130                     snake.direction('l');
131                     break;
132                 case 's':
133                     snake.direction('d');
```

```
133                     snake.direction('d');
134                     break;
135                 case 'd':
136                     snake.direction('r');
137                     break;
138                 case 'q':
139                     snake.save(score);
140                     exit(1);
141                 }
142             }
143
144             if (snake.collided()) // lose
145                 game_over = true;
146
147             if (snake.eaten(food.get_pos())) // eat food and plus score
148             {
149                 food.gen_food();
150                 snake.grow();
151                 score  +=10 ;
152             }
153
154             snake.move_snake();
155             Sleep(50); // delay in game
156             SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), {0, 0});
157         }
158     }
159     else if (select == 'b')
160     {
161         ifstream inFile("player.txt"); // open file
162         if (inFile.is_open())
163         {
164             inFile >> score; // cin score last game
165         }
166         else
167         {
168             cerr << "Unable to open file for reading." << endl;
169         }
170         srand(time(NULL));
171
172         food.gen_food();
173
174         bool game_over = false;
175
176         while (!game_over)
177         {
```

Main.cpp

```
        while (!game_over)
        {
            board();

            if (kbhit())
            {
                switch (getch()) // cin kebord
                {
                case 'w':
                    snake.direction('u');
                    break;
                case 'a':
                    snake.direction('l');
                    break;
                case 's':
                    snake.direction('d');
                    break;
                case 'd':
                    snake.direction('r');
                    break;
                case 'q':
                    snake.save(score);
                    exit(1);
                }
            }

            if (snake.collided())
                game_over = true;

            if (snake.eaten(food.get_pos()))
            {
                food.gen_food();
                snake.grow();
                score += 10;
            }

            snake.move_snake();
            Sleep(50); // delae in game
            SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), {0, 0});
        }
    }

    return 0;
}

```

App.exe