SAAD Rayen

Jeu de Serpent

Présentation du Projet

Description :

Le jeu du serpent (Snake) consiste à guider un serpent pour qu'il mange des pommes et s'allonge à chaque fois qu'il en mange une. La partie se termine si le serpent touche l'une des parois où se mord la queue.

L'objectif est de programmer ce jeu en langage C en respectant les spécifications suivantes :

- Génération des Pommes : Les pommes sont générées par trois à chaque fois que le serpent les mange.
- Croissance du Serpent : Le serpent s'agrandit après avoir mangé trois pommes.
- Fin de Partie : La partie est arrêtée si le serpent touche l'une des parois ou se mord la queue.
- Sauvegarde/Reprise de Partie : Programmer la possibilité de sauvegarder une partie dans un fichier et pouvoir la reprendre ultérieurement.
- Mode Joueur contre Ordinateur : Implémenter le jeu avec l'ordinateur comme adversaire, avec deux vitesses possibles.
- Bibliothèques Utilisées : Utiliser la bibliothèque Conio.h ou la SDL pour la gestion de l'interface utilisateur.

Implémentation du jeu :

Le jeu du serpent a été implémenté en langage C en respectant les spécifications énoncées. Les fonctionnalités principales incluent la génération de pommes, la croissance du serpent, la gestion de la fin de partie, la sauvegarde/reprise de partie, et le mode joueur contre ordinateur avec deux vitesses différentes.

Technologies Utilisées :

Le jeu a été développé en langage C en utilisant la bibliothèque [mentionne la bibliothèque utilisée, par exemple, Conio.h ou SDL pour la gestion de l'interface utilisateur.

```c
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <time.h>
#include <windows.h>

#define UP 72
#define DOWN 80
#define LEFT 75
#define RIGHT 77
#define ESC 27

int length;
int bend_no;
int len;
char key;
int life;
int score;

void gameover(); // DÈclaration de gameover

void record(); // DÈclaration de record

struct coordinate {
    int x, y;
    int direction;  // Ajout de la direction
};

struct food{
int x,y;
};

typedef struct coordinate coordinate;

coordinate head, bend[500];
food food,food1,food2;

int height = 20, width = 40;

bool VerifPosFoodSnake(int xf ,int yf){

    for(int i=0;i<length;i++){

        if(xf== bend[i].x && yf == bend[i].y)
        {
            return true;
        }
    }
        return false;


};

void GeneratePosFood(food *f) {
    int foodX, foodY;
    do {
        foodX = (rand() % height + height) % height;
        foodY = (rand() % width + width) % width;
    } while (foodX == f->x && foodY == f->y || foodX == food1.x && foodY == food1.y || foodX == food2.x &&
             foodY == food2.y || foodX >= 20 || foodY >= 40 || VerifPosFoodSnake(foodX, foodY));

    f->x = foodX;
    f->y = foodY;
}


void FoodEaten(){
    bend_no++;
    length++;
    if (bend_no >= length) bend_no = 0;
}

void gotoxy(int x, int y) {
    COORD coord;
    coord.X = x;
    coord.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coord);
}

void setup() {
    life = 3;
    length = 3;
    head.x = height / 2;
    head.y = width / 2;
    head.direction = RIGHT;
    srand(time(NULL));
    food.x = rand() % height;
    food.y = rand() % width;

    food1.x = rand() % height;
    food1.y = rand() % width;

    food2.x = rand() % height;
    food2.y = rand() % width;
    score = 0;

}

void draw() {
    system("cls");
    for (int i = 0; i < height; i++) {
        for (int j = 0; j < width; j++) {
            if (i == 0 || i == height - 1 || j == 0 || j == width - 1) {
                printf("#");
            } else {
                if (i == head.x && j == head.y) {
                    printf("@");
                }

                else if (i == food.x && j == food.y || i == food1.x && j == food1.y || i == food2.x && j == food2.y) {


                    if (i == food.x && j == food.y)
                    printf("1");

                    if (i == food1.x && j == food1.y)
                    printf("2");

                    if (i == food2.x && j == food2.y)
                    printf("3");


                } else {
                    int isprint = 0;
                    for (int k = 0; k < length; k++) {
                        if (i == bend[k].x && j == bend[k].y) {
                            printf("o");
                            isprint = 1;
                        }
                    }
                    if (isprint == 0) {
                        printf(" ");
                    }
                }
            }
        }
        printf("\n");
    }
    gotoxy(width + 2, 2);
    printf("Score: %d", score);
    gotoxy(width + 2, 4);
    printf("Life: %d", life);
}

void input() {
    if (_kbhit()) {
        key = _getch();
        switch (key) {
            case UP:
                if (head.direction != DOWN) {
                    head.direction = UP;
                }
                break;
            case DOWN:
                if (head.direction != UP) {
                    head.direction = DOWN;
                }
                break;
            case LEFT:
                if (head.direction != RIGHT) {
                    head.direction = LEFT;
                }
                break;
            case RIGHT:
                if (head.direction != LEFT) {
                    head.direction = RIGHT;
                }
                  break;
              case ESC:
                life = 0;
                break;
        }
    }
}

void logic() {
    Sleep(150);
    coordinate prev = {head.x, head.y};
    coordinate prev2;
    bend[0] = prev;

    for (int i = 1; i <= length; i++) {
        prev2 = bend[i];
        bend[i] = prev;
        prev = prev2;
    }

    switch (head.direction) {
        case UP:
            head.x--;
            break;
        case DOWN:
            head.x++;
            break;
        case LEFT:
            head.y--;
            break;
        case RIGHT:
            head.y++;
            break;
    }

    if (head.x < 0 || head.x >= height - 1 || head.y < 0 || head.y >= width - 1)
    {
        life--;
        if (life == 0) {
            gameover();
        } else {
            head.x = height / 2;
            head.y = width / 2;
            bend_no = 0;
            length = 5;
            for (int i = 0; i < length; i++) {
                bend[i].x = 0;
                bend[i].y = 0;
            }
            system("cls");
        }
    }

    for (int i = 0; i < length; i++) {
        if (head.x == bend[i].x && head.y == bend[i].y) {
            life--;
            if (life == 0) {
                gameover();
            } else {
                head.x = height / 2;
                head.y = width / 2;
                bend_no = 0;
                length = 5;
                for (int j = 0; j < length; j++) {
                    bend[j].x = 0;
                    bend[j].y = 0;
                }
            }
        }
    }

    if (head.x == food.x && head.y == food.y) {
        FoodEaten();
        score += 10;

        food.x = (rand() % height + height) % height;
        food.y = (rand() % width + width) % width;
    }

    if (head.x == food1.x && head.y == food1.y){
        FoodEaten();
        score += 10;

        food1.x = (rand() % height + height) % height;
        food1.y = (rand() % width + width) % width;
    }

     if (head.x == food2.x && head.y == food2.y) {
        FoodEaten();
        score += 10;

        GeneratePosFood(&food2);
    }

     // ...

}
// ...

int main() {
    setup();
    while (life > 0) {
        draw();
        input();
        logic();
    }
    gameover();
    return 0;
}

void gameover() {
    system("cls");
    printf("Game Over!\n");
    printf("Your Score: %d\n", score);
    record();
    exit(0);
}

void record() {
    FILE *file = fopen("record.txt", "a");
    if (file != NULL) {
        char name[20];
        printf("Enter your name: ");
        scanf("%s", name);
        fprintf(file, "Player Name: %s\n", name);
        fprintf(file, "Score: %d\n", score);
              fprintf(file, "Date and Time: ");
        time_t now;
        time(&now);
        fprintf(file, "%s", ctime(&now));
        fclose(file);
    } else {
        printf("Failed to save the record.");
    }
    exit(0); // Terminer le programme aprÈs avoir enregistrÈ le score
}
```