

Exercício Jogo da Serpente

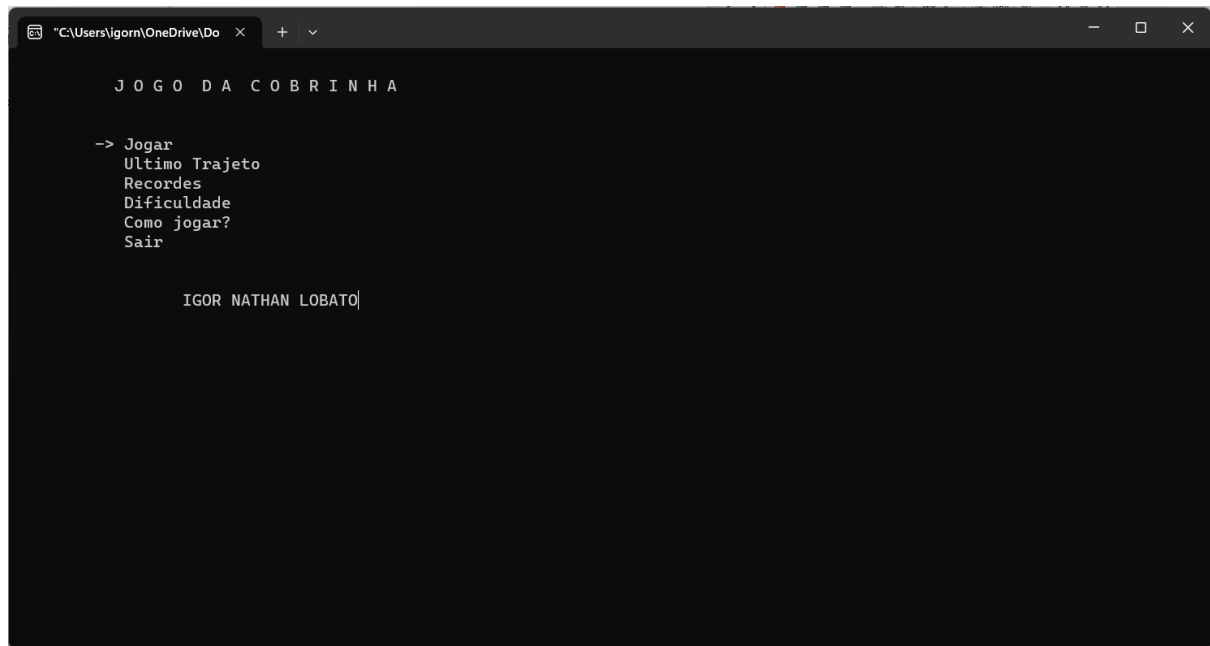
IGOR NATHAN LOBATO GRR20210549

VÍDEO DEMONSTRAÇÃO:

<https://www.youtube.com/watch?v=F7RezOfvfJs>

TELAS:

MENU PRINCIPAL:



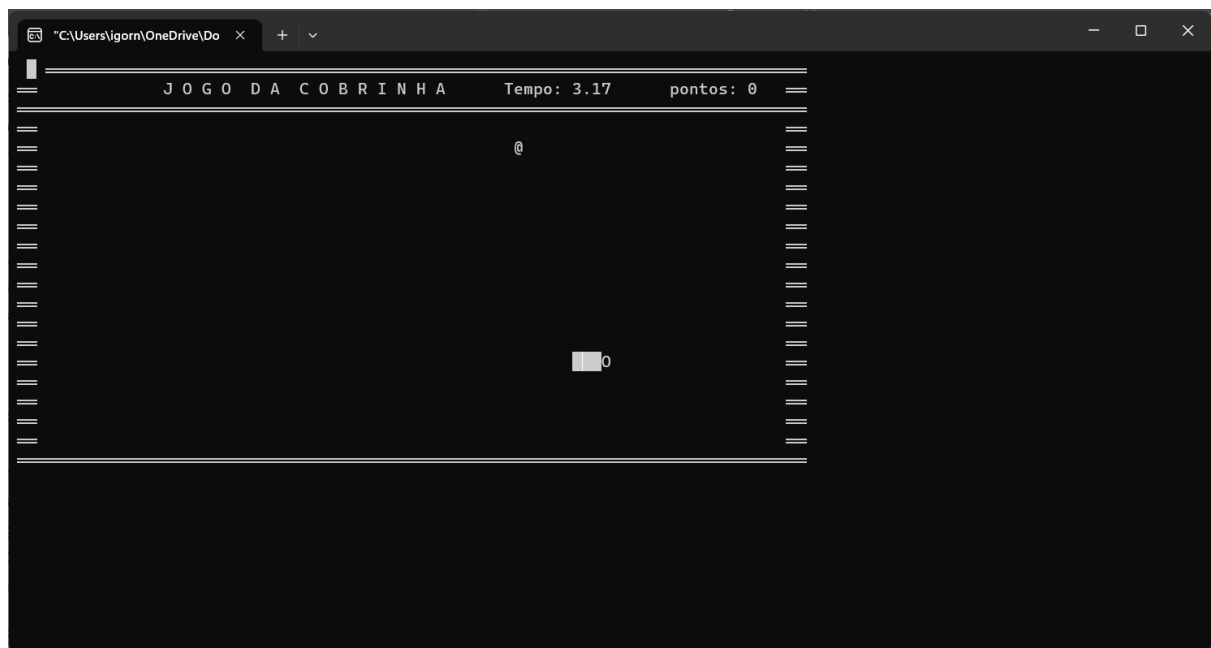
```
"C:\Users\igorn\OneDrive\Do x + v

J O G O   D A   C O B R I N H A

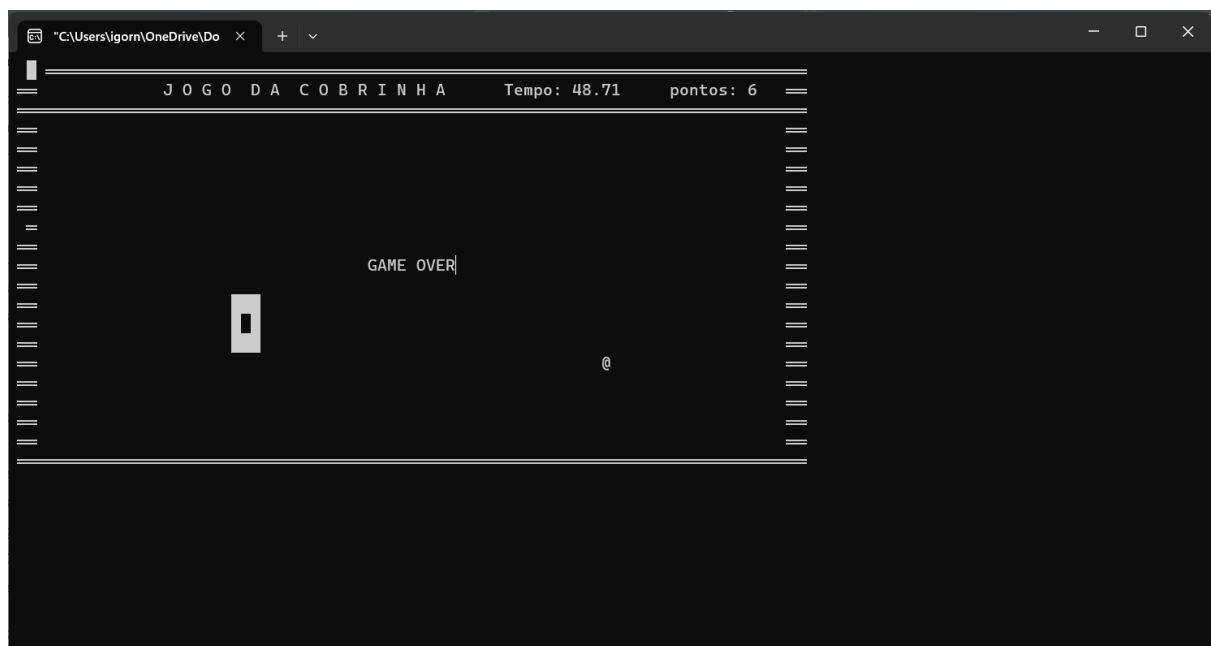
-> Jogar
   Ultimo Trajeto
   Recordes
   Dificuldade
   Como jogar?
   Sair

IGOR NATHAN LOBATO|
```

JOGAR:



GAME OVER:



INSTRUÇÕES:

```
"C:\Users\igorn\OneDrive\Do x + v
J O G O   D A   C O B R I N H A

INSTRUÇÕES:
- O objetivo do jogo é comer 20 macas
no menor tempo possível!
- Se a cobra bater nela mesma o jogo acaba!

CONTROLES:
- Use as setas do teclado ou as letras "wsad"
para alterar a direção da cobra!

Pressione uma tecla para voltar!|
```

```
"C:\Users\igorn\OneDrive\Do x + v
J O G O   D A   C O B R I N H A

Recordes:
1. Nome: igor
   Tempo: 17.58 segundos
2. Nome: igordemostracao
   Tempo: 113.25 segundos
3. Nome: igor
   Tempo: 140.44 segundos
4. Nome: gor
   Tempo: 284.27 segundos

Pressione Enter para voltar!|
```

ALTERAR DIFICULDADE:

```
"C:\Users\igorn\OneDrive\Do x + v

J O G O   D A   C O B R I N H A

Digite qual sera o nivel de dificuldade:
Nivel: [1] Facil [2] Medio [3] Dificil
|
```

DIFICULDADE ALTERADA:

```
"C:\Users\igorn\OneDrive\Do x + v

Selecionado nivel: Dificil
Pressione Enter para voltar!|
```

ENCERRAR JOGO:

```
"C:\Users\igorn\OneDrive\Do x + v

J O G O   D A   C O B R I N H A

Jogar
Ultimo Trajeto
Recordes
Dificuldade
Como jogar?
-> Sair

IGOR NATHAN LOBATO
Process returned 0 (0x0)   execution time : 121.320 s
Press any key to continue.
```

CÓDIGO DO JOGO:

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

#define MAX_RECORDS 5

int largura = 80, altura = 20;
int velocidade = 100;

typedef struct
{
    char nome[50];
    double tempo;
} Recorde;

typedef struct
{
    Recorde recordes[MAX_RECORDS];
    int quantidade;
} PilhaRecordes;

void ordenaRecordes(PilhaRecordes* pilha)
{
    for (int i = 0; i < pilha->quantidade - 1; i++)
    {
        for (int j = 0; j < pilha->quantidade - i - 1; j++)
        {
            if (pilha->recordes[j].tempo > pilha->recordes[j + 1].tempo)
            {
                Recorde temp = pilha->recordes[j];
                pilha->recordes[j] = pilha->recordes[j + 1];
                pilha->recordes[j + 1] = temp;
            }
        }
    }
}

void inicializarPilhaRecordes(PilhaRecordes* pilha)
{
    pilha->quantidade = 0;
}

void carregarRecordes(PilhaRecordes* pilha)
```

```

{
    FILE* arquivo = fopen("recordes.txt", "r");
    if (arquivo == NULL)
    {
        printf("Erro ao abrir o arquivo de recordes.\n");
        return;
    }

    Recorde recorde;

    while (fread(&recorde, sizeof(Recorde), 1, arquivo) == 1)
    {
        if (pilha->quantidade < MAX_RECORDS)
        {
            pilha->recordes[pilha->quantidade] = recorde;
            pilha->quantidade++;
        }
        else
        {
            break;
        }
    }

    fclose(arquivo);
}

void salvarRecordes(PilhaRecordes* pilha)
{
    FILE* arquivo = fopen("recordes.txt", "w");
    if (arquivo == NULL)
    {
        printf("Erro ao abrir o arquivo de recordes.\n");
        return;
    }

    for (int i = 0; i < pilha->quantidade; i++)
    {
        fwrite(&pilha->recordes[i], sizeof(Recorde), 1, arquivo);
    }

    fclose(arquivo);
}

void exibirRecordes(PilhaRecordes* pilha)
{
    printf("\nRecordes:\n");

    if (pilha->quantidade == 0)
    {
        printf("Nenhum recorde encontrado.\n");
    }
    else
    {
        ordenaRecordes(pilha);
        for (int i = 0; i < pilha->quantidade; i++)
        {
            printf("%d. Nome: %s\n", i + 1, pilha->recordes[i].nome);
            printf("    Tempo: %.2f segundos\n", pilha->recordes[i].tempo);
        }
    }

    printf("\n");
}

void salvarPontuacao(double tempo_decorrido, PilhaRecordes* pilha)
{
    char nome[50];

    system("cls");
    printf("Qual o seu nome? ");
    scanf("%s", nome);

    Recorde recorde;
    strcpy(recorde.nome, nome);
    recorde.tempo = tempo_decorrido;

    if (pilha->quantidade < MAX_RECORDS)
    {
        pilha->recordes[pilha->quantidade] = recorde;
        pilha->quantidade++;
    }
    else
    {
        for (int i = MAX_RECORDS - 1; i >= 1; i--)
        {

```

```

        pilha->recordes[i] = pilha->recordes[i - 1];
    }
    pilha->recordes[0] = recorde;
}

salvarRecordes(pilha);

printf("Registro de recorde adicionado com sucesso!\n");
getchar(); // Espera o usuário pressionar Enter para continuar
}

// Definição da estrutura para armazenar as posições (x, y) da cobra
typedef struct {
    int x;
    int y;
} Posicao;

void salvarTrajetoJogo(Posicao trajeto[], int contador, int tamanhoCobra) {
    FILE* arquivo = fopen("trajeto_jogo.txt", "w");
    if (arquivo != NULL) {
        for (int i = 0; i < contador; i++) {
            fprintf(arquivo, "%d,%d,%d\n", trajeto[i].x, trajeto[i].y, tamanhoCobra);
        }
        fclose(arquivo);
    }
}

char menus[6][1000] =
{
    "Jogar",
    "Ultimo Trajeto",
    "Recordes",
    "Dificuldade",
    "Como jogar?",
    "Sair"
};

void posicao(int x, int y) //posiciona o cursor na tela
{
    COORD coord;
    coord.X = x;
    coord.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coord);
}

void mapa(int largura, int altura) //Desenha o mapa
{
    int i, j;

    for(i=0; i<=altura; i++)
    {
        for(j=0; j<=largura; j++)
        {
            if(i==0||i==2||j==0||j==1||i==altura||j==largura||j==largura-1)
            {
                printf("%c", 205);
            }
            else
            {
                printf("%c", ' ');
            }
        }
        printf("\n");
    }

    posicao((largura/2)-25,1);

    printf("%s", "J O G O D A C O B R I N H A");
}

int gameOver(int tam, int x[], int y[] )
{
    int game=0;

    for(int i=3; i<tam; i++)
    {
        if(x[0]==x[i]&& y[0]==y[i])
        {
            game = 1;
            posicao((largura/2)-4, altura/2);
            printf("GAME OVER");
            getchar();
        }
    }
    return game;
}

```

```

}

void snake(int x[100], int y[100], int tam, char direcao) //desenha a cobrinha
{
    posicao(x[1], y[1]); //desenha a cabeca
    printf("%c", 'O');

    for(int i=2; i<tam; i++) //desenha o corpo
    {
        posicao(x[i], y[i]);
        printf("%c", 219);
    }
    for(int i=tam; i>1; i--) //atualiza as posições do corpo
    {
        x[i]=x[i-1];
        y[i]=y[i-1];
    }
}

void refazerTrajetoJogo(int tamanhoTrajeto, Posicao trajeto[]) {
    FILE* arquivo = fopen("trajeto_jogo.txt", "r");
    if (arquivo == NULL) {
        printf("Erro ao abrir o arquivo do trajeto.\n");
        return;
    }

    int tamanhoCobrinha;
    int contador = 0;
    int i = 0;

    while (contador < tamanhoTrajeto && fscanf(arquivo, "%d,%d,%d\n", &trajeto[contador].x, &trajeto[contador].y, &tamanhoCobrinha) ==
        contador++)
    }

    fclose(arquivo);

    while(i < tamanhoTrajeto){
        posicao(trajeto[i].x, &trajeto[i].y); //desenha a cabeca
        printf("%c", 'O');

        Sleep(100);

        posicao(trajeto[i].x, &trajeto[i].y); // apaga o rastro
        printf("%s", " ");
    }

}

// for (int i = 0; i < contador; i++) {
//     snake(trajeto[i].x, trajeto[i].y); // Chame a função 'snake' para imprimir a cobra na posição correta
// }

char tecla(char direcao)
{
    char tecla;

    if(kbhit()) //kbhit verifica se alguma tecla foi pressionada
    {
        tecla = getch();

        switch (tecla)
        {
            case 72:
                if(direcao!='s')
                {
                    direcao = 'n';
                }
                break;

            case 80:
                if(direcao!='n')
                {
                    direcao = 's';
                }
                break;
        }
    }
}

```



```

        }
        break;

    case 75:
        if(direcao!='l')
        {
            direcao = 'o';
        }
        break;

    case 77:
        if(direcao!='o')
        {
            direcao = 'l';
        }
        break;

    case 'w':
        if(direcao!='s')
        {
            direcao = 'n';
        }
        break;

    case 's':
        if(direcao!='n')
        {
            direcao = 's';
        }
        break;

    case 'a':
        if(direcao!='l')
        {
            direcao = 'o';
        }
        break;

    case 'd':
        if(direcao!='o')
        {
            direcao = 'l';
        }
        break;

    }
}
return direcao;
}

void maca(int mx,int my)
{
    posicao(mx,my);
    printf("%c", '@');
}

void pontuacao(int pontos)
{
    posicao(largura-13,1);
    printf("%s%d", "pontos: ", pontos);
}

double calculaTempo(time_t start_time, time_t current_time)
{
    double total;

    total = (double)(current_time - start_time) / CLOCKS_PER_SEC;

    posicao(largura - 30, 1);
    printf("%s%.2f", "Tempo: ", total);

    return total;
}

void exibirMenu(PilhaRecordes* recordes, int tamanhoTrajeto, Posicao trajeto[])
{
    int enter = 0;
    int count;
    int opcao = 0;
    int opcaoAnterior = -1;

    while(1)

```

```

{
    if (opcao != opcaoAnterior)
    {
        system("cls");

        posicao(10,1);

        printf("%s","J O G O D A C O B R I N H A");
        printf("\n\n\n");
        for (count=0; count<6; count++)
        {
            if(opcao==count)
            {
                printf ("\t-> %s\n",menus[count]);
            }
            else
            {
                printf ("\t   %s\n",menus[count]);
            }
        }
        opcaoAnterior = opcao;

        posicao((largura/2)-23,altura-8);
        printf("IGOR NATHAN LOBATO");

    }
    Sleep(100);
    if(kbhit())          //kbhit verifica se alguma tecla foi pressionada
    {
        char tecla = getch();

        switch (tecla)
        {
            case 72:
                if(opcao!=0)
                {
                    opcao --;
                }

                break;

            case 80:
                if(opcao!=5)
                {
                    opcao ++;
                }

                break;

            case 13:

                enter = 1;

                switch(opcao)
                {
                    case 0:
                        system("cls");
                        return;

                        break;

                    case 1:
                        system("cls");

                        refazerTrajetoJogo(tamanhoTrajeto, trajeto);

                        break;

                    case 2:
                        system("cls");
                        posicao(10,1);

                        printf("%s","J O G O D A C O B R I N H A");
                        printf("\n\n\n");

                        exibirRecordes(recordes);

                        posicao(10,18);

                        printf("Pressione Enter para voltar!");
                        getchar();

                        system("cls");
                        exibirMenu(recordes, 100000, trajeto);
                        return;

```

```

        break;

case 3:
    system("cls");

    posicao(10,1);

    printf("%s","J O G O D A C O B R I N H A");

    posicao(10,(altura/2)-6);
    printf("Digite qual sera o nivel de dificuldade: ");
    posicao(10,(altura/2)-4);
    printf("Nivel: [1] Facil [2] Medio [3] Dificil\n");

    char tecla = getch();

    switch (tecla)
    {
    case '1':
        velocidade = 100; // Velocidade fácil
        system("cls");
        posicao(10,(altura/2)-6);
        printf("Selecioneado nivel: Facil");
        posicao(10,(altura/2)-4);
        printf("\Pressione Enter para voltar!");
        getch();

        break;

    case '2':
        velocidade = 50; // Velocidade média
        system("cls");
        posicao(10,(altura/2)-6);
        printf("Selecioneado nivel: Medio");
        posicao(10,(altura/2)-4);
        printf("\Pressione Enter para voltar!");
        getch();

        break;

    case '3':
        velocidade = 25; // Velocidade dificil
        system("cls");
        posicao(10,(altura/2)-6);
        printf("Selecioneado nivel: Dificil");
        posicao(10,(altura/2)-4);
        printf("\Pressione Enter para voltar!");
        getch();

        break;
    }
    system("cls");
    exibirMenu(recordes, 100000, trajeto);

    return 0;

break;

case 4:
    system("cls");

    posicao(10,1);

    printf("%s","J O G O D A C O B R I N H A");

    posicao(10,(altura/2)-6);
    printf("INSTRUcoes:\n\n");
    printf("- O objetivo do jogo e comer 20 macas \nno menor tempo possivel!\n");
    printf("- Se a cobra bater nela mesma o jogo acaba!\n\n");
    printf("CONTROLES: \n\n");
    printf("- Use as setas do teclado ou as letras \"wsad\" \npara alterar a direcao da cobra!\n\n\n");

    printf("\Pressione uma tecla para voltar!");
    getch();

    system("cls");
    exibirMenu(recordes, 100000, trajeto);
    return 0;

```

```

        break;

        case 5:
            exit(0);
            break;

    }

    break;

    }

}

}

}

int main()
{
    int tamanhoTrajeto = 100000; // Defina o tamanho máximo do trajeto aqui
    Posicao trajeto[tamanhoTrajeto];

    int opcao;

    PilhaRecordes recordes;
    inicializarPilhaRecordes(&recordes);
    carregarRecordes(&recordes);

    while(1)
    {

        int x[30], y[30], mx, my, tam = 5;
        char direcao = 'l';
        int pontos = 0, over=0;
        time_t start_time, current_time;
        double tempo_decorrido;
        int contador = 0;

        exibirMenu(&recordes, 100000, trajeto);

        x[0]=30;    //[0] posição antiga
        y[0]=15;

        x[1]=x[0]; //[1] posicao nova
        y[1]=y[0];

        srand(time(NULL));
        mx = (rand() % ((largura-3) - 3)) + 3;
        my = (rand() % ((altura -2) - 4)) + 4;

        mapa(largura, altura); //desenha o mapa
        maca(mx, my);

        start_time = clock();

        while(over==0)
        {
            // Simulação do trajeto da cobrinha (exemplo)
            Posicao posicaoAtual;
            posicaoAtual.x = x[1];
            posicaoAtual.y = y[1];

            // Adicione a posição atual ao trajeto
            trajeto[contador] = posicaoAtual;
            contador++;

            over = gameOver(tam, x, y);

            switch(direcao)
            {
                case 'n':
                    y[1] = y[0]-1;

```

```

        if(y[1]==2)
        {
            y[1] = altura-1;
        }

        snake(x, y, tam, direcao);

        Sleep(velocidade);
        y[0] = y[1];

        break;

    case 's':
        y[1] = y[0]+1;

        if(y[1] == altura)
        {
            y[1] = 3;
        }

        snake(x, y, tam, direcao);

        Sleep(velocidade);
        y[0] = y[1];

        break;

    case 'l':
        x[1] = x[0]+1;
        if(x[1] == largura-2)
        {
            x[1] = 2;
        }

        snake(x, y, tam, direcao);

        x[0] = x[1];
        Sleep(velocidade);

        break;

    case 'o':
        x[1] = x[0]-1;

        if(x[1]==1)
        {
            x[1] = largura-3;
        }

        snake(x, y, tam, direcao);

        Sleep(velocidade);
        x[0] = x[1];

        break;
    }

    direcao = tecla(direcao);

    if(x[1]==mx && y[1]==my) //Verifica se a cobra comeu a maca
    {
        mx = (rand() % ((largura-3) - 3)) + 3;
        my = (rand() % ((altura -2) - 4)) + 4;
        maca(mx,my);

        pontos++;
        tam++;
    }

    if(contador<(100000-1)){
        salvarTrajetoJogo(trajeto, contador, tam);
    }

    posicao(x[tam], y[tam]); // apaga o rastro
    printf("%s", " ");

    pontuacao(pontos);

    current_time = clock();

    tempo_decorrido = calculaTempo(start_time, current_time);

    if(tam==25)
    {

```

```

        over = 1;
        posicao((largura/2)-15, altura/2);
        printf("PARABENS VOCE CONCLUIU O JOGO!");

        posicao((largura/2)-12, (altura/2)+2);
        printf("%.2f", "o seu tempo foi : ", tempo_decorrido);
        getchar();

        salvarPontuacao(tempo_decorrido, &recordes);
    }
}

return 0;
}

```