```
#include <stdio.h> #include <stdlib.h> #include <unistd.h>

#define estacao 51 #define robo 10 #define sujeira 20 #define parede 30 #define vazio 40 #define caminho 80

int map[8][8]; int i, j;

int linhaS; int colunaS;

struct robot{

int x; int y; int linhaR; int colunaR; int contPassos; };

void limpar(){

    system("clear"); }

void mapa(){

struct robot r;

limpar();

printf("\n");

for(i = 0; i < 8; i++){

for(j = 0; j <= 8; j++){

if(j == 8){

printf("|");
```

```
} else if(map[i][j] == estacao){
    printf("| E "); } else if(map[i][j] == robo){
    printf("| @ "); } else if(map[i][j] == caminho || map[i][j] == vazio){
    printf("| "); }else if(map[i][j] == sujeira){
    printf("|***"); } else if(linhaS == -1 && colunaS == -1 && map[i][j] != robo && map[i][j] != sujeira
&& map[i][j] != estacao && map[i][j] != vazio){
map[i][j] = parede; printf("|---"); } else {
    printf("| ", map[i][j]); } } printf("\n"); }
   printf("\n"); }
void jogo(){
struct robot r;
int contSujeira = 0; r.contPassos = 0; int contador = 0;
int a; int vetor[64][2];
mapa();
printf("Digite a linha do robô: "); scanf("%d", & r.linhaR); printf("Digite a coluna do robô: ");
scanf("%d", & r.colunaR);
map[r.linhaR][r.colunaR] = robo;
```

```
do{
mapa(); printf("Quantidade de sujeiras: %d \n", contSujeira); printf("Digite a linha da sujeira: "); scanf("%d", & linhaS); printf("Digite a coluna da sujeira: "); scanf("%d", & colunaS); for(a = 0; a < 8; a++){
map[linhaS][colunaS] = sujeira; vetor[a][a] = sujeira; mapa(); } contador++; contSujeira++; }while(linhaS != -1 && colunaS != -1);
do{
mapa(); printf("Quantidade de sujeiras: %d \n", contSujeira -1); printf("Quantidade de Passos: %d \n", r.contPassos); printf("O robô dará: %d ", contador * 2 - 2 * (100/100)); printf("passos, até limpar completamente o cômodo. \n"); while(contSujeira != 0){
        if(map[r.linhaR][r.colunaR+1] == sujeira && map[r.linhaR][r.colunaR+1] != parede){
map[r.linhaR][r.colunaR] = vazio; map[r.linhaR][r.colunaR+1] = robo; r.colunaR++; contSujeira--; r.contPassos++; break; } else if(map[r.linhaR][r.colunaR-1] == sujeira && map[r.linhaR][r.colunaR-1] != parede){
map[r.linhaR][r.colunaR] = vazio; map[r.linhaR][r.colunaR-1] = robo; r.colunaR--; contSujeira--; r.contPassos++; break;
```

```
} else if(map[r.linhaR+1][r.colunaR] == sujeira && map[r.linhaR+1][r.colunaR] != parede){
map[r.linhaR][r.colunaR] = vazio; map[r.linhaR+1][r.colunaR] = robo; r.linhaR++; r.contPassos++; contSujeira--; break; } else if(map[r.linhaR-1][r.colunaR] == sujeira && map[r.linhaR-1][r.colunaR] != parede){
map[r.linhaR][r.colunaR] = vazio; map[r.linhaR-1][r.colunaR] = robo; r.linhaR--; r.contPassos++; contSujeira--; // break; } // volta pelos lugares vazios else if(map[r.linhaR][r.colunaR-1] == vazio && map[r.linhaR][r.colunaR-1] != parede){
map[r.linhaR][r.colunaR] = vazio; map[r.linhaR][r.colunaR] = caminho; map[r.linhaR][r.colunaR-1] = robo; r.colunaR--; r.contPassos++; break; } else if(map[r.linhaR][r.colunaR+1] == vazio && map[r.linhaR][r.colunaR+1] != parede){
map[r.linhaR][r.colunaR] = vazio; map[r.linhaR][r.colunaR] = caminho; map[r.linhaR][r.colunaR+1] = robo; r.colunaR++; r.contPassos++; break; } else if(map[r.linhaR-1][r.colunaR] == vazio && map[r.linhaR-1][r.colunaR] != parede){
map[r.linhaR][r.colunaR] = vazio; map[r.linhaR][r.colunaR] = caminho; map[r.linhaR-1][r.colunaR] = robo; r.linhaR--;
```

```
r.contPassos++; break; } else if(map[r.linhaR+1][r.colunaR] == vazio && map[r.linhaR+1][r.colunaR] != parede){
map[r.linhaR][r.colunaR] = vazio; map[r.linhaR][r.colunaR] = caminho; map[r.linhaR+1][r.colunaR] = robo; r.linhaR++; r.contPassos++; break; } else {
mapa(); sleep(1); printf("\n - Limpeza concluída! - \n"); printf("\n \\o/ \n\n\n"); sleep(2); return 0; } }
    sleep(1); }while(contSujeira >= 1 && r.contPassos <= (2*contador - 2) * 100/100); system("exit"); }
int main(void){
jogo();
   return 0; }
```