```c
/*************************************************************
 *                                                           *
 * HW07 Q3                                                   *
 * HAZIRLAYAN: HASAN MEN                                     *
 * NUMARASI  : 131044009                                     *
 * TARIH       : 14/04/2015                                  *
 *                                                           *
 * BILGI                                                     *
 * Dosyadan okunun verilere gore 5*5lik bir labirentten      *
 * belli kurallara gore cisik yolunu bulan program           *
 *************************************************************/
#include<stdio.h>
#include<string.h>

#define N_SIZE 5 /* labirent sinirlari */

#define IN_F "table.txt"    /* input dosyamiz */

/* sayisal degiskenlerimiz */
typedef enum{FALSE,TRUE} Bool;
typedef enum{notavailable,available,right_down} Grid_t;

/* fonksiyon prototipleri */
void read_table(FILE *input_file,Grid_t table[][N_SIZE]);
void print_path(char path[][N_SIZE],int n);
Bool find_path(Grid_t table[][N_SIZE],char path[][N_SIZE],
               int size,int location_x,int location_y);
void fill_space(char path[][N_SIZE],int n);

int main()
{

    Grid_t table[N_SIZE][N_SIZE];
    char path[N_SIZE][N_SIZE]={0};
    int loc_x=0;
    int loc_y=0;

    FILE *inF;
    inF=fopen(IN_F,"r");

    /*degiskenlerin sonu */

    if(inF == NULL) /* dosya acilmamasi durumunda hata dondurur */
        printf("%s couldn't opened to read table",IN_F);
    else{
            read_table(inF,table);  /* dosyadan okuma ile table doldurulur*/
            fill_space(path,N_SIZE);/* yol arrayimiz once boslukla doldurulur*/
            /* en uygun yol bulunur ve sonuc ekrana basilir */
            if(find_path(table,path,N_SIZE,loc_x,loc_y))
            {
                printf("I FOUND RIGHT WAY FOLLOW ME :)\n");
                print_path(path,N_SIZE);}
            else
            {   printf("I couldn't find right way :(\n");
                print_path(path,N_SIZE);}
        }
    return 0;
    /* main fonksiyonu sonu */
}

/* Dosyadan okuma yaparak gerekli array doldurulur */
/* FILE * input_file = input dosyamiz -> file pointer */
/* Grid_t table -> degerlerin oturtulacagi array */

void read_table(FILE *input_file,Grid_t table[][N_SIZE])
{
    int i=0,j=0,temp;

    /* 5*5lik tablo arrayi doldurulur*/
    /* degerleri grid_t ye cast edilir*/
    for(i=0;i<N_SIZE;i++)
        for(j=0;j<N_SIZE;j++)
```

```
73          {   fscanf(input_file,"%d",&temp);
74              table[i][j]=(Grid_t)temp;}
75  }
76  /* daha onceden doldurulan table arrayindeki bilgilere gore path arrayini */
77  /* belli kurallar ile dolduran recursive fonksiyon */
78
79  /*Parametreler*/
80  /* Grid_t table->doldurulmus array - input parametre olarak */
81  /* path -> doldurulacak array -> out put array olarak */
82  /* size -> tablolarin satir sayilari */
83  /* location_X - location_y -> labirentin baslangic noktalari */
84  Bool find_path(Grid_t table[][N_SIZE],char path[][N_SIZE],
85                      int size,int location_x,int location_y)
86  {
87      Bool res=TRUE;
88
89      /* maze icindeki ise devam et*/
90      if(location_x<N_SIZE && location_y<N_SIZE )
91      {
92          /* yol kullanima aciksa devam et */
93          /* yol 1ise duz git, 2ise capraz, 0 ise yol kapali */
94          if(table[location_x][location_y]==available)
95          {
96              /* sag ve alt yollarda musait ise yoldan devam et */
97              if(find_path(table,path,N_SIZE,location_x,location_y+1))
98                  path[location_x][location_y]='*';
99
100             if(find_path(table,path,N_SIZE,location_x+1,location_y))
101                 path[location_x][location_y]='*';
102         }
103         else if(table[location_x][location_y]==right_down)
104         {
105             /* yolda 2 varsa capraz hareket et */
106             path[location_x+1][location_y+1]='*';
107             if(find_path(table,path,N_SIZE,location_x+1,location_y+1))
108                 path[location_x][location_y]='*';
109         }
110         else if(table[location_x][location_y]==notavailable)
111         /* yol kapali ise false dondur */
112             res=FALSE;
113
114     }else res=FALSE;
115     /* son cikis noktasii sag en alt ise true dondur */
116     if(location_x==N_SIZE-1 && location_y==N_SIZE-1)
117     {
118         path[location_x][location_y]='*';
119         res=TRUE;
120     }
121     return res;
122 }
123 /* find_path ile doldurulan path dizisini ekrana basar */
124 void print_path(char path[][N_SIZE],int n)
125 {
126     int i,j;
127     for(i=0;i<n;i++)
128     {
129         for(j=0;j<N_SIZE;j++)
130             printf("%c",path[i][j]);
131         printf("\n");
132     }
133 }
134
135 /* yollarin anlasilabilir olmasi icin path dizisin once '|' ile doldurulur*/
136 void fill_space(char path[][N_SIZE],int n)
137 {
138     int i,j;
139     for(i=0;i<n;i++)
140         for(j=0;j<N_SIZE;j++)
141             path[i][j]='|';
142 }
143 /* H07_HASAN_MEN_131044009_part3.c sonu */
```