```
/*********************************************************************************/
/*HW07_part3.c                                                                   */
/*_______________________________                                                 */
/*Written by Mustafa Akilli on April 13, 2015                                    */
/*                                                                               */
/*Description                                                                    */
/*___________                                                                    */
/*  Find a path on a grid maze.                                                  */
/*Inputs:                                                                        */
/*  -Row number.                                                                 */
/*  -Columns number.                                                             */
/*  -Txt file which have grid maze                                               */
/*Outputs:                                                                       */
/*  -A path on a grid maze.                                                      */
/*********************************************************************************/
/*                                                                               */
/*-------------------------------------------------------------------------------*/
/*                          Includes                                             */
#include <stdio.h>
/*-------------------------------------------------------------------------------*/

/*                          Define                                               */
#define COL_SIZE 5
#define ONE 1
#define TWO 2
#define START 0
/*-------------------------------------------------------------------------------*/

typedef enum{notavailable,available,right_down}Grid_t;
typedef enum{not,found}Bool;

void read_table(FILE *input_file, Grid_t table[][COL_SIZE]);
void print_path(char path[][COL_SIZE], int n);
Bool find_path(Grid_t table[][COL_SIZE], char path[][COL_SIZE],
int size, int location_x, int location_y);


int
main(void){

    int k,i;
    char path[COL_SIZE][COL_SIZE];

    Grid_t table[COL_SIZE][COL_SIZE];
    FILE *table_txt;

    table_txt = fopen("table.txt","r");

    read_table(table_txt,table);

    fclose(table_txt);

    find_path(table,path,COL_SIZE,START,START);

    print_path(path,COL_SIZE);

    return 0;
}

/*  An NxN table read from a file called table.txt.                              */
/*  Assign to array.                                                             */
void read_table(FILE *input_file, Grid_t table[][COL_SIZE])
{
    int number,status,i=0,k=0;

    status = fscanf(input_file,"%d",&number);
    table[i][k]=number;

    while(status!=EOF )
    {

        if(k<COL_SIZE-1)
```

```
        {
            ++k;
        }

        else
        {
            if(i<COL_SIZE-1)
            {
                k=0;
                ++i;
            }
        }

        status = fscanf(input_file,"%d",&number);
        table[i][k]=number;
    }
}

/*  An NxN char draw the path on this char array using “*”'s                    */
void print_path(char path[][COL_SIZE], int n)
{
    int i,k;

    for(i=0;i<n;++i)
    {
        for(k=0;k<n;++k)
        {
            if(path[i][k]=='+')
            {
                printf("* ");
            }

            else
            {
                printf("- ");
            }
        }
        printf("\n");
    }
}

/*  A recursive function to draw a path from first coordinate                   */
/*  of the grid to the last coordinate.                                         */
Bool find_path(Grid_t table[][COL_SIZE], char path[][COL_SIZE],
int size, int location_x, int location_y)
{
    int result;

    if((location_x+ONE)==COL_SIZE && (location_y+ONE)==COL_SIZE)
    {

        path[location_x][location_y]='+';
        result=found;
    }

    else
    {

        if(result!=found)
        {

            /*  if(table[location_x+ONE][location_y]==available)                */
            if(table[location_x+ONE][location_y]==available)
            {
                result=find_path(table,path,COL_SIZE,location_x+ONE,location_y);

                if(result==found)
                {
                    path[location_x][location_y]='+';
                }
            }
```

```
        else
        {
            if(result!=found)
            {
                path[location_x+ONE][location_y]='-';
            }

        }

        /*  if(table[location_x+ONE][location_y]==right_down)               */
        if(table[location_x+ONE][location_y]==right_down)
        {

            if(table[location_x+TWO][location_y+ONE]==available)
            {
                result=find_path(table,path,COL_SIZE,location_x+TWO,
                       location_y+ONE);

                if(result==found)
                {
                    path[location_x][location_y]='+';
                    path[location_x+ONE][location_y]='+';
                }
            }
        }

        else
        {
            if(result!=found)
            {
                path[location_x+ONE][location_y]='-';
            }
        }

        /*  if(table[location_x][location_y+ONE]==available)                */
        if(table[location_x][location_y+ONE]==available)
        {
            result=find_path(table,path,COL_SIZE,location_x,location_y+ONE);

            if(result==found)
            {
                path[location_x][location_y]='+';
            }
        }

        else
        {
            if(result!=found)
            {
                path[location_x+1][location_y]='-';
            }

        }

        /*  if(table[location_x][location_y+ONE]==right_down)               */
        if(table[location_x][location_y+ONE]==right_down)
        {
            if(table[location_x+ONE][location_y+TWO]!=notavailable)
            {
                result=find_path(table,path,COL_SIZE,location_x+ONE,
                       location_y+TWO);

                if(result==found)
                {
                    path[location_x][location_y]='+';
                    path[location_x][location_y+ONE]='+';
                }
            }
        }

        else
        {
```

```
                if(result!=found)
                {
                    path[location_x+ONE][location_y]='-';
                }
            }

        }
    }
    return result;
}
/*################################################################################*/
/*                              End of HW07_part3.c                               */
/*################################################################################*/
```