

CSE108**HW05****Part 1 100pts**

In this homework you will write a game called "minesweeper" that player is going to open cells until the game will be terminated by "**win**" and returns the number of moves is made until win when the player finds every "un-mined" cells or by "**loose**" when they hits a mine.

The player will choose a cell either to open or to flag it until the game is terminated. Every time the player made a move the program should print whole grid with current moves, ask the player their next move and get their choice.

If a **closed** cell is chosen; it can be either opened or flagged.

If a **closed_empty** cell is chosen to open; it will be open, its 8 neighbor will be checked for emptiness and **empty neighbors** are also opened.

If a **closed_mined** cell is chosen to open; the game will be terminated by a loose message.

If a **flagged** cell is chosen; it can only be un-flagged.

If **all empty** cells are found; the game will be terminated by a win message.

You will print the grid with 'e' for empty cells; 'f' for flagged cells; '.' for closed cells.

In your code you will use an enumerated data type called "**cell**" as following

```
typedef enum {mined=4,
             empty=1,
             closedEmpty=0,
             closedMined=2,
             flagged=3
            }cell;
```

The grid will be a GRIDSIZE x GRIDSIZE multi-dim array of **cell**.

Example : Assume we have a grid as below (**Initial Grid**), each move and its effect on the grid is colored by a different color.

.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.

Assume player made a choice to open the location of **(0,0)** (**1st Move**)

e	.	.	.
e	e	.	.
.	.	.	.
.	.	.	.

Now assume player wants to flag the location of (0,1) (2nd Move)

e	f	.	.
e	e	.	.
.	.	.	.
.	.	.	.

Now the player wants to open the location of (3,3) (3rd Move)

e	f	.	.
e	e	.	.
.	.	e	e
.	.	e	e

Now the player wants to open the location of (0,3) (4th Move)

e	f	.	e
e	e	.	.
.	.	e	e
.	.	e	e

Now the player wants to open (3,0) (5th Move)

e	f	.	e
e	e	.	.
e	e	e	e
e	e	e	e

Every closed-empty cells are open (number of moves : 5)

A message will be printed "Congratulations! You win the game with 5 moves"
(Here three closed-mined and one flagged-mined cells was not opened.)

Signature

```
void printGrid (cell grid[][GRIDSIZE]);
```

```
int openCell(cell grid[][GRIDSIZE], int x, int y); // return value int result =  
-2 when the cell is not opened with the case of illegal location; or flagged  
cell.
```

```
int flagCell(cell grid[][GRIDSIZE], int x, int y); // return value int result =  
-2 when the cell is not flagged with the case of illegal location; or already  
flagged cell.
```

```
int isEmptyCell(cell grid[][GRIDSIZE], int x, int y); //return value int result=0  
if the cell is not an empty cell and result=1 the cell is an empty cell.
```

```
int isLocationLegal(int x, int y); //return value int result=0 if the location is  
illegal and result=1 if the location is legal(in the grid).
```

```
void initGrid(cell grid[][GRIDSIZE]);
```

HINT

```
initGrid (); is a function to initialize your matrix as an arbitrary initial  
game board with closed-empty and closed-mined cells.
```

Good luck!