

After you add
ONE new
direction, Test it.
Test each time.

Save
before
you test.

This is
recursive,
so it crashes
a lot

Cautionary notes about the Open Method

```
public void open(int i, int j) {  
    if (b[i][j] != 0)  
        return;  
    if (i-1 >= 0 && j-1 >= 0 && show[i-1][j-1] == 0) {  
        show[i - 1][j - 1] = 1;  
        if (b[i - 1][j - 1] == 0)  
            open(i - 1, j - 1);  
    }  
    if (i-1 >= 0 && show[i - 1][j] == 0) {  
        show[i - 1][j] = 1;  
        if (b[i - 1][j] == 0)  
            open(i - 1, j);  
    }  
    //etc for all 8 neighbours  
}
```

Open Method

Up, Left

Guard:
 $i-1 \geq 0$
 $j-1 \geq 0$

$i-1,$
 $j-1$

Up

Guard:
 $i-1 \geq 0$

$i-1, j$

Up, Right

Guard:
 $i-1 \geq 0$
 $j+1 < \text{col}$

$i-1,$
 $j+1$

Left

Guard:
 $j-1 \geq 0$

$i, j-1$

i, j

Right

Guard:
 $j+1 < \text{col}$

$i, j+1$

$i+1,$
 $j-1$

$i+1, j$

$i+1,$
 $j+1$

Down, Left

Guard:
 $i+1 < \text{row}$
 $j-1 \geq 0$

Down

Guard:
 $i+1 < \text{row}$

Down, Right

Guard:
 $i+1 < \text{row}$
 $j+1 < \text{col}$

```
public void open(int i, int j) {
    if (b[i][j] != 0)
        return;
    if (i-1 >= 0 && j-1 >= 0 && show[i-1][j-1] == 0) {
        show[i-1][j-1] = 1;
        if (b[i-1][j-1] == 0)
            open(i-1, j-1);
    }
    if (i-1 >= 0 && show[i-1][j] == 0) {
        show[i-1][j] = 1;
        if (b[i-1][j] == 0)
            open(i-1, j);
    }
    //etc for all 8 neighbours
}
```

Call the Open
Method in the last
else of the
ActionPerformed

```
//TO DO: Fill this comment in
```

```
else
```

```
{
```

```
    int n = Integer.parseInt (e.getActionCommand ());
```

```
    int i = n / col;
```

```
    int j = n % col;
```

```
    open (i, j);
```

```
    show [i] [j] = 1;
```

```
}
```