

## My Project

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>File Index</b>	<b>1</b>
1.1	File List	1
<b>2</b>	<b>File Documentation</b>	<b>3</b>
2.1	build/CMakeFiles/3.17.2/CompilerIdC/CMakeCCompilerId.c File Reference	3
2.1.1	Macro Definition Documentation	3
2.1.1.1	ARCHITECTURE_ID	3
2.1.1.2	C_DIALECT	4
2.1.1.3	COMPILER_ID	4
2.1.1.4	DEC	4
2.1.1.5	HEX	4
2.1.1.6	PLATFORM_ID	4
2.1.1.7	STRINGIFY	5
2.1.1.8	STRINGIFY_HELPER	5
2.1.2	Function Documentation	5
2.1.2.1	main()	5
2.1.3	Variable Documentation	5
2.1.3.1	info_arch	5
2.1.3.2	info_compiler	6
2.1.3.3	info_language_dialect_default	6
2.1.3.4	info_platform	6
2.2	cunit_tests.c File Reference	6
2.2.1	Macro Definition Documentation	7
2.2.1.1	MAX	7

2.2.2	Function Documentation	7
2.2.2.1	getBottom()	7
2.2.2.2	getLeft()	8
2.2.2.3	getLeftBottom()	9
2.2.2.4	getLeftTop()	9
2.2.2.5	getRight()	10
2.2.2.6	getRightBottom()	10
2.2.2.7	getRightTop()	11
2.2.2.8	getTop()	11
2.2.2.9	main()	12
2.2.2.10	test_point_location()	12
2.2.3	Variable Documentation	13
2.2.3.1	array	13
2.2.3.2	epsilon	14
2.3	main.c File Reference	14
2.3.1	Detailed Description	14
2.4	mylib/copy3.c File Reference	15
2.4.1	Function Documentation	15
2.4.1.1	decide_mode()	15
2.4.1.2	getBottom()	17
2.4.1.3	getLeft()	18
2.4.1.4	getLeftBottom()	19
2.4.1.5	getLeftTop()	19
2.4.1.6	getRight()	20
2.4.1.7	getRightBottom()	21
2.4.1.8	getRightTop()	21
2.4.1.9	getTop()	22
2.5	mylib/header.h File Reference	23
2.5.1	Macro Definition Documentation	24
2.5.1.1	MAX	24
2.5.2	Function Documentation	24
2.5.2.1	decide_mode()	24
2.5.2.2	getBottom()	25
2.5.2.3	getLeft()	26
2.5.2.4	getLeftBottom()	27
2.5.2.5	getLeftTop()	27
2.5.2.6	getRight()	28
2.5.2.7	getRightBottom()	29
2.5.2.8	getRightTop()	29
2.5.2.9	getTop()	30

# Chapter 1

## File Index

### 1.1 File List

Here is a list of all files with brief descriptions:

<a href="#">cunit_tests.c</a> . . . . .	6
<a href="#">main.c</a>	
Main function Main file of Conway's Game Of Life project . . . . .	14
build/CMakeFiles/3.17.2/CompilerIdC/ <a href="#">CMakeCCompilerId.c</a> . . . . .	3
mylib/ <a href="#">copy3.c</a> . . . . .	15
mylib/ <a href="#">header.h</a> . . . . .	23



## Chapter 2

# File Documentation

## 2.1 build/CMakeFiles/3.17.2/CompilerIdC/CMakeCCompilerId.c File Reference

### Macros

- `#define COMPILER_ID ""`
- `#define STRINGIFY_HELPER(X) #X`
- `#define STRINGIFY(X) STRINGIFY_HELPER(X)`
- `#define PLATFORM_ID`
- `#define ARCHITECTURE_ID`
- `#define DEC(n)`
- `#define HEX(n)`
- `#define C_DIALECT`

### Functions

- `int main (int argc, char *argv[ ])`

### Variables

- `char const * info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"`
- `char const * info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"`
- `char const * info_arch = "INFO" ":" "arch[" ARCHITECTURE_ID "]"`
- `const char * info_language_dialect_default`

### 2.1.1 Macro Definition Documentation

#### 2.1.1.1 ARCHITECTURE\_ID

```
#define ARCHITECTURE_ID
```

### 2.1.1.2 C\_DIALECT

```
#define C_DIALECT
```

### 2.1.1.3 COMPILER\_ID

```
#define COMPILER_ID ""
```

### 2.1.1.4 DEC

```
#define DEC(  
    n )
```

**Value:**

```
('0' + (((n) / 10000000) % 10)), \
('0' + (((n) / 1000000) % 10)), \
('0' + (((n) / 100000) % 10)), \
('0' + (((n) / 10000) % 10)), \
('0' + (((n) / 1000) % 10)), \
('0' + (((n) / 100) % 10)), \
('0' + (((n) / 10) % 10)), \
('0' + ((n) % 10))
```

### 2.1.1.5 HEX

```
#define HEX(  
    n )
```

**Value:**

```
('0' + ((n) >> 28 & 0xF)), \
('0' + ((n) >> 24 & 0xF)), \
('0' + ((n) >> 20 & 0xF)), \
('0' + ((n) >> 16 & 0xF)), \
('0' + ((n) >> 12 & 0xF)), \
('0' + ((n) >> 8 & 0xF)), \
('0' + ((n) >> 4 & 0xF)), \
('0' + ((n) & 0xF))
```

### 2.1.1.6 PLATFORM\_ID

```
#define PLATFORM_ID
```



### 2.1.1.7 STRINGIFY

```
#define STRINGIFY(  
    X ) STRINGIFY\_HELPER(X)
```

### 2.1.1.8 STRINGIFY\_HELPER

```
#define STRINGIFY_HELPER(  
    X ) #X
```

## 2.1.2 Function Documentation

### 2.1.2.1 main()

```
int main (  
    int argc,  
    char * argv[] )  
  
647 {  
648     int require = 0;  
649     require += info\_compiler[argc];  
650     require += info\_platform[argc];  
651     require += info\_arch[argc];  
652     #ifdef COMPILER_VERSION_MAJOR  
653     require += info\_version[argc];  
654     #endif  
655     #ifdef COMPILER_VERSION_INTERNAL  
656     require += info\_version\_internal[argc];  
657     #endif  
658     #ifdef SIMULATE_ID  
659     require += info\_simulate[argc];  
660     #endif  
661     #ifdef SIMULATE_VERSION_MAJOR  
662     require += info\_simulate\_version[argc];  
663     #endif  
664     #if defined(__CRAYXE) || defined(__CRAYXC)  
665     require += info\_cray[argc];  
666     #endif  
667     require += info\_language\_dialect\_default[argc];  
668     (void)argv;  
669     return require;  
670 }
```

## 2.1.3 Variable Documentation

### 2.1.3.1 info\_arch

```
char const* info\_arch = "INFO" ":" "arch[" ARCHITECTURE\_ID "]"
```

### 2.1.3.2 info\_compiler

```
char const* info_compiler = "INFO" ":" "compiler[" COMPILER_ID "]"
```

### 2.1.3.3 info\_language\_dialect\_default

```
const char* info_language_dialect_default
```

**Initial value:**

```
=  
"INFO" ":" "dialect_default[" C_DIALECT "]"
```

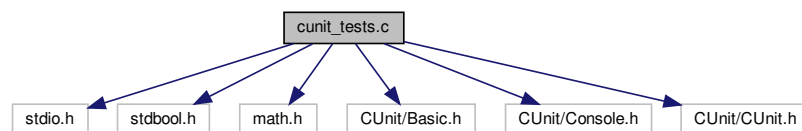
### 2.1.3.4 info\_platform

```
char const* info_platform = "INFO" ":" "platform[" PLATFORM_ID "]"
```

## 2.2 cunit\_tests.c File Reference

```
#include <stdio.h>  
#include <stdbool.h>  
#include <math.h>  
#include <CUnit/Basic.h>  
#include <CUnit/Console.h>  
#include <CUnit/CUnit.h>
```

Include dependency graph for cunit\_tests.c:



## Macros

- `#define` `MAX` 5

## Functions

- int [getTop](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getBottom](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getLeft](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getRight](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getLeftTop](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getRightTop](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getLeftBottom](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getRightBottom](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- void [test\\_point\\_location](#) (void)
- int [main](#) ()

## Variables

- static const double [epsilon](#) =0.000000001
- int [array](#) [[MAX](#)][[MAX](#)]

## 2.2.1 Macro Definition Documentation

### 2.2.1.1 MAX

```
#define MAX 5
```

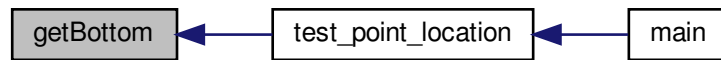
## 2.2.2 Function Documentation

### 2.2.2.1 getBottom()

```
int getBottom (
    int i,
    int j,
    int arr[MAX][MAX] )
```

```
41                                     {
42
43     int y;
44     if (i == MAX - 1)
45     {
46         y = arr[0][j];
47     }
48     else
49     {
50         y = arr[i+1][j];
51     }
52
53     return y;
54
55 }
```

Here is the caller graph for this function:

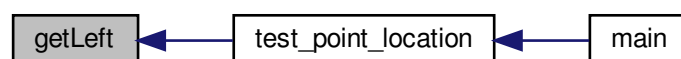


#### 2.2.2.2 `getLeft()`

```
int getLeft (  
    int i,  
    int j,  
    int arr[MAX][MAX] )
```

```
56                                     {  
57  
58     int y;  
59     if (j == 0)  
60     {  
61         y = arr[i][MAX - 1];  
62     }  
63     else  
64     {  
65         y = arr[i][j-1];  
66     }  
67     return y;  
68  
69  
70 }
```

Here is the caller graph for this function:



### 2.2.2.3 getLeftBottom()

```
int getLeftBottom (
    int i,
    int j,
    int arr[MAX][MAX] )

{
    int y;
    if(i == MAX - 1 && j != 0)
    {
        y = arr[0][j-1];
    }
    else if(i != MAX - 1 && j == 0)
    {
        y = arr[i+1][MAX-1];
    }
    else if(i == MAX - 1 && j == 0)
    {
        y = arr[0][MAX-1];
    }
    else
    {
        y = arr[i+1][j-1];
    }
    return y;
}
```

### 2.2.2.4 getLeftTop()

```
int getLeftTop (
    int i,
    int j,
    int arr[MAX][MAX] )

{
    int y;
    if(i == 0 && j != 0)
    {
        y = arr[MAX-1][j-1];
    }
    else if(i != 0 && j == 0)
    {
        y = arr[i-1][MAX-1];
    }
    else if(i == 0 && j == 0)
    {
        y = arr[MAX-1][MAX-1];
    }
    else
    {
        y = arr[i-1][j-1];
    }
    return y;
}
```

### 2.2.2.5 getRight()

```
int getRight (
    int i,
    int j,
    int arr[MAX][MAX] )
```

```
71                                     {
72
73     int y;
74     if(j == MAX - 1)
75     {
76         y = arr[i][0];
77     }
78     else
79     {
80         y = arr[i][j+1];
81     }
82
83     return y;
84
85 }
```

### 2.2.2.6 getRightBottom()

```
int getRightBottom (
    int i,
    int j,
    int arr[MAX][MAX] )
```

```
155                                     {
156
157     int y;
158     if(i == MAX - 1 && j != MAX - 1)
159     {
160         y = arr[0][j+1];
161     }
162     else if(i != MAX - 1 && j == MAX - 1)
163     {
164         y = arr[i+1][0];
165     }
166     else if(i == MAX - 1 && j == MAX - 1)
167     {
168         y = arr[0][0];
169     }
170     else
171     {
172         y = arr[i+1][j+1];
173     }
174
175     return y;
176
177 }
```

## 2.2.2.7 getRightTop()

```
int getRightTop (
    int i,
    int j,
    int arr[MAX][MAX] )

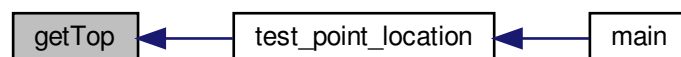
110                                     {
111
112     int y;
113     if(i == 0 && j != MAX - 1)
114     {
115         y = arr[MAX-1][j+1];
116     }
117     else if(i != 0 && j == MAX - 1)
118     {
119         y = arr[i-1][0];
120     }
121     else if(i == 0 && j == MAX - 1)
122     {
123         y = arr[MAX-1][0];
124     }
125     else
126     {
127         y = arr[i-1][j+1];
128     }
129
130     return y;
131 }
```

## 2.2.2.8 getTop()

```
int getTop (
    int i,
    int j,
    int arr[MAX][MAX] )

26                                     {
27
28     int y;
29     if(i == 0)
30     {
31         y = arr[MAX-1][j];
32     }
33     else
34     {
35         y = arr[i-1][j];
36     }
37
38     return y;
39
40 }
```

Here is the caller graph for this function:



### 2.2.2.9 main()

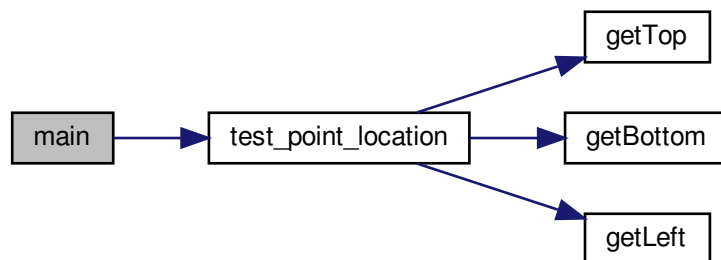
```
main ( )
```

fill new array with values of array to use it in future

```

212     {
213     if(CUE_SUCCESS !=CU_initialize_registry())
214         return CU_get_error();
215
216     CU_pSuite pS1=CU_add_suite("Testing points",NULL,NULL);
217
218     CU_ADD_TEST(pS1,test_point_location);
219
220     CU_basic_set_mode(CU_BRM_VERBOSE);
221
222     CU_basic_run_tests();
223     return 0;
224 }
```

Here is the call graph for this function:



### 2.2.2.10 test\_point\_location()

```

test_point_location (
    void )
```

For checking if circular table is working well or no check some of critical points

#### Warning

it mustn't give error bc the left of the most left value is most right one  
left of the most left must return to right

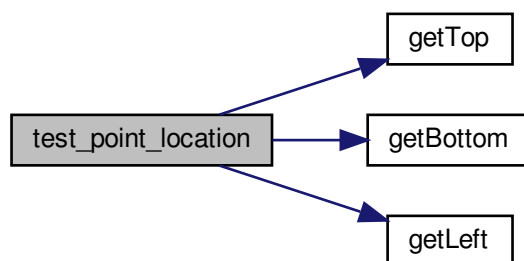


```

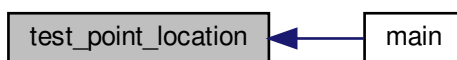
184         {
185
186     CU_ASSERT_DOUBLE_EQUAL(getTop(2, 1, array), 0, epsilon);
187
188     CU_ASSERT_DOUBLE_EQUAL(getTop(1, 2, array), 1, epsilon);
189
190     CU_ASSERT_DOUBLE_EQUAL(getBottom(2, 1, array), 0, epsilon);
191
192     CU_ASSERT_DOUBLE_EQUAL(getLeft(1, 1, array), 0, epsilon);
193
194         CU_ASSERT_DOUBLE_EQUAL(getLeft(1, 4, array), 0, epsilon);
195
196     CU_ASSERT_DOUBLE_EQUAL(getTop(0, 2, array), 0, epsilon);
197
198     CU_ASSERT_DOUBLE_EQUAL(getLeft(3, 0, array), 0, epsilon);
206 CU_ASSERT_DOUBLE_EQUAL(getBottom(4, 2, array), 1, epsilon);
207
208
209
210 }

```

Here is the call graph for this function:



Here is the caller graph for this function:



## 2.2.3 Variable Documentation

### 2.2.3.1 array

```
int array[MAX][MAX]
```

**Initial value:**

```
= {
    {0,0,1,0,0},
    {0,0,1,0,0},
    {0,1,1,1,0},
    {0,0,1,0,0},
    {0,0,0,0,0},
}
```

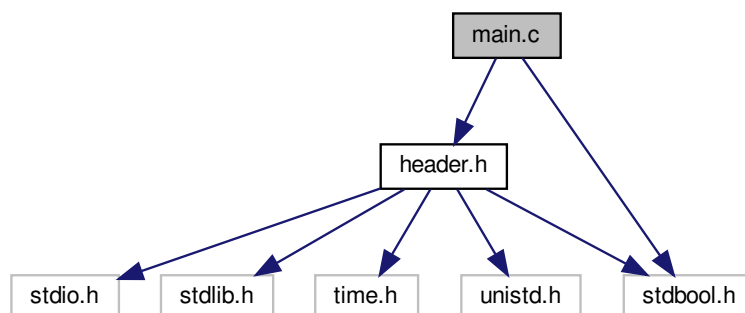
### 2.2.3.2 epsilon

```
const double epsilon =0.000000001 [static]
```

## 2.3 main.c File Reference

main function Main file of Conway's Game Of Life project

```
#include <header.h>
#include <stdbool.h>
Include dependency graph for main.c:
```



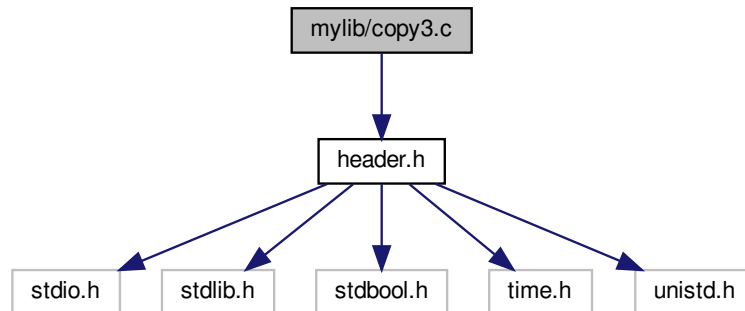
### 2.3.1 Detailed Description

main function Main file of Conway's Game Of Life project

## 2.4 mylib/copy3.c File Reference

```
#include "header.h"
```

Include dependency graph for copy3.c:



### Functions

- int [getTop](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getBottom](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getRight](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getLeft](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getLeftTop](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getRightTop](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getLeftBottom](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- int [getRightBottom](#) (int i, int j, int arr[[MAX](#)][[MAX](#)])
- bool [decide\\_mode](#) (int i, int j, int array[[MAX](#)][[MAX](#)])

### 2.4.1 Function Documentation

#### 2.4.1.1 [decide\\_mode\(\)](#)

```

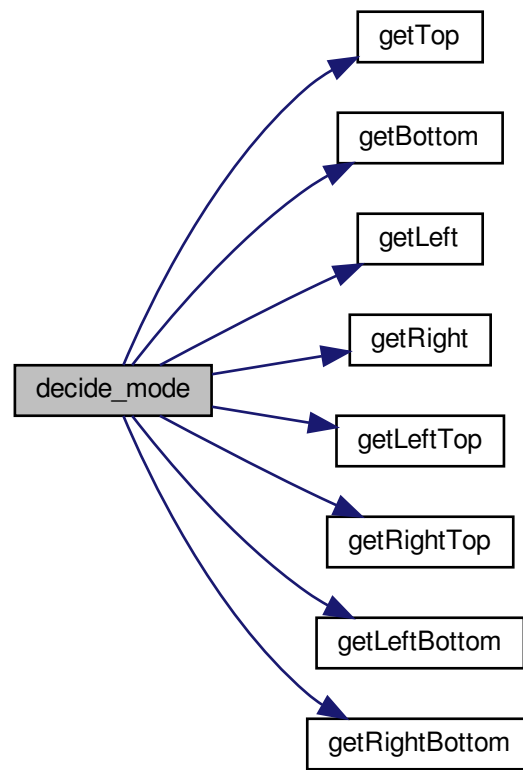
bool decide_mode (
    int i,
    int j,
    int array[MAX][MAX] )

163 {
164     int counter = 0;
165
166     // ----- //
167
168     int top = getTop(i, j, array);
169     counter = counter + top;
170
171     int bottom = getBottom(i, j, array);

```

```
172     counter = counter + bottom;
173
174     int left = getLeft( i, j, array);
175     counter = counter + left;
176
177     int right = getRight(i, j, array);
178     counter = counter + right;
179
180     // ----- //
181
182     int ltop = getLeftTop(i, j, array);
183     counter = counter + ltop;
184
185     int rtop = getRightTop(i, j, array);
186     counter = counter + rtop;
187
188     int lbottom = getLeftBottom(i, j, array);
189     counter = counter + lbottom;
190
191     int rbottom = getRightBottom(i, j, array);
192     counter = counter + rbottom;
193
194     // ----- //
195
196
197     if(array[i][j] == 0)
198     {
199         if(counter == 3)
200         {
201             //printf("%d %d --- ", i, j);
202             //printf("live \n");
203             array[i][j] = 1;
204             return true;
205         }
206     }
207     else if(array[i][j] == 1)
208     {
209         if(counter != 2 && counter != 3)
210         {
211             //printf("%d %d --- ", i, j);
212             //printf("dead \n");
213             array[i][j] = 0;
214             return true;
215         }
216     }
217
218     return false;
219 }
```

Here is the call graph for this function:



### 2.4.1.2 getBottom()

```

int getBottom (
    int i,
    int j,
    int arr[MAX][MAX] )

```

```

19                                     { //for reaching down of given coordinate
20
21     int y;
22     if (i == MAX - 1)
23     {
24         y = arr[0][j];
25     }
26     else
27     {
28         y = arr[i+1][j];
29     }
30
31     return y;
32
33 }

```

Here is the caller graph for this function:



#### 2.4.1.3 getLeft()

```
int getLeft (
    int i,
    int j,
    int arr[MAX][MAX] )
```

```
51                                     ///for reaching left of given coordinate
52
53     int y;
54     if (j == 0)
55     {
56         y = arr[i][MAX - 1];
57     }
58     else
59     {
60         y = arr[i][j-1];
61     }
62
63     return y;
64
65 }
```

Here is the caller graph for this function:



## 2.4.1.4 getLeftBottom()

```

int getLeftBottom (
    int i,
    int j,
    int arr[MAX][MAX] )

113                                     { //for reaching left top of given coordinate
114
115     int y;
116     if(i == MAX - 1 && j != 0)
117     {
118         y = arr[0][j-1];
119     }
120     else if(i != MAX - 1 && j == 0)
121     {
122         y = arr[i+1][MAX-1];
123     }
124     else if(i == MAX - 1 && j == 0)
125     {
126         y = arr[0][MAX-1];
127     }
128     else
129     {
130         y = arr[i+1][j-1];
131     }
132
133     return y;
134 }

```

Here is the caller graph for this function:



## 2.4.1.5 getLeftTop()

```

int getLeftTop (
    int i,
    int j,
    int arr[MAX][MAX] )

68                                     { //for reaching left top of given coordinate
69
70     int y;
71     if(i == 0 && j != 0)
72     {
73         y = arr[MAX-1][j-1];
74     }
75     else if(i != 0 && j == 0)
76     {
77         y = arr[i-1][MAX-1];
78     }
79     else if(i == 0 && j == 0)
80     {

```

```
81     y = arr[MAX-1][MAX-1];
82 }
83 else
84 {
85     y = arr[i-1][j-1];
86 }
87
88 return y;
89
90 }
```

Here is the caller graph for this function:



#### 2.4.1.6 getRight()

```
int getRight (
    int i,
    int j,
    int arr[MAX][MAX] )

36                                     { //for reaching right of given coordinate
37
38     int y;
39     if(j == MAX - 1)
40     {
41         y = arr[i][0];
42     }
43     else
44     {
45         y = arr[i][j+1];
46     }
47
48     return y;
49
50 }
```

Here is the caller graph for this function:





## 2.4.1.7 getRightBottom()

```

int getRightBottom (
    int i,
    int j,
    int arr[MAX][MAX] )

136                                     ///for reaching right bottom of given coordinate
137
138     int y;
139     if(i == MAX - 1 && j != MAX - 1)
140     {
141         y = arr[0][j+1];
142     }
143     else if(i != MAX - 1 && j == MAX - 1)
144     {
145         y = arr[i+1][0];
146     }
147     else if(i == MAX - 1 && j == MAX - 1)
148     {
149         y = arr[0][0];
150     }
151     else
152     {
153         y = arr[i+1][j+1];
154     }
155
156     return y;
157 }
158 }

```

Here is the caller graph for this function:



## 2.4.1.8 getRightTop()

```

int getRightTop (
    int i,
    int j,
    int arr[MAX][MAX] )

91                                     ///for reaching right top of given coordinate
92
93     int y;
94     if(i == 0 && j != MAX - 1)
95     {
96         y = arr[MAX-1][j+1];
97     }
98     else if(i != 0 && j == MAX - 1)
99     {
100         y = arr[i-1][0];
101     }
102     else if(i == 0 && j == MAX - 1)

```

```
103     {
104         y = arr[MAX-1][0];
105     }
106     else
107     {
108         y = arr[i-1][j+1];
109     }
110
111     return y;
112 }
```

Here is the caller graph for this function:



#### 2.4.1.9 getTop()

```
int getTop (
    int i,
    int j,
    int arr[MAX][MAX] )

4                                     ///for reaching up of given coordinate
5
6     int y;
7     if(i == 0)
8     {
9         y = arr[MAX-1][j];
10    }
11    else
12    {
13        y = arr[i-1][j];
14    }
15
16    return y;
17
18 }
```

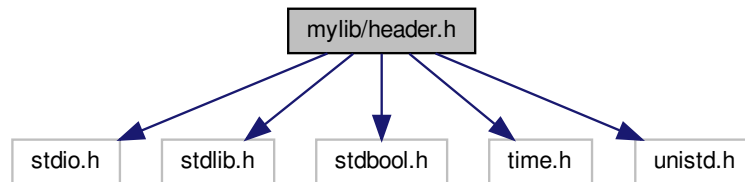
Here is the caller graph for this function:



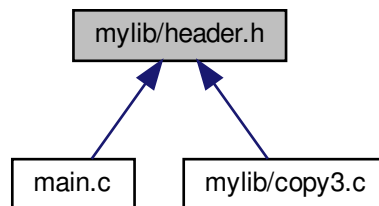
## 2.5 mylib/header.h File Reference

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

Include dependency graph for header.h:



This graph shows which files directly or indirectly include this file:



### Macros

- `#define MAX 5`

### Functions

- `int getTop (int i, int j, int arr[MAX][MAX])`
- `int getBottom (int i, int j, int arr[MAX][MAX])`
- `int getRight (int i, int j, int arr[MAX][MAX])`
- `int getLeft (int i, int j, int arr[MAX][MAX])`
- `int getLeftTop (int i, int j, int arr[MAX][MAX])`
- `int getRightTop (int i, int j, int arr[MAX][MAX])`
- `int getLeftBottom (int i, int j, int arr[MAX][MAX])`
- `int getRightBottom (int i, int j, int arr[MAX][MAX])`
- `bool decide_mode (int i, int j, int array[MAX][MAX])`

## 2.5.1 Macro Definition Documentation

### 2.5.1.1 MAX

```
#define MAX 5
```

## 2.5.2 Function Documentation

### 2.5.2.1 decide\_mode()

```
bool decide_mode (
    int i,
    int j,
    int array[MAX][MAX] )

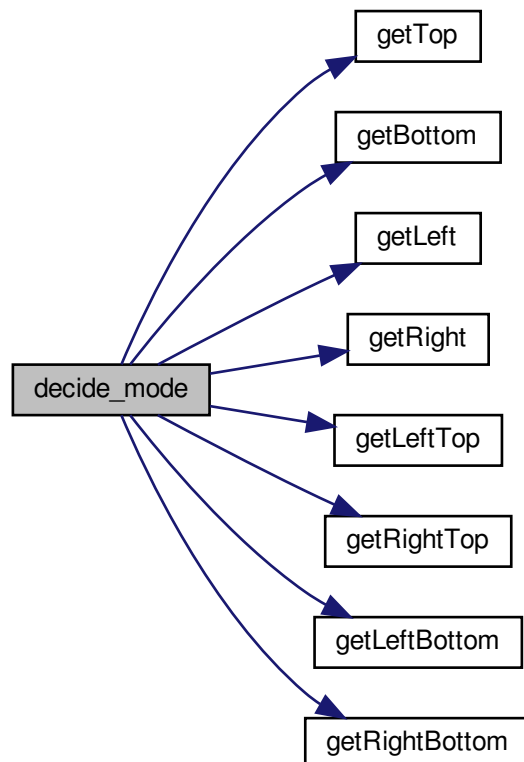
163 {
164     int counter = 0;
165
166     // ----- //
167
168     int top = getTop(i, j, array);
169     counter = counter + top;
170
171     int bottom = getBottom(i, j, array);
172     counter = counter + bottom;
173
174     int left = getLeft(i, j, array);
175     counter = counter + left;
176
177     int right = getRight(i, j, array);
178     counter = counter + right;
179
180     // ----- //
181
182     int ltop = getLeftTop(i, j, array);
183     counter = counter + ltop;
184
185     int rtop = getRightTop(i, j, array);
186     counter = counter + rtop;
187
188     int lbottom = getLeftBottom(i, j, array);
189     counter = counter + lbottom;
190
191     int rbottom = getRightBottom(i, j, array);
192     counter = counter + rbottom;
193
194     // ----- //
195
196     if(array[i][j] == 0)
197     {
198         if(counter == 3)
199         {
200             //printf("%d %d --- ", i, j);
201             //printf("live \n");
202             array[i][j] = 1;
203             return true;
204         }
205     }
206 }
207 else if(array[i][j] == 1)
208 {
209     if(counter != 2 && counter != 3)
210     {
211         //printf("%d %d --- ", i, j);
```

```

212         //printf("dead \n");
213         array[i][j] = 0;
214         return true;
215     }
216 }
217
218 return false;
219 }

```

Here is the call graph for this function:



### 2.5.2.2 getBottom()

```

int getBottom (
    int i,
    int j,
    int arr[MAX][MAX] )

```

```

41     {
42
43     int y;
44     if (i == MAX - 1)
45     {
46         y = arr[0][j];

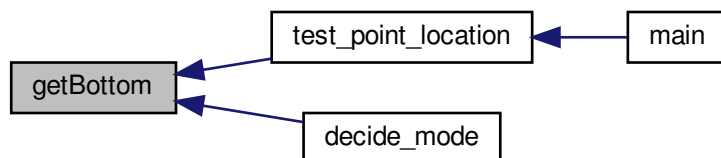
```

```

47     }
48     else
49     {
50         y = arr[i+1][j];
51     }
52
53     return y;
54
55 }

```

Here is the caller graph for this function:



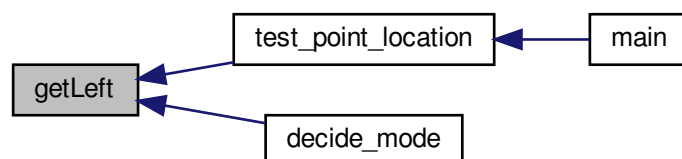
### 2.5.2.3 getLeft()

```

int getLeft (
    int i,
    int j,
    int arr[MAX][MAX] )
{
56
57
58     int y;
59     if (j == 0)
60     {
61         y = arr[i][MAX - 1];
62     }
63     else
64     {
65         y = arr[i][j-1];
66     }
67
68     return y;
69
70 }

```

Here is the caller graph for this function:



## 2.5.2.4 getLeftBottom()

```

int getLeftBottom (
    int i,
    int j,
    int arr[MAX][MAX] )

132                                     {
133
134     int y;
135     if(i == MAX - 1 && j != 0)
136     {
137         y = arr[0][j-1];
138     }
139     else if(i != MAX - 1 && j == 0)
140     {
141         y = arr[i+1][MAX-1];
142     }
143     else if(i == MAX - 1 && j == 0)
144     {
145         y = arr[0][MAX-1];
146     }
147     else
148     {
149         y = arr[i+1][j-1];
150     }
151
152     return y;
153 }

```

Here is the caller graph for this function:



## 2.5.2.5 getLeftTop()

```

int getLeftTop (
    int i,
    int j,
    int arr[MAX][MAX] )

87                                     {
88
89     int y;
90     if(i == 0 && j != 0)
91     {
92         y = arr[MAX-1][j-1];
93     }
94     else if(i != 0 && j == 0)
95     {

```

```

96     y = arr[i-1][MAX-1];
97 }
98 else if(i == 0 && j == 0)
99 {
100     y = arr[MAX-1][MAX-1];
101 }
102 else
103 {
104     y = arr[i-1][j-1];
105 }
106
107 return y;
108 }
109 }

```

Here is the caller graph for this function:



### 2.5.2.6 getRight()

```

int getRight (
    int i,
    int j,
    int arr[MAX][MAX] )

{
71
72
73     int y;
74     if(j == MAX - 1)
75     {
76         y = arr[i][0];
77     }
78     else
79     {
80         y = arr[i][j+1];
81     }
82
83     return y;
84
85 }

```

Here is the caller graph for this function:





## 2.5.2.7 getRightBottom()

```

int getRightBottom (
    int i,
    int j,
    int arr[MAX][MAX] )

155                                     {
156
157     int y;
158     if(i == MAX - 1 && j != MAX - 1)
159     {
160         y = arr[0][j+1];
161     }
162     else if(i != MAX - 1 && j == MAX - 1)
163     {
164         y = arr[i+1][0];
165     }
166     else if(i == MAX - 1 && j == MAX - 1)
167     {
168         y = arr[0][0];
169     }
170     else
171     {
172         y = arr[i+1][j+1];
173     }
174
175     return y;
176
177 }
```

Here is the caller graph for this function:



## 2.5.2.8 getRightTop()

```

int getRightTop (
    int i,
    int j,
    int arr[MAX][MAX] )

110                                     {
111
112     int y;
113     if(i == 0 && j != MAX - 1)
114     {
115         y = arr[MAX-1][j+1];
116     }
117     else if(i != 0 && j == MAX - 1)
118     {
119         y = arr[i-1][0];
120     }
121     else if(i == 0 && j == MAX - 1)
```

```

122     {
123         y = arr[MAX-1][0];
124     }
125     else
126     {
127         y = arr[i-1][j+1];
128     }
129
130     return y;
131 }

```

Here is the caller graph for this function:



### 2.5.2.9 getTop()

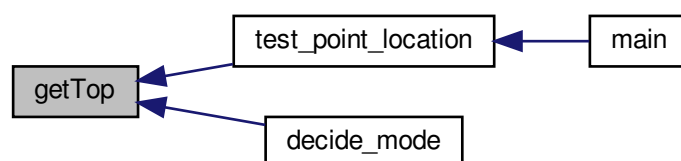
```

int getTop (
    int i,
    int j,
    int arr[MAX][MAX] )

{
26
27
28     int y;
29     if(i == 0)
30     {
31         y = arr[MAX-1][j];
32     }
33     else
34     {
35         y = arr[i-1][j];
36     }
37
38     return y;
39
40 }

```

Here is the caller graph for this function:



# Index

ARCHITECTURE\_ID  
    CMakeCCompilerId.c, 3

array  
    cunit\_tests.c, 13

build/CMakeFiles/3.17.2/CompilerIdC/CMakeC↵  
    CompilerId.c, 3

C\_DIALECT  
    CMakeCCompilerId.c, 3

CMakeCCompilerId.c  
    ARCHITECTURE\_ID, 3  
    C\_DIALECT, 3  
    COMPILER\_ID, 4  
    DEC, 4  
    HEX, 4  
    info\_arch, 5  
    info\_compiler, 5  
    info\_language\_dialect\_default, 6  
    info\_platform, 6  
    main, 5  
    PLATFORM\_ID, 4  
    STRINGIFY\_HELPER, 5  
    STRINGIFY, 4

COMPILER\_ID  
    CMakeCCompilerId.c, 4

copy3.c  
    decide\_mode, 15  
    getBottom, 17  
    getLeft, 18  
    getLeftBottom, 18  
    getLeftTop, 19  
    getRight, 20  
    getRightBottom, 20  
    getRightTop, 21  
    getTop, 22

cunit\_tests.c, 6  
    array, 13  
    epsilon, 14  
    getBottom, 7  
    getLeft, 8  
    getLeftBottom, 8  
    getLeftTop, 9  
    getRight, 9  
    getRightBottom, 10  
    getRightTop, 10  
    getTop, 11  
    MAX, 7  
    main, 11  
    test\_point\_location, 12

DEC  
    CMakeCCompilerId.c, 4

decide\_mode  
    copy3.c, 15  
    header.h, 24

epsilon  
    cunit\_tests.c, 14

getBottom  
    copy3.c, 17  
    cunit\_tests.c, 7  
    header.h, 25

getLeft  
    copy3.c, 18  
    cunit\_tests.c, 8  
    header.h, 26

getLeftBottom  
    copy3.c, 18  
    cunit\_tests.c, 8  
    header.h, 27

getLeftTop  
    copy3.c, 19  
    cunit\_tests.c, 9  
    header.h, 27

getRight  
    copy3.c, 20  
    cunit\_tests.c, 9  
    header.h, 28

getRightBottom  
    copy3.c, 20  
    cunit\_tests.c, 10  
    header.h, 28

getRightTop  
    copy3.c, 21  
    cunit\_tests.c, 10  
    header.h, 29

getTop  
    copy3.c, 22  
    cunit\_tests.c, 11  
    header.h, 30

HEX  
    CMakeCCompilerId.c, 4

header.h  
    decide\_mode, 24  
    getBottom, 25  
    getLeft, 26  
    getLeftBottom, 27  
    getLeftTop, 27

- getRight, [28](#)
- getRightBottom, [28](#)
- getRightTop, [29](#)
- getTop, [30](#)
- MAX, [24](#)
- info\_arch
  - CMakeCCompilerId.c, [5](#)
- info\_compiler
  - CMakeCCompilerId.c, [5](#)
- info\_language\_dialect\_default
  - CMakeCCompilerId.c, [6](#)
- info\_platform
  - CMakeCCompilerId.c, [6](#)
- MAX
  - cunit\_tests.c, [7](#)
  - header.h, [24](#)
- main
  - CMakeCCompilerId.c, [5](#)
  - cunit\_tests.c, [11](#)
- main.c, [14](#)
- mylib/copy3.c, [15](#)
- mylib/header.h, [23](#)
- PLATFORM\_ID
  - CMakeCCompilerId.c, [4](#)
- STRINGIFY\_HELPER
  - CMakeCCompilerId.c, [5](#)
- STRINGIFY
  - CMakeCCompilerId.c, [4](#)
- test\_point\_location
  - cunit\_tests.c, [12](#)