

QUADRATIC_EQUATION_PROJECT

Generated by Doxygen 1.9.2

1 File Index	1
1.1 File List	1
2 File Documentation	3
2.1 eq_solver.c File Reference	3
2.1.1 Function Documentation	3
2.1.1.1 eq_solver()	3
2.1.1.2 lin_eq_solver()	4
2.2 eq_solver.h File Reference	4
2.2.1 Enumeration Type Documentation	4
2.2.1.1 CODES	4
2.2.2 Function Documentation	5
2.2.2.1 eq_solver()	5
2.2.2.2 lin_eq_solver()	5
2.3 eq_solver.h	6
2.4 main.c File Reference	6
2.4.1 Function Documentation	6
2.4.1.1 main()	6
2.5 tests.c File Reference	6
2.5.1 Function Documentation	7
2.5.1.1 eq_test()	7
2.5.1.2 unit_eq_test()	7
2.6 tests.h File Reference	7
2.6.1 Function Documentation	7
2.6.1.1 eq_test()	7
2.6.1.2 unit_eq_test()	7
2.7 tests.h	7
Index	9

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

eq_solver.c	3
eq_solver.h	4
main.c	6
tests.c	6
tests.h	7

Chapter 2

File Documentation

2.1 eq_solver.c File Reference

```
#include <math.h>
#include "eq_solver.h"
```

Functions

- int [eq_solver](#) (double a, double b, double c, double *x1, double *x2)
- double [lin_eq_solver](#) (double b, double c)

2.1.1 Function Documentation

2.1.1.1 eq_solver()

```
int eq_solver (
    double a,
    double b,
    double c,
    double * x1,
    double * x2 )
```

Solves quadratic equation $ax^2 + bx + c = 0$

Parameters

in	<i>a,b,c</i>	Coefficients
out	<i>x1,x2</i>	Pointers to the roots

Returns

Number of root or code of the error

2.1.1.2 lin_eq_solver()

```
double lin_eq_solver (
    double b,
    double c )
```

Solves linear equation $bx + c = 0$

Parameters

in	b,c	Coefficients
----	-----	--------------

Returns

Solution

2.2 eq_solver.h File Reference**Enumerations**

- enum [CODES](#) {
 [ZERO_ROOTS](#) = 0 , [ONE_ROOT](#) = 1 , [TWO_ROOTS](#) = 2 , [INF_ROOTS](#) = 3 ,
 [NULL_PTR_ERR](#) = 5 , [EQ_PTR_ERR](#) = 6 , [INCORRECT_COEFF_ERR](#) = 7 }

Codes, returned by eq_solver function.

Functions

- int [eq_solver](#) (double a, double b, double c, double *x1, double *x2)
- double [lin_eq_solver](#) (double b, double c)

2.2.1 Enumeration Type Documentation**2.2.1.1 CODES**

enum [CODES](#)

Codes, returned by eq_solver function.

Enumerator

ZERO_ROOTS	There are no real solutions.
ONE_ROOT	One real solution.
TWO_ROOTS	Two real solutions.
INF_ROOTS	Infinitely many roots (a = b = c = 0 case)
NULL_PTR_ERR	function got null pointer as argument
EQ_PTR_ERR	function got same pointers
INCORRECT_COEFF_ERR	function got incorrect coefficients

2.2.2 Function Documentation

2.2.2.1 eq_solver()

```
int eq_solver (
    double a,
    double b,
    double c,
    double * x1,
    double * x2 )
```

Solves quadratic equation $ax^2 + bx + c = 0$

Parameters

in	a, b, c	Coefficients
out	$x1, x2$	Pointers to the roots

Returns

Number of root or code of the error

2.2.2.2 lin_eq_solver()

```
double lin_eq_solver (
    double b,
    double c )
```

Solves linear equation $bx + c = 0$

Parameters

in	b, c	Coefficients
----	--------	--------------

Returns

Solution

2.3 eq_solver.h

[Go to the documentation of this file.](#)

```
1
2 enum CODES {
3     ZERO_ROOTS = 0,
4     ONE_ROOT = 1,
5     TWO_ROOTS = 2,
6     INF_ROOTS = 3,
7     NULL_PTR_ERR = 5,
8     EQ_PTR_ERR = 6,
9     INCORRECT_COEFF_ERR = 7,
10 };
11
12 int eq_solver(double a, double b, double c, double* x1, double* x2);
13
14 double lin_eq_solver(double b, double c);
```

2.4 main.c File Reference

```
#include "tests.h"
```

Functions

- int [main](#) ()

2.4.1 Function Documentation

2.4.1.1 main()

```
int main ( )
```

2.5 tests.c File Reference

```
#include <stdio.h>
#include "eq_solver.h"
```

Functions

- void [eq_test](#) (double a, double b, double c, int test_num)
- void [unit_eq_test](#) ()

2.5.1 Function Documentation

2.5.1.1 eq_test()

```
void eq_test (
    double a,
    double b,
    double c,
    int test_num )
```

2.5.1.2 unit_eq_test()

```
void unit_eq_test ( )
```

2.6 tests.h File Reference

Functions

- void [eq_test](#) (double, double, double, int)
- void [unit_eq_test](#) ()

2.6.1 Function Documentation

2.6.1.1 eq_test()

```
void eq_test (
    double a,
    double b,
    double c,
    int test_num )
```

2.6.1.2 unit_eq_test()

```
void unit_eq_test ( )
```

2.7 tests.h

[Go to the documentation of this file.](#)

```
1 void eq\_test(double, double, double, int);
2
3 void unit\_eq\_test();
```


Index

- CODES
 - eq_solver.h, [4](#)
- EQ_PTR_ERR
 - eq_solver.h, [5](#)
- eq_solver
 - eq_solver.c, [3](#)
 - eq_solver.h, [5](#)
- eq_solver.c, [3](#)
 - eq_solver, [3](#)
 - lin_eq_solver, [4](#)
- eq_solver.h, [4](#)
 - CODES, [4](#)
 - EQ_PTR_ERR, [5](#)
 - eq_solver, [5](#)
 - INCORRECT_COEFF_ERR, [5](#)
 - INF_ROOTS, [5](#)
 - lin_eq_solver, [5](#)
 - NULL_PTR_ERR, [5](#)
 - ONE_ROOT, [5](#)
 - TWO_ROOTS, [5](#)
 - ZERO_ROOTS, [5](#)
- eq_test
 - tests.c, [7](#)
 - tests.h, [7](#)
- INCORRECT_COEFF_ERR
 - eq_solver.h, [5](#)
- INF_ROOTS
 - eq_solver.h, [5](#)
- lin_eq_solver
 - eq_solver.c, [4](#)
 - eq_solver.h, [5](#)
- main
 - main.c, [6](#)
- main.c, [6](#)
 - main, [6](#)
- NULL_PTR_ERR
 - eq_solver.h, [5](#)
- ONE_ROOT
 - eq_solver.h, [5](#)
- tests.c, [6](#)
 - eq_test, [7](#)
 - unit_eq_test, [7](#)
- tests.h, [7](#)
 - eq_test, [7](#)
 - unit_eq_test, [7](#)
- TWO_ROOTS
 - eq_solver.h, [5](#)
- ZERO_ROOTS
 - eq_solver.h, [5](#)