Solver square equations

Generated by Doxygen 1.8.20

Chapter 1

File Index

1.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 2

File Documentation

2.1 main.cpp File Reference

The main file of project.

```
#include <iostream>
#include <stdlib.h>
#include <stdio.h>
#include <assert.h>
#include <math.h>
```

Functions

- bool isZero (const double t, const double accuracy)

  Compares value with 0 in current accuracy.
- int SolveLinear (const double a, const double b, double ∗x, double accuracy=1e-7)

  Solves a linear equation ax + b = 0.
- int SolveSquare (const double a, const double b, const double c, double ∗x_1, double ∗x_2, double accuracy=1e-7)

  Solves a square equation ax^2 + bx + c = 0.
- int **main** ()

Variables

- const int SE_INFTY = -1

  Constant which returns when equation has infinite many roots.

2.1.1 Detailed Description

The main file of project.

This file consists main functions which demands for solving square equation.

2.1.2 Function Documentation

2.1.2.1 isZero()

```
bool isZero (
            const double t,
            const double accuracy )  [inline]
```

Compares value with 0 in current accuracy.

Parameters

in	*t*	Rounding variable
in	*accuracy*	Accuracy of rounding to zero

Returns

Next to a variable with zero or not.

Note

Accuracy should be gain than DBL_MIN constant.

2.1.2.2 SolveLinear()

```
int SolveLinear (
            const double a,
            const double b,
            double * x,
            double accuracy = 1e-7 )
```

Solves a linear equation ax + b = 0.

Parameters

in	*a*	a-coefficient
in	*b*	b-coefficient
out	*x*	Pointer to the root
in	*accuracy*	Accuracy of rounding to zero, by default equals to 1e-7

Returns

Number of roots.

Note

Function demands allocated memory for pointer x. In case of infinite number of roots, returns SE_INFTY. Accuracy should be gain than DBL_MIN constant.

Example of usage:

```
int main()
{
    printf("Enter the coefficients of linear equation ax+b = 0\n");
    double a = 0, b = 0;
    scanf("%lg %lg", &a, &b);
    double x = 0;
    int nRoots = SolveLinear(a, b, &x, 1e-6);
    switch (nRoots)
    {
        case 0:
            printf("There aren’t any roots\n");
            break;
        case 1:
            printf("There is a root: x = %.4f\n", x1);
            break;
        case SE_INFTY:
            printf("Any number is a root of current equation.\n");
            break;
        default:
            printf("Strange number of roots... \n");
            break;
    }
    return 0;
}
```

2.1.2.3 SolveSquare()

```
int SolveSquare (
            const double a,
            const double b,
            const double c,
            double * x_1,
            double * x_2,
            double accuracy = 1e-7 )
```

Solves a square equation ax^2 + bx + c = 0.

Parameters

in	*a*	a-coefficient
in	*b*	b-coefficient
in	*c*	c-coefficient
out	*x_1*	Pointer to the 1st root
out	*x_2*	Pointer to the 2nd root
in	*accuracy*	Accuracy of rounding to zero, by default equals to 1e-7

Returns

Number of roots

Note

Function demands allocated memory for both pointers x_1,x_2. In case of infinite number of roots, returns SE_INFTY. Accuracy should be gain than DBL_MIN constant.

Example of usage:

```
int main()
{
    printf("Enter the coefficients of square equation ax^2+bx+c = 0\n");
    double a = 0, b = 0, c = 0;
    scanf("%lg %lg %lg", &a, &b, &c);
    double x1 = 0, x2 = 0;
    int nRoots = SolveSquare(a, b, c, &x1, &x2, 1e-6);
    switch (nRoots)
    {
        case 0:
            printf("There aren’t any roots\n");
            break;
        case 1:
            printf("There is a root: x_1 = %.4f\n", x1);
            break;
        case 2:
            printf("There is two roots: x_1 = %.4f, x_2 = %.4f \n", x1, x2);
            break;
        case SE_INFTY:
            printf("Any number is a root of current equation.\n");
            break;
        default:
            printf("Strange number of roots... \n");
            break;
    }
    return 0;
}
```

Index