LAMPIRAN 1: PROGRAM JAVA UNTUK EKSPERIMEN AWAL

```
public class QuadraticSolver {


    static final int ONE_ROOT= 200;
    static final int TWO_ROOTS = 201;
    static final int NO_REAL_ROOT = 202;

    static final double INITIAL_GUESS = 1;
    static final double D_VALUE = 1E-4;
    static final double E_VALUE = 1E-15;


    //equation variables
    double a, b, c;

    //newton variables
    double x, d;

    //supporting vars
    double roots[], aroot, disc;

    public QuadraticSolver(String sa, String sb, String sc) {

        a = Double.parseDouble(sa);
        b = Double.parseDouble(sb);
        c = Double.parseDouble(sc);

        roots = new double [2];

        System.out.println("a = "+a+" b = "+b+" c = "+c);

        switch (discriminant(a,b,c)) {

            case ONE_ROOT:
                roots[0] = rootOf(a,b,c);
                roots[1] = roots[0];
                break;

            case TWO_ROOTS:
                do {
                    roots[0] = rootOf(a,b,c);
                    roots[1] = rootOf(a,b,c);
                } while (Math.abs(roots[0] - roots[1]) < 1E-4);
                break;

            case NO_REAL_ROOT:
                System.out.println("no real root.");
                return;
        }

        System.out.println("roots are: "+roots[0]+" and "+roots[1]);
    }
    public double rootOf(double a, double b, double c) {

        x = new java.util.Random().nextInt();
        c = c/a; b=b/a; a = 1;

        do {

           //~ d = (Math.pow(a*x, 2)+b*x+c)/(2*a*x+b);
           d = (Math.pow(x, 2)+b*x+c)/(2*x+b);
           x = x-d;

       } while (Math.abs(d) >= D_VALUE);

       return x;
    }

    public int discriminant(double a, double b, double c) {
        double disc = (Math.pow(b,2))-(4*a*c);

        if (disc > 0) return TWO_ROOTS;
        else if (disc == 0) return ONE_ROOT;
        else return NO_REAL_ROOT;

    }


    public static void main (String[] argv) {
        new QuadraticSolver(argv[0], argv[1], argv[2]);
    }


    //T2 vars and specs here

    private boolean classinv() {
        boolean b = true;
        return b;
    }

    public double rootOf_spec(double a, double b, double c) {
        assert a != 0 : "PRE";
        double d = rootOf(a, b, c);
        assert Math.abs(a*(Math.pow(d, 2)) + b*d + c) < E_VALUE : "POST";
        return d;
    }

}
```

LAMPIRAN 2: IMPLEMENTASI MOCK CLASS DALAM JAVA

```java
/* Testing out the e-Calculator #RELOAD

   made by Prastudy
   modified by yud1

*/

import voting.*;

class TestReloaded {

    static final int CANDIDATE_MAX = 3;

    Timing tim = new Timing();
    Calculator calc = new Calculator();
    Central cen  = new Central(CANDIDATE_MAX);
    Voting vot = new Voting();


    public TestReloaded(){

        vot = new Voting();
        vot.setVote(01+"",CANDIDATE_MAX);
        vot.addVoteListener(calc);
        vot.addAskTimeListener(tim);

        tim.addTimeListener(vot);


        tim.addTimeListener(calc);
        tim.addTimeListener(cen);

        calc.addAskTimeListener(tim);
        cen.addAskTimeListener(tim);
        calc.addCalculationListener( cen );
        calc.setCalculation(01+"", CANDIDATE_MAX);

    }

    // T2-purpose

    public static void main(String [] args) {

        //~ new TestReloaded();
        String testString;

        //T2 configurable options
        nmax = 500;
        lenexec = 4;
        itr = 1;

        for (int i = 0; i< itr; i++ ) {
            testString = "java -ea -cp .;TT_0.4_silverlance.jar"+
                " U2.T2.RndEngine TestReloaded"+
                " --nullprob=-1"+
                " --excldefault"+
                " --nmax="+nmax+
                " --lenexec="+lenexec+
                " --excldefault"+
                " --outfile=T2-test-0"+i+".txt";

            try {
            Runtime.getRuntime().exec(testString);
            } catch (Exception e) {System.out.println("whatever");}
        }
    }

    public void timStartStop() {
        tim.startStop();
    }

    public boolean votIsValidTime() {
        return vot.isValidTime();
    }

    public void votGenerateVote(String voteId , int cand ) {
        //added #1a: candidate ID are to be within range
        assert cand > 0 && cand <= CANDIDATE_MAX : "PRE";
        aux_voteMe = cenGetTotalVote(cand);

        vot.generateVote(voteId, cand);

        //added #1b: assures generateVote complies to stopflag
        if(aux_state != VOTE)
            assert cenGetTotalVote(cand) == aux_voteMe : "POST";
    }

    public int cenGetTotalVote(int cand){
        //added #2a: candidate ID are to be within range
        assert cand >= 0 && cand < CANDIDATE_MAX : "PRE" ;

        int ret = cen.getTotalVote(cand);

        //added #2b: votes are always non-negative
        assert ret >= 0 : "POST";
        return ret;
    }

    public int cenGetTotalCount() {
        int ret = cen.totalCount();
        return ret;
    }

    public java.util.Calendar timGetTime(){
        java.util.Calendar ret = tim.getTime();
        return ret;
    }


    //T2-related specifications

    private static final int NOTIME = 301;
    private static final int VOTE = 302;
    private static final int LIMIT = 303;
    private static int nmax, lenexec, itr;

    private int aux_state = NOTIME;
    private int aux_count = 0;
    private int aux_votes = 0;
    private int aux_voteMe = 0;
    private String aux_laststep = "";

    private java.util.Calendar aux_cal = java.util.Calendar.getInstance();

    private boolean classinv() {

        if (aux_laststep.equals("timStartStop"))
            aux_count++;

        if (aux_count % 2 == 1)
            aux_state = VOTE;
        else if (aux_count % 2 == 0)
            aux_state = LIMIT;

        switch (aux_state) {
            case NOTIME :

                //script #1: NOTIME is followed by t.start()
                //...(needed?)

                //property #4: votes are empty upon NOTIME
                for (int i = 0; i < CANDIDATE_MAX; i++)
                    assert cenGetTotalVote(i) == 0 : "PROP#4a";
                assert cenGetTotalCount() == 0 : "PROP#4b";

            break;

            case VOTE :

                //property #3: t.tick always increment outside NOTIME
                aux_cal = timGetTime();

            break;

            case LIMIT :

                //script #1: stop calculation by v past limit
                //property #2: raise stopflag immediately after limit
                assert !votIsValidTime() : "PROP#2";

                //property #1: votes are superset or equivalent past limit
                aux_votes = 0;
                for (int i=0; i<CANDIDATE_MAX; i++)
                    aux_votes += cenGetTotalVote(i);
                assert cenGetTotalCount() >= aux_votes : "PROP#1";

                //property #3 with weaker assumption -- unused
                //assert !aux_cal.after(timGetTime()) : "PROP#3";

                //property #3: t.tick always increments outside NOTIME
                assert timGetTime().after(aux_cal) : "PROP#3";

            break;
        }

        return true;
    }

}
```

LAMPIRAN 3: CONTOH KELUARAN T2 FRAMEWORK

Eksperimen Awal: defect-01

```
----------------------------------------------------
**  T2 Testing Tool for Java. Version 0.4, 2007.
----------------------------------------------------
----------------------------------------------------
** Engine CONFIGURATION:
** Max. number of steps : 500
** Max. execution depth : 4
** Stop after this many violations : 1
** Max. object depth (on creation) : 4
** Max. show depth                 : 5
** Max. array/collection size : 5
** Default members are included.
** Protected members are included.
** Static members are included.
** Members of superclasses are included.
** Prob. of updating field : 0.1
** Prob. of auto generation of NULL : 0.1
** Prob. of passing target obj as parameter : 0.4
** Prob. of trying pool before constuctor : 0.7
** CUT           : QuadraticSolver
** TYVAR0        : QuadraticSolver
** Pool          : U2.T2.Pool
** Base domain   : U2.T2.BaseDomainSmall
** Interface map : U2.T2.InterfaceMap0
----------------------------------------------------
** Testing QuadraticSolver ...
a = 1.0 b = 1.0 c = 1.0
no real root.
** Time elapsed: 78 ms

** 1 VIOLATIONS found.
----------------------------------------------------
** total attempted execution steps : 139
** total generated executions      : 1
** number of irrelevant checks     : 0
----------------------------------------------------



** Violating trace [1] :
a = 1.0 b = 1.0 c = 1.0
no real root.
  ** CREATING target object.
    ** Return value:
       (QuadraticSolver) @ 0
       ONE_ROOT (Integer) : 200
       TWO_ROOTS (Integer) : 201
       NO_REAL_ROOT (Integer) : 202
       INITIAL_GUESS (Double) : 1.0
       a (Double) : 1.0
       b (Double) : 1.0
```

```
     c (Double) : 1.0
     x (Double) : 0.0
     d (Double) : 0.0
     roots (ARRAY)  @ 6
           [0] (Double) : 0.0
           [1] (Double) : 0.0
     aroot (Double) : 0.0
     disc (Double) : 0.0
     ROUNDING_LENGTH (Integer) : 20
** STEP 1.
** Calling method discriminant with:
  ** Receiver: target-obj
  ** Arg [0:]
     (Double) : 3.93019962310791
  ** Arg [1:]
     (Double) : 2.9288992881774902
  ** Arg [2:]
     (Double) : -2.588395118713379
  ** Target object after the step:
     (QuadraticSolver) @ 0
     ONE_ROOT (Integer) : 200
     TWO_ROOTS (Integer) : 201
     NO_REAL_ROOT (Integer) : 202
     INITIAL_GUESS (Double) : 1.0
     a (Double) : 1.0
     b (Double) : 1.0
     c (Double) : 1.0
     x (Double) : 0.0
     d (Double) : 0.0
     roots (ARRAY)  @ 6
           [0] (Double) : 0.0
           [1] (Double) : 0.0
     aroot (Double) : 0.0
     disc (Double) : 0.0
     ROUNDING_LENGTH (Integer) : 20
  ** Return value:
     (Integer) : 201
** STEP 2.
** Calling method rootOf_spec with:
  ** Receiver: target-obj
  ** Arg [0:]
     (Double) : -1.2139594554901123
  ** Arg [1:]
     (Double) : 0.5255091190338135
  ** Arg [2:]
     (Double) : 1.523733139038086
  ** Target object after the step:
     (QuadraticSolver) @ 0
     ONE_ROOT (Integer) : 200
     TWO_ROOTS (Integer) : 201
     NO_REAL_ROOT (Integer) : 202
     INITIAL_GUESS (Double) : 1.0
     a (Double) : 1.0
     b (Double) : 1.0
     c (Double) : 1.0
     x (Double) : NaN
     d (Double) : NaN
     roots (ARRAY)  @ 6
           [0] (Double) : 0.0
```

```
             [1] (Double) : 0.0
       aroot (Double) : 0.0
       disc (Double) : 0.0
       ROUNDING_LENGTH (Integer) : 20
  xx Assertion VIOLATED!
  ** Strack trace:
java.lang.AssertionError: POST
      at QuadraticSolver.rootOf_spec(QuadraticSolver.java:102)
      at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
      at sun.reflect.NativeMethodAccessorImpl.invoke(Unknown Source)
      at      sun.reflect.DelegatingMethodAccessorImpl.invoke(Unknown
Source)
      at java.lang.reflect.Method.invoke(Unknown Source)
      at U2.T2.Trace.exec(Trace.java:804)
      at U2.T2.Trace.report(Trace.java:899)
      at U2.T2.RndEngine.report(RndEngine.java:752)
      at U2.T2.RndEngine.RndTest(RndEngine.java:891)
      at U2.T2.RndEngine.RndTest(RndEngine.java:954)
      at U2.T2.RndEngine.main(RndEngine.java:1314)
  -----------------------------------
** Saving 1 traces...
** DONE.
```

Eksperimen Awal: stable-01

```
----------------------------------------------------
**  T2 Testing Tool for Java. Version 0.4, 2007.
----------------------------------------------------
----------------------------------------------------
** Engine CONFIGURATION:
** Max. number of steps : 500
** Max. execution depth : 4
** Stop after this many violations : 1
** Max. object depth (on creation) : 4
** Max. show depth                 : 5
** Max. array/collection size : 5
** Default members are included.
** Protected members are included.
** Static members are included.
** Members of superclasses are included.
** Prob. of updating field : 0.1
** Prob. of auto generation of NULL : 0.1
** Prob. of passing target obj as parameter : 0.4
** Prob. of trying pool before constuctor : 0.7
** CUT           : QuadraticSolver
** TYVAR0        : QuadraticSolver
** Pool          : U2.T2.Pool
** Base domain   : U2.T2.BaseDomainSmall
** Interface map : U2.T2.InterfaceMap0
----------------------------------------------------
** Testing QuadraticSolver ...
a = 1.0 b = 1.0 c = 1.0
no real root.
** Time elapsed: 172 ms

** NO violation found.
```

```
----------------------------------------------------
** total attempted execution steps : 500
** total generated executions      : 1
** number of irrelevant checks     : 0
----------------------------------------------------

** DONE.
```

Eksperimen Studi Kasus: Pengujian Tanpa Property (3)

```
----------------------------------------------------
**  T2 Testing Tool for Java. Version 0.4, 2007.
----------------------------------------------------
----------------------------------------------------
** Engine CONFIGURATION:
** Max. number of steps : 500
** Max. execution depth : 4
** Stop after this many violations : 1
** Max. object depth (on creation) : 4
** Max. show depth                 : 5
** Max. array/collection size : 5
** Protected members are included.
** Static members are included.
** Members of superclasses are included.
** Prob. of updating field : 0.1
** Auto generation of NULL disabled.
** Prob. of passing target obj as parameter : 0.4
** Prob. of trying pool before constuctor : 0.7
** CUT           : TestReloaded
** TYVAR0        : TestReloaded
** Pool          : U2.T2.Pool
** Base domain   : U2.T2.BaseDomainSmall
** Interface map : U2.T2.InterfaceMap0
----------------------------------------------------
** Testing TestReloaded ...
** Time elapsed: 2703 ms

** NO violation found.
----------------------------------------------------
** total attempted execution steps : 500
** total generated executions      : 127
** number of irrelevant checks     : 64
----------------------------------------------------

** DONE.
```