Assign #3 posted

1) vector dot product recursively

2) binary powering (recursively) x^{51}

bonus 3) matrix mult. recursively

c

Testing Automated

Defn: black box - from specs

white box - from code (transparent)

coverage: is a statement executed?

```
if (cond) {
    stmt1;
} else {
    stmt2;
}
```

always true in your tests

add tests where cond is false

```
if (cond1) { S1 } else { S2 }

if (cond2) { S3 } else { S4 }
```

all paths. $2^3 = 8$

cond cond2 cond3

~100 => 2^{100} exhaustive

[statistical sampling]

⇒ generate problems

⇒ verify answers

ex: d.e. solver
differential equation

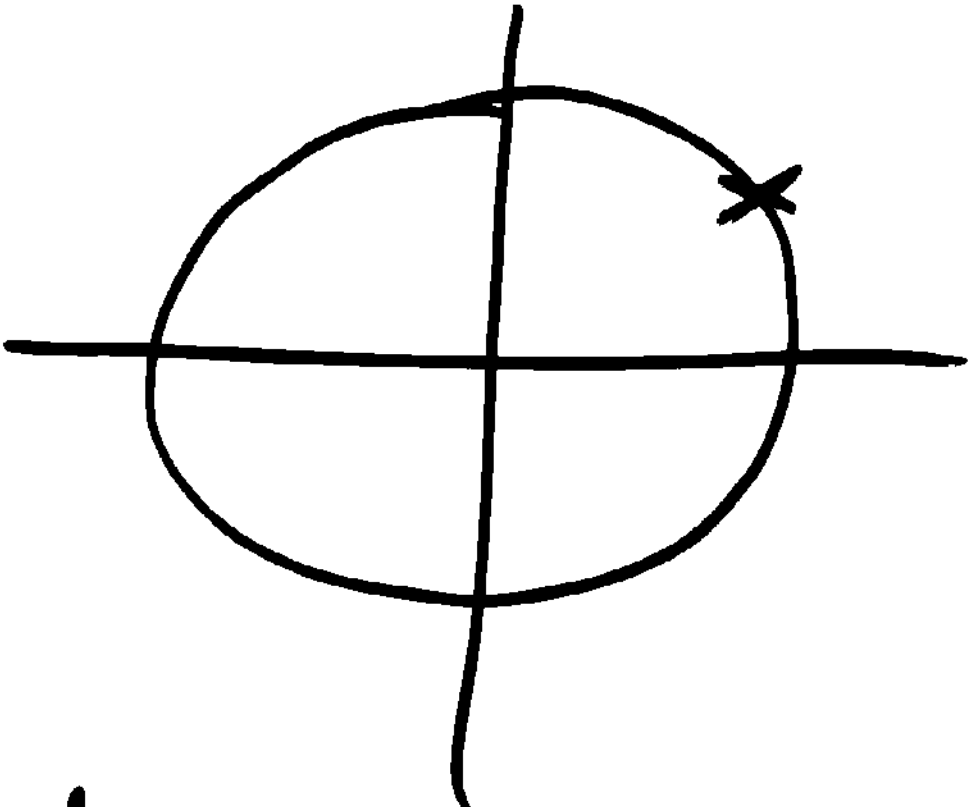
ans → plug-in & simplify
if ≡ 0 —
else bug?

⇒ automate testing
Frameworks

Defn: unit testing is
testing of 1 function in
isolation.

statistical testing

```
double f(double x, double y) {
    return 1/(x^2+y^2-1);
}
```



random x & y

white box → sampling non-uniform

→ user-oriented

Java → JUnit

C → CuTest

repeat / chaining:

- define a problem
- answer = f(problem)
- test if answer == expected answer. | how?
- if not then output diagnostic

verbose →
tells you exactly how to reproduce the failure

problem(n) = function(answer(n-1))

output is code to reproduce

Configuration Management

→ person X changed file foo35.c

→ test #1235
  calls fns in foo35.c

→ that test failed!

→ email person (& boss) broke the ✱!! build.

→ auto back out change & re-test

XP eXtreme Programming

1. tests written first
2. tests <u>never</u> fail