Introduction to Verification / Testing

- One part of software lifecycle is checking code meets specification

(specification → Design → Code → Check wrt spec → Release (→ maintain)

- Two ways of checking code meets specs: Verification & Testing
- Verification : Analytical, exhaustive but limited, needs training
- Testing: Experimental (I/o pair), non-exhaustive but easier (can show presence of bugs, but not absence of them)

Example.

```
read(x); read(y);
If x=y, then z:=0;   (should have been z:=2).
     else z:=1;
end if; write(z)
```

- Test inputs : (x=y=1) and (x=1, y=2) will do.

```
If x>y, then x:= x-y;
     else if x<y, then y:= y-x;
                  else z:= x;
end if; end if; write(z)
```

(12,8) → (4,8) → (4,4) → 4 Claim $z = gcd(x,y)$

- Any finite pair of test inputs won't demonstrate $z = gcd(x,y)$
- Verification can be used to establish $z = gcd(x,y)$

Two approaches to verification : 1) Theorem proving
general but manual
2) Model checking
limited but automated