

CS 6110 – Formal Methods in System Design | Spring 2015  
Mar-2

## Lecture 9 Program Correctness: Strategy

Zvonimir Rakamarić  
University of Utah

slides acknowledgements: Z. Manna, R. Leino

### Last Time

- ▶ Procedures
- ▶ Loops
- ▶ Loop Invariants

## While Loop with Invariant

```

while E
  invariant J
do
  S
end

```

Annotations:

- loop condition (points to E)
- loop invariant (points to J)
- loop body (points to S)

- ▶ Loop body S executed as long as loop condition E holds
- ▶ Loop invariant J must hold on every iteration
  - ▶ J must hold initially and is evaluated before E
  - ▶ J must hold even on final iteration when E is false
  - ▶ Provided by a user or inferred automatically

## Desugaring While Loop Using Invariant

- ▶ while E invariant J do S end

```

assert J;
havoc x; assume J;
(
  assume E; S; assert J; assume false
□
  assume ¬E
)

```

Annotations:

- check that the loop invariant holds initially (points to `assert J;`)
- jump to an arbitrary iteration of the loop (points to `assume J;`)
- where x denotes the assignment targets of S (points to `havoc x;`)
- check that the loop invariant is maintained by the loop body (points to `assert J;`)
- exit the loop (points to `assume ¬E`)

## This Time

- ▶ Examples, examples, examples...
- ▶ Some strategies for proving correctness

## (Dumb) Example: Multiply by 2

```
method Multiply2(n:int) returns (r:int)
{
  r := 0;
  var i:int := 0;
  while (i < n)
  {
    r := r + 2;
    i := i + 1;
  }
}
```

- ▶ Specification:
  - ▶ Given a non-negative integer `n`, function `Multiply2` multiplies it by 2

### Example: Initialize Array

- ▶ Signature:  
`InitializeArray(a:array<int>, e:int)`
- ▶ Specification:
  - ▶ Initializes elements of array `a` to `e`

### Example: Linear Search

- ▶ Signature:  
`LinearSearch(a:array<int>, l:int, u:int, e:int) returns (r:bool)`
- ▶ Specification:
  - ▶ Returns `true` if `e` is found in array `a` between `l` and `u`, otherwise returns `false`