Analysis of Software Artifacts

Hoare Logic: Proving Programs Correct (continued)

Jonathan Aldrich

Review: Hoare Logic Rules

- $wp(x := E, P) = [E/x]\, P$
- $wp(S;T, Q) = wp(S, wp(T, Q))$
- $wp(\text{if } B \text{ then } S \text{ else } T, Q)$

  $= B \Rightarrow wp(S,Q)\ \&\&\ \neg B \Rightarrow wp(T,Q)$

Proving loops correct

- *Partial correctness*
  - The loop may not terminate, but if it does, the postcondition will hold
- {P} while B do S {Q}
  - Find an invariant Inv such that:
    - $P \Rightarrow Inv$
      - The invariant is initially true
    - { Inv && B } S {Inv}
      - Each execution of the loop preserves the invariant
    - $(Inv \;\&\&\; \neg B) \Rightarrow Q$
      - The invariant and the loop exit condition imply the postcondition

Quick Quiz

Consider the following program:

```
{ N >= 0 }
i := 0;
while (i < N) do
    i := N
{ i = N }
```

Correctness Conditions

$P \Rightarrow Inv$

The invariant is initially true

{ Inv && B } S {Inv}

Loop preserves the invariant

$(Inv \;\&\&\; \neg B) \Rightarrow Q$

Invariant and exit implies postcondition

Which of the following loop invariants are correct? For those that are incorrect, explain why.

A) i = 0
B) i = N
C) N >= 0
D) i <= N

Loop Example

- Prove array sum correct

```
{ N ≥ 0 }
j := 0;
s := 0;

while (j < N) do

    j := j + 1;
    s := s + a[j];

end
{ s = (Σi | 0≤i<N • a[i]) }
```

How can we find a loop invariant?

Replace N with j
Add information on range of j
Result: $0 \le j \le N$ && $s = (\Sigma i \mid 0 \le i < j \bullet a[i])$

Loop Example

- Prove array sum correct

```
{ N ≥ 0 }
j := 0;
s := 0;
{ 0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }
while (j < N) do
    {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}
    j := j + 1;
    s := s + a[j];
    {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }
end
{ s = (Σi | 0≤i<N • a[i]) }
```

Loop Example

- Prove array sum correct

```
{ N ≥ 0 }
j := 0;
s := 0;
{ 0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }        } Proof obligation #1
while (j < N) do
    {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}
    j := j + 1;
    s := s + a[j];
    {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }    } Proof obligation #2
end                       ⇓ Proof obligation #3
{ s = (Σi | 0≤i<N • a[i]) }
```

Proof Obligations

- Invariant is initially true

  $\{ N \geq 0 \}$
  j := 0;
  s := 0;
  $\{ 0 \leq j \leq N \ \&\& \ s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \}$

Proof Obligations

- Invariant is initially true

```
{ N ≥ 0 }
j := 0;
s := 0;
{ 0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }
```

- Invariant is maintained

```
{0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}
j := j + 1;
s := s + a[j];
{0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }
```

Proof Obligations

- Invariant is initially true

  $\{ N \geq 0 \}$
  j := 0;
  s := 0;
  $\{ 0 \leq j \leq N \;\&\&\; s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \}$

- Invariant is maintained

  $\{0 \leq j \leq N \;\&\&\; s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \;\&\&\; j < N\}$
  j := j + 1;
  s := s + a[j];
  $\{0 \leq j \leq N \;\&\&\; s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \}$

- Invariant and exit condition imply postcondition

  $0 \leq j \leq N \;\&\&\; s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \;\&\&\; j \geq N$
  $\Rightarrow s = (\Sigma i \mid 0 \leq i < N \bullet a[i])$

Proof Obligations

- Invariant is initially true

  $\{ N \geq 0 \}$

  j := 0;

  s := 0;

  $\{ 0 \leq j \leq N \ \&\& \ s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \}$

Proof Obligations

- Invariant is initially true

  $\{ N \geq 0 \}$

  j := 0;

  $\{ 0 \leq j \leq N \;\&\&\; \mathbf{0} = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \}$ *// by assignment rule*

  s := 0;

  $\{ 0 \leq j \leq N \;\&\&\; s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \}$

Proof Obligations

- Invariant is initially true

  ```
  { N ≥ 0 }
  { 0 ≤ 0 ≤ N && 0 = (Σi | 0≤i<0 • a[i]) } // by assignment rule
  j := 0;
  { 0 ≤ j ≤ N && 0 = (Σi | 0≤i<j • a[i]) }   // by assignment rule
  s := 0;
  { 0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }
  ```

Proof Obligations

- Invariant is initially true

  { N ≥ 0 }

  { 0 ≤ **0** ≤ N && 0 = (Σi | 0≤i<**0** • a[i]) } *// by assignment rule*

  j := 0;

  { 0 ≤ j ≤ N && **0** = (Σi | 0≤i<j • a[i]) } *// by assignment rule*

  s := 0;

  { 0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }

- Need to show that:

  (N ≥ 0) ⇒ (0 ≤ 0 ≤ N && 0 = (Σi | 0≤i<0 • a[i]))

Proof Obligations

- Invariant is initially true

  { N ≥ 0 }
  { 0 ≤ **0** ≤ N && 0 = (Σi | 0≤i<**0** • a[i]) } *// by assignment rule*
  j := 0;
  { 0 ≤ j ≤ N && **0** = (Σi | 0≤i<j • a[i]) } *// by assignment rule*
  s := 0;
  { 0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }

- Need to show that:

  (N ≥ 0) ⇒ (0 ≤ 0 ≤ N && 0 = (Σi | 0≤i<0 • a[i]))

= (N ≥ 0) ⇒ (0 ≤ N && 0 = **0**) *// 0 ≤ 0 is true, empty sum is 0*

Proof Obligations

- Invariant is initially true

  $\{ N \geq 0 \}$

  $\{ 0 \leq \mathbf{0} \leq N \ \&\& \ 0 = (\Sigma i \mid 0 \leq i < \mathbf{0} \bullet a[i]) \}$ *// by assignment rule*

  j := 0;

  $\{ 0 \leq j \leq N \ \&\& \ \mathbf{0} = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \}$ *// by assignment rule*

  s := 0;

  $\{ 0 \leq j \leq N \ \&\& \ s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \}$

- Need to show that:

  $(N \geq 0) \Rightarrow (0 \leq 0 \leq N \ \&\& \ 0 = (\Sigma i \mid 0 \leq i < 0 \bullet a[i]))$

= $(N \geq 0) \Rightarrow (0 \leq N \ \&\& \ 0 = \mathbf{0})$ *// 0 ≤ 0 is true, empty sum is 0*

= $(N \geq 0) \Rightarrow (0 \leq N)$ *// 0=0 is true, P && true is P*

Proof Obligations

- Invariant is initially true

  $\{ N \geq 0 \}$
  $\{ 0 \leq \mathbf{0} \leq N \;\&\&\; 0 = (\Sigma i \mid 0 \leq i < \mathbf{0} \bullet a[i]) \}$ *// by assignment rule*
  j := 0;
  $\{ 0 \leq j \leq N \;\&\&\; \mathbf{0} = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \}$ *// by assignment rule*
  s := 0;
  $\{ 0 \leq j \leq N \;\&\&\; s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \}$

- Need to show that:

$(N \geq 0) \Rightarrow (0 \leq 0 \leq N \;\&\&\; 0 = (\Sigma i \mid 0 \leq i < 0 \bullet a[i]))$
= $(N \geq 0) \Rightarrow (0 \leq N \;\&\&\; 0 = \mathbf{0})$ *// 0 ≤ 0 is true, empty sum is 0*
= $(N \geq 0) \Rightarrow (0 \leq N)$ *// 0=0 is true, P && true is P*
= **true**

Proof Obligations

- Invariant is maintained

  $\{0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \ \&\& \ j < N\}$

  j := j + 1;

  s := s + a[j];

  $\{0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \}$

Proof Obligations

- Invariant is maintained

  $\{0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \ \&\& \ j < N\}$

  j := j + 1;

  $\{0 \le j \le N \ \&\& \ \mathbf{s+a[j]} = (\Sigma i \mid 0 \le i < j \bullet a[i]) \}$ *// by assignment rule*

  s := s + a[j];

  $\{0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \}$

Proof Obligations

- Invariant is maintained

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}
  {0 ≤ **j +1** ≤ N && s+a[**j+1**] = (Σi | 0≤i<**j+1** • a[i]) }   *// by assignment rule*
  j := j + 1;
  {0 ≤ j ≤ N && **s+a[j]** = (Σi | 0≤i<j • a[i]) }   *// by assignment rule*
  s := s + a[j];
  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }

Proof Obligations

- Invariant is maintained

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}

  {0 ≤ **j +1** ≤ N && s+a[**j+1**] = (Σi | 0≤i<**j+1** • a[i]) } *// by assignment rule*

  j := j + 1;

  {0 ≤ j ≤ N && **s+a[j]** = (Σi | 0≤i<j • a[i]) } *// by assignment rule*

  s := s + a[j];

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }

- Need to show that:

  (0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N)

  ⇒ (0 ≤ j +1 ≤ N && s+a[j+1] = (Σi | 0≤i<j+1 • a[i]))

Proof Obligations

- Invariant is maintained

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}

  {0 ≤ **j +1** ≤ N && s+a[**j+1**] = (Σi | 0≤i<**j+1** • a[i]) }   *// by assignment rule*

  j := j + 1;

  {0 ≤ j ≤ N && **s+a[j]** = (Σi | 0≤i<j • a[i]) }   *// by assignment rule*

  s := s + a[j];

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }

- Need to show that:

  (0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N)

  ⇒ (0 ≤ j +1 ≤ N && s+a[j+1] = (Σi | 0≤i<j+1 • a[i]))

= (0 ≤ j < N && s = (Σi | 0≤i<j • a[i]))

  ⇒ (**-1 ≤ j < N** && s+a[j+1] = (Σi | 0≤i<j+1 • a[i]))   *// simplify bounds of j*

Proof Obligations

- Invant is maintained

  $\{0 \le j \le N \;\&\&\; s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \;\&\&\; j < N\}$

  $\{0 \le \mathbf{j+1} \le N \;\&\&\; s+a[\mathbf{j+1}] = (\Sigma i \mid 0 \le i < \mathbf{j+1} \bullet a[i])\}$ *// by assignment rule*

  j := j + 1;

  $\{0 \le j \le N \;\&\&\; \mathbf{s+a[j]} = (\Sigma i \mid 0 \le i < j \bullet a[i])\}$ *// by assignment rule*

  s := s + a[j];

  $\{0 \le j \le N \;\&\&\; s = (\Sigma i \mid 0 \le i < j \bullet a[i])\}$

- Need to show that:

  $(0 \le j \le N \;\&\&\; s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \;\&\&\; j < N)$

  $\quad \Rightarrow (0 \le j+1 \le N \;\&\&\; s+a[j+1] = (\Sigma i \mid 0 \le i < j+1 \bullet a[i]))$

$=$ $(0 \le j < N \;\&\&\; s = (\Sigma i \mid 0 \le i < j \bullet a[i]))$

  $\quad \Rightarrow (\mathbf{-1 \le j < N} \;\&\&\; s+a[j+1] = (\Sigma i \mid 0 \le i < j+1 \bullet a[i]))$ *// simplify bounds of j*

$=$ $(0 \le j < N \;\&\&\; s = (\Sigma i \mid 0 \le i < j \bullet a[i]))$

  $\quad \Rightarrow (-1 \le j < N \;\&\&\; s+a[j+1] = (\Sigma i \mid 0 \le i < j \bullet a[i]) \mathbf{+ a[j]})$ *// separate last element*

Proof Obligations

- Invariant is maintained

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}

  {0 ≤ **j +1** ≤ N && s+a[**j+1**] = (Σi | 0≤i<**j+1** • a[i]) } *// by assignment rule*

  j := j + 1;

  {0 ≤ j ≤ N && **s+a[j]** = (Σi | 0≤i<j • a[i]) } *// by assignment rule*

  s := s + a[j];

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }

- Need to show that:

  (0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N)

  ⇒ (0 ≤ j +1 ≤ N && s+a[j+1] = (Σi | 0≤i<j+1 • a[i]))

= (0 ≤ j < N && s = (Σi | 0≤i<j • a[i]))

  ⇒ (**-1 ≤ j < N** && s+a[j+1] = (Σi | 0≤i<j+1 • a[i])) *// simplify bounds of j*

= (0 ≤ j < N && s = (Σi | 0≤i<j • a[i]))

  ⇒ (-1 ≤ j < N && s+a[j+1] = (Σi | 0≤i<j • a[i]) **+ a[j]**) *// separate last element*

// we have a problem – we need a[j+1] and a[j] to cancel out

Where's the error?

- Prove array sum correct

```
{ N ≥ 0 }
j := 0;
s := 0;

while (j < N) do

    j := j + 1;
    s := s + a[j];

end
{ s = (Σi | 0≤i<N • a[i]) }
```

Where's the error?

- Prove array sum correct

```
{ N ≥ 0 }
j := 0;
s := 0;

while (j < N) do

    j := j + 1;
    s := s + a[j];

end
{ s = (Σi | 0≤i<N • a[i]) }
```

Need to add element ***before*** incrementing j

Corrected Code

- Prove array sum correct

```
{ N ≥ 0 }
j := 0;
s := 0;

while (j < N) do

    s := s + a[j];
    j := j + 1;

end
{ s = (Σi | 0≤i<N • a[i]) }
```

Proof Obligations

- Invariant is maintained

  $\{0 \leq j \leq N \ \&\& \ s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \ \&\& \ j < N\}$

  s := s + a[j];

  j := j + 1;

  $\{0 \leq j \leq N \ \&\& \ s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \}$

Proof Obligations

- Invariant is maintained

$\{0 \le j \le N \;\&\&\; s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \;\&\&\; j < N\}$

s := s + a[j];

$\{0 \le \mathbf{j+1} \le N \;\&\&\; s = (\Sigma i \mid 0 \le i < \mathbf{j+1} \bullet a[i])\}$ *// by assignment rule*

j := j + 1;

$\{0 \le j \le N \;\&\&\; s = (\Sigma i \mid 0 \le i < j \bullet a[i])\}$

Proof Obligations

- Invariant is maintained

```
{0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}
{0 ≤ j +1 ≤ N && s+a[j] = (Σi | 0≤i<j+1 • a[i]) }      // by assignment rule
s := s + a[j];
{0 ≤ j +1 ≤ N && s = (Σi | 0≤i<j+1 • a[i]) }            // by assignment rule
j := j + 1;
{0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }
```

Proof Obligations

- Invariant is maintained

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}

  {0 ≤ j +1 ≤ N && **s+a[j]** = (Σi | 0≤i<j+1 • a[i]) }   *// by assignment rule*

  s := s + a[j];

  {0 ≤ **j +1** ≤ N && s = (Σi | 0≤i<**j+1** • a[i]) }   *// by assignment rule*

  j := j + 1;

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }

- Need to show that:

  (0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N)

  ⇒ (0 ≤ j +1 ≤ N && s+a[j] = (Σi | 0≤i<j+1 • a[i]))

Proof Obligations

- Invariant is maintained

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}
  {0 ≤ j +1 ≤ N && **s+a[j]** = (Σi | 0≤i<j+1 • a[i]) }   *// by assignment rule*
  s := s + a[j];
  {0 ≤ **j +1** ≤ N && s = (Σi | 0≤i<**j+1** • a[i]) }   *// by assignment rule*
  j := j + 1;
  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }

- Need to show that:

  (0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N)
  ⇒ (0 ≤ j +1 ≤ N && s+a[j] = (Σi | 0≤i<j+1 • a[i]))

= (0 ≤ j < N && s = (Σi | 0≤i<j • a[i]))
  ⇒ (**-1 ≤ j < N** && s+a[j] = (Σi | 0≤i<j+1 • a[i])) *// simplify bounds of j*

Proof Obligations

- Invariant is maintained

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}
  {0 ≤ j +1 ≤ N && **s+a[j]** = (Σi | 0≤i<j+1 • a[i]) } *// by assignment rule*
  s := s + a[j];
  {0 ≤ **j +1** ≤ N && s = (Σi | 0≤i<**j+1** • a[i]) } *// by assignment rule*
  j := j + 1;
  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }

- Need to show that:

  (0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N)
  ⇒ (0 ≤ j +1 ≤ N && s+a[j] = (Σi | 0≤i<j+1 • a[i]))

= (0 ≤ j < N && s = (Σi | 0≤i<j • a[i]))
  ⇒ (**-1 ≤ j < N** && s+a[j] = (Σi | 0≤i<j+1 • a[i])) *// simplify bounds of j*

= (0 ≤ j < N && s = (Σi | 0≤i<j • a[i]))
  ⇒ (-1 ≤ j < N && s+a[j] = (Σi | 0≤i<j • a[i]) **+ a[j]**) *// separate last part of sum*

Proof Obligations

- Invariant is maintained

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}
  {0 ≤ j +1 ≤ N && **s+a[j]** = (Σi | 0≤i<j+1 • a[i]) } *// by assignment rule*
  s := s + a[j];
  {0 ≤ **j +1** ≤ N && s = (Σi | 0≤i<**j+1** • a[i]) } *// by assignment rule*
  j := j + 1;
  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }

- Need to show that:

  (0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N)
  ⇒ (0 ≤ j +1 ≤ N && s+a[j] = (Σi | 0≤i<j+1 • a[i]))

= (0 ≤ j < N && s = (Σi | 0≤i<j • a[i]))
  ⇒ (**-1 ≤ j < N** && s+a[j] = (Σi | 0≤i<j+1 • a[i])) *// simplify bounds of j*

= (0 ≤ j < N && s = (Σi | 0≤i<j • a[i]))
  ⇒ (-1 ≤ j < N && s+a[j] = (Σi | 0≤i<j • a[i]) **+ a[j]**) *// separate last part of sum*

= (0 ≤ j < N && s = (Σi | 0≤i<j • a[i]))
  ⇒ (-1 ≤ j < N && s = (Σi | 0≤i<j • a[i])) *// subtract a[j] from both sides*

Proof Obligations

- Invariant is maintained

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}
  {0 ≤ j +1 ≤ N && **s+a[j]** = (Σi | 0≤i<j+1 • a[i]) } *// by assignment rule*
  s := s + a[j];
  {0 ≤ **j +1** ≤ N && s = (Σi | 0≤i<**j+1** • a[i]) } *// by assignment rule*
  j := j + 1;
  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }

- Need to show that:

  (0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N)
  ⇒ (0 ≤ j +1 ≤ N && s+a[j] = (Σi | 0≤i<j+1 • a[i]))

= (0 ≤ j < N && s = (Σi | 0≤i<j • a[i]))
  ⇒ (**-1 ≤ j < N** && s+a[j] = (Σi | 0≤i<j+1 • a[i])) *// simplify bounds of j*

= (0 ≤ j < N && s = (Σi | 0≤i<j • a[i]))
  ⇒ (-1 ≤ j < N && s+a[j] = (Σi | 0≤i<j • a[i]) **+ a[j]**) *// separate last part of sum*

= (0 ≤ j < N && s = (Σi | 0≤i<j • a[i]))
  ⇒ (-1 ≤ j < N && s = (Σi | 0≤i<j • a[i])) *// subtract a[j] from both sides*

= **true** // 0 ≤ j ⇒ -1 ≤ j

Proof Obligations

- Invariant and exit condition implies postcondition

$0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \ \&\& \ j \ge N$

$\Rightarrow s = (\Sigma i \mid 0 \le i < N \bullet a[i])$

Proof Obligations

- Invariant and exit condition implies postcondition

$0 \le j \le N$ && $s = (\Sigma i \mid 0 \le i < j \bullet a[i])$ && $j \ge N$

$\Rightarrow s = (\Sigma i \mid 0 \le i < N \bullet a[i])$

= $0 \le j$ && **$j = N$** && $s = (\Sigma i \mid 0 \le i < j \bullet a[i])$

$\Rightarrow s = (\Sigma i \mid 0 \le i < N \bullet a[i])$

// because ($j \le N$ && $j \ge N$) = ($j = N$)

Proof Obligations

- Invariant and exit condition implies postcondition

$0 \le j \le N \;\&\&\; s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \;\&\&\; j \ge N$

$\Rightarrow s = (\Sigma i \mid 0 \le i < N \bullet a[i])$

$= \quad 0 \le j \;\&\&\; \mathbf{j = N} \;\&\&\; s = (\Sigma i \mid 0 \le i < j \bullet a[i])$

$\Rightarrow s = (\Sigma i \mid 0 \le i < N \bullet a[i])$

// because $(j \le N \;\&\&\; j \ge N) = (j = N)$

$= \quad 0 \le \mathbf{N} \;\&\&\; s = (\Sigma i \mid 0 \le i < \mathbf{N} \bullet a[i]) \Rightarrow s = (\Sigma i \mid 0 \le i < N \bullet a[i])$

// by substituting N for j, since j = N

Proof Obligations

- Invariant and exit condition implies postcondition

$0 \le j \le N \;\&\&\; s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \;\&\&\; j \ge N$

$\quad \Rightarrow s = (\Sigma i \mid 0 \le i < N \bullet a[i])$

$= \quad 0 \le j \;\&\&\; \mathbf{j = N} \;\&\&\; s = (\Sigma i \mid 0 \le i < j \bullet a[i])$

$\quad \Rightarrow s = (\Sigma i \mid 0 \le i < N \bullet a[i])$

// because $(j \le N \;\&\&\; j \ge N) = (j = N)$

$= \quad 0 \le \mathbf{N} \;\&\&\; s = (\Sigma i \mid 0 \le i < \mathbf{N} \bullet a[i]) \Rightarrow s = (\Sigma i \mid 0 \le i < N \bullet a[i])$

// by substituting N for j, since j = N

= **true** *// because P && Q ⇒ Q*

Quick Quiz

- For the program below and the invariant i <= N, write the proof obligations. The form of your answer should be three mathematical implications.

```
{ N >= 0 }

i := 0;

while (i < N) do

    i := N

{ i = N }
```

- Invariant is initially true:
- Invariant is preserved by the loop body:
- Invariant and exit condition imply postcondition:

Invariant Intuition

- For code without loops, we are simulating execution directly
  - We prove one Hoare Triple for each statement, and each statement is executed once
- For code with loops, we are doing *one* proof of correctness for *multiple* loop iterations
  - Proof must cover all iterations
    - Don't know how many there will be
  - The invariant must be *general* yet *precise*
    - general enough to be true for every execution
    - precise enough to imply the postcondition we need
  - This tension makes inferring loop invariants challenging

Total Correctness for Loops

- {P} while B do S {Q}
- Partial correctness:
  - Find an invariant Inv such that:
    - $P \Rightarrow Inv$
      - The invariant is initially true
    - { Inv && B } S {Inv}
      - Each execution of the loop preserves the invariant
    - $(Inv \;\&\&\; \neg B) \Rightarrow Q$
      - The invariant and the loop exit condition imply the postcondition
- Total correctness
  - Loop will terminate

We haven’t proven termination

- Consider the following program:

```
{ true }
i := 0
while (true) do          { true }
    i := i + 1;
{ i == -1 }
```

We haven’t proven termination

- Consider the following program:

```
{ true }
i := 0
while (true) do            { true }
    i := i + 1;
{ i == -1 }
```

- This program verifies (as partially correct)
  - Loop invariant trivially true initially and trivially preserved
  - Postcondition check:
    - (not(true) && true) => (i == -1)
    - = (false && true) => (i == -1)
    - = (false) => (i == -1)
    - = true

We haven't proven termination

- Consider the following program:

```
{ true }
i := 0
while (true) do            { true }
    i := i + 1;
{ i == -1 }
```

- This program verifies (as partially correct)
  - Loop invariant trivially true initially and trivially preserved
  - Postcondition check:
    - (not(true) && true) => (i == -1)
    - = (false && true) => (i == -1)
    - = (false) => (i == -1)
    - = true
  - Partial correctness: if the program terminates, then the postcondition will hold
    - Doesn't say anything about the postcondition if the program does not terminate—any postcondition is OK.
    - We need a stronger correctness property

Termination

```
{ N ≥ 0 }
j := 0;
s := 0;

while (j < N) do

    s := s + a[j];
    j := j + 1;

end
{ s = (Σi | 0≤i<N • a[i]) }
```

- How would you prove this program terminates?

Termination

```
{ N ≥ 0 }
j := 0;
s := 0;

while (j < N) do

    s := s + a[j];
    j := j + 1;

end
{ s = (Σi | 0≤i<N • a[i]) }
```

- How would you prove this program terminates?
- Consider the loop
  - What is the maximum number of times it could execute?
  - Use induction to prove this bound is correct

Total Correctness for Loops

- {P} while B do S {Q}
- Partial correctness:
  - Find an invariant Inv such that:
    - $P \Rightarrow Inv$
      - The invariant is initially true
    - { Inv && B } S {Inv}
      - Each execution of the loop preserves the invariant
    - $(Inv \&\& \neg B) \Rightarrow Q$
      - The invariant and the loop exit condition imply the postcondition
- Termination bound
  - Find a *variant function* v such that:
    - v is an upper bound on the number of loops remaining

Total Correctness for Loops

- {P} while B do S {Q}
- Partial correctness:
  - Find an invariant Inv such that:
    - $P \Rightarrow Inv$
      - The invariant is initially true
    - { Inv && B } S {Inv}
      - Each execution of the loop preserves the invariant
    - $(Inv \&\& \neg B) \Rightarrow Q$
      - The invariant and the loop exit condition imply the postcondition
- Termination bound
  - Find a *variant function* v such that:
    - v is an upper bound on the number of loops remaining
    - { Inv && B && v=V } S {v < V}
      - The variant function decreases each time the loop body executes

Total Correctness for Loops

- {P} while B do S {Q}
- Partial correctness:
  - Find an invariant Inv such that:
    - $P \Rightarrow Inv$
      - The invariant is initially true
    - { Inv && B } S {Inv}
      - Each execution of the loop preserves the invariant
    - $(Inv \ \&\& \ \neg B) \Rightarrow Q$
      - The invariant and the loop exit condition imply the postcondition

- Termination bound
  - Find a *variant function* v such that:
    - v is an upper bound on the number of loops remaining
    - { Inv && B && v=V } S {v < V}
      - The variant function decreases each time the loop body executes
    - $(Inv \ \&\& \ v \leq 0) \Rightarrow \neg B$
      - If we the variant function reaches zero, we must exit the loop

Total Correctness Example

```
while (j < N) do
    {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}
    s := s + a[j];
    j := j + 1;
    {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }
end
```

- Variant function for this loop?

Total Correctness Example

```
while (j < N) do
    {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N}
    s := s + a[j];
    j := j + 1;
    {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) }
end
```

- Variant function for this loop?
  - N-j

Guessing Variant Functions

- Loops with an index
  - $N \pm i$
  - Applies if you always add or always subtract a constant, and if you exit the loop when the index reaches some constant
  - Use $N-i$ if you are incrementing i, $N+i$ if you are decrementing i
  - Set N such that $N \pm i \leq 0$ at loop exit

- Other loops
  - Find an expression that is an upper bound on the number of iterations left in the loop

Additional Proof Obligations

- Variant function for this loop: N-j
- To show: variant function is decreasing

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N && N-j = V}
  s := s + a[j];
  j := j + 1;
  {N-j < V}

Additional Proof Obligations

- Variant function for this loop: N-j
- To show: variant function is decreasing

  $\{0 \leq j \leq N \;\&\&\; s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \;\&\&\; j < N \;\&\&\; N\text{-}j = V\}$

  s := s + a[j];

  j := j + 1;

  $\{N\text{-}j < V\}$

- To show: exit the loop once variant function reaches 0

  $(0 \leq j \leq N \;\&\&\; s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \;\&\&\; N\text{-}j \leq 0)$

  $\quad \Rightarrow j \geq N$

Additional Proof Obligations

- To show: variant function is decreasing

  $\{0 \leq j \leq N \; \&\& \; s = (\Sigma i \mid 0 \leq i < j \bullet a[i]) \; \&\& \; j < N \; \&\& \; N-j = V\}$

  s := s + a[j];

  j := j + 1;

  $\{N-j < V\}$

Additional Proof Obligations

- To show: variant function is decreasing

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N && N-j = V}

  s := s + a[j];

  {N-(**j+1**) < V}   *// by assignment*

  j := j + 1;

  {N-j < V}

Additional Proof Obligations

- To show: variant function is decreasing

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N && N-j = V}

  {N-(j+1) < V} *// by assignment*

  s := s + a[j];

  {N-(**j+1**) < V} *// by assignment*

  j := j + 1;

  {N-j < V}

Additional Proof Obligations

- To show: variant function is decreasing

```
{0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N && N-j = V}
{N-(j+1) < V}          // by assignment
s := s + a[j];
{N-(j+1) < V}          // by assignment
j := j + 1;
{N-j < V}
```

- Need to show:

```
(0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N && N-j = V)
      ⇒ (N-(j+1) < V)
```

Additional Proof Obligations

- To show: variant function is decreasing

  $\{0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \ \&\& \ j < N \ \&\& \ N-j = V\}$

  $\{N-(j+1) < V\}$ *// by assignment*

  s := s + a[j];

  $\{N-(\mathbf{j+1}) < V\}$ *// by assignment*

  j := j + 1;

  $\{N-j < V\}$

- Need to show:

  $(0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \ \&\& \ j < N \ \&\& \ N-j = V)$

  $\Rightarrow (N-(j+1) < V)$

Assume $0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \ \&\& \ j < N \ \&\& \ N-j = V$

Additional Proof Obligations

- To show: variant function is decreasing

```
{0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N && N-j = V}
{N-(j+1) < V}           // by assignment
s := s + a[j];
{N-(j+1) < V}           // by assignment
j := j + 1;
{N-j < V}
```

- Need to show:

(0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N && N-j = V)
    ⇒ (N-(j+1) < V)

Assume 0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N && N-j = V

By weakening we have N-j = V

Additional Proof Obligations

- To show: variant function is decreasing

  {0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N && N-j = V}
  {N-(j+1) < V} *// by assignment*
  s := s + a[j];
  {N-(**j+1**) < V} *// by assignment*
  j := j + 1;
  {N-j < V}

- Need to show:

  (0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N && N-j = V)
  ⇒ (N-(j+1) < V)

Assume 0 ≤ j ≤ N && s = (Σi | 0≤i<j • a[i]) && j < N && N-j = V

By weakening we have N-j = V

Therefore N-j-1 < V

Additional Proof Obligations

- To show: variant function is decreasing

  $\{0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \ \&\& \ j < N \ \&\& \ N-j = V\}$

  $\{N-(j+1) < V\}$ *// by assignment*

  s := s + a[j];

  $\{N-(\mathbf{j+1}) < V\}$ *// by assignment*

  j := j + 1;

  $\{N-j < V\}$

- Need to show:

  $(0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \ \&\& \ j < N \ \&\& \ N-j = V)$

  $\Rightarrow (N-(j+1) < V)$

Assume $0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \ \&\& \ j < N \ \&\& \ N-j = V$

By weakening we have $N-j = V$

Therefore $N-j-1 < V$

But this is equivalent to $N-(j+1) < V$, so we are done.

Additional Proof Obligations

- To show: exit the loop once variant function reaches 0
  $(0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \ \&\& \ N-j \le 0)$
  $\quad \Rightarrow j \ge N$

Additional Proof Obligations

- To show: exit the loop once variant function reaches 0

  $(0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \ \&\& \ N\text{-}j \le 0)$

  $\Rightarrow j \ge N$

  $(0 \le j \le N \ \&\& \ s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \ \&\& \ \mathbf{N \le j})$

  $\Rightarrow j \ge N$ *// added j to both sides*

Additional Proof Obligations

- To show: exit the loop once variant function reaches 0

  $(0 \le j \le N \;\&\&\; s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \;\&\&\; N\text{-}j \le 0)$

  $\quad \Rightarrow j \ge N$

  $(0 \le j \le N \;\&\&\; s = (\Sigma i \mid 0 \le i < j \bullet a[i]) \;\&\&\; \mathbf{N \le j})$

  $\quad \Rightarrow j \ge N$ *// added j to both sides*

= **true** *// $(N \le j) = (j \ge N)$, $P \;\&\&\; Q \Rightarrow P$*

Quick Quiz

For each of the following loops, is the given variant function correct? If not, why not?

A) Loop:

```
n := 256;
while (n > 1) do
    n := n / 2
```

Variant Function: $\log_2 n$

B) Loop:

```
n := 100;
while (n > 0) do
    if (random())
        then n := n + 1;
        else n := n – 1;
```

Variant Function: n

C) Loop:

```
n := 0;
while (n < 10) do
    n := n + 1;
```

Variant Function: -n

Session Summary

- While testing can find bugs, formal verification can assure their absence

- Hoare Logic is a mechanical approach for verifying software
  - Creativity is required in finding loop invariants, however

Further Reading

- C.A.R. Hoare. **An Axiomatic Basis for Computer Programming.** *Communications of the ACM* 12(10):576-580, October 1969.