CSC3310 Algorithms

Correctness

William Retert

Milwaukee School of Engineering

Algorithms

Algorithm a step-by-step procedure for performing a task in a finite amount of time.

Algorithms

Algorithm a step-by-step procedure for solving a computational problem in a finite amount of time.

Computational Problem Specifies in general terms relationship between input and output

Algorithms

Algorithm a step-by-step procedure for solving a computational problem in a finite amount of time.

Computational Problem Specifies in general terms relationship between input and output

- Brief description
- Constraints on input
- Desired output

Correctness

- An algorithm is *partially correct* if it always generates a correct result if/when it halts
  - "correct" meaning that the output matches the problem's specification
- An algorithm is totally correct if it is partially correct, and can be proven to terminate

  Thus, correctness proofs tend to have two parts
  - A proof of partial correctness
  - A proof of termination

Correctness

- An algorithm is *partially correct* if it always generates a correct result if/when it halts
  - “correct” meaning that the output matches the problem’s specification
- An algorithm is totally correct if it is partially correct, and can be proven to terminate
  Thus, correctness proofs tend to have two parts
  - A proof of partial correctness
  - A proof of termination

Example: Absolute Value

Name	ABSOLUTE VALUE
Input	$a \in \mathbb{Z}$
Output	$\lvert a \rvert$

1: **if** $a < 0$ **then**
2: $\quad r \leftarrow -a$
3: **else**
4: $\quad r \leftarrow a$
5: **end if**

Example: Absolute Value

Name ABSOLUTE VALUE
Input $a \in \mathbb{Z}$
Output $|a|$

1: **if** $a < 0$ **then** ▷ $a < 0$
2: $\quad r \leftarrow -a$
3: **else** ▷ $a \geq 0$
4: $\quad r \leftarrow a$
5: **end if**

Example: Absolute Value

Name ABSOLUTE VALUE
Input $a \in \mathbb{Z}$
Output $|a|$

1: **if** $a < 0$ **then** ▷ $a < 0$
2: $\quad r \leftarrow -a$ ▷ $a < 0$ and $r = -a$
3: **else** ▷ $a \geq 0$
4: $\quad r \leftarrow a$
5: **end if**

Example: Absolute Value

Name ABSOLUTE VALUE
Input $a \in \mathbb{Z}$
Output $|a|$

1: **if** $a < 0$ **then** ▷ $a < 0$
2: $r \leftarrow -a$ ▷ $a < 0$ and $r = -a$
3: **else** ▷ $a \geq 0$
4: $r \leftarrow a$ ▷ $a \geq 0$ and $r = a$
5: **end if**

Example: Absolute Value

Name ABSOLUTE VALUE
Input $a \in \mathbb{Z}$
Output $|a|$

1: **if** $a < 0$ **then** ▷ $a < 0$
2: $\quad r \leftarrow -a$ ▷ $a < 0$ and $r = -a \Rightarrow r = |a|$
3: **else** ▷ $a \geq 0$
4: $\quad r \leftarrow a$ ▷ $a \geq 0$ and $r = a \Rightarrow r = |a|$
5: **end if**

Example: Absolute Value

Name ABSOLUTE VALUE
Input $a \in \mathbb{Z}$
Output $|a|$

1: **if** $a < 0$ **then** ▷ $a < 0$
2: $\quad r \leftarrow -a$ ▷ $a < 0$ and $r = -a \Rightarrow r = |a|$
3: **else** ▷ $a \geq 0$
4: $\quad r \leftarrow a$ ▷ $a \geq 0$ and $r = a \Rightarrow r = |a|$
5: **end if** ▷ $r = |a|$

Example: Maximum Value

Name MAXIMUMVALUE
Description Find a value in an array that is $\geq$ every other value in the array
Input Array A
Output $a \in A$ such that $a \geq a_i$ for all $a_i \in A$

```
procedure FINDMAX(A)
    m ← A[0]
    i ← 1
    while i < A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Name MAXIMUMVALUE
Description Find a value in an array that is $\geq$ every other value in the array
Input Array A
Output $a \in A$ such that $a \geq a_i$ for all $a_i \in A$

```
procedure FINDMAX(A)
    m ← A[0]                          ▷ m = A[0]
    i ← 1
    while i < A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Name MAXIMUMVALUE
Description Find a value in an array that is $\geq$ every other value in the array
Input Array A
Output $a \in A$ such that $a \geq a_i$ for all $a_i \in A$

```
procedure FINDMAX(A)
    m ← A[0]                          ▷ m = A[0]
    i ← 1                             ▷ i = 1 and m = A[0]
    while i < A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Name MAXIMUMVALUE
Description Find a value in an array that is $\geq$ every other value in the array
Input Array A
Output $a \in A$ such that $a \geq a_i$ for all $a_i \in A$

```
procedure FINDMAX(A)
    m ← A[0]                                          ▷ m = A[0]
    i ← 1                                             ▷ i = 1 and m = A[0]
    while i < A.length do                             ▷ i < A.length and i = 1 and m = A[0]
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Name MAXIMUMVALUE
Description Find a value in an array that is $\geq$ every other value in the array
Input Array A
Output $a \in A$ such that $a \geq a_i$ for all $a_i \in A$

```
procedure FINDMAX(A)
    m ← A[0]                                              ▷ m = A[0]
    i ← 1                                                 ▷ i = 1 and m = A[0]
    while i < A.length do                                 ▷ i < A.length and i = 1 and m = A[0]
        if m < A[i] then                                  ▷ i < A.length and i = 1 and m = A[0] and m < A[1]
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Name MAXIMUMVALUE
Description Find a value in an array that is $\geq$ every other value in the array
Input Array A
Output $a \in A$ such that $a \geq a_i$ for all $a_i \in A$

```
procedure FINDMAX(A)
    m ← A[0]                                              ▷ m = A[0]
    i ← 1                                                 ▷ i = 1 and m = A[0]
    while i < A.length do                                 ▷ i < A.length and i = 1 and m = A[0]
        if m < A[i] then                                  ▷ i < A.length and i = 1 and m = A[0] and A[0] < A[1]
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Name MAXIMUMVALUE
Description Find a value in an array that is $\geq$ every other value in the array
Input Array A
Output $a \in A$ such that $a \geq a_i$ for all $a_i \in A$

```
procedure FINDMAX(A)
    m ← A[0]                                              ▷ m = A[0]
    i ← 1                                                 ▷ i = 1 and m = A[0]
    while i < A.length do                                 ▷ i < A.length and i = 1 and m = A[0]
        if m < A[i] then                                  ▷ i < A.length and i = 1 and m = A[0] and A[0] < A[1]
            m ← A[i]                                      ▷ i < A.length and i = 1 and m = A[1] and A[0] < A[1]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Name MAXIMUMVALUE
Description Find a value in an array that is $\geq$ every other value in the array
Input Array A
Output $a \in A$ such that $a \geq a_i$ for all $a_i \in A$

procedure FINDMAX(A)
  $m \leftarrow A[0]$ ▷ $m = A[0]$
  $i \leftarrow 1$ ▷ $i = 1$ and $m = A[0]$
  while $i <$ A.length **do** ▷ $i <$ A.length and $i = 1$ and $m = A[0]$
    if $m < A[i]$ **then** ▷ $i <$ A.length and $i = 1$ and $m = A[0]$ and $A[0] < A[1]$
      $m \leftarrow A[i]$ ▷ $i <$ A.length and $i = 1$ and $m = A[1]$ and $A[0] < A[1]$
    end if ▷ $i <$ A.length and $i = 1$ and $[(m = A[0]$ and $A[0] \geq A[1])$ or $(m = A[1]$ and $A[0] < A[1])]$
    $i \leftarrow i + 1$
  end while
  return m
end procedure

Example: Maximum Value

Name MAXIMUMVALUE
Description Find a value in an array that is $\geq$ every other value in the array
Input Array A
Output $a \in A$ such that $a \geq a_i$ for all $a_i \in A$

```
procedure FINDMAX(A)
    m ← A[0]                                                          ▷ m = A[0]
    i ← 1                                                             ▷ i = 1 and m = A[0]
    while i < A.length do                                             ▷ i < A.length and i = 1 and m = A[0]
        if m < A[i] then                                              ▷ i < A.length and i = 1 and m = A[0] and A[0] < A[1]
            m ← A[i]                                                  ▷ i < A.length and i = 1 and m = A[1] and A[0] < A[1]
        end if          ▷ i < A.length and i = 1 and [(m = A[0] and A[0] ≥ A[1]) or (m = A[1] and A[0] < A[1])]
        i ← i + 1       ▷ i < A.length and i = 2 and [(m = A[0] and A[0] ≥ A[1]) or (m = A[1] and A[0] < A[1])]
    end while
    return m
end procedure
```

Example: Maximum Value

Name MAXIMUMVALUE
Description Find a value in an array that is $\geq$ every other value in the array
Input Array A
Output $a \in A$ such that $a \geq a_i$ for all $a_i \in A$

```
procedure FINDMAX(A)
    m ← A[0]                                                          ▷ m = A[0]
    i ← 1                                                             ▷ i = 1 and m = A[0]
    while i < A.length do                                             ▷
i < A.length and i = 2 and [(m = A[0] and A[0] ≥ A[1]) or (m = A[1] and A[0] < A[1])]
        if m < A[i] then                                              ▷ i < A.length and i = 1 and m = A[0] and A[0] < A[1]
            m ← A[i]                                                  ▷ i < A.length and i = 1 and m = A[1] and A[0] < A[1]
        end if                ▷ i < A.length and i = 1 and [(m = A[0] and A[0] ≥ A[1]) or (m = A[1] and A[0] < A[1])]
        i ← i + 1             ▷ i < A.length and i = 2 and [(m = A[0] and A[0] ≥ A[1]) or (m = A[1] and A[0] < A[1])]
    end while
    return m
end procedure
```

Example: Maximum Value

Name MAXIMUMVALUE
Description Find a value in an array that is $\geq$ every other value in the array
Input Array A
Output $a \in A$ such that $a \geq a_i$ for all $a_i \in A$

procedure FINDMAX(A)
  $m \leftarrow A[0]$ ▷ $m = A[0]$
  $i \leftarrow 1$ ▷ $i = 1$ and $m = A[0]$
  while $i <$ A.length **do** ▷ $i <$ A.length and $i = 2$ and [($m = A[0]$ and $A[0] \geq A[1]$) or ($m = A[1]$ and $A[0] < A[1]$)]
    if $m < A[i]$ **then** ▷ $i <$ A.length and $i = 2$ and [($m = A[0]$ and $A[0] \geq A[1]$) or ($m = A[1]$ and $A[0] < A[1]$)] and $m < A[2]$
      $m \leftarrow A[i]$ ▷ $i <$ A.length and $i = 1$ and $m = A[1]$ and $A[0] < A[1]$
    end if ▷ $i <$ A.length and $i = 1$ and [($m = A[0]$ and $A[0] \geq A[1]$) or ($m = A[1]$ and $A[0] < A[1]$)]
    $i \leftarrow i + 1$ ▷ $i <$ A.length and $i = 2$ and [($m = A[0]$ and $A[0] \geq A[1]$) or ($m = A[1]$ and $A[0] < A[1]$)]
  end while
  return m
end procedure

Example: Maximum Value

Name MAXIMUMVALUE
Description Find a value in an array that is $\geq$ every other value in the array
Input Array A
Output $a \in A$ such that $a \geq a_i$ for all $a_i \in A$

procedure FINDMAX(A)
  $m \leftarrow A[0]$ ▷ $m = A[0]$
  $i \leftarrow 1$ ▷ $i = 1$ and $m = A[0]$
  while $i < A$.length **do** ▷
$i < A$.length and $i = 2$ and [($m = A[0]$ and $A[0] \geq A[1]$) or ($m = A[1]$ and $A[0] < A[1]$)]
    if $m < A[i]$ **then** ▷
$i < A$.length and $i = 2$ and [($m = A[0]$ and $A[0] \geq A[1]$) or ($m = A[1]$ and $A[0] < A[1]$)] and $m < A[2]$
      $m \leftarrow A[i]$ ▷ $i < A$.length and $i = 1$ and $m = A[1]$ and $A[0] < A[1]$
    end if ▷ $i < A$.length and $i = 1$ and [($m = A[0]$ and $A[0] \geq A[1]$) or ($m = A[1]$ and $A[0] < A[1]$)]
    $i \leftarrow i + 1$ ▷ $i < A$.length and $i = 2$ and [($m = A[0]$ and $A[0] \geq A[1]$) or ($m = A[1]$ and $A[0] < A[1]$)]
  end while
  return m
end procedure

We need a way to *generalize* loop behavior

Loop Invariants

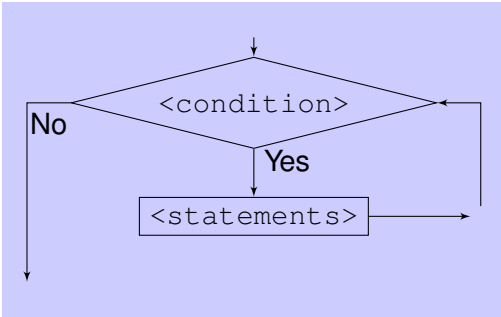
- Property that is **true** before each iteration of a loop
  - Can be broken in the loop body
  - Must be restored by end of loop
- Describe state during loop execution
  - What is the loop doing?
  - How does it accomplish that?

  Describe the state *at that moment!*
  - Do not summarize the behavior of the loop
- Inductive
  - Generally in terms of the loop counter
  - i out of n elements processed after iteration i

Loop Invariants

- Property that is **true** before each iteration of a loop
  - Can be broken in the loop body
  - Must be restored by end of loop
- Describe state during loop execution
  - What is the loop doing?
  - How does it accomplish that?

  Describe the state *at that moment!*
  - Do not summarize the behavior of the loop
- Inductive
  - Generally in terms of the loop counter
  - *i* out of *n* elements processed after iteration *i*



Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i < A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]                                  ▷ m = A[0]
    i ← 1
    while i < A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]                          ▷ m = A[0]
    i ← 1                             ▷ i = 1 and m = A[0]
    while i < A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]                                  ▷ m = A[0]
    i ← 1                                     ▷ i = 1 and m = A[0]

    while i < A.length do                     ▷ Invariant: i < n + 1
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]                                  ▷ m = A[0]
    i ← 1                                     ▷ i = 1 and m = A[0]

    while i < A.length do                     ▷ Invariant: i < n + 1
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1                             ▷ i < A.length and i < n + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]                                  ▷ m = A[0]
    i ← 1                                     ▷ i = 1 and m = A[0]

    while i < A.length do                     ▷ Invariant: i < n + 1
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1                             ▷ i < n and i < n + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]                                   ▷ m = A[0]
    i ← 1                                      ▷ i = 1 and m = A[0]

    while i < A.length do                      ▷ Invariant: i < n + 1
        if m < A[i] then                       ▷ i < n and i < n + 1
            m ← A[i]
        end if
        i ← i + 1                              ▷ i − 1 < n and i − 1 < n + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]                                  ▷ m = A[0]
    i ← 1                                     ▷ i = 1 and m = A[0]

    while i < A.length do                     ▷ Invariant: i < n + 1
        if m < A[i] then                      ▷ i < n and i < n + 1
            m ← A[i]
        end if
        i ← i + 1                             ▷ i − 1 < n and i − 1 < n + 1
                                              ▷ i − 1 < n ⇒ i < n + 1
    end while
    return m
end procedure
```

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]                                   ▷ m = A[0]
    i ← 1                                      ▷ i = 1 and m = A[0]

    while i < A.length do                      ▷ Invariant: i < n + 1
        if m < A[i] then                       ▷ i < n and i < n + 1
            m ← A[i]
        end if
        i ← i + 1                              ▷ i − 1 < n and i − 1 < n + 1

    end while                                  ▷ i − 1 < n ⇒ i < n + 1
    return m                                   ▷ i ≥ n and i < n + 1
end procedure
```

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]                                          ▷ m = A[0]
    i ← 1                                             ▷ i = 1 and m = A[0]

    while i < A.length do                             ▷ Invariant: i < n + 1
        if m < A[i] then                              ▷ i < n and i < n + 1
            m ← A[i]
        end if
        i ← i + 1                                     ▷ i − 1 < n and i − 1 < n + 1

    end while                                         ▷ i − 1 < n ⇒ i < n + 1
                                                      ▷ i ≥ n and i < n + 1
    return m                                          ▷ ⇒ i = n
end procedure
```

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]                                   ▷ m = A[0]
    i ← 1                                      ▷ i = 1 and m = A[0]
                                               ▷ Invariant: i < n + 1
    while i < A.length do                      ▷ i < n and i < n + 1
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1                              ▷ i − 1 < n and i − 1 < n + 1
                                               ▷ i − 1 < n ⇒ i < n + 1
    end while                                  ▷ i ≥ n and i < n + 1
                                               ▷ ⇒ i = n
    return m                                   ▷ ????
end procedure
```

What is the problem with this invariant?

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$

$$m = 3$$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 3$$
$$i = 1$$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$

$$m = 3$$

$$i = 1$$

Invariant: $m = a_0$?

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 3$$
$$i = 1$$

Invariant: $m = a_0$?
Test is **true**; $1 < 4$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 3$$
$$i = 1$$

Invariant: $m = a_0$?
Test is **true**; $3 < 5$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$

$$m = 5$$

$$i = 1$$

Invariant: $m = a_0$?

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 2$$

Invariant: $m = a_0$?

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 2$$

Invariant: $m = a_0? \sqcap m = a_1 \geq a_0?$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 2$$

Invariant: $m \geq a_0? \sqcap m = a_1 \geq a_0?$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 2$$

Invariant: $m \geq a_0$?
AND $m \geq a_0$ and $m \geq a_1$?

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 2$$

Invariant: $m \geq a_j$ for $0 \leq j < i$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 2$$

Invariant: $m \geq a_j$ for $0 \leq j < i$
Test is **true**; $2 < 4$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 2$$

Invariant: $m \geq a_j$ for $0 \leq j < i$
Test is **false**; $5 \not< 2$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 3$$

Invariant: $m \geq a_j$ for $0 \leq j < i$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 3$$

Invariant: $m \geq a_j$ for $0 \leq j < i$ Still true for $i = 3$?

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 3$$

Invariant: $m \geq a_j$ for $0 \leq j < i$
Test is **true**; $3 < 4$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 3$$

Invariant: $m \geq a_j$ for $0 \leq j < i$
Test is **false**; $5 \not< 4$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 4$$

Invariant: $m \geq a_j$ for $0 \leq j < i$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 4$$

Invariant: $m \geq a_j$ for $0 \leq j < i$ Still true for $i = 4$?

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 4$$

Invariant: $m \geq a_j$ for $0 \leq j < i$
Test is **false**; $4 \not< 4$

Example: Maximum Value

Loop Invariant:

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- Inductive — i out of n elements processed after iteration i

```
procedure FindMax(A)
    m ← A[0]
    i ← 1
    while i ≠ A.length do
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

$$A = \langle 3, 5, 2, 4 \rangle$$
$$m = 5$$
$$i = 4$$

Invariant: $m \geq a_j$ for $0 \leq j < i$
$i = 4 = n$, so m is at least as large as every array element

Example: Maximum Value

- Property that is **true** before each iteration of a loop
- Describe state during loop execution

```
procedure FindMax(A)
    m ← A[0]
    i ← 1

    while i ≠ A.length do                ▷ Invariant: m ≥ a_j for all 0 ≤ j < i
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Example: Maximum Value

- Property that is **true** before each iteration of a loop
  To prove, show is true:
  - At the start of the loop (first iteration)
  - At the end of the loop (later iterations)

```
procedure FindMax(A)
    m ← A[0]                                   ▷ m = a_0
    i ← 1                                      ▷ i = 1 and m = a_0

    while i ≠ A.length do                      ▷ Invariant: m ≥ a_j for all 0 ≤ j < i
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Example: Maximum Value

- Property that is **true** before each iteration of a loop
  To prove, show is true:
  - At the start of the loop (first iteration)
  - At the end of the loop (later iterations)

```
procedure FindMax(A)
    m ← A[0]
    i ← 1

    while i ≠ A.length do                          ▷ Invariant: m ≥ a_j for all 0 ≤ j < i

        if m < A[i] then
            m ← A[i]
        end if

        i ← i + 1                                  ▷ i ≠ n and (m ≥ a_j for all 0 ≤ j < i)

    end while                                      ▷ m ≥ a_j for all 0 ≤ j ≤ i
    return m
end procedure                                      ▷ m ≥ a_j for all 0 ≤ j < i
```

Example: Maximum Value

- Describe state during loop execution
- *Use* invariant to help show output postcondition

```
procedure FindMax(A)
    m ← A[0]
    i ← 1

    while i ≠ A.length do                    ▷ Invariant: m ≥ a_j for all 0 ≤ j < i
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while

    return m                                 ▷ i = n and (m ≥ a_j for all 0 ≤ j < i)
end procedure                                ▷ m ≥ a_j for all 0 ≤ j < n
```

Example: Maximum Value

- Property that is **true** before each iteration of a loop
- Describe state during loop execution
- *Use* invariant to help show output postcondition
- Separately, show that the loop terminates

```
procedure FindMax(A)
    m ← A[0]
    i ← 1

    while i ≠ A.length do                ▷ Invariant: m ≥ a_j for all 0 ≤ j < i
        if m < A[i] then
            m ← A[i]
        end if
        i ← i + 1
    end while
    return m
end procedure
```

Proving Correctness

- To formally prove correctness, need the step-by-step walkthrough
- For class purposes, a less-formal argument will often suffice
  - Clearly state the loop invariant

> $m \geq a_j$ for all $0 \leq j < i$

  - Argue convincingly that it is true at the start of the loop

> Initially, i is 1, and m is a_0. The only j less than 1 is 0, and $a_0 \geq a_0$.

  - Argue convincingly that it is preserved across an arbitrary iteration of the loop.

> We know from the loop invariant that all values up to a_i are no larger than m. If the i^{th} value is also less-than-or-equal-to m, then the invariant will hold after incrementing i. If the i^{th} value is larger than, then we update m, making it equal to a_i and transitively greater than all elements before a_i, so again the invariant holds.