Writing Invariants

Invariants

- An **invariant** is a statement that is true during all steps of a procedure.
- In other words, your invariant should be true at Step 0, Step 1, and Step 1000 of your algorithm.
- If the invariant holds for an arbitrary step of your algorithm, this means it'll always hold! (What does that sound like?)
- If the invariant always holds, this means that it'll hold at the last step, which tells us something about our output!

Saying Something About Your Output (Without an Invariant)

My algorithm takes as input [Input(s)]

and outputs [Output(s)]

such that:

[Properties of Output(s)]

Saying Something About Your Output (Without an Invariant)

My algorithm takes as input $\text{list } A$

and outputs $\text{max\_idx} \in [0, length(A) - 1]$

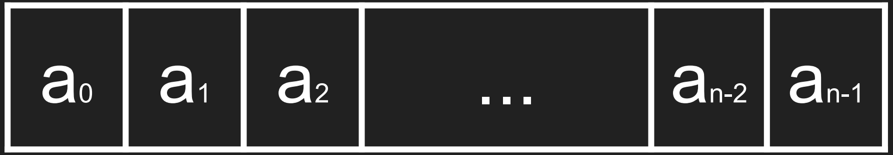
such that:

$$\forall j \in [0, length(A) - 1],\ A[j] \leq A[\text{max\_idx}]$$

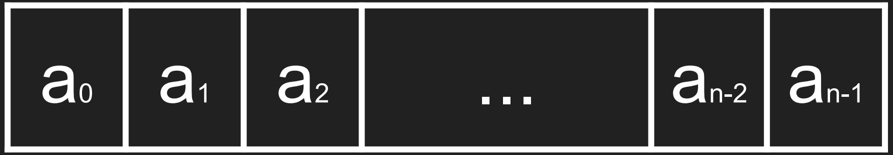
Writing Invariants

An invariant says something about your algorithm at *every step* of execution, not just at output.

```
1. INPUT: list A
2. max_idx = 0
3. max_elem = A[0]
4. n = length(A)
5. FOR i in range[1,n-1]:
6.     IF A[i] > max_elem:
7.         max_elem = A[i]
8.         max_idx = i
9. RETURN max_idx
```



```
1. INPUT: list A
2. max_idx = 0
3. max_elem = A[0]
4. n = length(A)
5. FOR i in range[1,n-1]:
6.     IF A[i] > max_elem:
7.         max_elem = A[i]
8.         max_idx = i
9. RETURN max_idx
```



Invariant: $\exists\ \text{max\_idx} \in [0, i-1],$
$\forall j \in [0, i-1], A[j] \leq A[\text{max\_idx}]$

Writing Invariants

An invariant says something about your algorithm at *every step* of execution, not just at output.

My algorithm takes as input [Input(s)]

And behaves such that [Invariant Definition]

[Properties of Invariant]

Writing Invariants

An invariant says something about your algorithm at *every step* of execution, not just at output.

My algorithm takes as input $\text{list } A$

And behaves such that $\forall i \in [0, length(A)]$

$$\exists \text{ max\_idx} \in [0, i-1],$$
$$\forall j \in [0, i-1], A[j] \leq A[\text{max\_idx}]$$

Indexing In Invariants

$$\forall i \in [0, \boxed{length(A)}], \exists \text{ max\_idx} \in [0, i-1],$$
$$\forall j \in [0, i-1], A[j] \leq A[\text{max\_idx}]$$

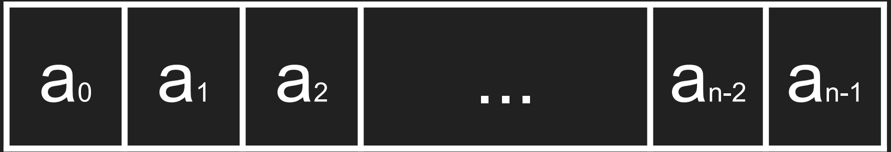
If the elements of A only go up to $A[n-1]$,

why does i go up to n?

```
1. INPUT: list A
2. max_idx = 0
3. max_elem = A[0]
4. n = length(A)
5. FOR i in range[1,n-1]:
6.     IF A[i] > max_elem:
7.         max_elem = A[i]
8.         max_idx = i
9. RETURN max_idx
```

Invariant: $\exists\ \text{max\_idx} \in [0, i-1]$,
$\forall j \in [0, i-1], A[j] \leq A[\text{max\_idx}]$

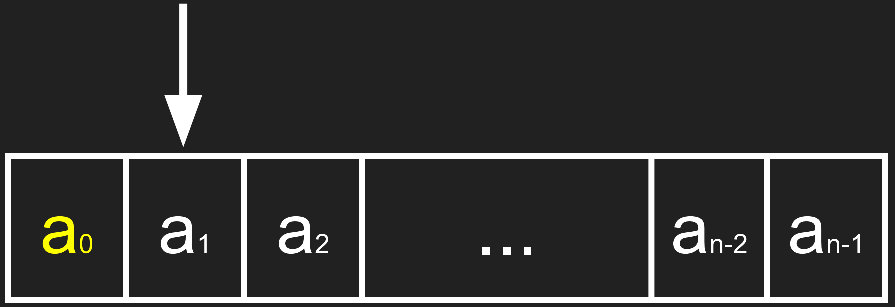
```
1. INPUT: list A
2. max_idx = 0
3. max_elem = A[0]
4. n = length(A)
5. FOR i in range[1,n-1]:
6.     IF A[i] > max_elem:
7.         max_elem = A[i]
8.         max_idx = i
9. RETURN max_idx
```



Invariant: $\exists\ \text{max\_idx} \in [0, i-1],$
$\forall j \in [0, i-1], A[j] \leq A[\text{max\_idx}]$

```
1. INPUT: list A
2. max_idx = 0
3. max_elem = A[0]
4. n = length(A)
5. FOR i in range[1,n-1]:
6.     IF A[i] > max_elem:
7.         max_elem = A[i]
8.         max_idx = i
9. RETURN max_idx
```

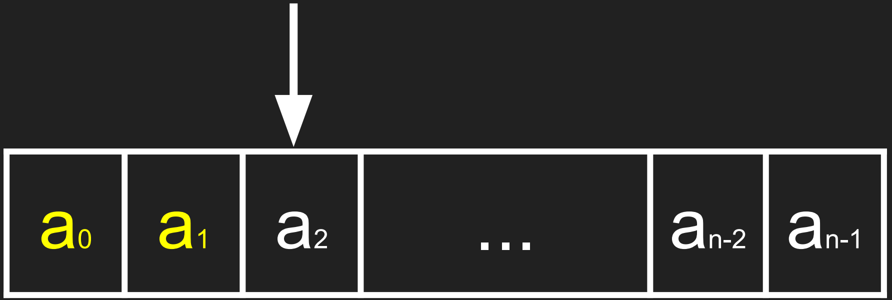
Base case: $i = 1$



Invariant: $\exists$ max_idx $\in [0, 0]$,
$\forall j \in [0, 0], A[j] \leq A[\text{max\_idx}]$

```
1. INPUT: list A
2. max_idx = 0
3. max_elem = A[0]
4. n = length(A)
5. FOR i in range[1,n-1]:
6.     IF A[i] > max_elem:
7.         max_elem = A[i]
8.         max_idx = i
9. RETURN max_idx
```

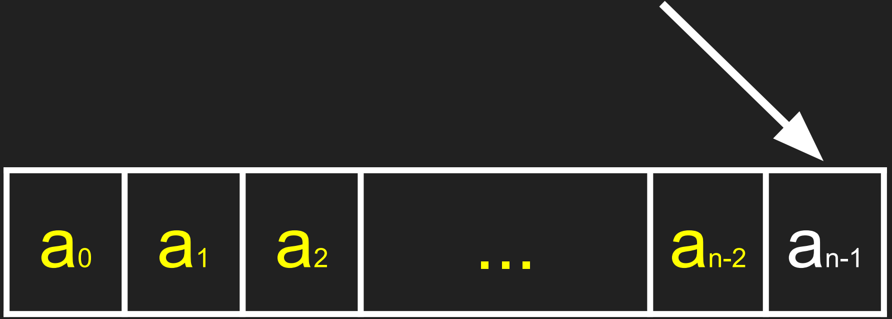
$$i = 2$$



Invariant: $\exists\, \text{max\_idx} \in [0, 1]$,
$\forall j \in [0, 1], A[j] \leq A[\text{max\_idx}]$

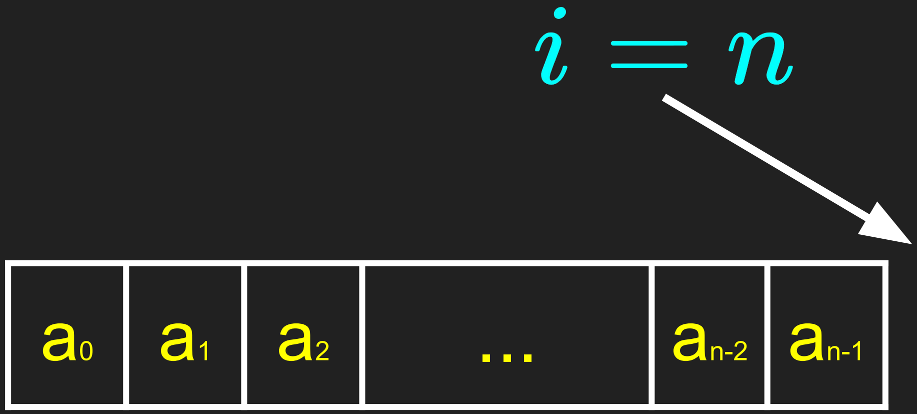
```
1. INPUT: list A
2. max_idx = 0
3. max_elem = A[0]
4. n = length(A)
5. FOR i in range[1,n-1]:
6.     IF A[i] > max_elem:
7.         max_elem = A[i]
8.         max_idx = i
9. RETURN max_idx
```

$i = n - 1$



Invariant: $\exists$ max_idx $\in [0, n-2]$,
$\forall j \in [0, n-2], A[j] \leq A[\text{max\_idx}]$

```
1. INPUT: list A
2. max_idx = 0
3. max_elem = A[0]
4. n = length(A)
5. FOR i in range[1,n-1]:
6.     IF A[i] > max_elem:
7.         max_elem = A[i]
8.         max_idx = i
9. RETURN max_idx
```



Invariant: $\exists\, \text{max\_idx} \in [0, n-1]$, $\forall j \in [0, n-1], A[j] \leq A[\text{max\_idx}]$

```
1. INPUT: list A
2. max_idx = 0
3. max_elem = A[0]
4. n = length(A)
5. FOR i in range[1,n-1]:
6.     IF A[i] > max_elem:
7.         max_elem = A[i]
8.         max_idx = i
9. RETURN max_idx
```

Loop invariant:

$\forall i \in [0, length(A)],$

$\exists$ max_idx $\in [0, i-1],$

$\forall j \in [0, i-1], A[j] \leq A[\text{max\_idx}]$

a_0
a_1
a_2
…
a_{n-2}
a_{n-1}

Invariant: $\exists$ max_idx $\in [0, n-1],$

$\forall j \in [0, n-1], A[j] \leq A[\text{max\_idx}]$