Proofs by induction → proving running time, correctness for recursive algo.

Set Sort: pick max → put in last pos.

"Silly" Selection Sort recursive version

Algorithm SelectionSort (a, n)

Input: Array a of size n
Output: a should be sorted

// "Base case"
if n <= 1 return

int indmax ← find MaxIndex (a, m)
swap (a, indmax, m)
SelectionSort (a, m-1)

"Tail recursion" → consider terms such things into loops (for)

"Base case" for code → "base case for induction recursive call → induction step"
Proof by induction

For all natural numbers \( n \), \( P(n) \) is true

\[ 1 + 2 + \cdots + n = \frac{n(n+1)}{2} \quad \forall n \geq 1 \]

Show \( P(1) \) true \( (\text{base case}) \)

Show if \( P(n) \) true, then \( P(n+1) \) must also be true \( (\text{induction step}) \)

Later on: do this with more complicated structures

E.g. Show that \( \sum_{i=1}^{n} i = \frac{n(n+1)}{2} \)

Base case: \( m = 1 \)

\[ 1 = \frac{1 \cdot 2}{2} \quad \text{Yay!} \]

Induction step: suppose \( \sum_{i=1}^{m} i = \frac{m(m+1)}{2} \) "\( P(m) \)"

to show \( \sum_{i=1}^{m+1} i = \frac{(m+1)(m+2)}{2} \) "\( P(m+1) \)"

\[ \sum_{i=1}^{m+1} i = \sum_{i=1}^{m} i + (m+1) = \frac{m(m+1)}{2} + (m+1) = \frac{(m+1)(m+2)}{2} \]

\[ \text{ind hyp} \]
E.g. Show that $2^m \leq m! \quad \forall m \geq 4$

$m! = 1 \cdot 2 \cdot 3 \cdots m$

Base case: $m = 4 \quad 2 \cdot 2 \cdot 2 \cdot 2 \leq 1 \cdot 2 \cdot 3 \cdot 4\quad 16 \leq 24 \quad \text{yay!}$

Induction step: Suppose $2^m \leq m!$

To show: $2^{m+1} \leq (m+1)!$

$2^m \cdot 2 \leq m! \cdot (m+1)$

We know from inductive hyp $2^m \leq m!$

$2 \leq (m+1)$

$2^{m+1} \leq (m+1)! \quad \text{yay!}$
E.g., prove that \( 1^3 + 2^3 + \ldots + n^3 = (1+2+\ldots+n)^2 \)

**Base case:** \( n = 1 \) \( 1^3 = 1^2 \) **Yay!**

**Induction step:** Suppose \( \sum_{i=1}^{m} i^3 = \left( \sum_{i=1}^{m} i \right)^2 \)

\[
\sum_{i=1}^{m+1} i^3 = \sum_{i=1}^{m} i^3 + (m+1)^3 = \left( \sum_{i=1}^{m} i \right)^2 + (m+1)^3
\]

and hyp

\[
= \left( \sum_{i=1}^{m} i \right)^2 + (m+1) \left( \sum_{i=1}^{m} i \right)^2 = (m+1) (m+1) \left( \sum_{i=1}^{m} i \right)^2
\]

Assume for all \( \sum_{i=1}^{m} i + (m+1)^3 \)

\[
= \left( \sum_{i=1}^{m} i \right)^2 + (m+1)^2 + \frac{m(m+1)(m+1)}{2} + \frac{m(m+1)}{2} \cdot \frac{2}{2}
\]

From before: \( \frac{m(m+1)}{2} = \sum_{i=1}^{m} i \)

\[
= \left( \sum_{i=1}^{m} i \right)^2 + (m+1)^2 + \frac{m(m+1)(m+1)}{2} + \frac{m(m+1)}{2} \cdot \frac{2}{2}
\]

\[
= \left( \sum_{i=1}^{m} i + (m+1) \right)^2 = \left( \sum_{i=1}^{m} i \right)^2 \quad **Yay!**
\]
Blueprint for proving correctness of recursive algos

Prove base case is correct

"Put our faith in the recursion!"
Assume recursive call worked correctly → prove that when we called from else works

Sorting: \( a[j] \leq a[i] \quad \forall j \leq i \) Correctness property

Base case: \( n = 1 \) "Smallest input structure"
Typically true

Induction step:

\[
\text{int \; indmax} \leftarrow \text{find Max index} (a, m)
\]

\[
\rightarrow a[\text{indmax}] \geq a[i] \quad \forall i \neq \text{indmax} \quad \text{by correctness of find Max index}
\]

\[
\text{swap} (a, m, \text{indmax})
\]

\[
\rightarrow a[m] \geq a[i] \quad \forall i \leq m
\]

Selection Sort \((a, m-1)\)

- Induction hypothesis: Rec. call worked so:

\[
\forall j \leq i, \; i \leq m-1, \; a[j] \leq a[i]
\]

\[
\rightarrow \forall j \leq i, \; i \leq m \quad a[j] \leq a[i]
\]
Comparing this proof technique to loop invariants, this is more "automated" (no need to guess at a loop invariant).

What is true before/after each piece (e.g., calls etc.)

what is true before: precondition
what is true after: postcondition

At any point, we have a "collection" of statements true so far → deduce more ones

Programming languages may even produce proofs in some cases (e.g., at McGill on this)