Induction and Recursion

Guest Lecturer:

Fatemeh Asgarinejad (PhD Candidate at UCSD)

Useful Links

- Chapter # in Rosen’s Discrete Mathematics and Its application book: 5
- GitHub (slides) https://github.com/Fasgarinejad/Machine-Learning

Prerequisites:

Basic arithmetic in mathematics

Quantifiers

Basis logic

Definition of functions

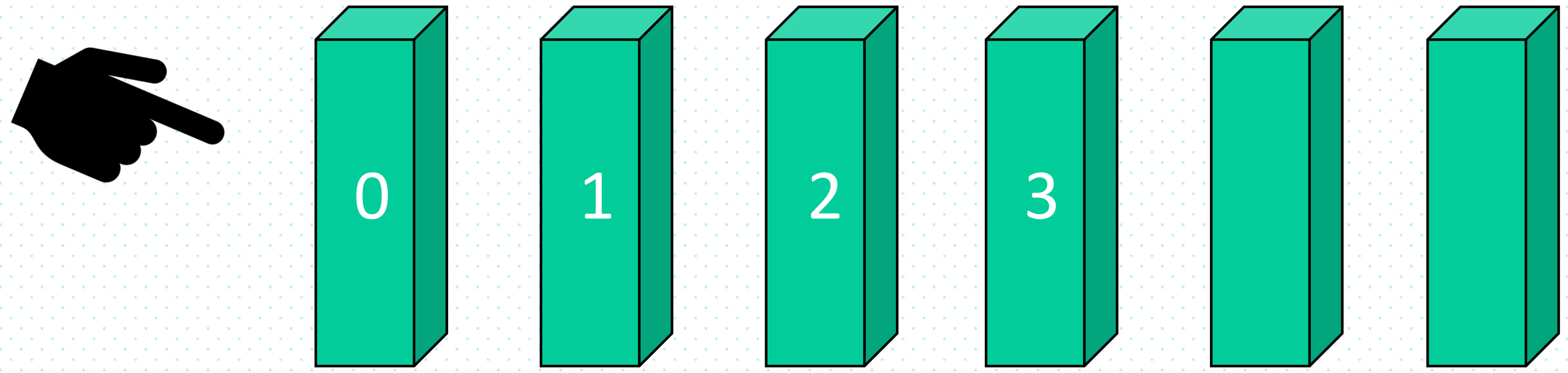
Merge Sort

Basic programming

Today’s Learning Goals: Induction and Recursion

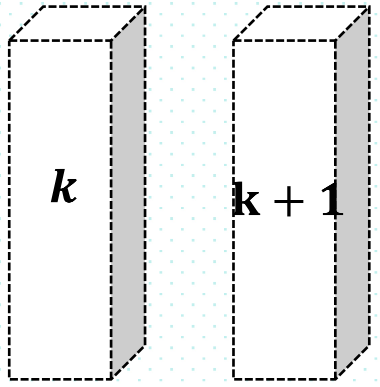
- Principles of proof by induction
    - Mathematical Example
    - Computer Science Example
- Common mistakes made by CS majors in a proof by induction
- Relationship between Recursion and Induction
- Q&A

Dominoes



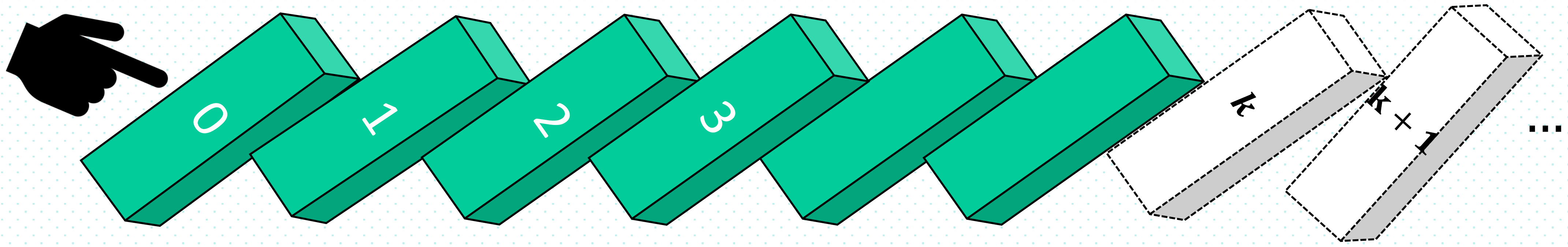
Assume equally spaced dominoes, and assume that spacing between two dominoes is less than a domino length.

How do you argue that all dominoes would fall?



- Domino 0 falls because we push it over.
- Domino 1 falls because domino 0 falls (domino 0 knocks over domino 1).
- Domino 2 falls because domino 1 falls (domino 1 knocks over domino 2).
- …
- Domino $k+1$ falls because domino k falls (domino k knocks over domino $k+1$)

Dominoes



Assume equally spaced dominoes, and assume that spacing between two dominoes is less than a domino length.

How do you argue that all dominoes would fall?

- Domino 0 falls because we push it over.
- Domino 1 falls because domino 0 falls (domino 0 knocks over domino 1).
- Domino 2 falls because domino 1 falls (domino 1 knocks over domino 2).
- …
- Domino $k+1$ falls because domino k falls (domino k knocks over domino $k+1$)
- **So, we can argue that all dominoes will fall.**

Induction

In dominoes example, p is falling of the first domino

From modus ponens (affirming the antecedents):

$$\begin{array}{r} p \text{ basis assertion} \\ p \to q \text{ conditional assertion} \\ \hline q \text{ conclusion} \end{array}$$

q: falling of second domino
If the first domino falls, the the second one falls.

We can easily derive "double modus ponens"

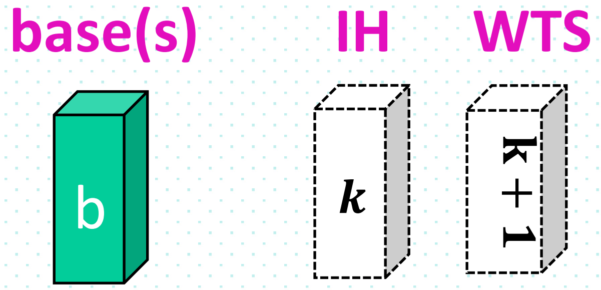
$$\begin{array}{r} p_0 \text{ basis assertion} \\ p_0 \to p_1 \text{ conditional assertion} \\ p_1 \to p_2 \text{ conditional assertion} \\ \hline p_2 \text{ conclusion} \end{array}$$

We might also derive triple and quadruple modus ponens, and so on.
If any of the first dominoes falls, then so does its successor → all of the successors will fall.

Mathematical Induction

Assume you have a set **S** and you want to prove that for all elements of this set, a statement holds ($\forall s \in S\ statement(s)$), if you can fall the first domino (statement holds for first element of the set), and if you can show that falling of the k^{th} domino will result in falling of its successor $k+1$, then you are using induction to prove $\forall s \in S\ statement(s)$.

$\boldsymbol{element}$	$\boldsymbol{s_0}$	$\boldsymbol{s_1}$	…	$\boldsymbol{s_k}$	$\boldsymbol{s_{k+1}}$	…	$\boldsymbol{s_n}$
Statement	✓	✓	…	$\longrightarrow$		…	✓



To prove a universal quantification over the set of all integers greater than or equal to some base integer b:
Basis Step: Show the property holds for b.
Inductive Hypothesis (IH): Consider an arbitrary integer k greater than or equal to b, assume that the property holds for k, and
Inductive Step: Use Inductive Hypothesis and other facts to prove property holds for $k+1$.
Now you have proven the statement for all integers $\geq$ base case(s).

Mathematical Induction

So can I use induction for proving any universal quantified statement $\forall s \in S\ P(s)$?

Note: $P(s)$ is a predicate. A statement with variable(s) that is True or False but not both.

No, take the following statement:

"For all real numbers r, if $r^2 + r$ is not an even integer then r is not an integer."

It is defined over real numbers rather than integers (no clear next element in the set of real numbers as it is uncountable and there is no natural progression from r to $r + 1$). Also, we do not have the base case (first domino to fall).
We can use contrapositive proof for the above statement.

Proof by Mathematical Induction Example

Example: Prove that for all $n \geq 5, 2^n > n^2$ $(\forall n \geq 5\ \ 2^n > n^2)$

Basis Step: We show that the given statement $(2^{\mathrm{n}} > \mathrm{n}^2)$ for $n = 5$: $2^5 > 5^2$ **base(s)**

Inductive Hypothesis: Assume $2^k > k^2$ for some $k \geq 5$ **IH**

Inductive Step: We want to show that assuming the statement holds for k, it will holds for $k+1$:

$$(\mathrm{i.e.}, 2^{\mathrm{k}+1} > (\mathrm{k}+1)^2)$$ **WTS**

$$
\begin{aligned}
2^{k+1} &= 2.2^k && \text{arithmetic} \\
&= 2^k + 2^k && \text{arithmetic} \\
&> k^2 + k^2 && \text{Induction Hypothesis} \\
&> k^2 + (2k+1) && (k \geq 5) \\
&= (k+1)^2 && \text{arithmetic} \quad \text{QED}
\end{aligned}
$$

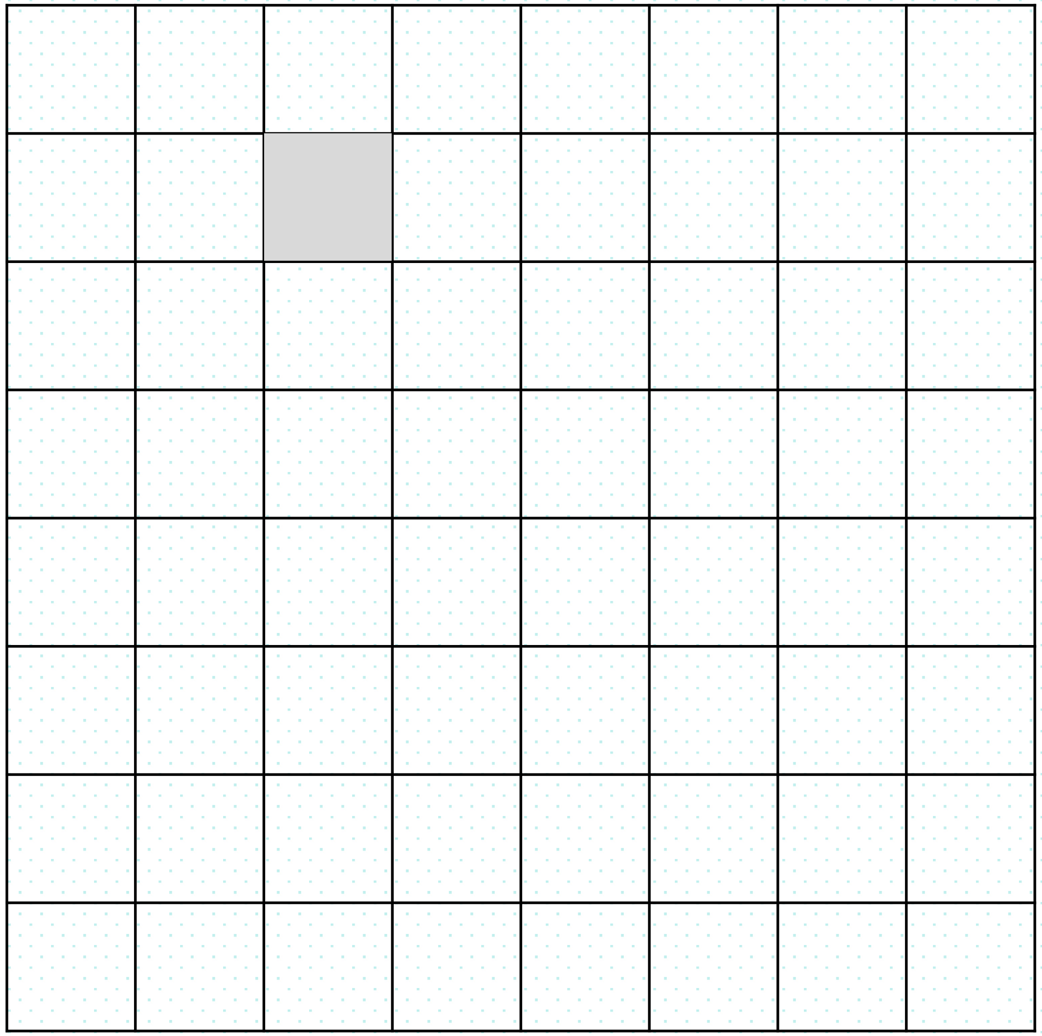
Proof by Mathematical Induction (More Formal)

In order to prove that $\forall n \in \mathbb{Z}^{\geq b}\ P(n)$ (for some predicate $P(n)$), then
Basis Step: Show $P(b)$
Inductive Hypothesis: Assume for some arbitrary k**,** $\mathrm{P(k)}$
Inductive Step: Prove that $P(k) \rightarrow P(k+1)$ ($P(k)$ infers $P(k+1)$).
Then you can claim that $\forall n \in \mathbb{Z}^{\geq b}\ P(n)$.

Note: For proving the implication $p \rightarrow q$, we assume p holds, and prove q.

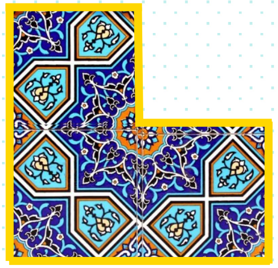
Proof by Mathematical Induction Example (Nonalgebraic)

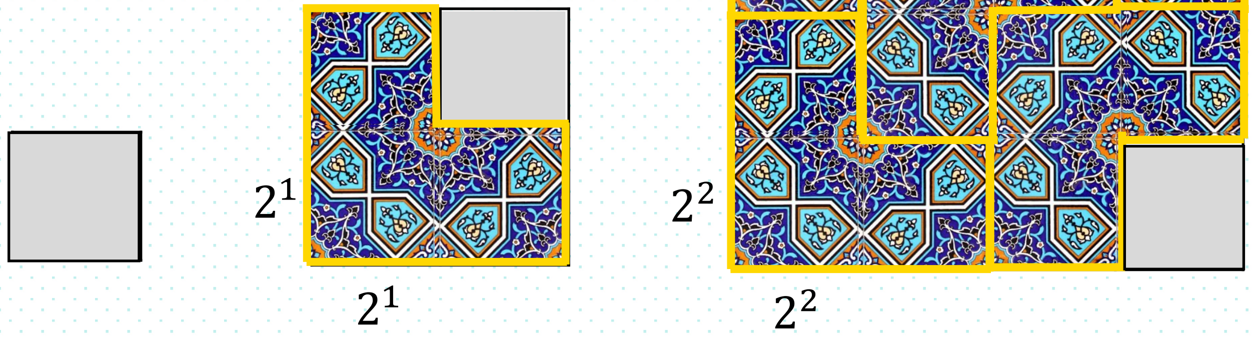
Prove that any 2^n by 2^n chessboard with one square removed can be tiled using the following L-shaped tiles. In the following chessboard, $n = 3$



Proof by Mathematical Induction Example (Nonalgebraic)

Prove that any 2^n by 2^n chessboard with one square removed can be tiled using the following L-shaped tiles. In the following chessboard, $n = 3$





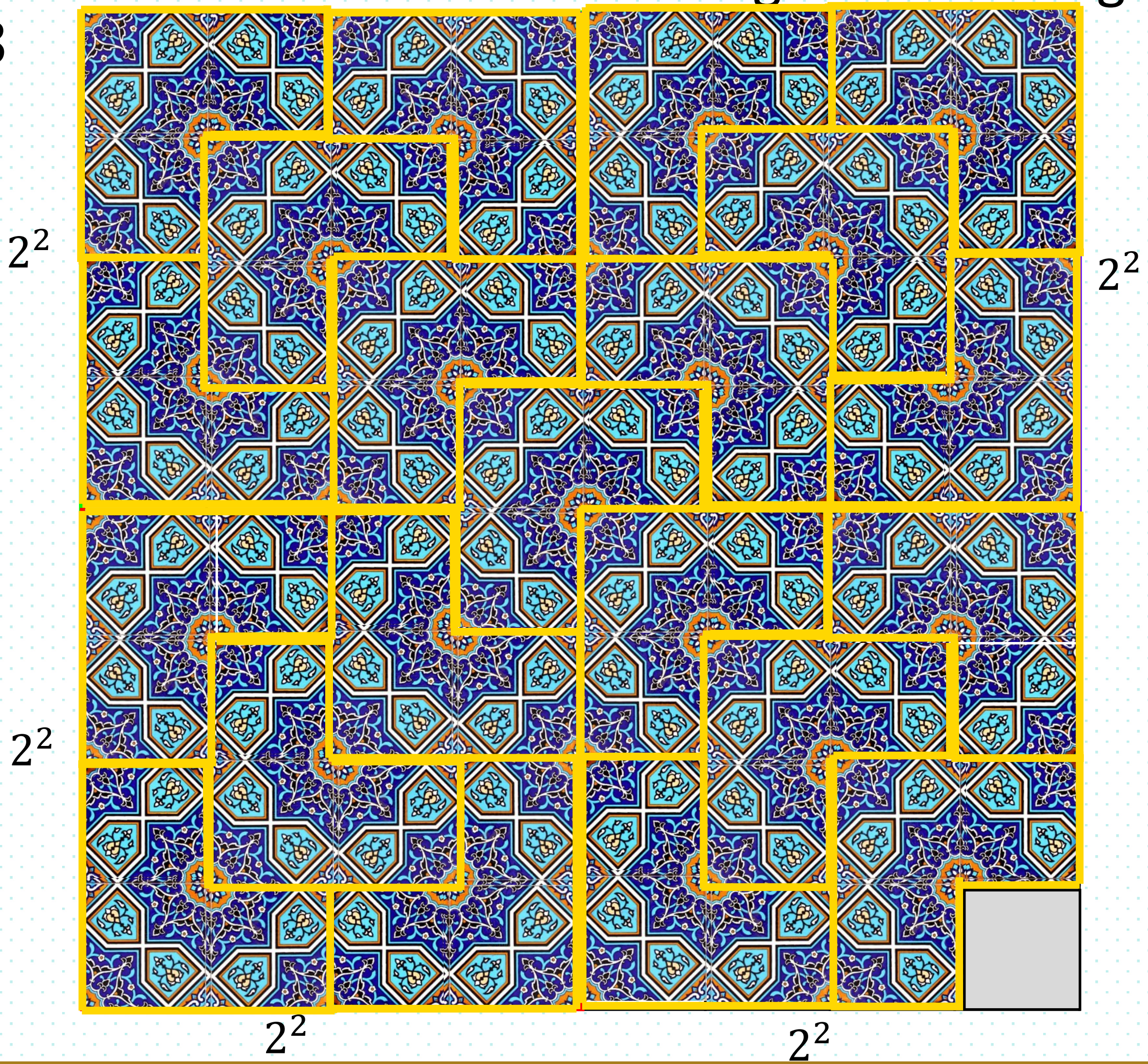
$\boldsymbol{n = 0}$ $\boldsymbol{n = 1}$ $\boldsymbol{n = 2}$

Being inspired with the $2^1 by\ 2^1$ chessboard, tile the $2^2 by\ 2^2$ chessboard. Remember you van have one square removed ⬜

And, you are allowed to rotate the L-shaped tile.

Proof by Mathematical Induction Example (Nonalgebraic)

Prove that any 2^n by 2^n chessboard with one square removed can be tiled using the following L-shaped tiles. In the following chessboard, $n = 3$



Proof by Mathematical Induction Example (Nonalgebraic)

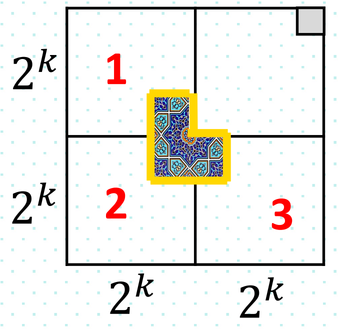
Prove that any 2^n by 2^n chessboard with one square removed can be tiled using the following L-shaped tiles. In the following chessboard, $n = 3$

Basis Step: We show that for base $n = 0$, we can tile a $2^0 by$ 2^0 by L-shaped tiles wit one square removed. **base(s)**

Inductive Hypothesis: Assume for some arbitrary k, we can tile any 2^k by 2^k chessboard with one removed square. **IH**

Inductive Step: We want to show that assuming the statement for k, it will holds for $k + 1$, meaning, we can tile any $2^{k+1}$by 2^{k+1}. **WTS**

Proof of inductive step: divide the $2^{k+1} by$ 2^{k+1} to four $2^k by 2^k$ squares. Put an L-shaped tile in the middle of this square as following

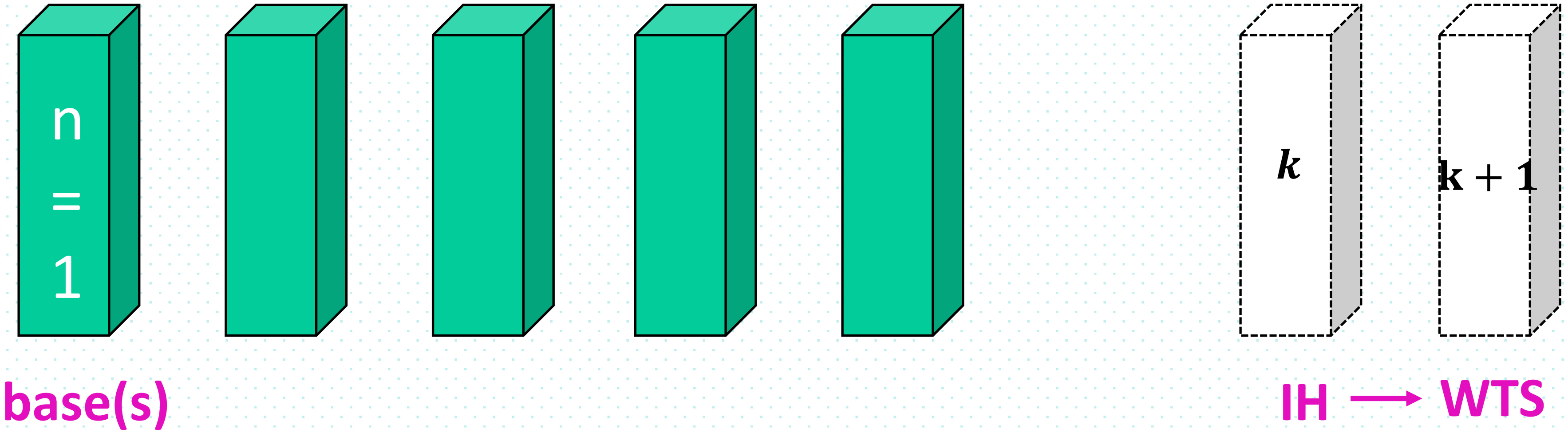


Based on the **IH**, we can tile the areas 1, 2 and 3 and for the remaining area, we can tile it with one square removed.

Therefore we can tile any $2^n by 2^n chessboard$. QED

Proof by Mathematical Induction Example

Prove for all $n \geq 1$, that 133 divides $11^{n+1} + 12^{2n-1}$.



$133 \mid 11^{k+1} + 12^{2*k-1}$

$133 \mid 11^{(k+1)+1} + 12^{2*(k+1)-1}$

IH: The statement holds for k
*k is an arbitrary variable

We want to show the statement holds true for k+1.

Proof by Mathematical Induction Example

Prove for all $n \geq 1$, that 133 divides $11^{n+1} + 12^{2n-1}$.

Base Case(s):

$n \geq 1 \rightarrow$ base case is $\mathrm{n} = 1$. When $\mathrm{n} = 1, 11^{1+1} + 12^{2*1-1} = 133 \rightarrow 133\,|11^{1+1} + 12^{2*1-1}$

Induction Hypothesis:

We assume that $133\,|11^{k+1} + 12^{2*k-1}$

Induction Step:

We want to show that for $\mathrm{k} + 1$ the statement holds: $133\,|11^{(k+1)+1} + 12^{2*(k+1)-1}$

$$\begin{aligned} 11^{(k+1)+1} + 12^{2*(k+1)-1} &= 11^{k+1+1} + 12^{2*k-1+2} \\ &= 11^{k+1}\times 11 + 12^{2*k-1}\times 12^2 \\ &= 11^{k+1}\times 11 + 12^{2*k-1}\times(133 + 11) \\ &= (11^{k+1} + 12^{2*k-1})\times 11 + 12^{2*k-1}\times 133 \end{aligned}$$

Based on assumption 133 $divides$ $(11^{k+1} + 12^{2*k-1})$, hence $(11^{k+1} + 12^{2*k-1})\times 11$

Also, $133|12^{2*(n-1)-1}\times 133$

Thereby 133 $divides$ $11^{n+1} + 12^{2*n-1}$ which is their summation! QED

Recursive Definitions

Why is induction important to CS majors?
It is the method used to prove that a loop or a recursively defined function correctly calculates the intended result.

Functions, sets and algorithms can be defined recursively.
The simplest form of recursive definition of a function f on the $\mathbb{N}$ specifies:
(B) a basis rule for the value of $f(0)$
(R) a recursion rule that determines how to obtain $f(n)$ from $f(n-1), \forall n \geq 1$

Example: Factorial Function $f(n) = n!$
(B) Basis: $f(0) = 1$ $(0! = 1)$
(R) Recursive Step: $f(n+1) = (n+1) * f(n)$ $(n! = \mathrm{n}*(\mathrm{n}-1)!)$

Example: The set of RNA strands S is defined recursively by:
(B) Basis: $A \in S, C \in S, U \in S, G \in S$
(R) Recursive Step: If $s \in S, then\ sA, sC, sU, sG \in S$

Recursive Algorithms

An **algorithm** is a computation representation of a function.

```
Recursive function: factorial(n)
Input: integer n≥ 0
Output: n!

if n=0 then return (1) (B)
   else return prod(n, factorial(n-1)) (R)
```

n	0	1	2	3	$k-1$	k	…
factorial(n)	1	1	$1*2$	$2*3$			…

(B)

$f(k) = k * f(k-1)$ (R)

```
Recursive function: Fibonacci(n)
Input: integer n≥ 0
Output: n!

if n=0 return 0   (B)
if n=1 return 1
    else return Fibonacci(n-1)+ Fibonacci(n-2) (R)
```

n	0	1	2	3	$k-2$	$k-1$	k
Fibonacci(n)	1	1	$1+1$	$1+2$			

(B)

$f(k) = f(k-1) + f(k-2)$(R)

Strong Induction

Sometimes, in order to do the inductive step:

$$\forall n \in \mathbb{Z}^{\geq b}\ (P(n))$$

$P(n-1)$ is not enough information, you may need to know $P(n-2)$ or $P(n-3)$ or $P(\frac{n}{2})$.

When this is the case, we can use **strong induction**.

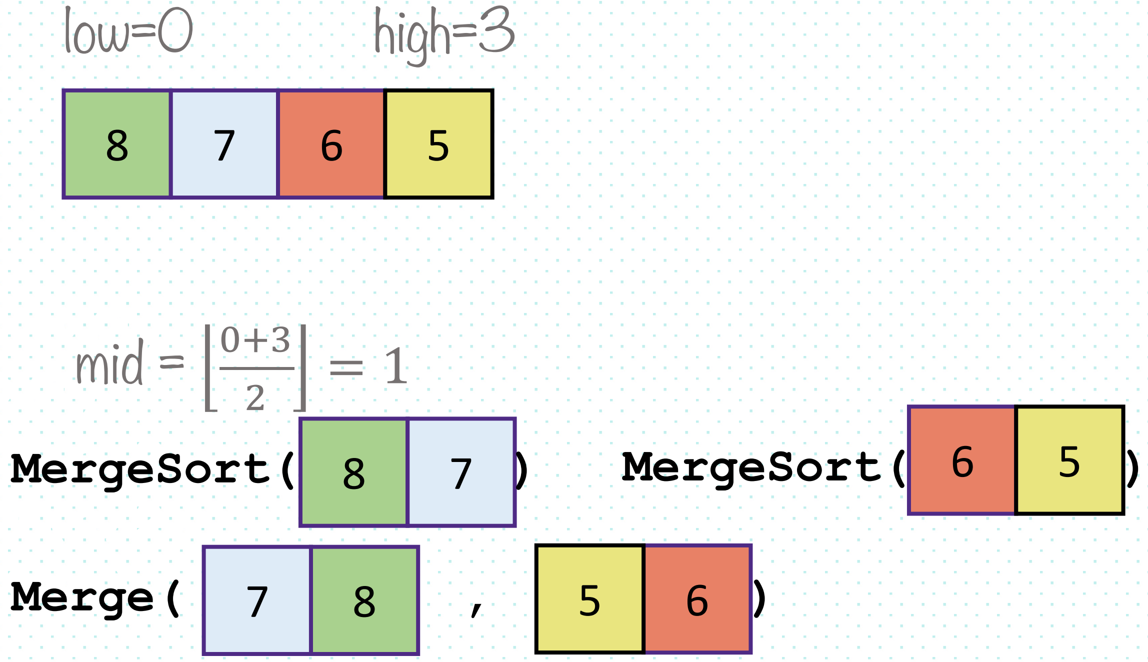
Basis Step: Show the property holds for $b, b+1, \ldots$
Inductive Hypothesis (IH): Assume $P(b) \land P(b+1) \land \cdots \land P(h-1)$
Inductive Step: Prove $\forall k \in \mathbb{Z}^{\geq b}\ P(b) \land P(b+1) \land \cdots \land P(k-1) \ \rightarrow P(k)$
Now you have proven the statement for all integers $\geq$ base case(s).

Recursive Algorithms

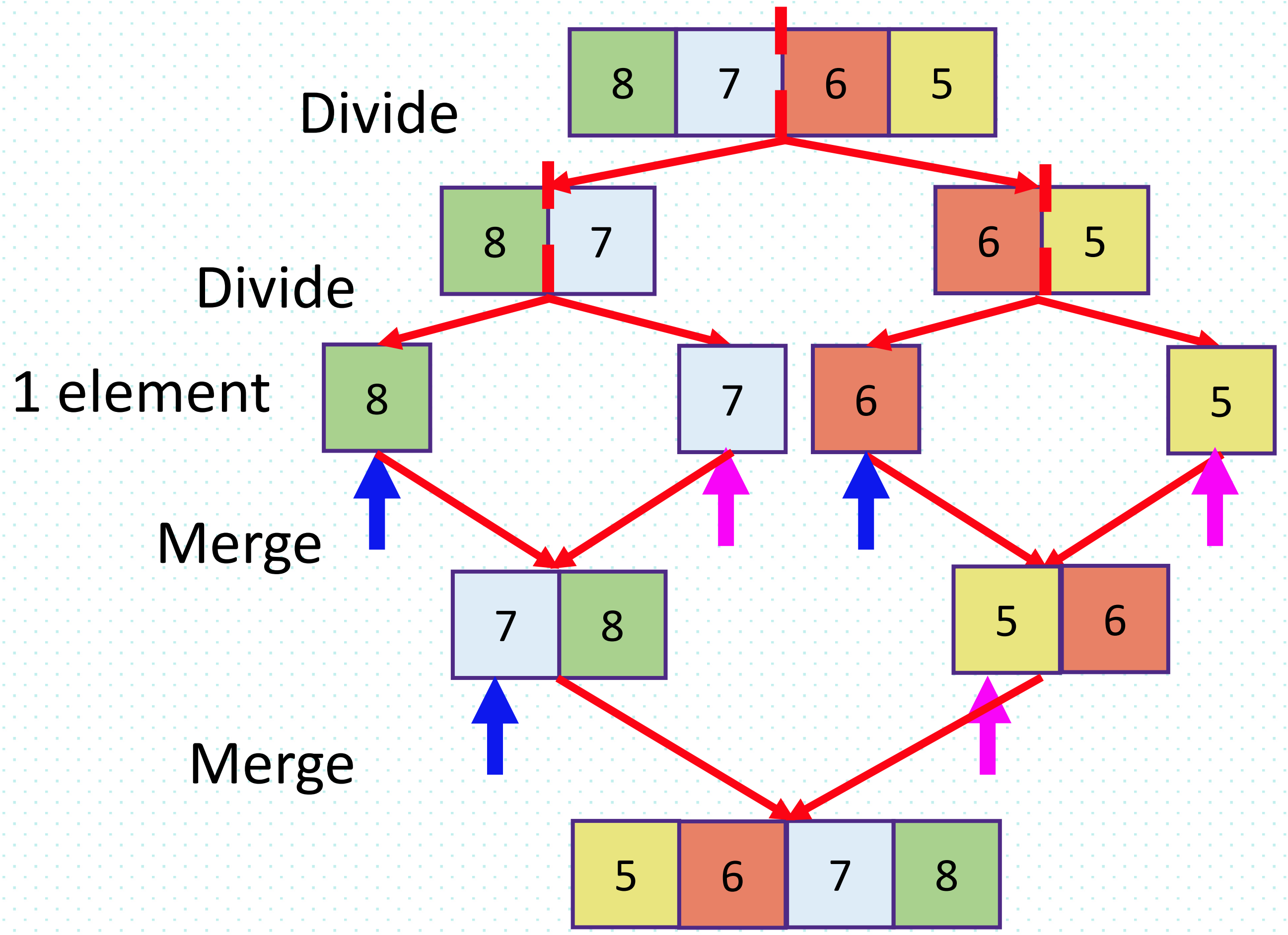
Remember: Merge Sort (Top Down) Algorithm's Pseudocode

- Divide array into two roughly equal halves.
- Conquer each half in turn.
- Merge two halves.

```
MergeSort(the_list, low, high)
    if low >= high:
(B)     return the_list
    if (low < high) {
        mid = floor((low + high) / 2)
(R)     MergeSort(A, low, mid)
        MergeSort(A, mid+1, high)
        Merge(A, left, mid, high)
```



Merge Sort (Top Down) on array [8, 7, 6, 5]



Merge Sort Runtime using Master’s Theorem

Let T(n) be the running time of Merge Sort of n elements $T(n) = 2T\left(\frac{n}{2}\right) + O(n)$

```
MergeSort(the_list, low, high)              T(n)
   if low >= high:                          O(1)
      return the_list                       O(1)
   if (low < high):                         O(1)
      mid = floor((low + high) / 2);        O(1)
      MergeSort(A, low, mid);               T(n/2)
      MergeSort(A, mid+1, high);            T(n/2)
   Merge(A, left, mid, high);               O(n)
```

$a = 2, b = 2, k = 1, and\ a = b^k$

$Using\ Master's\ Theorem, T(n) \in O(n^k \log_b n) \rightarrow \boldsymbol{T(n) \in O(nlogn)}$

Proof of Runtime of Merge Sort using Induction

```
MergeSort(the_list, low, high)              T(n)
    if low >= high:                         O(1)
        return the_list                     O(1)
    if (low < high):                        O(1)
        mid = floor((low + high) / 2);      O(1)
        MergeSort(A, low, mid);             T(n/2)
        MergeSort(A, mid+1, high);          T(n/2)
    Merge(A, left, mid, high);              O(n)
```

Statement: Merge Sort takes $O(nlogn)$ to sort a list of size n.

Base Case(s):

$n = 1 \rightarrow \mathrm{T(n)} = \mathrm{O(0)}$ also $\mathrm{O}(nlogn) = O(1 * log1) = 0$

Induction Hypothesis:

For input size of $\frac{n}{2}$ we assume that $T\left(\frac{n}{2}\right) = O(\frac{n}{2}\log\frac{n}{2})$

Induction Step:

We want to show that for n the statement holds: $\mathrm{T(n)} = \mathrm{O(nlogn)}$

Since $T(n) = 2T\left(\frac{n}{2}\right) + O(n) = 2O(\frac{n}{2}\log\frac{n}{2}) + O(n) = O(nlogn\ - nlog\,2 + n) = O(nlogn)$

Proof of Correctness of **Merge** in Merge Sort using Induction and Loop Invariant

To prove Merge, we can use loop invariants. A loop invariant is a statement that we want to prove is satisfied at the beginning of every iteration of a loop.
Initialization: The loop invariant is satisfied at the beginning of the for loop.
Maintenance: If the loop invariant is true before the ith iteration, loop invariant will hold before the $i + 1st$ iteration.
Termination: When the loop ends, the invariant gives us a useful property that helps show that the algorithm is correct.

Loop invariant: At the start of each iteration k of the for loop, the nonempty part of S contains the k − 1 smallest elements of L and R, in sorted order.
Initialization: i = j = 1, and S is completely empty. L[1] is the smallest element of L, while R[1] is the smallest element of R, so the initialization step holds.
Maintenance: Suppose without loss of generality that L[i] ≤ R[j]. Then L[i] is the smallest element not yet copied to S. The current nonempty part of S consists of the k − 1 smallest elements, so after the loop is over and L[i] is copied to S, the nonempty part of S will consist of the k smallest elements. Incrementing k (in the for loop update) and i reestablishes the loop invariant for the next iteration.
Termination: At termination, k = m+ 1. By the loop invariant, S contains the m smallest elements of L and R, in sorted order.

```
Merge(L, R):
    m = length(L) + length(R)
    S = empty array of size m
    i = 1; j = 1
    for k = 1 to m:
        if L[i] <= R[j]:
            S[k] = L[i]
            i = i + 1
        else: (L[i] > R[j])
            S[k] = R[j]
            j = j + 1
    return S
```

Proof of Correctness of Merge Sort using Induction

```
MergeSort(the_list, low, high)
   if low >= high:
      return the_list
   if (low < high):
      mid = floor((low + high) / 2)
      MergeSort(A, low, mid)
      MergeSort(A, mid+1, high)
   Merge(A, left, mid, high)
```

Base Case(s):

$n = 1 \rightarrow \text{Array of size 1 is already sorted}$

Induction Hypothesis:

Assume Merge Sort correctly sorts arrays of size $\sim\frac{n}{2}$ (the_list[low, mid] and [mid+1, high])

Induction Step:

We want to show that merge sort correctly sorts arrays of size n

Since merge is correct, therefore we can conclude that the_list is sorted correctly.

Common mistakes of students in proof by induction

Challenge: Not understanding the notion of induction
Instead of seeing it as a **logical chain**, they may view it as a memorized technique.

Observation: Sometimes students are confused proving $\mathrm{P}(h-1) \to P(h)$ vs $\mathrm{P}(h) \to P(h+1), etc$.
How to address: Proving $P(arbitrary\ element\ in\ set) \to P(its\ successor\ element)$

Observation: Lack of clear/correct induction hypothesis or do not know they need to use it for inductive step.
Observation: Proof misuses the induction predicate in a way that assumes the conclusion.
Observation: Proof assumes $P(k) \to P(k+1)$ in the induction hypothesis instead of assuming $P(k)$
How to address: Showing how properly formulate and apply the induction step in the inductive step.

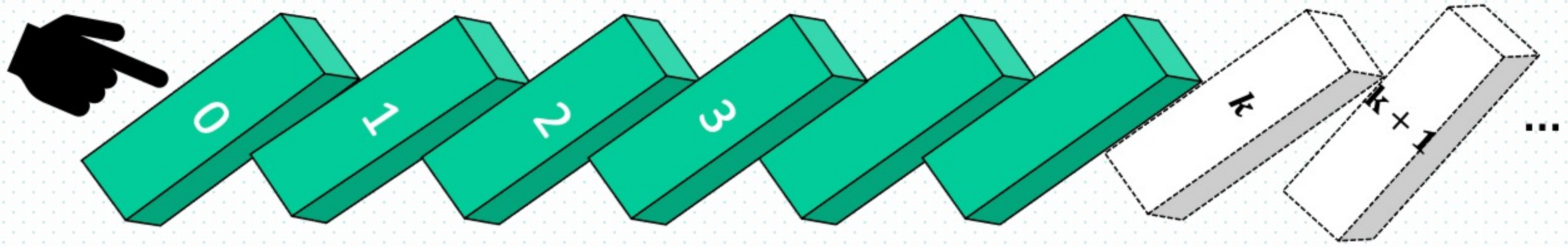
Observation: Missing the base cases or mistake in where to start.
How to address: Emphasizing on the set we want to prove the statement for.

Observation: Reversing the implication.
How to address: Stressing that the base case serves as the foundation of the entire inductive statement and explaining the logical chain of the proof.

How to Address common mistakes?

Converting the abstract notions to more visual content would help in understanding in all those cases.

$element$	s_0	s_1	…	s_k	s_{k+1}	…	s_n
Statement			…			…	



Common mistakes of students in proof by induction

Challenge: Not having the logical and arithmetic skills to get from $P(k) \rightarrow P(k+1)$
Example: Not knowing how to show $133 |\ 11^{(k+1)+1} + 12^{2*(k+1)-1}$ assuming that $11^{k+1} + 12^{2*k-1}$ (modular arithmetics)

How to address: Providing the prerequisite lists / slides for the course and clear examples.
Prerequisites:

Sets, Functions, Algorithms
Propositional logic
Quantification
Basic Algebra and arithmetics
Summation
Operations on Sets
Implication Proofs, direct proofs

Common mistakes of students in proof by induction

Challenge: Not knowing when to use mathematical (regular), strong and structural induction.

How to address: Train students to analyze the problem structure:

- If the problem involves natural numbers with a clear step-by-step progression, consider regular induction.
- If it requires assuming P(k) for all $\mathrm{b} \leq k \leq n$ to prove P(n+1), especially when the step size varies or multiple previous cases are needed.
- For recursive data structures, structural induction is often the best choice

Challenge: Not knowing how many base case(s) are needed in strong induction.

How to address: Determine how far back the inductive step depends on previous cases.

Example: How many base cases are needed in traditional coin problem.

Suppose you only have 5 cent coins and 9 cent coins, Fill in the blanks of the following strong induction proof that you can make change for any integer amount n ≥ 32.

32 = 5*(1) + 9*(3) 33 = 5*(2) + 9*(4) 34 = 5*(5) + 9*(1) 35 = 5*(7)+9*(0) 36 = 5*(0) + 9*(4)

then 37 is 32+5, 38 is 33+5, etc.

Exit Ticket

We want to prove that $P(n) = 3^n - 1$ is an even number, for all positive integers n using mathematical induction.
What should be the first step?

A) Basis step: $3^1 - 1 = 2$, which is an even number.
B) Basis step: $3^0 - 1 = 0$, which is which is an even number
C) Basis step: $3^2 - 1 = 8$, which is an even number.
D) Basis step: We assume that for an arbitrary positive integer k, P(k) is an even number.

Resources

UIUC, Algorithms & Models of Computation CS/ECE 374
University of Colombia, CS 3203

Proof by Induction Example

Use Induction to show that:

For all integers $k \geq 1, \;\; 4 \mid (5^k - 1)$

Proof by Induction Example

Use Induction to show that:

For all integers $k \geq 1, \ \ 4 \mid (5^k - 1)$

What does the above statement mean?

- That $4 \mid 5^1 - 1, 4 \mid 5^2 - 1, 4 \mid 5^3 - 1, \ldots, 4 \mid 5^n - 1 \; n \in \mathbb{Z}$

What does $a \mid b$ mean?

- That a divides b or $b = a \times c$ where c is an integer.

Which approach should we take?

- Induction

What are the base case(s)?

- $k \geq 1$, so the base case is $k = 1$. When $k = 1, 5^1 - 1 = 4 = 4 \times 1 \ \rightarrow 4 \mid 5^1 - 1 = 4$

For proving that for $n \in \mathbb{Z}$, $4 \mid 5^n - 1$, do we need to know if for all $1 \leq \mathrm{m} \leq \mathrm{n}$, , $4 \mid 5^m - 1$?

- No, only assuming that $4 | 5^{n-1} - 1$ is enough.

  We assume $4 | 5^{n-1} - 1$ and we'll show $4 | 5^n - 1$

  $4 | 5^n - 1 = 5^{n-1} \times 5 \ - 1 = \ 5^{n-1} \times 5 \ - 5 + 4 = 5 \times (5^{n-1} - 1) + 4$

  Since based on the assumption, $4 | 5^{n-1} - 1$, So, $4 \mid 5 \times (5^{n-1} - 1) + 4 = 5 \times 4x + 4 \quad x \in \mathbb{Z}$

Proof by Induction

Use Induction to show that:

For all integers $k \geq 1, \quad 4 \mid (5^k - 1)$

Base Case(s):

- $k \geq 1$, so the base case is $k = 1$. When $k = 1, 5^1 - 1 = 4 = 4 \times 1 \;\rightarrow 4 \mid 5^1 - 1 = 4$

Induction Hypothesis:

- We assume that $4|5^{n-1} - 1$

Induction Step:

- We assume $4|5^{n-1} - 1$ and we'll show $4|5^n - 1$

$4|5^n - 1 = 5^{n-1} \times 5 \; - 1 = \; 5^{n-1} \times 5 \; - 5 + 4 = 5 \times (5^{n-1} - 1) + 4$

Since based on the assumption, $4|5^{n-1} - 1$, So, $4 \mid 5 \times (5^{n-1} - 1) + 4 = 5 \times 4x + 4 \quad x \in \mathbb{Z}$

Backup of slide 14: Proof by Induction

Prove for all $n \geq 1$, that 133 divides $11^{n+1} + 12^{2n-1}$.

What does the above statement mean?

- That $133 \mid 11^{1+1} + 12^{2*1-1}, 133 \mid 11^{2+1} + 12^{2*2-1}, \dots, 133 \mid 11^{n+1} + 12^{2*n-1} \quad n \in \mathbb{Z}$

What does $a \mid b$ mean?

- That a divides b or $b = a \times c$ where c is an integer.

Which approach should we take?

- Induction

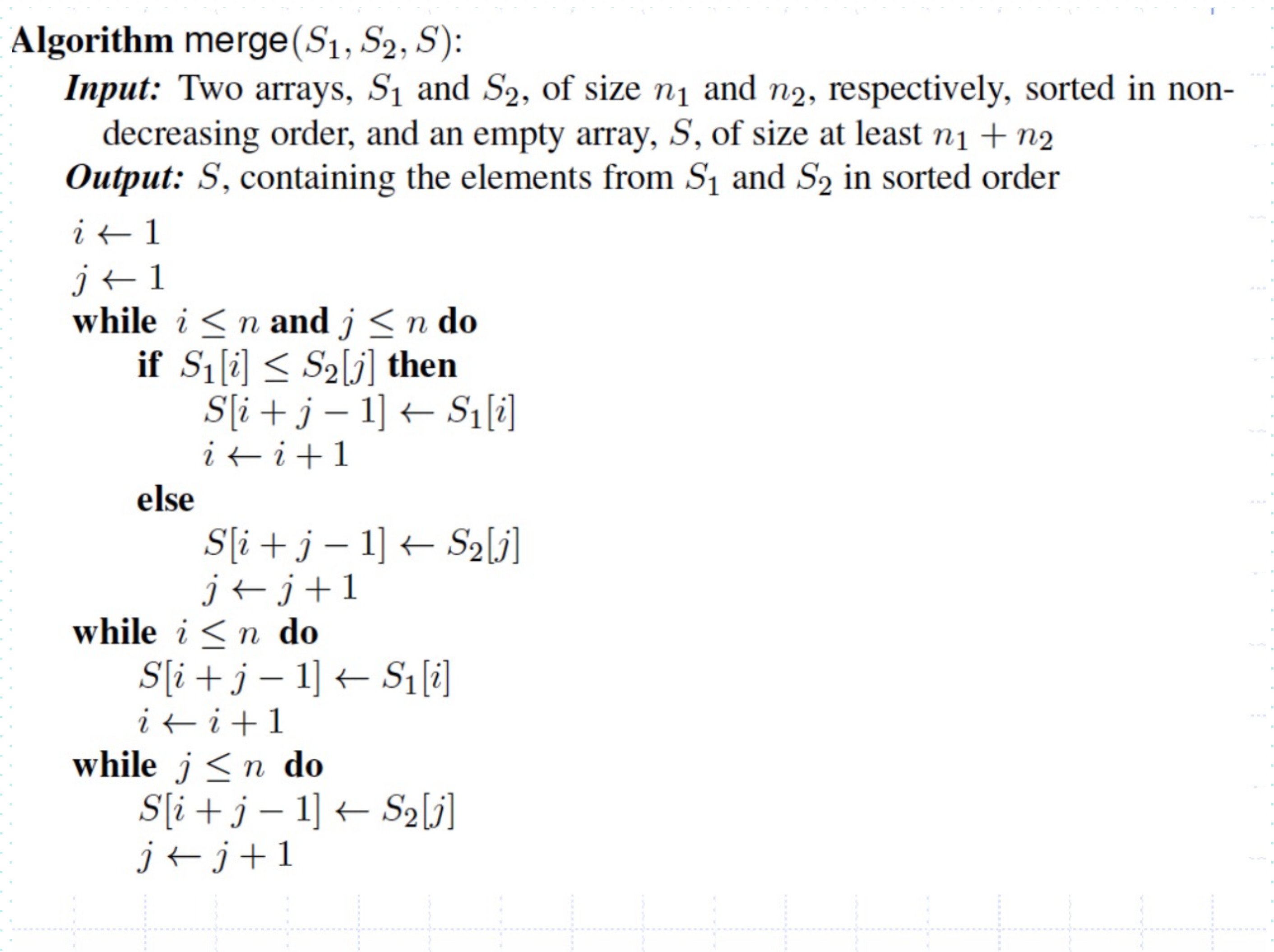
What are the base case(s)?

- $n \geq 1 \rightarrow$ base case is $\mathrm{n} = 1$. When $\mathrm{n} = 1, 11^{1+1} + 12^{2*1-1} = 133 \rightarrow 133 \mid 11^{1+1} + 12^{2*1-1}$

For proving that for $n \in \mathbb{Z}$, $133 \mid 11^{1+1} + 12^{2*1-1}$, do we need to know if for all $1 \leq \mathrm{m} \leq \mathrm{n}$, $, 133 \mid 11^{m+1} + 12^{2*m-1}$?

- No, only assuming that $133 \mid 11^{(n-1)+1} + 12^{2*(n-1)-1}$ is enough.

Backup of slide 17: Merge Sort (Top Down) Pseudocode



```
**Algorithm** merge($S_1, S_2, S$):
    *Input:* Two arrays, $S_1$ and $S_2$, of size $n_1$ and $n_2$, respectively, sorted in non-
      decreasing order, and an empty array, $S$, of size at least $n_1 + n_2$
    *Output:* $S$, containing the elements from $S_1$ and $S_2$ in sorted order

    $i \leftarrow 1$
    $j \leftarrow 1$
    while $i \leq n$ and $j \leq n$ do
        if $S_1[i] \leq S_2[j]$ then
            $S[i + j - 1] \leftarrow S_1[i]$
            $i \leftarrow i + 1$
        else
            $S[i + j - 1] \leftarrow S_2[j]$
            $j \leftarrow j + 1$
    while $i \leq n$ do
        $S[i + j - 1] \leftarrow S_1[i]$
        $i \leftarrow i + 1$
    while $j \leq n$ do
        $S[i + j - 1] \leftarrow S_2[j]$
        $j \leftarrow j + 1$
```

Back-up of slide 17: Merge Sort’s Time Complexity using Master’s Theorem

Let T(n) be the running time of Merge Sort of n elements $T(n) = 2T\left(\frac{n}{2}\right) + O(n)$

$$a = 2, b = 2, k = 1, and\ a = b^k\ therefore, T(n) \in O(n^k \log_b n) \rightarrow \boldsymbol{T(n) \in O(nlogn)}$$

Note: The recurrence relation $T(n) = aT\left(\frac{n}{b}\right) + cn^k$

where $a \geq 1, b > 1,$ and c are all constants, is solved (asymptotic complexity) as:

$$\begin{cases} If\ \ a < b^k & \rightarrow & T(n) \in O(n^k) \\ If\ \ a = b^k & \rightarrow & T(n) \in n^k \log_b n \\ If\ \ a > b^k & \rightarrow & T(n) \in n^{\log_b a} \end{cases}$$

Backup of slide 17: Merge-sort's Runtime using Unraveling

$$T(n) = 2\boldsymbol{T}\left(\frac{\boldsymbol{n}}{\boldsymbol{2}}\right) + O(n)$$

$$\vdots$$

$$= 2^k\, T\left(\frac{n}{2^k}\right) + \cdots + 4O\left(\frac{n}{4}\right) + 2O\left(\frac{n}{2}\right) + O(n)$$

$$= 2^k\, T\left(\frac{n}{2^k}\right) + 2^{k-1}O\left(\frac{n}{2^{k-1}}\right) + \ \ldots\ + 2^2O\left(\frac{n}{2^2}\right) + 2^1O\left(\frac{n}{2^1}\right) + 2^0O(\frac{n}{2^0})$$

Set $\boldsymbol{\frac{n}{2^k} = base = 1}$ ***as T(1)=0 and simplify above*** $\boldsymbol{(n = 2^k}$***, hence*** $\boldsymbol{k = logn)}$

$$= 2^{\log_2 n}\, T(1) + 2^{logn\,-1}O\left(\frac{n}{2^{logn\,-1}}\right) + \ \ldots\ + 4O\left(\frac{n}{2^{log4\,-1}}\right) + 2^{1}{}^{1}O\left(\frac{n}{2^1}\right) + 2^0O\left(\frac{n}{2^0}\right)$$

$$= \sum_{i=1}^{\log_2 n} 2^iO(\frac{n}{2^i}) = \sum_{i=1}^{\log_2 n} O(n) = \boldsymbol{O(nlogn)}$$

Proof by Strong Induction (Stronger than Mathematical)

Sending old-fashioned mail with postage stamps

Suppose we had postage stamps worth 5 cents and 3 cents. Which number of cents can we form using these stamps? In other words, which postage can we pay?

11? 2*(3) + 1*(5)

15? 5*(3) or 3*(5)

4? We cannot pay this with postage stamps of 3 and 5 cents.

The predicate $CanPay$ with domain $\mathbb{N}$ is

$$CanPay(n) = \exists x \exists y (5x + 3y = n)$$

Claim: $\forall n \in \mathbb{Z}^{\geq 8}(CanPay(n))$

strong induction:

Basis Step: Show that you can make change for n cents where $n = 8, 9, 10$.

Proof by Strong Induction (Stronger than Mathematical)

Claim: $\forall n \in \mathbb{Z}^{\geq 8}(CanPay(n))$

strong induction:

Basis Step: Show that you can make change for n cents where $n = 8, 9, 10$.

8 = 1*(3) + 1*(5)

9 = 3*(3) + 0*(5)

10 = 0*(3) + 2*(5)

Induction Step:

$$\forall h \in \mathbb{Z}^{\geq 11}((CanPay(8) \wedge CanPay(9) \wedge \cdots \wedge CanPay(h-1)) \rightarrow CanPay(h))$$

Let h be an arbitrary integer $h > 10$.

Assume that you can make change for m cents for all m in the range $8 \leq m < h$. or $8 \leq m \leq h-1$

(Want to show that you can make change for h cents.)

Proof by Strong Induction (Stronger than Mathematical)

Induction Step:

$$\forall h \in \mathbb{Z}^{\geq 11}((CanPay(8) \wedge CanPay(9) \wedge \cdots \wedge CanPay(h-1)) \rightarrow CanPay(h))$$

Let h be an arbitrary integer $h > 10$.

Assume that you can make change for m cents for all m in the range $8 \leq m < h$.

(Want to show that you can make change for h cents.)

By the induction hypothesis, since $8 \leq h - 3 \leq h - 1$, I can make change for $h - 3$ cents, i.e., $h - 3 = 3x + 5y$
Now, I can just add one more 3-cent coin to get $h - 3 + 3 \ - h = 3 * (x + 1) + 5y$.
Therefore, I can make change for h cent. And, by strong induction, I can make change for n cents for all $n \geq 8$.

Proof by Strong Induction (Stronger than Mathematical)

mathematical (regular) induction:

Basis Step: Show that you can make change for n cents where $n = 8, 9$.

8 = 1*(3) + 1*(5)

9 = 3*(3) + 0*(5)

Induction Step:

$$\forall k \in \mathbb{Z}^{\geq 9}((CanPay(k)) \rightarrow CanPay(k+1))$$

Let k be an arbitrary integer $k \geq 9$.

Assume that you can make change for k cents.

(Want to show that you can make change for $k + 1$ cents.)

Proof by Strong Induction (Stronger than Mathematical)

Induction Step:

$$\forall k \in \mathbb{Z}^{\geq 9}((CanPay(k)) \rightarrow CanPay(k+1))$$

Let k be an arbitrary integer $k \geq 9$.

Assume that you can make change for k cents.

(Want to show that you can make change for $k+1$ cents.)

Case 1: the way to make change for k cents includes at least one 5 cent stamp. For k+1 cents, take one 5 cent away and replace it with two 3-cents.

Therefore, if k = 3x+5y, then k+1 = 3*(x+2) +5*(y-1)

Case 2: the way to make change for k cents includes no 5 cents but at least three 3 cent stamp (why at least 3? because k >=9 (base) and it is only composed of 3-cents, so at least it has three 3-cent.). For k+1 cents, take three 3- cent away and replace it with two 5-cents.

Therefore, if k = 3x+5y, then k+1 = 3*(x-3) +5*(y-2)