Week 6 at a glance

We will be learning and practicing to:

- Clearly and unambiguously communicate computational ideas using appropriate formalism. Translate across levels of abstraction.
  - Translating between symbolic and English versions of statements using precise mathematical language
  - Using appropriate signpost words to improve readability of proofs, including 'arbitrary' and 'assume'
- Know, select and apply appropriate computing knowledge and problem-solving techniques. Reason about computation and systems. Use mathematical techniques to solve problems. Determine appropriate conceptual tools to apply to new situations. Know when tools do not apply and try different approaches. Critically analyze and evaluate candidate solutions.
  - Judging logical equivalence of compound propositions using symbolic manipulation with known equivalences, including DeMorgan's Law
  - Writing the converse, contrapositive, and inverse of a given conditional statement
  - Determining what evidence is required to establish that a quantified statement is true or false
  - Evaluating quantified statements about finite and infinite domains
- Apply proof strategies, including direct proofs and proofs by contradiction, and determine whether a proposed argument is valid or not.
  - Identifying the proof strategies used in a given proof
  - Identifying which proof strategies are applicable to prove a given compound proposition based on its logical structure
  - Carrying out a given proof strategy to prove a given statement
  - Carrying out a universal generalization argument to prove that a universal statement is true
  - Using proofs as knowledge discovery tools to decide whether a statement is true or false

TODO:

Review quiz based on Week 5 class material (due Monday 02/09/2026)

Midquarter feedback: please let us know what's working well for you and what isn't. `https://canvas.ucsd.edu/courses/71479/quizzes`

Homework 3 due on Gradescope https://www.gradescope.com/ (Thursday 02/12/2026)

Review quiz based on Week 6 class material (due Monday 02/16/2026)

No class Monday of Week 7 (02/16/2026) in observance of UCSD holiday.

Week 6 Monday: Proofs for properties of sets and numbers

Facts about numbers

We now have propositional and predicate logic that can help us express statements about any domain. We will develop proof strategies to craft valid argument for proving that such statements are true or disproving them (by showing they are false). We will practice these strategies with statements about sets and numbers, both because they are familiar and because they can be used to build cryptographic systems. Then we will apply proof strategies more broadly to prove statements about data structures and machine learning applications.

1. Addition and multiplication of real numbers are each commutative and associative.

   $\forall x \in \mathbb{R}\ \forall y \in \mathbb{R}\ (\ x+y = y+x \ \wedge \ x \cdot y = y \cdot x\ )$ commutativity

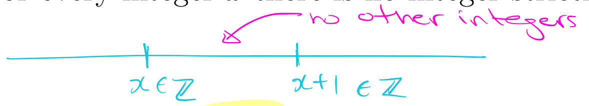
   $\forall x \in \mathbb{R}\ \forall y \in \mathbb{R}\ \forall z \in \mathbb{R}(\ (x+y)+z = x+(y+z) \wedge (x\cdot y)\cdot z = x \cdot (y \cdot z))$ associativity

2. The product of two positive numbers is positive, of two negative numbers is positive, and of a positive and a negative number is negative.

3. The sum of two integers, the product of two integers, and the difference between two integers are each integers.

   $\mathbb{Z} \subseteq \mathbb{R}$

4. For every integer x there is no integer strictly between x and $x+1$,



5. When x, y are positive integers, $xy \geq x$ and $xy \geq y$.

Factoring

Definition: When a and b are integers and a is nonzero, a **divides** b means there is an integer c such that $b = ac$.

quotient

Symbolically, $F(\ (a,b)\ ) = \exists c \in \mathbb{Z} (b = ac)$ and is a predicate over the domain $\mathbb{Z}^{\neq 0} \times \mathbb{Z}$

Other (synonymous) ways to say that $F(\ (a,b)\ )$ is true:

a is a **factor** of b $\quad$ a is a **divisor** of b $\quad$ b is a **multiple** of a (integer) $\quad$ $a|b$ "a divides b"

By the division theorem:

remainder upon division of b by a.

When a is a positive integer and b is any integer, $a|b$ exactly when b **mod** $a = 0$

When a is a positive integer and b is any integer, $a|b$ exactly when $b = a \cdot (b$ **div** $a)$

Notation: $\mathbb{Z}^{\neq 0} = \{ x \in \mathbb{Z} \mid x \neq 0 \}$

Translate these quantified statements by matching to English statement on right.

$\exists a \in \mathbb{Z}^{\neq 0}\ (\ F(\ (a,a)\ )\ )$	Every nonzero integer is a factor of itself.
$\exists a \in \mathbb{Z}^{\neq 0}\ (\ \neg F(\ (a,a)\ )\ )$	No nonzero integer is a factor of itself.
$\forall a \in \mathbb{Z}^{\neq 0}\ (\ F(\ (a,a)\ )\ )$	At least one nonzero integer is a factor of itself.
$\forall a \in \mathbb{Z}^{\neq 0}\ (\ \neg F(\ (a,a)\ )\ )$	Some nonzero integer is not a factor of itself.

Claim: Every nonzero integer is a factor of itself. ♤

Proof: Towards universal generalization, let a be an arbitrary nonzero integer. WTS $F((a,\overset{b}{a}))$ is true. In other words, we WTS $\exists c \in \mathbb{Z}\ (\underset{b}{a} = a \cdot c)$

We're looking for witness, consider $c = 1$.

- in domain? Yes because 1 is an integer
- makes predicate evaluate to true? Plugging in $c = 1$

$RHS = a \cdot c = a \cdot 1 = a = LHS$ ✓

Thus c is a witness to the existential statement and we've shown that the original predicate is true of an arbitrary element of domain ☒

~~Prove~~ or Disprove: There is a nonzero integer that does not divide its square.

WTS Each nonzero integer divides its square. ♧

WTS $\forall a \in \mathbb{Z}^{\neq 0}\ (\ F(\ (a, a^2)\ )\ )$

Towards universal generalization, let a be an arbitrary nonzero integer. WTS $\exists c \in \mathbb{Z}\ (a^2 = a \cdot c)$.. Consider $c = a$, an integer. It witnesses the existential statement because $LHS = a^2 = a \cdot a = a \cdot c = RHS$ as required ☒

Prove or **~~Disprove~~**: Every positive factor of a positive integer is less than or equal to it.

WTS $\forall b \in \mathbb{Z}^+\ \forall a \in \mathbb{Z}^+\ (\ F(\ (a,b)\ ) \rightarrow a \leq b\ )$

Towards universal generalization let b and a be positive integers. Towards direct proof, assume $F((a,b))$. WTS $a \leq b$. By definition of factor, we know $\exists c \in \mathbb{Z}(\ b = ca)$. Call such a witness c. Because a and b are both positive integers, Fact 2 from previous page gives that c is positive. Fact 5 gives that, therefore $ac \geq a$, so since $b = ac$, $b \geq a$ ☒

Claim: Every nonzero integer is a factor of itself and every nonzero integer divides its square. ♤ ♧ Bonus

Pf: We have already proved the conjuncts (♤ , ♧) separately, so the conjunction has been proved ☒

Definition: an integer n is **even** means that there is an integer a such that $n = 2a$; an integer n is **odd** means that there is an integer a such that $n = 2a + 1$. Equivalently, an integer n is **even** means $n \textbf{ mod } 2 = 0$; an integer n is **odd** means $n \textbf{ mod } 2 = 1$. Also, an integer is even if and only if it is not odd.

Definition: An integer p greater than 1 is called **prime** means the only positive factors of p are 1 and p. A positive integer that is greater than 1 and is not prime is called composite.

Extra examples: Use the definition to prove that 1 is not prime, 2 is prime, 3 is prime, 4 is not prime, 5 is prime, 6 is not prime, and 7 is prime.

"one after the other"

$\exists$

~~**True**~~ [✓ **or False**]: The statement "There are three consecutive positive integers that are prime."

Hint: These numbers would be of the form $p, p + 1, p + 2$ (where p is a positive integer).

Proof: We need to show $\forall p \in \mathbb{Z}^+ (\neg (p \text{ is prime} \wedge p+1 \text{ is prime} \wedge p+2 \text{ is prime}))$

or equivalenty WTS $\forall p \in \mathbb{Z}^+ (p \text{ is not prime} \vee p+1 \text{ is not prime} \vee p+2 \text{ is not prime}))$

Notice: to disprove $\exists p \in \mathbb{Z}^+ (p \text{ is prime} \wedge p+1 \text{ is prime} \wedge p+2 \text{ is prime})$

we will prove $\neg \exists p \in \mathbb{Z}^+ (\ldots \text{ --- --- --- --- --- })$

equivalently we WTS $\forall p \in \mathbb{Z}^+ \neg (\ldots \text{ --- --- --- --- })$

Towards proof by universal generalization, let p be arbitrary positive integer. By definition, since p≠0, p is either odd or even.

Case ① Assume p is even. Then WTS p+2 is not prime, DETAILS OMITTED...

Case ② Assume p is odd. Then WTS p+1 is not prime. DETAILS OMITTED_

[✓ **True**] ~~**or False**~~: The statement "There are three consecutive odd positive integers that are prime."

Hint: These numbers would be of the form $p, p + 2, p + 4$ (where p is an odd positive integer).

Proof: We need to show $\exists p \in \mathbb{Z}^+ (p \text{ is odd} \wedge p \text{ is prime} \wedge p+2 \text{ is prime} \wedge p+4 \text{ is prime})$

WTS p=3 is a witness.

DETAILS OMITTED.

Pre-class reading for Week 6 Wednesday

At this point, we've seen the proof strategies

- A **counterexample** to prove that $\forall x P(x)$ is **false**.
- A **witness** to prove that $\exists x P(x)$ is **true**.
- **Proof of universal by exhaustion** to prove that $\forall x\, P(x)$ is true when P has a finite domain
- **Proof by universal generalization** to prove that $\forall x\, P(x)$ is true using an arbitrary element of the domain.
- To prove that $\exists x P(x)$ is **false**, write the universal statement that is logically equivalent to its negation and then prove it true using universal generalization.
- To prove that $p \wedge q$ is true, have two subgoals: subgoal (1) prove p is true; and, subgoal (2) prove q is true. To prove that $p \wedge q$ is false, it's enough to prove that p is false. To prove that $p \wedge q$ is false, it's enough to prove that q is false.
- Proof of conditional by **direct proof**
- Proof of conditional by **contrapositive proof**
- Proof of disjuction using equivalent conditional: To prove that the disjunction $p \vee q$ is true, we can rewrite it equivalently as $\neg p \to q$ and then use direct proof or contrapositive proof.
- **Proof by cases**.

Recall the definitions: The set of RNA strands S is defined (recursively) by:

$$\begin{array}{ll}
\text{Basis Step:} & \texttt{A} \in S, \texttt{C} \in S, \texttt{U} \in S, \texttt{G} \in S \\
\text{Recursive Step:} & \text{If } s \in S \text{ and } b \in B \text{, then } sb \in S
\end{array}$$

where sb is string concatenation.

The function *rnalen* that computes the length of RNA strands in S is defined recursively by:

$$\begin{array}{llll}
 & & \mathit{rnalen} : S & \to \mathbb{Z}^+ \\
\text{Basis Step:} & \text{If } b \in B \text{ then} & \mathit{rnalen}(b) & = 1 \\
\text{Recursive Step:} & \text{If } s \in S \text{ and } b \in B \text{, then} & \mathit{rnalen}(sb) & = 1 + \mathit{rnalen}(s)
\end{array}$$

The function *basecount* that computes the number of a given base b appearing in a RNA strand s is defined recursively by:

$$\begin{array}{llll}
 & & \mathit{basecount} : S \times B & \to \mathbb{N} \\
\text{Basis Step:} & \text{If } b_1 \in B, b_2 \in B & \mathit{basecount}(\ (b_1, b_2)\ ) & = \begin{cases} 1 & \text{when } b_1 = b_2 \\ 0 & \text{when } b_1 \neq b_2 \end{cases} \\
\text{Recursive Step:} & \text{If } s \in S, b_1 \in B, b_2 \in B & \mathit{basecount}(\ (sb_1, b_2)\ ) & = \begin{cases} 1 + \mathit{basecount}(\ (s, b_2)\ ) & \text{when } b_1 = b_2 \\ \mathit{basecount}(\ (s, b_2)\ ) & \text{when } b_1 \neq b_2 \end{cases}
\end{array}$$

Which proof strategies could be used to prove each of the following statements?

Hint: first translate the statements to English and identify the main logical structure.

$\forall b \in B\ \exists s \in S\ (\ basecount(\ (s,b)\ )\ > 0\ )$

For each base, there's at least one strand with at least one occurrence of that base

Exhaustion, then witness

$\exists s \in S\ \forall b \in B\ (\ basecount(\ (s,b)\ )\ > 0\ )$

There is a strand in which each base occurs at least once.

Witness, then exhaustion

$\exists s \in S\ (rnalen(s) = basecount(\ (s, \texttt{A})\ )\ )$

There is a strand whose length matches the number of occurrences of A in the strand.

Witness

$\forall s \in S\ \exists b \in B\ (\ basecount(\ (s,b)\ ) > 0\ )$

For each strand there's a base whose number of occurrences in the strand is greater than zero.

??

$\forall s \in S\ (rnalen(s) \geq basecount(\ (s, \texttt{A})\ ))$

The length of any strand is greater than or equal to the number of occurrences of A in that strand.

??

$\forall s \in S\ (\ rnalen(s) > 0\ )$

All strands have positive length.

??

Week 6 Wednesday: Structural Induction

Claim $\forall s \in S\ (\ rnalen(s) > 0\ )$

Proof: Let s be an arbitrary RNA strand. By the recursive definition of S, either $s \in B$ or there is some strand s_0 and some base b such that $s = s_0 b$. We will show that the inequality holds for both cases.

Base **Case**: Assume $s \in B$. We need to show $rnalen(s) > 0$. By the basis step in the definition of $rnalen$,

$$rnalen(s) = 1$$

which is greater than 0, as required.

Recursive **Case**: Assume there is some strand s_0 and some base b such that $s = s_0 b$. We will show *(the stronger claim)* that

$$\forall u \in S\ \forall b \in B\ (\ \underset{\text{HYP}}{\underline{rnalen(u) > 0}} \to \underset{\text{CONC}}{\underline{rnalen(ub) > 0}}\ )$$

No matter what strand I start at, if property is true about, it's also true about strand's extensions.

U.G. Consider an arbitrary RNA strand u and an arbitrary base b, and assume towards a direct proof, that

$$rnalen(u) > 0 \qquad \text{HYP}$$

We need to show that $\underset{\text{CONC}}{\underline{rnalen(ub)}} > 0$.

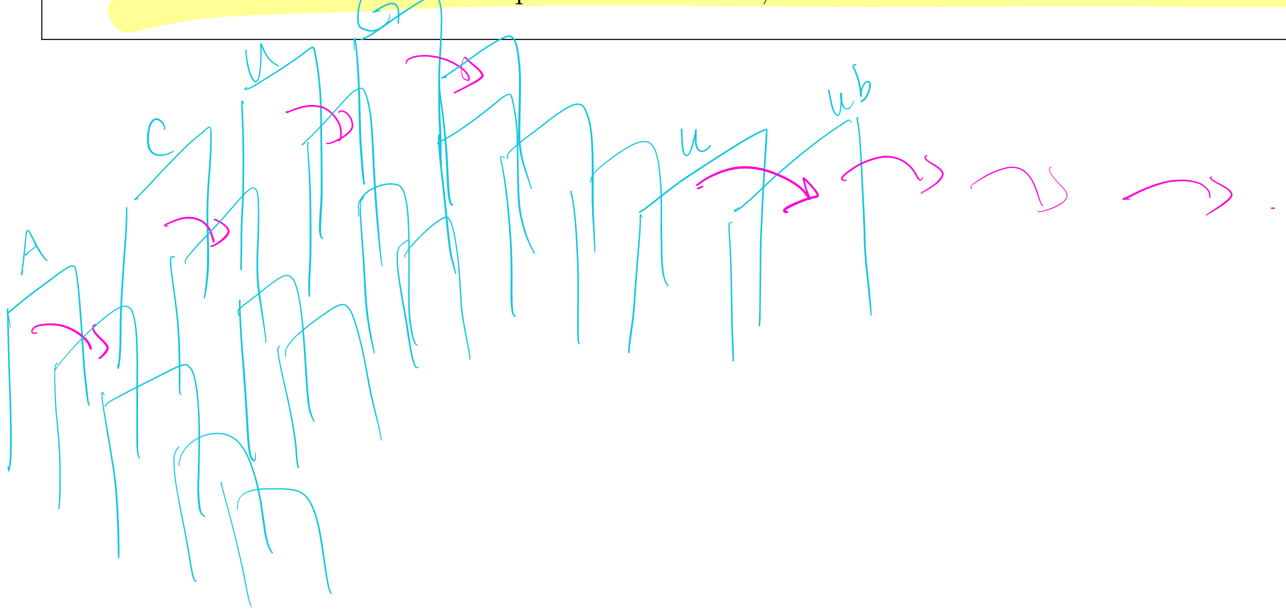
$$rnalen(ub) \overset{\text{by definition of rnalen}}{=} 1 + rnalen(u) \underset{\text{by HYP that } rnalen(u) > 0}{>} 1 + 0 = 1 > 0$$

as required.

Proof by Structural Induction To prove a universal quantification over a recursively defined set:

Basis Step: Show the statement holds for elements specified in the basis step of the definition.

Recursive Step: Show that if the statement is true for each of the elements used to construct new elements in the recursive step of the definition, the result holds for these new elements.



ex. S= ACUG rnalen(s)= 4 basecount ((s,A))= 1
S= AAAA rnalen(s) = 4 basecount((s,A)) = 4
S= CCCC rnalen(s) = 4 basecount((s,A))= 0

Claim $\forall s \in S\,(rnalen(s) \geq basecount(\ (s, \mathtt{A})\ ))$:

Proof: We proceed by structural induction on the recursively defined set S.

Basis Case: We need to prove that the inequality holds for each element in the basis step of the recursive definition of S. Need to show

$$(\ rnalen(\mathtt{A}) \overset{\textcircled{1}}{\geq} basecount(\ (\mathtt{A},\mathtt{A})\ )\ ) \land (\ rnalen(\mathtt{C}) \overset{\textcircled{2}}{\geq} basecount(\ (\mathtt{C},\mathtt{A})\ )\ )$$
$$\land (\ rnalen(\mathtt{U}) \underset{\textcircled{3}}{\geq} basecount(\ (\mathtt{U},\mathtt{A})\ )\ ) \land (\ rnalen(\mathtt{G}) \underset{\textcircled{4}}{\geq} basecount(\ (\mathtt{G},\mathtt{A})\ )\ )$$

We calculate, using the definitions of *rnalen* and *basecount*:

Goal ① rnalen(A) ≥ basecount ((A,A)).

Calculate LHS = rnalen(A) = 1 by definition of rnalen (basis case)

RHS = basecount ((A,A)) = 1 by definition of basecount (basis case, $b_1 = b_2$)

Since 1=1, we have 1≥1, so LHS ≥ RHS ☺.

Similarly, for other goals. ②, ③, ④.

Recursive Case: We will prove that

$$\forall u \in S\ \forall b \in B\ (\ \underbrace{rnalen(u) \geq basecount(\ (u,\mathtt{A})\ )}_{\text{HYP}} \rightarrow \underbrace{rnalen(ub) \geq basecount(\ (ub,\mathtt{A})\ )}_{\text{CONC}}\ )$$

Consider arbitrary RNA strand u and arbitrary base b. Assume, as the **induction hypothesis**, that $rnalen(u) \geq basecount(\ (u,\mathtt{A})\ )$. We need to show that $rnalen(ub) \geq basecount(\ (ub,\mathtt{A})\ )$.

Using the recursive step in the definition of the function *rnalen*:

$$LHS = rnalen(ub) = 1 + rnalen(u) \qquad RHS = basecount((ub,A))$$

The recursive step in the definition of the function *basecount* has two cases. We notice that $b = \mathtt{A} \lor b \neq \mathtt{A}$ and we proceed by cases.

Case i. Assume $b = \mathtt{A}$.

Using the first case in the recursive step in the definition of the function *basecount*:

$$RHS = basecount(\ (ub,\mathtt{A})\ ) = 1 + basecount(\ (u,\mathtt{A})\ )$$

By the **induction hypothesis**, we know that $basecount(\ (u,\mathtt{A})\ ) \leq rnalen(u)$ so:

$$RHS = basecount(\ (ub,\mathtt{A})\ ) = 1 + basecount(\ (u,\mathtt{A})\ ) \overset{IH}{\leq} 1 + rnalen(u) = rnalen(ub) \qquad LHS$$

and, thus, $basecount(\ (ub,\mathtt{A})\ ) \leq rnalen(ub)$, as required.

Case ii. Assume $b \neq \mathtt{A}$.

Using the second case in the recursive step in the definition of the function *basecount*:

$$basecount(\ (ub,\mathtt{A})\ ) = basecount(\ (u,\mathtt{A})\ )$$

By the **induction hypothesis**, we know that $basecount(\ (u,\mathtt{A})\ ) \leq rnalen(u)$ so:

$$basecount(\ (ub,\mathtt{A})\ ) = basecount(\ (u,\mathtt{A})\ ) \leq rnalen(u) < 1 + rnalen(u) = rnalen(ub)$$

and, thus, $basecount(\ (ub,\mathtt{A})\ ) \leq rnalen(ub)$, as required.

New vocab today: Structural induction, induction hypothesis, and ...

To organize our proofs, it's useful to highlight which claims are most important for our overall goals. We use some terminology to describe different roles statements can have.

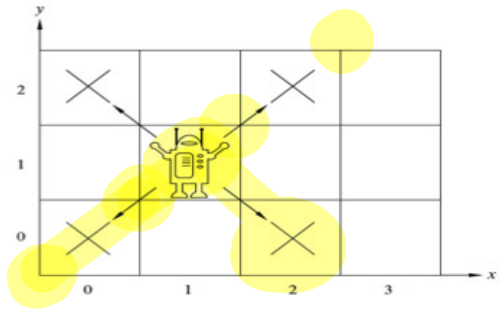
Theorem: Statement that can be shown to be true, usually an important one.

Less important theorems can be called **proposition**, **fact**, **result**, **claim**.

Lemma: A less important theorem that is useful in proving a theorem.

Corollary: A theorem that can be proved directly after another one has been proved, without needing a lot of extra work.

Invariant: A theorem that describes a property that is true about an algorithm or system no matter what inputs are used.



Theorem: A robot on an infinite 2-dimensional integer grid starts at $(0,0)$ and at each step moves to diagonally adjacent grid point. This robot ~~can~~ / [✓ cannot] (*circle one*) reach $(1,0)$.

Definition The set of positions the robot can visit *Pos* is defined by:

Basis Step: $(0,0) \in Pos$

Recursive Step: If $(x,y) \in Pos$, then $(x+1, y+1)$, $(x+1, y-1)$, $(x-1, y+1)$, $(x-1, y-1)$ are also in *Pos*

Example elements of Pos are:

$(0,0)$, $(1,1)$, $(2,0)$

WTS $(1,0) \notin Pos$

Lemma: $\forall (x,y) \in Pos \;\; (x+y \text{ is an even integer })$

universal about recursively defined set

Why are we calling this a lemma?

Proof of theorem using lemma: To show is $(1,0) \notin Pos$. Rewriting the lemma to explicitly restrict the domain of the universal, we have $\forall (x,y)\ (\ (x,y) \in Pos \;\; \to \;\; (x+y \text{ is an even integer})\ )$. Since the universal is true, $(\ (1,0) \in Pos \;\; \to \;\; (1+0 \text{ is an even integer})\ )$ is a true statement. Evaluating the conclusion of this conditional statement: By definition of long division, since $1 = 0 \cdot 2 + 1$ (where $0 \in \mathbb{Z}$ and $1 \in \mathbb{Z}$ and $0 \leq 1 < 2$ mean that 0 is the quotient and 1 is the remainder), 1 **mod** $2 = 1$ which is not 0 so the conclusion is false. A true conditional with a false conclusion must have a false hypothesis: $(1,0) \notin Pos$, QED. □

Proof of lemma by structural induction:

Basis Step: WTS when $x=0$, $y=0$ we have that $x+y$ is an even integer.

WTS $0+0=0$ is an even integer.
By long division, $0 = 2 \cdot \underbrace{0}_{q} + \underbrace{0}_{r}$ so $0 \bmod 2 = 0$ and so by definition of even, 0 is even. ☒

Recursive Step: Consider arbitrary $(x,y) \in Pos$. To show is:

$$\underbrace{(x+y \text{ is an even integer})}_{\text{HYP}} \to \underbrace{(\text{sum of coordinates of next position is even integer})}_{\text{CONCLUSION}}$$

Assume **as the induction hypothesis, IH** that: $x+y$ is an even integer.
By definition of even, this means there's an integer, call it q, so that $x+y=2q$.
WTS sum of coords of next position is an even integer
We have four cases for what next position is.

Goal ① WTS if $(x-1, y+1)$ is next position then sum of coords $x-1$ and $y+1$ is even.
Calculate: $x-1+y+1 = x+y-1+1 = x+y$, which we assumed is even ☺

Goal ② WTS if $(x+1, y+1)$ is next position then sum of coords $x+1$ and $y+1$ is even.
Calculate: $x+1+y+1 = x+y+1+1 = x+y+2$
$= 2q+2 = 2(q+1)$
and since q is an integer so is $q+1$
so $x+1+y+1$ is even ☺

Goal ③ WTS if $(x-1, y-1)$ is next position then sum of coords $x-1$ and $y-1$ is even.
Calculate: $x-1+y-1 = x+y-1-1 = x+y-2$
$= 2q-2 = 2(q-1)$
and since q is an integer so is $q-1$
so $x-1+y-1$ is even ☺

Goal ④ WTS $(x+1, y-1)$ is next position then sum of coords $x+1$ and $y-1$ is even.
Calculate: $x+1+y-1 = x+y+1-1 = x+y$, which we assumed is even ☺

Week 6 Friday: Mathematical and Strong Induction

Proof by Mathematical Induction
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.

Recursive Step: Consider an arbitrary integer n greater than or equal to b, assume (as the **induction hypothesis**) that the property holds for n, and use this and other facts to prove that the property holds for $n+1$.

domain codomain

The set $\mathbb{N}$ is recursively defined. Therefore, the function $sumPow : \mathbb{N} \to \mathbb{N}$ which computes, for input i, the sum of the nonnegative powers of 2 up to and including exponent i is defined recursively by

Basis step: $sumPow(0) = 1$

Recursive step: If $x \in \mathbb{N}$, then $sumPow(x+1) = sumPow(x) + 2^{x+1}$

$sumPow(0) =$ 1 basis step. $2^0 = 2^1 - 1$

$sumPow(1) =$ $SumPow(0) + 2^{0+1} = 1 + 2^1 = 1+2 = 3$ $2^0 + 2^1 = 2^2 - 1$

$1 = x+1$
$x = 0$

$sumPow(2) =$ $sumPow(1) + 2^{1+1} = 3 + 2^2 = 3+4 = 7$ $2^0 + 2^1 + 2^2 = 2^3 - 1$

$2 = x+1$
$x = 1$

Fill in the blanks in the following proof of

$$\forall n \in \mathbb{N}\ (sumPow(n) = 2^{n+1} - 1)$$

Proof: Since $\mathbb{N}$ is recursively defined, we proceed by structural induction.

Basis case: We need to show that $sumPow(0) = 2^{0+1} - 1$. Evaluating each side: $LHS = sumPow(0) = 1$ by the basis case in the recursive definition of $sumPow$; $RHS = 2^{0+1} - 1 = 2^1 - 1 = 2 - 1 = 1$. Since $1 = 1$, the equality holds.

$\forall n\ (sumPow(n) = 2^{n+1} - 1 \longrightarrow sumPow(n+1) = 2^{(n+1)+1} - 1)$

Recursive case: Consider arbitrary natural number n and assume, as the induction hypothesis that $sumPow(n) = 2^{n+1} - 1$. We need to show that $sumPow(n+1) = 2^{(n+1)+1} - 1$. Evaluating each side:

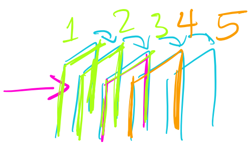
Prop is true at n

prop is true at n+1

$$LHS = sumPow(n+1) \overset{\text{rec def}}{=} sumPow(n) + 2^{n+1} \overset{\text{IH}}{=} (2^{n+1} - 1) + 2^{n+1}.$$

$$RHS = 2^{(n+1)+1} - 1 \overset{\text{exponent rules}}{=} 2 \cdot 2^{n+1} - 1 = \left(2^{n+1} + 2^{n+1}\right) - 1 \overset{\text{regrouping}}{=} (2^{n+1} - 1) + 2^{n+1}$$

Thus, $LHS = RHS$. The structural induction is complete and we have proved the universal generalization. □



$(n+1)$ div2 ↑ ↑ $n+1$

Proof by Strong Induction

To prove that a universal quantification over the set of all integers greater than or equal to some base integer b holds, pick a fixed nonnegative integer j and then: $\mathbb{Z}^{\geq b}$

Basis Step: Show the statement holds for $b, b+1, \ldots, b+j$.

Recursive Step: Consider an arbitrary integer n greater than or equal to $b+j$, assume (as the **strong induction hypothesis**) that the property holds for **each of** $b, b+1, \ldots, n$, and use this and other facts to prove that the property holds for $n+1$.

coefficient in columns is 0 or 1

Theorem: Every positive integer is a sum of (one or more) distinct powers of 2. *Binary expansions exist!*

$n = (\underbrace{\_\ \_\ \_\ \_\ \_\ \_}_{n \text{ div } 2}\ \overset{n \bmod 2}{\_}\ )_2$

The idea in the "Least significant first" algorithm for computing binary expansions is that the binary expansion of *half* a number becomes *part* of the binary expansion of the number of itself. We can use this idea in a proof by strong induction that binary expansions exist for all positive integers n.

Proof by strong induction, with $b = 1$ and $j = 0$. We need to prove that for each positive integer n, there is a positive integer k and coefficients $a_0, \ldots, a_{k-1}$ where each a_i is 0 or 1 and $a_{k-1} \neq 0$, and

$$n = \sum_{i=0}^{k-1} a_i 2^i$$

Basis step: WTS property is true about 1.

$$n = 1 = 1 \cdot 2^0 = \sum_{i=0}^{1-1} a_i 2^i \qquad a_0 = 1$$

Recursive step: Consider an arbitrary integer $n \geq 1$.

Assume (as the strong induction hypothesis, IH) that the property is true about each of $1, \ldots, n$.

WTS that the property is true about $n+1$.

Idea: We will apply the IH to $(n+1)$ **div** 2.

Why is this ok? Need to confirm that $(n+1)$ div 2 is an integer between 1 and n (inclusive).

- int? yes by definition of div.
- $(n+1)$ div $2 \geq 1$? yes because $n \geq 1$ so $n+1 \geq 2$.
- $(n+1)$ div $2 \leq n$? yes ~~because~~ $n+1 \leq n+n = 2n$

Why is this helpful?

$$\underbrace{\_\ \_\ \_\ \_\ \_}_{(n+1)\ \text{div}\ 2}\ \underline{\overset{(n+1)}{\text{mod } 2}}$$

By the IH, we can write $(n+1)$ **div** 2 as a sum of powers of 2. In other words, there are values $a_{k-1}, \ldots, a_0$ such that each a_i is 0 or 1, $a_{k-1} = 1$, and

$$\sum_{i=0}^{k-1} a_i 2^i = (n+1)\ \textbf{div}\ 2$$

Define the collection of coefficients

$$c_j = \begin{cases} a_{j-1} & \text{if } 1 \leq j \leq k \\ (n+1)\ \textbf{mod}\ 2 & \text{if } j = 0 \end{cases}$$

Calculating:

$$\begin{aligned}
\sum_{j=0}^{k} c_j 2^j &= c_0 + \sum_{j=1}^{k} c_j 2^j = c_0 + \sum_{i=0}^{k-1} c_{i+1} 2^{i+1} && \text{re-indexing the summation} \\
&= c_0 + 2 \cdot \sum_{i=0}^{k-1} c_{i+1} 2^i && \text{factoring out a 2 from each term in the sum} \\
&= c_0 + 2 \cdot \sum_{i=0}^{k-1} a_i 2^i && \text{by definition of } c_{i+1} \\
&= c_0 + 2\,(\ (n+1)\ \textbf{div}\ 2\ ) && \text{by IH} \\
&= (\ (n+1)\ \textbf{mod}\ 2\ ) + 2\,(\ (n+1)\ \textbf{div}\ 2\ ) && \text{by definition of } c_0 \\
&= n+1 && \text{by definition of long division}
\end{aligned}$$

Thus, $n+1$ can be expressed as a sum of powers of 2, as required.

Representing positive integers with primes

Theorem: Every positive integer *greater than 1* is a product of (one or more) primes.

Before we prove, let's try some examples:

20 = $2 \cdot 10 = 2 \cdot 2 \cdot 5$

100 = $4 \cdot 25 = 2 \cdot 2 \cdot 5 \cdot 5$

5 = 5

Proof by strong induction, with $b = 2$ and $j = 0$.

Basis step: WTS property is true about 2.

Since 2 is itself prime, it is already written as a product of (one) prime.

Recursive step: Consider an arbitrary integer $n \geq 2$. Assume (as the strong induction hypothesis, IH) that the property is true about each of $2, \ldots, n$. WTS that the property is true about $n + 1$: We want to show that $n + 1$ can be written as a product of primes. Notice that $n + 1$ is itself prime or it is composite.

Case 1: assume $n + 1$ is prime and then immediately it is written as a product of (one) prime so we are done.

Case 2: assume that $n + 1$ is composite so there are integers x and y where $n + 1 = xy$ and each of them is between 2 and n (inclusive). Therefore, the induction hypothesis applies to each of x and y so each of these factors of $n + 1$ can be written as a product of primes. Multiplying these products together, we get a product of primes that gives $n + 1$, as required.

Since both cases give the necessary conclusion, the proof by cases for the recursive step is complete.

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? $1 \cdot 5 + 2 \cdot 3$

15? $3 \cdot 5$

4? Not possible!

14: $1 \cdot 5 + 3 \cdot 3$

$$\begin{aligned}&CanPay(0) \wedge \neg CanPay(1) \wedge \neg CanPay(2) \wedge \\ &CanPay(3) \wedge \neg CanPay(4) \wedge CanPay(5) \wedge CanPay(6) \\ &\neg CanPay(7) \wedge \forall n \in \mathbb{Z}^{\geq 8} CanPay(n)\end{aligned}$$

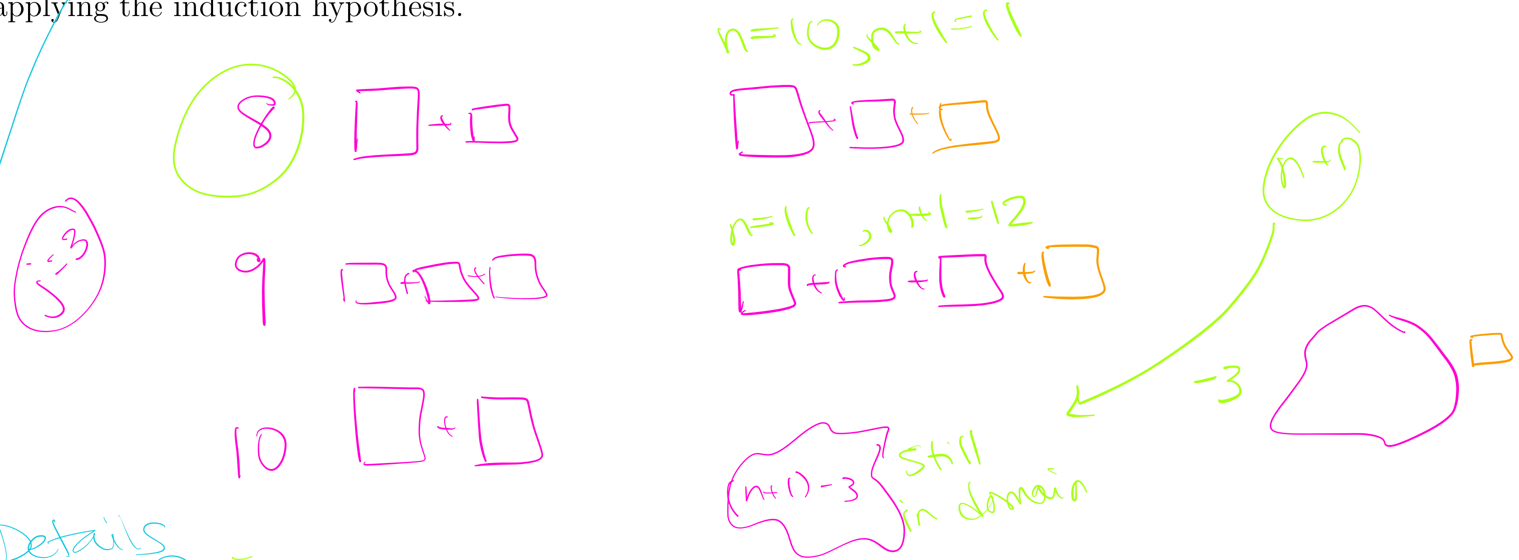
where the predicate $CanPay$ with domain $\mathbb{N}$ is

$$CanPay(n) = \exists x \in \mathbb{N} \exists y \in \mathbb{N} (5x + 3y = n)$$

Proof (idea): First, explicitly give witnesses or general arguments for postages between 0 and 7. To prove the universal claim, we can use mathematical induction or strong induction.

Approach 1, mathematical induction: if we have stamps that add up to n cents, need to use them (and others) to give $n + 1$ cents. How do we get 1 cent with just 3-cent and 5-cent stamps?
Either take away a 5-cent stamps and add two 3-cent stamps,
or take away three 3-cent stamps and add two 5-cent stamps.
The details of this proof by mathematical induction are making sure we have enough stamps to use one of these approaches.

Approach 2, strong induction: assuming we know how to make postage for **all** smaller values (greater than or equal to 8), when we need to make $n+1$ cents, add one 3 cent stamp to however we make $(n + 1) - 3$ cents. The details of this proof by strong induction are making sure we stay in the domain of the universal when applying the induction hypothesis.



Details in RQ