Monday November 1

Today's session is on Zoom, log in with your @ucsd.edu account https://ucsd.zoom.us/j/97431852722 Meeting ID: 974 3185 2722

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

Basis Step:	$\mathtt{A} \in S, \mathtt{C} \in S, \mathtt{U} \in S, \mathtt{G} \in S$
Recursive Step:	If $s \in S$ and $b \in B$, then $sb \in S$

where sb is string concatenation.

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

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

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

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

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**.

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.

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

Every RNA strand has length greater than zero.

Notice rnalen: $S \to \mathbb{Z}^+$
since codomain is a collection of positive numbers
any output of rnalen has to be positive

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

predicate

[For every base there is a strand where the number of occurrences of this base in this strand is at least 1]

Every base occurs in some strand

Proof by exhaustive

WTS ① $\exists s \in S\ (basecount(\ (s,A)\ ) > 0)$
WTS ② $\exists s \in S\ (basecount(\ (s,C)\ ) > 0)$
WTS ③ $\exists s \in S\ (basecount(\ (s,G)\ ) > 0)$
WTS ④ $\exists s \in S\ (basecount(\ (s,U)\ ) > 0)$

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

For every strand there's at least one base that occurs in it (at least once).

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

predicate

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

witness A ✓

Confirm
- $A \in S$ (domain)
- LHS = rnalen(A) $\boxed{=}$ 1 (by basis step)
- RHS = basecount((A,A)) $\boxed{=}$ 1 (by 1st case in basis step)

There is a strand all of whose bases are A.

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

For every strand its length is greater than or equal to the number of occurrences of A in the strand.

The set S is infinite
The set B is finite (4 distinct elements)

arbitrary: fixed but unknown, from given set.

Recursive set

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

Structural Induction

Proof: Let s be an arbitrary RNA strand. By the recursive definition of S, either $\boxed{s \in B}$ (basis) 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.

Basis **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.

$rnalen(s) = rnalen(s_0 b) \overset{def}{=} 1 + rnalen(s_0)$

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

Predicate is true at u — Predicate is true at ub

$$\forall u \in S\ \forall b \in B\ (\ rnalen(u) > 0 \to rnalen(ub) > 0\ )$$

Consider an arbitrary RNA strand u and an arbitrary base b, and $\boxed{\text{assume}}$ towards a direct proof, that

$$\boxed{\text{Induction Hypothesis} \quad rnalen(u) > 0}$$

predicate is true at u

We need to show that $rnalen(ub) > 0$.

GOAL

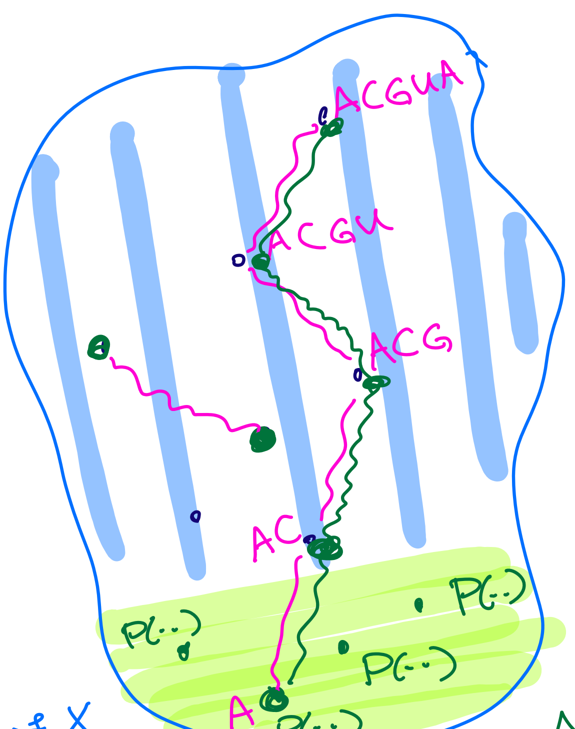
$$\text{LHS} = rnalen(ub) = 1 + rnalen(u) > 1 + 0 = 1 > 0 = \text{RHS}$$

def of function — assumption IH — predicate is true at ub

as required.

> **Proof by Structural Induction** To prove a universal quantification over a recursively defined set: $\forall x \in X\ P(x)$
>
> **Basis Step**: Show the statement holds for elements specified in the basis step of the definition. (P(..) is T — P(...) is T — of X)
>
> **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.

X



Recursive step in definition of X.

Basis step in definition of X

Goal: Prove $\forall x \in X\ P(x)$.

Basis Step: WTS P(...) for each basis element.

Recursive Step: WTS

$$\forall u \in X \left(\boxed{P(u)}^{IH} \to P(\text{element built from } u) \right)$$

Take arbitrary u in X.

Assume IH, P(u).

WTS P(element built from u)

recursively defined set

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

predicate

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

$$\overset{①}{}(\underbrace{\mathit{rnalen}(\mathtt{A})}_{\text{LHS}} \geq \underbrace{\mathit{basecount}(\ (\mathtt{A}, \mathtt{A})\ )}_{\text{RHS}}) \wedge \overset{②}{}(\mathit{rnalen}(\mathtt{C}) \geq \mathit{basecount}(\ (\mathtt{C}, \mathtt{A})\ ))$$
$$\wedge \overset{③}{}(\mathit{rnalen}(\mathtt{U}) \geq \mathit{basecount}(\ (\mathtt{U}, \mathtt{A})\ )) \wedge \overset{④}{}(\mathit{rnalen}(\mathtt{G}) \geq \mathit{basecount}(\ (\mathtt{G}, \mathtt{A})\ ))$$

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

① $rnalen(A) = 1$ by.. $basecount((A,A)) = 1$ by.. so LHS = RHS, i.e. LHS ≥ RHS ✓

② $rnalen(C) = 1$ by.. $basecount((C,A)) = 0$ by... so LHS > RHS, i.e. LHS ≥ RHS ✓

③ $rnalen(G) = 1$ by.. $basecount((G,A)) = 0$ by.. so LHS > RHS, i.e. LHS ≥ RHS ✓

④ $rnalen(U) = 1$ by.. $basecount((U,A)) = 0$ by.. so LHS > RHS, i.e. LHS ≥ RHS ✓

Recursive Case: We will prove that

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

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

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

$$\text{LHS} = \mathit{rnalen}(ub) = 1 + \mathit{rnalen}(u)$$

The recursive step in the definition of the function *basecount* has two cases. We notice that $b = \mathtt{A} \vee b \neq \mathtt{A}$ 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*:

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

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

$$\text{RHS} = \mathit{basecount}(\ (ub, \mathtt{A})\ ) = 1 + \mathit{basecount}(\ (u, \mathtt{A})\ ) \underset{\text{IH}}{\leq} 1 + \mathit{rnalen}(u) = \mathit{rnalen}(ub) = \text{LHS}.$$

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

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

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

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

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

$$\text{RHS} = \mathit{basecount}(\ (ub, \mathtt{A})\ ) = \mathit{basecount}(\ (u, \mathtt{A})\ ) \underset{\text{IH}}{\leq} \mathit{rnalen}(u) \underset{\text{by properties of addition}}{<} 1 + \mathit{rnalen}(u) = \mathit{rnalen}(ub) = \text{LHS}.$$

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

Review

Recall the definitions of the functions *rnalen* and *basecount* from class.

1. Select all and only options that give a witness for the existential quantification

$$\exists s \in S \ (\ rnalen(s) = basecount(\ (s, \mathtt{U})\ )\ )$$

   (a) `A`
   (b) `UU`
   (c) `CU`
   (d) $(\mathtt{U}, 1)$
   (e) None of the above.

2. Select all and only options that give a counterexample for the universal quantification

$$\forall s \in S \ (\ rnalen(s) > basecount(\ (s, \mathtt{G})\ )\ )$$

   (a) `U`
   (b) `GG`
   (c) `AG`
   (d) `CUG`
   (e) None of the above.

3. Select all and only the true statements

   (a) $\forall s \in S\ \exists b \in B\ \ (\ rnalen(s) = basecount(\ (s, b)\ )\ )$
   (b) $\exists s \in S\ \forall b \in B\ \ (\ rnalen(s) = basecount(\ (s, b)\ )\ )$
   (c)

$$\forall s_1 \in S\ \forall s_2 \in S\ \forall b \in B\ \Big(\ \big(rnalen(s_1) = basecount(\ (s_1, b)\ ) \\ \land rnalen(s_2) = basecount(\ (s_2, b)\ ) \land rnalen(s_1) = rnalen(s_2)\big) \to s_1 = s_2\Big)$$

   (d) None of the above.

Wednesday November 3

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.

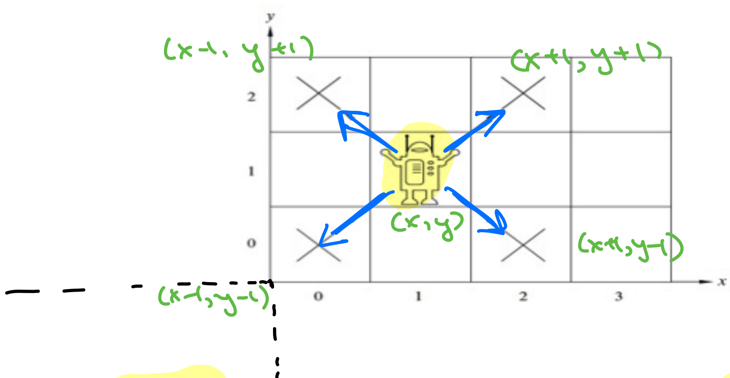
Theorem: Statement that can be shown to be true, usually an important one.

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 P is defined by:

Basis Step: $(0,0) \in P$ starting position

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

Example elements of P are: $(0,0)$ and $(0,2)$ and $(1000, 1000)$ and $(2,0)$ and $(1,1)$

$(0,0) \to (1,1) \to (0,2)$

Claim: $\forall x \in \mathbb{Z} \ ((x,x) \in P)$ Pf: exercise.

Lemma: $\forall (x,y) \in P (x+y$ is an even integer $)$ Invariant

Why are we calling this a lemma? Can be used to prove that $(1,0) \notin P$.

Proof of theorem using lemma: To show is $(1,0) \notin P$. Rewriting the lemma to explicitly restrict the domain of the universal, we have $\forall (x,y)$ ($(x,y) \in P \to (x+y$ is an even integer$)$). Since the universal is true, ($(1,0) \in P \to (1+0$ 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), $1 \textbf{ mod } 2 = 1$ which is not 0 so the conclusion is false. A true conditional with a false conclusion must have a false hypothesis. Thus, $(1,0) \notin P$, QED. □

Proof of lemma by structural induction: WTS $\forall (x,y) \in P$ ($x+y$ is even int)

Basis Step: WTS $0+0$ is even integer.

Calculating: $0+0=0$

By long division $0 = 2 \cdot 0 + 0$ (quotient $0 \text{ div } 2 = 0$, remainder $0 \bmod 2 = 0$)

since $0 \bmod 2 = 0$, 2 is a factor of 0

hence 0 is even ∎

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

property is true about (x,y) → property is true about next position.

$$(x + y \text{ is an even integer}) \rightarrow (\text{sum of coordinates of next position is even integer})$$

Assume **as the induction hypothesis, IH** that: $x+y$ is an even integer.

By definition of even, this means there's a witness, call it c, an integer such that

$$x+y = 2c$$

By rec step in definition of P

next position is $(x+1, y+1)$ $\vee$ next position is $(x+1, y-1)$ $\vee$ next position is $(x-1, y+1)$ $\vee$ next position is $(x-1, y-1)$

By cases

1. Assume next position is $(x+1, y+1)$. WTS sum of $x+1$ and $y+1$ is even. $(x+1)+(y+1) = x+y+1+1 = x+y+2 \overset{IH}{=} 2c+2 = 2(\underbrace{c+1}_{\text{int}})$
so 2 is a factor of $(x+1)+(y+1)$, as required.

2. Assume next position is $(x+1, y-1)$. WTS sum of $x+1$ and $y-1$ is even. $(x+1)+(y-1) = x+y\cancel{+1}\cancel{-1} = x+y$. By IH $x+y$ is even
so we are done this case.

3. Assume next position is $(x-1, y+1)$. WTS sum of $x-1$ and $y+1$ is even. $(x-1)+(y+1) = x+y-1+1 = x+y$ so IH guarantees it is even, as required.

4. Assume next position is $(x-1, y-1)$. WTS sum of $x-1$ and $y-1$ is even. $(x-1)+(y-1) = x+y-1-1 = x+y-2 \overset{IH}{=} 2c-2 = 2(\underbrace{c-1}_{\text{int}})$
so 2 is a factor of $(x-1)+(y-1)$, as required.

No matter which next position we take, the robot will land in ordered pair whose coords sum to even int.

Recall $\mathbb{N}$ is defined as Basis Step: $0 \in \mathbb{N}$
Rec Step: If $n \in \mathbb{N}$, $n+1 \in \mathbb{N}$.

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 $\frac{1}{2^0}$

$sumPow(1) =$ $sumPow(0) + 2^1 = 1+2 = 3$ Rec Step $\frac{1}{2^1}$ $\frac{1}{2^0}$

$sumPow(2) =$ $sumPow(1) + 2^2 = 3+4 = 7$ Rec Step $\frac{1}{2^2}$ $\frac{1}{2^1}$ $\frac{1}{2^0}$

Fill in the blanks in the following proof of

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

universal about property

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.

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:

property at n property at n+1

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

left hand side

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

right hand side

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

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 $n+1$.

Review

1.

Recall the set P defined by the recursive definition

Basis Step:	$(0,0) \in P$
Recursive Step:	If $(x,y) \in P$ then $(x+1,y+1) \in P$ and $(x+1,y-1) \in P$ and $(x-1,y-1) \in P$ and $(x-1,y+1) \in P$

(a) Select all and only the ordered pairs below that are elements of P

i. $(0,0)$

ii. $(4,0)$

iii. $(1,1)$

iv. $(1.5,2.5)$

v. $(0,-2)$

(b) What is another description of the set P ? (Select all and only the true descriptions.)

i. $\mathbb{Z} \times \mathbb{Z}$

ii. $\{(n,n) \mid n \in \mathbb{Z}\}$

iii. $\{(a,b) \in \mathbb{Z} \times \mathbb{Z} \mid (a+b) \textbf{ mod } 2 = 0\}$

2.

Select all and only the true statements below about the relationship between structural induction and mathematical induction.

(a) Both structural induction and mathematical induction are proof strategies that may be useful when proving universal claims about recursively defined sets.

(b) Mathematical induction is a special case of structural induction, for the case when the domain of quantification is $\{n \in \mathbb{Z} \mid n \geq b\}$ for some integer b.

(c) Universal claims about the set of all integers may be proved using structural induction but not using mathematical induction.

3.

Consider the following function definitions

$$2^n : \mathbb{N} \to \mathbb{N} \text{ given by } 2^0 = 1 \quad \text{and} \quad 2^{n+1} = 2 \cdot 2^n$$

$$n! : \mathbb{N} \to \mathbb{N} \text{ given by } 0! = 1 \quad \text{and} \quad (n+1)! = (n+1)n!$$

(a) Select all and only true statements below:

i. $2^0 < 0!$
ii. $2^1 < 1!$
iii. $2^2 < 2!$
iv. $2^3 < 3!$
v. $2^4 < 4!$
vi. $2^5 < 5!$
vii. $2^6 < 6!$
viii. $2^7 < 7!$

(b) Fill in the blanks in the following proof.

Claim: For all integers n greater than or equal to 4, $2^n < n!$

Proof: We proceed by mathematical induction on the set of integers greater than or equal to 4.
Basis step: Using the __BLANK 1__,

$$2^4 = 2 \cdot 2^3 = 2 \cdot 2 \cdot 2^2 = 2 \cdot 2 \cdot 2 \cdot 2^1 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2^0 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 1 = 16$$

and

$$4! = 4 \cdot 3! = 4 \cdot 3 \cdot 2! = 4 \cdot 3 \cdot 2 \cdot 1! = 4 \cdot 3 \cdot 2 \cdot 1 \cdot 0! = 4 \cdot 3 \cdot 2 \cdot 1 \cdot 1 = 24$$

Since $16 < 24$, we have proved that $2^4 < 4!$, as required.

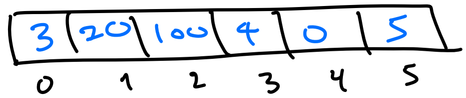
Recursive step: Consider an arbitrary integer k that is greater than or equal to 4 and assume as the __BLANK 2__, that $2^k < k!$. We want to show that $2^{k+1} < (k+1)!$.

$$\begin{aligned} 2^{k+1} &= 2 \cdot 2^k && \text{by } \underline{BLANK3} \\ &< 2 \cdot k! && \text{by } \underline{BLANK4} \\ &< k \cdot k! && \text{by } \underline{BLANK5} \\ &< (k+1) \cdot k! && \text{by } \underline{BLANK6} \\ &= (k+1)! && \text{by } \underline{BLANK7} \end{aligned}$$

as required.

i. properties of addition, multiplication, and $<$ for real numbers
ii. definitions of the functions 2^n and $n!$
iii. definition of k
iv. induction hypothesis

arrays



Friday November 5

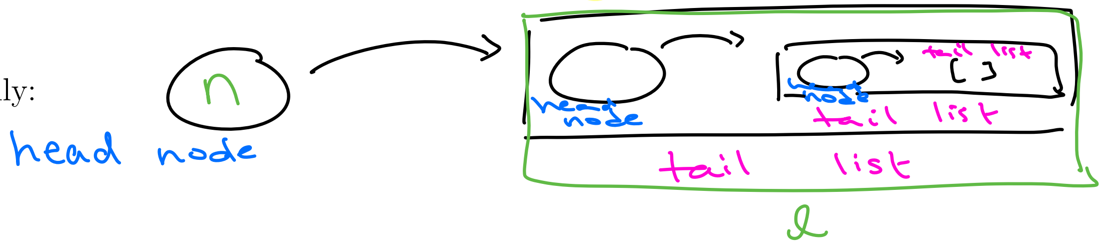
Definition The set of linked lists of natural numbers L is defined recursively by

Basis Step:	$[] \in L$	simplest
Recursive Step:	If $l \in L$ and $n \in \mathbb{N}$, then $(n, l) \in L$	

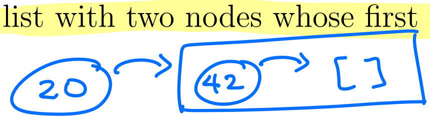
data for head node

tail in new list

Visually:



Example: the list with two nodes whose first node has 20 and whose second node has 42



(20, (42, [ ]))

Parentheses: +1 +2 +1 +0

Definition: The length of a linked list of natural numbers L, $length : L \to \mathbb{N}$ is defined by

Basis Step:		$length(\ [] \ ) = 0$
Recursive Step:	If $l \in L$ and $n \in \mathbb{N}$, then	$length(\ (n, l) \ ) = 1 + length(l)$

Expected behavior

length((20, (42, [ ])))
= 2

Function application example:

length((20, (42, [ ])))
= 1 + length((42, [ ]))
= 1 + 1 + length([ ]) = 1 + 1 + 0 = 2

Definition: The function $prepend : L \times \mathbb{N} \to L$ that adds an element at the front of a linked list is defined by prepend((l , m)) = (m, l) for each $m \in \mathbb{N}$, $l \in L$.

Expected behavior

prepend(((20, (42, [ ])) , 3))
l n
= (3, (20, (42, [ ])))

Function application example

prepend(((20, (42, [ ])) , 3))
= (3, (20, (42, [ ])))

Definition The function $append : L \times \mathbb{N} \to L$ that adds an element at the end of a linked list is defined by

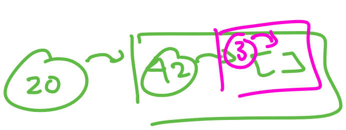
parameter

Basis Step:	If $m \in \mathbb{N}$ then	append(([ ], m)) = (m, [ ])
Recursive Step:	If $l \in L$ and $n \in \mathbb{N}$ and $m \in \mathbb{N}$, then	append(((n, l) , m)) = (n, append((l, m)))

simplest element in L is [ ]

Expected behavior

append(((20, (42, [ ])) , 3))
l n
= (20, (42, (3, [ ])))



Function application example

append(((20, (42, [ ])) , 3))
= (20, append(((42, [ ]) , 3)))
= (20, (42, append(([ ], 3))))
= (20, (42, (3, [ ])))

Claim: $\forall l \in L\ (\ length(\ append(\ (l, 100)\ )\ ) > length(l)\ )$

Proof: By structural induction on L, we have two cases:

(universal claim about L)

Basis Step

1.	**To Show** $length(\ append(\ ([], 100)\ )\ ) > length(\ []\ )$	Because $[]$ is the only element defined in the basis step of L, we only need to prove that the property holds for $[]$.
2.	**To Show** $length(\ (100, [])\ ) > length(\ []\ )$	By basis step in definition of *append*.
3.	**To Show** $(1 + length(\ []\ )) > length(\ []\ )$	By recursive step in definition of *length*.
4.	**To Show** $1 + 0 > 0$	By basis step in definition of *length*.
5.	T	By properties of integers
	QED	Because we got to T only by rewriting **To Show** to equivalent statements, using well-defined proof techniques, and applying definitions.

Recursive Step

Consider an arbitrary: $l' \in L$, $n \in \mathbb{N}$, and we assume as the **induction hypothesis** that:

$$\boxed{length(\ append(\ (l', 100\ )\ )\ ) > length(l')} \quad \text{IH}$$

Our goal is to show that $length(\ append(\ (\ (n, l'), 100\ )\ )\ ) > length(\ (n, l')\ )$ is also true. We start by working with one side of the candidate inequality:

$$\begin{aligned}
LHS &= length(\ append(\ (\ (n, l'), 100\ )\ )\ ) \\
&= length(\ (n, append(\ (l', 100)\ )\ )\ ) && \text{by the recursive definition of } append \\
&= 1 + length(\ append(\ (l', 100)\ )\ ) && \text{by the recursive definition of } length \\
&> 1 + length(l') && \text{by the induction hypothesis} \\
&= length((n, l')) && \text{by the recursive definition of } length \\
&= RHS
\end{aligned}$$

Prove or disprove: $\forall n \in \mathbb{N}\ \exists l \in L\ (\ length(l) = n\ )$

For each natural number there is a list whose length is that number.

Pf: By mathematical induction

Basis Step: WTS $\exists l \in L\ (length(l) = 0)$

Consider $l = [\,]$ as a witness

- $[\,]$ is in the domain by basis step in definition of L
- $length(l) = 0$ by basis step in definition of length.

Recursive Step: Consider arbitrary $n \in \mathbb{N}$

Assume, as IH that

$\exists l \in L\ (length(l) = n)$

There is a witness $l_0 \in L$ with $length(l_0) = n$

WTS $\exists l \in L\ (length(l) = n+1)$

Build $l = (2, l_0)$

- Domain? Yes
- Predicate?

$length(\ (2, l_0)\ ) = 1 + length(l_0) = 1 + n$

Review

Recall the definition of linked lists from class.

Consider this (incomplete) definition:

Definition The function *increment* : ____________ that adds 1 to the data in each node of a linked list is defined by:

$$\begin{array}{llll} & & increment : \_\_\_\_\_\_ & \to \_\_\_\_\_\_ \\ \text{Basis Step:} & & increment([]) & = [] \\ \text{Recursive Step:} & \text{If } l \in L, n \in \mathbb{N} & increment((n,l)) & = (1+n, increment(l)) \end{array}$$

Consider this (incomplete) definition:

Definition The function $sum : L \to \mathbb{N}$ that adds together all the data in nodes of the list is defined by:

$$\begin{array}{llll} & & sum : L & \to \mathbb{N} \\ \text{Basis Step:} & & sum([]) & = 0 \\ \text{Recursive Step:} & \text{If } l \in L, n \in \mathbb{N} & sum((n,l)) & = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \end{array}$$

You will compute a sample function application and then fill in the blanks for the domain and codomain of each of these functions.

1. Based on the definition, what is the result of $increment((4,(2,(7,[]))))$? Write your answer directly with no spaces.

2. Which of the following describes the domain and codomain of *increment*?

   (a) The domain is L and the codomain is $\mathbb{N}$
   (b) The domain is L and the codomain is $\mathbb{N} \times L$
   (c) The domain is $L \times \mathbb{N}$ and the codomain is L
   (d) The domain is $L \times \mathbb{N}$ and the codomain is $\mathbb{N}$
   (e) The domain is L and the codomain is L
   (f) None of the above

3. Assuming we would like $sum((5,(6,[])))$ to evaluate to 11 and $sum((3,(1,(8,[]))))$ to evaluate to 12, which of the following could be used to fill in the definition of the recursive case of *sum*?

   (a) $\begin{cases} 1 + sum(l) & \text{when } n \neq 0 \\ sum(l) & \text{when } n = 0 \end{cases}$
   (b) $1 + sum(l)$
   (c) $n + increment(l)$
   (d) $n + sum(l)$
   (e) None of the above

4. Choose only and all of the following statements that are **well-defined**; that is, they correctly reflect the domains and codomains of the functions and quantifiers, and respect the notational conventions we use in this class. Note that a well-defined statement may be true or false.

   (a) $\forall l \in L\,(sum(l))$

   (b) $\exists l \in L\,(sum(l) \wedge length(l))$

   (c) $\forall l \in L\,(sum(increment(l)) = 10)$

   (d) $\exists l \in L\,(sum(increment(l)) = 10)$

   (e) $\forall l \in L\,\forall n \in \mathbb{N}\,((n \times l) \subseteq L)$

   (f) $\forall l_1 \in L\,\exists l_2 \in L\,(increment(sum(l_1)) = l_2)$

   (g) $\forall l \in L\,(length(increment(l)) = length(l))$

5. Choose only and all of the statements in the previous part that are both well-defined and true.