Review of Chapter 5 Induction + recursion

We use mathematical induction to prove that all the propositions $P(1)$, $P(2)$, $P(3)$, ... are true. I want you to use these steps:

(A) Write down what $P(n)$ says.
(B) Basis step — check $P(1)$ is true
(C) Inductive step — check that you can use $P(k)$ being true to show that $P(k+1)$ is true.
(D) Write "So by mathematical induction $P(n)$ is true for all $n \geq 1$.

Example (1) Prove that $\frac{1}{1\cdot 2} + \frac{1}{2\cdot 3} + \cdots + \frac{1}{n(n+1)} = \frac{n}{n+1}$

for all $n \geq 1$.

Solution:

(A) $P(n)$ says that $\frac{1}{1\cdot 2} + \frac{1}{2\cdot 3} + \cdots + \frac{1}{n(n+1)} = \frac{n}{n+1}$

(B) Basis step. $P(1)$ says $\frac{1}{1\cdot 2} = \frac{1}{1+1}$ $(n=1)$

this is $\frac{1}{2} = \frac{1}{2}$ ✓ so $P(1)$ true.

(C) Inductive step.

Assume $P(k)$ is true.

So $\frac{1}{1\cdot 2} + \frac{1}{2\cdot 3} + \cdots + \frac{1}{k(k+1)} = \frac{k}{k+1}$ true

add to both sides $\frac{1}{(k+1)(k+2)}$ $\frac{1}{(k+1)(k+2)}$

So $\frac{1}{1\cdot 2} + \frac{1}{2\cdot 3} + \cdots + \frac{1}{k(k+1)} + \frac{1}{(k+1)(k+2)} = \frac{k}{k+1} + \frac{1}{(k+1)(k+2)}$

is true

the left side matches $P(k+1)$. How about the right side?

$$\frac{k}{k+1} + \frac{1}{(k+1)(k+2)} = \frac{k(k+2)}{(k+1)(k+2)} + \frac{1}{(k+1)(k+2)}$$

$$= \frac{k^2+2k+1}{(k+1)(k+2)}$$

$$= \frac{\cancel{(k+1)}(k+1)}{\cancel{(k+1)}(k+2)}$$

So $\frac{1}{1\cdot 2} + \frac{1}{2\cdot 3} + \cdots + \frac{1}{(k+1)(k+2)} = \frac{k+1}{k+2}$ is true

and this is $P(k+1)$. We have finished the inductive step.

(D) So by mathematical induction $P(n)$ is true for all $n \geq 1$.

Review the notes for section 5.1 to see many more examples of induction.

In the usual induction the inductive hypothesis is assuming $P(k)$ is true.

We also looked at strong induction. There the inductive hypothesis is that
$P(1), P(2), \ldots, P(k)$ are all true.

Either way you must use the inductive hypothesis to show that $P(k+1)$ is true.

Recursion is a way to define functions, sets, sequences or graphs step-by-step.
A recursive definition has two parts — a basis step and a recursive step.

Example (2) Define the function $q(n)$ with

Basis step: $q(0) = 4$
Recursive step: $q(n+1) = 10q(n)$

Compute $q(1), q(2), q(3)$. Can you guess a formula for $q(n)$?

Solution: Take $n=0$ in the recursive step to see

$$q(0+1) = 10q(0)$$
so $$q(1) = 10 \cdot 4 = 40$$

then $n=1$: $$q(1+1) = 10q(1)$$
so $$q(2) = 10 \cdot 40 = 400$$

with $n=2$: $$q(2+1) = 10q(2)$$
$$q(3) = 10 \cdot 400 = 4000$$

We can see that $q(n) = 4 \cdot 10^n$ is how the pattern goes.

This explicit formula $q(n) = 4 \cdot 10^n$ could be proved by induction.
Use $P(n)$ says $q(n) = 4 \cdot 10^n$

and use the recursive step to check the inductive step.

Next we give an example of a recursively defined set (of ordered pairs).

Example (3) Let S be defined by

Basis step: $(2,1) \in S$

Recursive step: If $(a,b) \in S$ then $(a+4, b+2) \in S$.

Find 4 different elements of S.

Solution: At the start we only know one element of S, that is $(2,1)$, so we can only use the recursive step with $a=2, b=1$

$$\text{then } (2+4, 1+2) \in S$$
$$= (6,3)$$

Can now take $a=6, b=3$ so

$$(6+4, 3+2) = (10, 5) \in S.$$

Next $(14, 7) \in S$. We've found that

$(2,1)$, $(6,3)$, $(10,5)$ and $(14,7)$ are in S.

We saw in the notes that bit strings like 1011 or 0100010110 can also be defined recursively. To make a new bit string just add a 0 or 1 on the right. So from 1011 we can make

$$1011 \to 10110 \text{ or } 1011 \to 10111$$

For the basis step start with the empty bit string (called λ).

Step-by-step we get $\lambda, 0, 1, 00, 01, 10, 11, 000 \ldots$

These are the set of strings Σ^* over the alphabet $\Sigma = \{0, 1\}$.

For any alphabet (set of symbols) Σ the recursive definition of Σ^* is

Basis step: empty string $\lambda \in \Sigma^*$
Recursive step: if $w \in \Sigma^*$ and $x \in \Sigma$ then $wx \in \Sigma^*$.

Example ④ If $\Sigma = \{9\}$ find 5 elements of Σ^*.

Solution: Start with $w = \lambda$ and $x = 9$

and the recursive step says $\lambda 9 = 9 \in \Sigma^*$.
Next take $w = 9$, $x = 9$ and $99 \in \Sigma^*$.
Next $w = 99$, $x = 9$ so $999 \in \Sigma^*$,
$w = 999$, $x = 9$ makes $9999 \in \Sigma^*$.

Five elements of Σ^* are $\lambda, 9, 99, 999, 9999$.

We just get strings of 9s in this Σ^*.

We also gave recursive definitions for rooted trees and full binary trees.

We can use structural induction to prove things about recursively defined objects.

It works like this: 1 show the basis objects have the property, 2 show that if old objects have the property then the new objects made in the recursive step must also have the property. If 1 and 2 can be verified then by structural induction all the objects have the property.

Example 5 For the set S in example 3 prove that if $(a,b) \in S$ then $a = 2b$.

Solution: The basis object is $(2,1)$ and $a=2$, $b=1$ means $a=2b$ is true.

Next, suppose $(a,b) \in S$ has $a = 2b$
old

then what about $(a+4, b+2)$?
new from recursive step

$$a+4 = 2b+4 = 2(b+2)$$

The new object has the property we want.

So by structural induction, for every $(a,b) \in S$ we have $a = 2b$.

Recursive algorithms execute a task by calling themselves with smaller inputs.

We saw examples in the notes that computed factorials and Fibonacci numbers.

The Euclidean Algorithm is a recursive algorithm to compute the gcd of two numbers:

```
procedure gcd(a,b : integers 0≤a<b)

If a=0 then return b
else return gcd(b mod a, a)

{output is gcd(a,b)}
```

Example (6) Show how this procedure computes the gcd of 391, 506.

Solution: $a = 391$, $b = 506$

$a \neq 0$

so next find $\gcd(b \bmod a, a)$

$= \gcd(\underbrace{506 \bmod 391}, 391)$

$$\begin{array}{r} 1 \\ 391\overline{)506} \\ -391 \\ \hline 115 \end{array} \leftarrow \text{rem}$$

$= \gcd(115, 391)$

calls itself

new $a = 115$, new $b = 391$

$a \neq 0$

find next $\gcd(\underbrace{391 \bmod 115}, 115)$

$$\begin{array}{r} 3 \\ 115 \overline{)391} \\ -345 \\ \hline 46 \end{array} \leftarrow \text{rem}$$

$= \gcd(46, 115)$

calls itself again

- $a = 46$, $b = 115$

  $a \neq 0$

  need $\gcd(\underbrace{115 \bmod 46}, 46)$

$$\begin{array}{r} 2 \\ 46 \overline{)115} \\ -92 \\ \hline 23 \end{array} \leftarrow \text{rem}$$

$= \gcd(23, 46)$

- $a = 23$, $b = 46$

  $a \neq 0$

  need $\gcd(46 \bmod 23, 23)$

  $= \gcd(0, 23)$

- $\underbrace{a = 0}$, $b = 23$

  return $b = 23$

Computation shows gcd of 391 and 506 is $\boxed{23}$.

We also looked at the recursive algorithm Merge Sort that effeciently sorts sets of integers into increasing order.