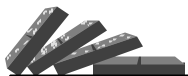
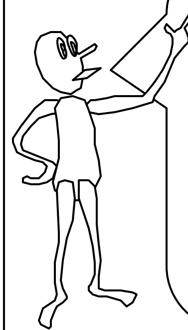
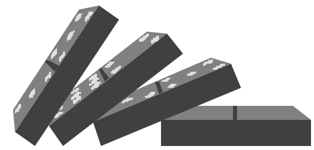
Great Theoretical Ideas In Computer Science		
Steven Rudich		CS 15-251 Spring 2005
Lecture 1	Jan 11, 2005	Carnegie Mellon University

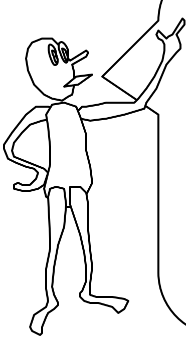
Induction: One Step At A Time





Today we will talk about INDUCTION



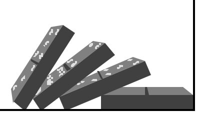


Induction is the primary way we:

1. Prove theorems
2. Construct and define objects

Let's start with dominoes

Domino Principle: Line up any number of dominos in a row; knock the first one over and they will all fall.



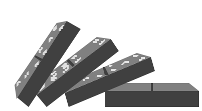
n dominoes numbered 1 to n

$F_k \equiv$ The k^{th} domino falls

If we set them all up in a row then we know that each one is set up to knock over the next one:

For all $1 \le k < n$:

$$F_k \Rightarrow F_{k+1}$$



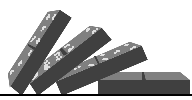
n dominoes numbered 1 to n

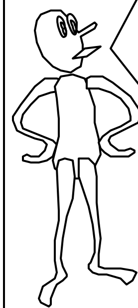
F_k ´ The k^{th} domino falls

For all 1 = k < n:

F_k) F_{k+1}

F_1) F_2) F_3) ...
F_1) All Dominoes Fall





Computer Scientists don't start numbering things at 1, they start at 0.

YOU will spend a career doing this, so GET USED TO IT NOW.

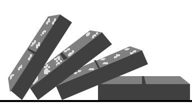
n dominoes numbered 0 to n-1

F_k ´ The k^{th} domino falls

For all 0 = k < n-1:

F_k) F_{k+1}

F_0) F_1) F_2) ...
F_0) All Dominoes Fall



Standard Notation/Abbreviation
"for all" is written "8"

Example:

For all k>0, P(k)

is equivalent to

8k>0, P(k)

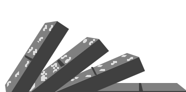
n dominoes numbered 0 to n-1

F_k ´ The k^{th} domino falls

8k, 0 = k < n-1:

F_k) F_{k+1}

F_0) F_1) F_2) ...
F_0) All Dominoes Fall



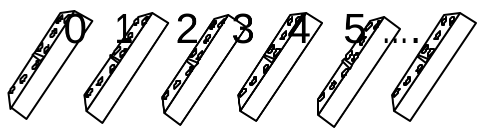
The Natural Numbers

ℕ = { 0, 1, 2, 3, . . .}

The Natural Numbers

ℕ = { 0, 1, 2, 3, . . .}

One domino for each natural number:



The Infinite Domino Principle

F_k ´ The k^{th} domino falls

Suppose F_0
Suppose for each natural number k,
F_k) F_{k+1}
Then All Dominoes Fall!

F_0) F_1) F_2) ...

The Infinite Domino Principle

F_k ´ The k^{th} domino falls

Suppose F_0
Suppose for each natural number k,
F_k) F_{k+1}
Then All Dominoes Fall!

Proof: If they do not all fall, there must be a least numbered domino d>0 that did not fall. Hence, F_{d-1} and not F_d. F_{d-1}) F_d. Hence, domino d fell and did not fall. Contradiction.

Mathematical Induction: statements proved instead of dominoes fallen

Infinite sequence of dominoes.	Infinite sequence of statements: S_0, S_1, ...
F_k ´ domino k falls	F_k ´ S_k proved

Establish 1) F_0
2) 8 k, F_k) F_{k+1}

Conclude that F_k is true for all k

Inductive Proof / Reasoning To Prove ∀k, S_k

Establish “Base Case”: S_0
Establish “Domino Property”: ∀k, S_k) S_{k+1}

∀k, S_k) S_{k+1} {
Assume hypothetically that S_k for *any* particular k;
Conclude that S_{k+1}

Inductive Proof / Reasoning To Prove ∀k, S_k

Establish “Base Case”: S_0
Establish “Domino Property”: ∀k, S_k) S_{k+1}

∀k, S_k) S_{k+1} {
“Induction Hypothesis” S_k
Use I.H. to show S_{k+1}

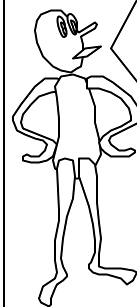
Inductive Proof / Reasoning
To Prove ∀k ¸ b, S_k

Establish "Base Case": S_b
Establish "Domino Property": ∀k ¸ b, S_k) S_{k+1}

Assume k ¸ b

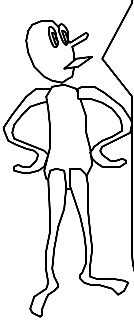
Assume "Inductive Hypothesis": S_k

Prove that S_{k+1} follows



Theorem?

The sum of the first n odd numbers is n^2.

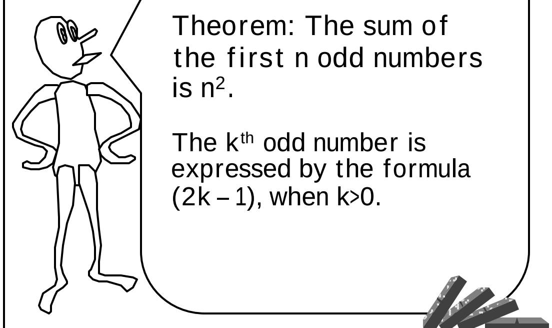


Theorem?

The sum of the first n odd numbers is n^2.

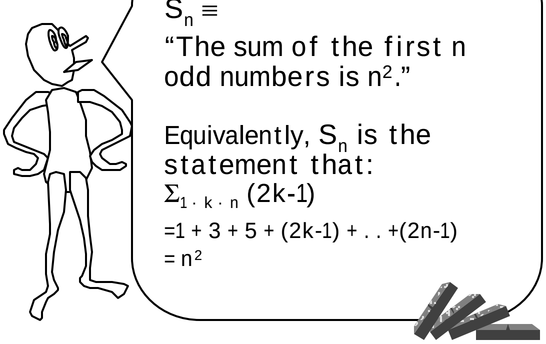
CHECK IT OUT ON SMALL VALUES:

1	= 1
1+3	= 4
1+3+5	= 9
1+3+5+7	= 16



Theorem: The sum of the first n odd numbers is n^2.

The k^{th} odd number is expressed by the formula $(2k – 1)$, when $k>0$.



$S_n \equiv$
"The sum of the first n odd numbers is n^2."

Equivalently, S_n is the statement that:
$\Sigma_{1 \cdot k \cdot n} (2k-1)$
$=1 + 3 + 5 + (2k-1) + . . +(2n-1)$
$= n^2$

$S_n \equiv$ "The sum of the first n odd numbers is n^2."
"$1 + 3 + 5 + (2k-1) + . . +(2n-1)= n^2$"

Trying to establish that: 8n ¸1 S_n

Base case: S_1 is true

The sum of the first 1 odd numbers is 1.

S_n ≡ "The sum of the first n odd numbers is n^2."

"1 + 3 + 5 + (2k-1) + . . +(2n-1)= n^2"

Trying to establish that: 8n ¸ 1 S_n

Assume "Induction Hypothesis": S_k (for any particular k ¸ 1)

1+3+5+…+ (2k-1) = k^2

S_n ≡ "The sum of the first n odd numbers is n^2."

"1 + 3 + 5 + (2k-1) + . . +(2n-1)= n^2"

Trying to establish that: 8n ¸ 1 S_n

Assume "Induction Hypothesis": S_k (for any particular k ¸ 1)

1+3+5+…+ (2k-1) = k^2

Add (2k+1) to both sides.

1+3+5+…+ (2k-1)+(2k+1) = k^2 +(2k+1)

Sum of first k+1 odd numbers = $(k+1)^2$

CONCLUSE: S_{k+1}

S_n ≡ "The sum of the first n odd numbers is n^2."

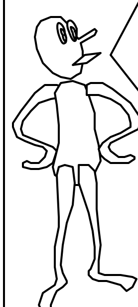
"1 + 3 + 5 + (2k-1) + . . +(2n-1)= n^2"

Trying to establish that: 8n ¸ 1 S_n

Established base case: S_1

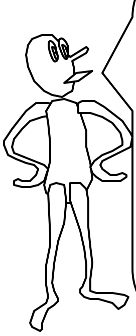
Established domino property: 8 k ¸ 1 S_k) S_{k+1}

By induction of n, we conclude that:
8n ¸ 1 S_n



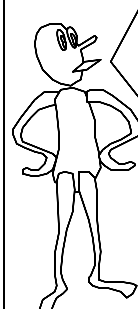
THEOREM:

The sum of the first n odd numbers is n^2.



Theorem?

The sum of the first n numbers is ½n(n+1).



Theorem? The sum of the first n numbers is ½n(n+1).

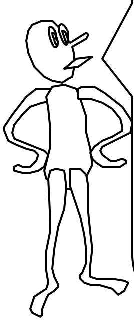
Try it out on small numbers!

1 = 1 = =½ 1(1+1).

1+2 = 3 =½ 2(2+1).

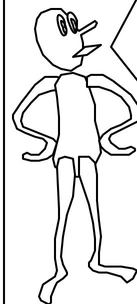
1+2+3 = 6 =½ 3(3+1).

1+2+3+4 = 10=½ 4(4+1).



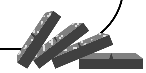
Theorem? The sum of the first n numbers is ½n(n+1).

= 0 = =½ 0(0+1).
1 = 1 = =½ 1(1+1).
1+2 = 3 =½ 2(2+1).
1+2+3 = 6 =½ 3(3+1).
1+2+3+4 = 10=½ 4(4+1).



Notation:
$\Delta_0 = 0$
$\Delta_n = 1 + 2 + 3 + . . . + n\text{-}1 + n$

Let S_n ´
"$\Delta_n = n(n+1)/2$"



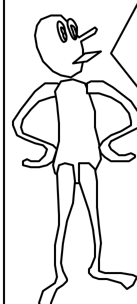
S_n ´ "$\Delta_n = n(n+1)/2$"
Use induction to prove ∀k ¸ 0, S_k

Establish "Base Case": S_0. Δ_0=The sum of the first 0 numbers = 0. Setting n=0 the formula gives 0(0+1)/2 = 0.

Establish "Domino Property": ∀k ¸ 0, S_k) S_{k+1}

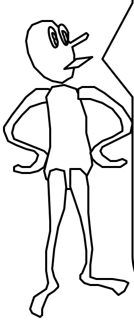
"Inductive Hypothesis" S_k: $\Delta_k = k(k+1)/2$

$\Delta_{k+1} = \Delta_k + (k+1)$
$= k(k+1)/2 + (k+1)$ [Using I.H.]
$= (k+1)(k+2)/2$ [which proves S_{k+1}]



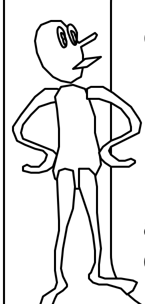
THEOREM:

The sum of the first n numbers is ½n(n+1).



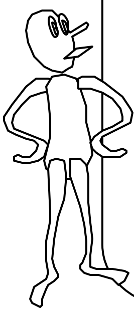
A natural number n>1 is <u>prime</u> if it has no divisors besides 1 and itself.

N.B.
1 is not considered prime.



Easy theorem:
Every natural number>1 can be factored into primes.

N.B.:
It is much more subtle to argue for the existence of a unique prime factorization

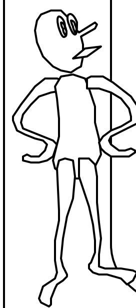


Easy theorem:

Every natural number>1 can be factored into primes.

$S_n \equiv$ "n can be factored into primes"

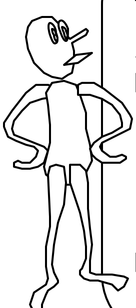
S_2 is true because 2 is prime.



Every natural number>1 can be factored into primes. Base case: 2

Assume 2,3,.....,k-1 all can be factored into primes.

Show k can be factored into primes.

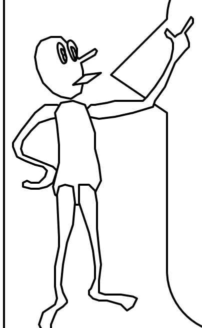


Assume 2,3,.....,k-1 all can be factored into primes.

Show k can be factored into primes.

If k is prime, we are done.

If not, k=ab where 1<a,b<k, hence a and b can be factored into primes. Thus, k is the product of the factors of a and the factors of b.



This illustrates a technical point about using and defining mathematical induction.

<u>All Previous</u> Induction
To Prove $\forall k, S_k$

Establish "Base Case": S_0

Establish that $\forall k, S_k \Rightarrow S_{k+1}$

Let k be any natural number.

Induction Hypothesis:

Assume $\forall j<k, S_j$

Derive S_k

"Strong" Induction
To Prove $\forall k, S_k$

Establish "Base Case": S_0

Establish that $\forall k, S_k \Rightarrow S_{k+1}$

Let k be any natural number.

Assume $\forall j<k, S_j$

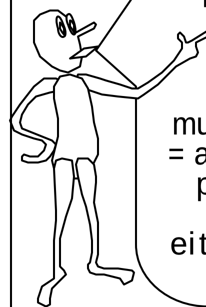
Prove S_k

Least Counter-Example Induction to Prove ∀k, S_k

Establish "Base Case": S_0
Establish that ∀k, S_k) S_{k+1}

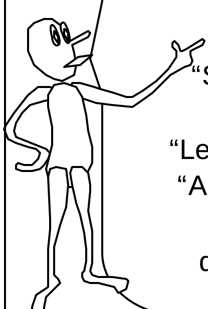
Assume that S_k is the least counter-example.

Derive the existence of a smaller counter-example



All numbers > 1 has a prime factorization.

Let n be the least counter-example. n must not be prime – so n = ab. If both a and b had prime factorizations, then n would. Thus either a or b is a smaller counter-example.



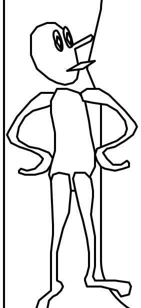
Inductive reasoning is the high level idea:

"Standard" Induction

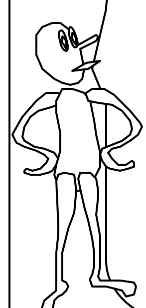
"Least Counter-example"
"All Previous" Induction
all just
different packaging.

"All Previous" Induction Can Be Repackaged As Standard Induction

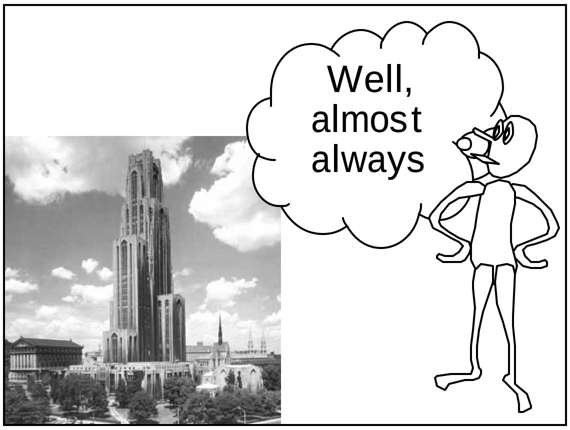
	Define T_i = ∀j · i, S_j
Establish "Base Case": S_0	Establish "Base Case": T_0
Establish that ∀k, S_k) S_{k+1}	Establish that ∀k, T_k) T_{k+1}
Let k be any natural number.	Let k be any natural number.
Assume ∀j<k, S_j	Assume T_{k-1}
Prove S_k	Prove T_k



Induction is also how we can define and construct our world.



So many things, from buildings to computers, are built up stage by stage, module by module, each depending on the previous stages.



Inductive Definition Of Functions

Stage 0, Initial Condition, or Base Case:
Declare the value of the function on some subset of the domain.

Inductive Rules
Define new values of the function in terms of previously defined values of the function

F(x) is defined if and only if it is implied by finite iteration of the rules.

Inductive Definition Of Functions

Stage 0, Initial Condition, or Base Case:
Declare the value of the function on some subset of the domain.

Inductive Rules
Define new values of the function in terms of previously defined values of the function

If there is an x such that F(x) has more than one value – then the whole inductive definition is said to be inconsistent.

Inductive Definition Recurrence Relation for F(X)

Initial Condition, or Base Case:
F(0) = 1

Inductive Rule
For n>0, F(n) = F(n-1) + F(n-1)

n	0	1	2	3	4	5	6	7
F(n)	1							

Inductive Definition Recurrence Relation for F(X)

Initial Condition, or Base Case:
F(0) = 1

Inductive Rule
For n>0, F(n) = F(n-1) + F(n-1)

n	0	1	2	3	4	5	6	7
F(n)	1	2						

Inductive Definition Recurrence Relation for F(X)

Initial Condition, or Base Case:
F(0) = 1

Inductive Rule
For n>0, F(n) = F(n-1) + F(n-1)

n	0	1	2	3	4	5	6	7
F(n)	1	2	4					

Inductive Definition
Recurrence Relation for F(X)

Initial Condition, or Base Case:
F(0) = 1

Inductive Rule
For n>0, F(n) = F(n-1) + F(n-1)

n	0	1	2	3	4	5	6	7
F(n)	1	2	4	8	16	32	64	128

Inductive Definition
Recurrence Relation for F(X) = 2^X

Initial Condition, or Base Case:
F(0) = 1

Inductive Rule
For n>0, F(n) = F(n-1) + F(n-1)

n	0	1	2	3	4	5	6	7
F(n)	1	2	4	8	16	32	64	128

Inductive Definition
Recurrence Relation

Initial Condition, or Base Case:
F(1) = 1

Inductive Rule
For n>1, F(n) = F(n/2) + F(n/2)

n	0	1	2	3	4	5	6	7
F(n)		1						

Inductive Definition
Recurrence Relation

Initial Condition, or Base Case:
F(1) = 1

Inductive Rule
For n>1, F(n) = F(n/2) + F(n/2)

n	0	1	2	3	4	5	6	7
F(n)		1	2					

Inductive Definition
Recurrence Relation

Initial Condition, or Base Case:
F(1) = 1

Inductive Rule
For n>1, F(n) = F(n/2) + F(n/2)

n	0	1	2	3	4	5	6	7
F(n)		1	2		4			

Inductive Definition
Recurrence Relation

Initial Condition, or Base Case:
F(1) = 1

Inductive Rule
For n>1, F(n) = F(n/2) + F(n/2)

n	0	1	2	3	4	5	6	7
F(n)	%	1	2	%	4	%	%	%

Inductive Definition
Recurrence Relation
F(X) = X for X a whole power of 2.

Initial Condition, or Base Case:
F(1) = 1

Inductive Rule
For n>1, F(n) = F(n/2) + F(n/2)

n	0	1	2	3	4	5	6	7
F(n)	%	1	2	%	4	%	%	%

Base Case: 8x2ℕ P(X,0) = X

Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

P(x,y)	0	1	2	3	4	5	6	7
0								
1								
2								
3								

Base Case: 8x2ℕ P(X,0) = X

Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

P(x,y)	0	1	2	3	4	5	6	7
0	0							
1	1							
2	2							
3	3							

Base Case: 8x2ℕ P(X,0) = X

Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

P(x,y)	0	1	2	3	4	5	6	7
0	0	1						
1	1	2						
2	2	3						
3	3	4						

Base Case: 8x2ℕ P(X,0) = X

Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

P(x,y)	0	1	2	3	4	5	6	7
0	0	1	2					
1	1	2	3					
2	2	3	4					
3	3	4	5					

Base Case: 8x2ℕ P(X,0) = X

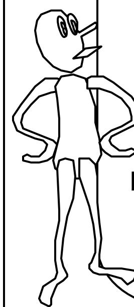
Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

P(x,y)	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	8
2	2	3	4	5	6	7	8	9
3	3	4	5	6	7	8	9	10

Base Case: 8x2ℕ P(X,0) = X

Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

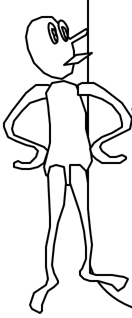
X+Y	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	8
2	2	3	4	5	6	7	8	9
3	3	4	5	6	7	8	9	10



Definition of P:

8x2ℕ P(X,0) = X
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

Any inductive definition with a finite number of base cases, can be translated into a program. The program simply calculates from the base cases on up.



Definition of P:

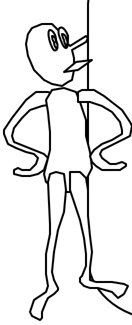
8x2{0,1,2,3} P(X,0) = X
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

What would be the <u>bottom up</u> implementation of P?

```
For k = 0 to 3
      P(k,0)=k
For j = 1 to 7
      For k = 0 to 3
            P(k,j) = P(k,j-1) + 1
```

Bottom-Up Program for P

P(x,y)	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	8
2	2	3	4	5	6	7	8	9
3	3	4	5	6	7	8	9	10



Suppose we wanted to know P(2,3) in particular, but we had not yet done any calculation.

Base Case: 8x2ℕ P(X,0) = X

Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

P(x,y)	0	1	2	3	4	5	6	7
0								
1								
2				?				
3								

Base Case: 8x2ℕ P(X,0) = X

Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

P(x,y)	0	1	2	3	4	5	6	7
0								
1								
2			?	?				
3								

Base Case: 8x2ℕ P(X,0) = X

Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

P(x,y)	0	1	2	3	4	5	6	7
0								
1								
2		?	?	?				
3								

Base Case: 8x2ℕ P(X,0) = X

Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

P(x,y)	0	1	2	3	4	5	6	7
0								
1								
2	?	?	?	?				
3								

Base Case: 8x2ℕ P(X,0) = X

Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

P(x,y)	0	1	2	3	4	5	6	7
0								
1								
2	2	?	?	?				
3								

Base Case: 8x2ℕ P(X,0) = X

Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

P(x,y)	0	1	2	3	4	5	6	7
0								
1								
2	2	3	?	?				
3								

Base Case: 8x2ℕ P(X,0) = X

Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

P(x,y)	0	1	2	3	4	5	6	7
0								
1								
2	2	3	4	?				
3								

Base Case: 8x2ℕ P(X,0) = X

Inductive Rule:
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

P(x,y)	0	1	2	3	4	5	6	7
0								
1								
2	2	3	4	5				
3								

Procedure P(x,y): Top Down
 If y=0 return x
 Otherwise return P(x,y-1)+1;

P(x,y)	0	1	2	3	4	5	6	7
0								
1								
2	2	3	4	5				
3								

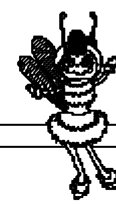
Procedure P(x,y): Recursive Programming
 If y=0 return x
 Otherwise return P(x,y-1)+1;

P(x,y)	0	1	2	3	4	5	6	7
0								
1								
2	2	3	4	5				
3								

Inductive Definition:
8x2ℕ P(X,0) = X
8x,y2ℕ, y>0, P(x,y) = P(x,y-1) + 1

Bottom-Up, Iterative Program:

```
For k = 0 to 3
    P(k,0)=k
For j = 1 to 7
    For k = 0 to 3
        P(k,j) = P(k,j-1) + 1
```

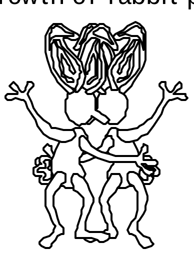


Top-Down, Recursive Program:

```
Procedure P(x,y):
    If y=0 return x
    Otherwise return P(x,y-1)+1;
```

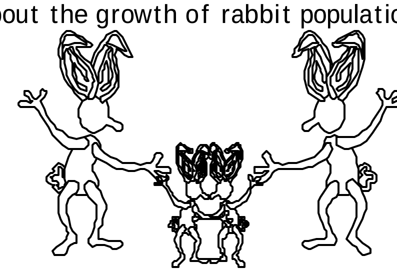
Leonardo Fibonacci

In 1202, Fibonacci proposed a problem about the growth of rabbit populations.



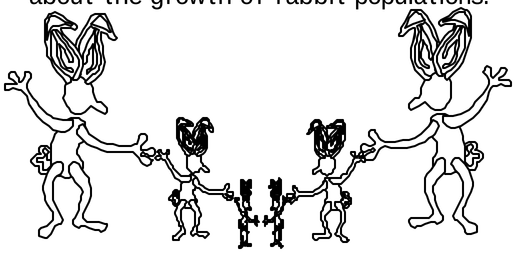
Leonardo Fibonacci

In 1202, Fibonacci proposed a problem about the growth of rabbit populations.



Leonardo Fibonacci

In 1202, Fibonacci proposed a problem about the growth of rabbit populations.



The rabbit reproduction model

- A rabbit lives forever
- The population starts as a single newborn pair
- Every month, each productive pair begets a new pair which will become productive after 2 months old

F_n= # of rabbit pairs at the beginning of the n^{th} month

month	1	2	3	4	5	6	7
rabbits							

The rabbit reproduction model

- A rabbit lives forever
- The population starts as a single newborn pair
- Every month, each productive pair begets a new pair which will become productive after 2 months old

F_n= # of rabbit pairs at the beginning of the n^{th} month

month	1	2	3	4	5	6	7
rabbits	1						

The rabbit reproduction model

- A rabbit lives forever
- The population starts as a single newborn pair
- Every month, each productive pair begets a new pair which will become productive after 2 months old

F_n= # of rabbit pairs at the beginning of the n^{th} month

month	1	2	3	4	5	6	7
rabbits	1	1					

The rabbit reproduction model

- A rabbit lives forever
- The population starts as a single newborn pair
- Every month, each productive pair begets a new pair which will become productive after 2 months old

F_n= # of rabbit pairs at the beginning of the n^{th} month

month	1	2	3	4	5	6	7
rabbits	1	1	2				

The rabbit reproduction model

- A rabbit lives forever
- The population starts as a single newborn pair
- Every month, each productive pair begets a new pair which will become productive after 2 months old

F_n= # of rabbit pairs at the beginning of the n^{th} month

month	1	2	3	4	5	6	7
rabbits	1	1	2	3			

The rabbit reproduction model

- A rabbit lives forever
- The population starts as a single newborn pair
- Every month, each productive pair begets a new pair which will become productive after 2 months old

F_n= # of rabbit pairs at the beginning of the n^{th} month

month	1	2	3	4	5	6	7
rabbits	1	1	2	3	5		

The rabbit reproduction model

- A rabbit lives forever
- The population starts as a single newborn pair
- Every month, each productive pair begets a new pair which will become productive after 2 months old

F_n= # of rabbit pairs at the beginning of the n^{th} month

month	1	2	3	4	5	6	7
rabbits	1	1	2	3	5	8	13

Inductive Definition or Recurrence Relation for the Fibonacci Numbers

Stage 0, Initial Condition, or Base Case:
Fib(1) = 1; Fib (2) = 1

Inductive Rule
For n>3, Fib(n) = Fib(n-1) + Fib(n-2)

n	0	1	2	3	4	5	6	7
Fib(n)	%	1	1	2	3	5	8	13

Inductive Definition or Recurrence Relation for the Fibonacci Numbers

Stage 0, Initial Condition, or Base Case:
Fib(0) = 0; Fib (1) = 1

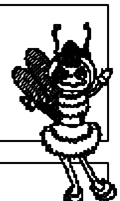
Inductive Rule
For n>1, Fib(n) = Fib(n-1) + Fib(n-2)

n	0	1	2	3	4	5	6	7
Fib(n)	0	1	1	2	3	5	8	13

Inductive Definition:
Fib(0)=0, Fib(1)=1, k>1, Fib(k)=Fib(k-1)+Fib(k-2)

Bottom-Up, Iterative Program:

```
Fib(0) = 0; Fib(1) =1;
Input x;
For k= 2 to x do Fib(x)=Fiib(x-1)+Fib(x-2);
Return Fib(k);
```



Top-Down, Recursive Program:

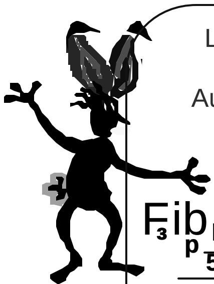
```
Procedure Fib(k)
        If k=0 return 0
        If k=1 return 1
        Otherwise return Fib(k-1)+Fib(k-2);
```

What is a closed form formula for Fib(n) ????

Stage 0, Initial Condition, or Base Case:
Fib(0) = 0; Fib (1) = 1

Inductive Rule
For n>1, Fib(n) = Fib(n-1) + Fib(n-2)

n	0	1	2	3	4	5	6	7
Fib(n)	0	1	1	2	3	5	8	13



Leonhard Euler (1765)
J. P. M. Binet (1843)
August de Moivre (1730)

$$Fib_n = \frac{\left(\frac{\sqrt{5}+1}{2}\right)^n - \left(\frac{\sqrt{5}+1}{2}\right)^{-n}}{\sqrt{5}}$$

Study Bee

Inductive Proof
- Standard Form
- All Previous Form
- Least-Counter Example Form
- Invariant Form

Inductive Definition
- Bottom-Up Programming
- Top-Down Programming
- Recurrence Relations
- Solving a Recurrence