CS375: Logic and Theory of Computing

Fuhua (Frank) Cheng

Department of Computer Science

University of Kentucky

Question 1:

Q: Why are you Here?   or,

why are you taking this course?

Ans: ???

Question 2:

Q: Who do you think is the most important scientist from the last (20-th) century?

Ans: keep your answer to yourself.

The same question will be asked at the end of the semester again.

For the first question: why are you here?

You are here to learn three things.

Three things you will learn in this class:

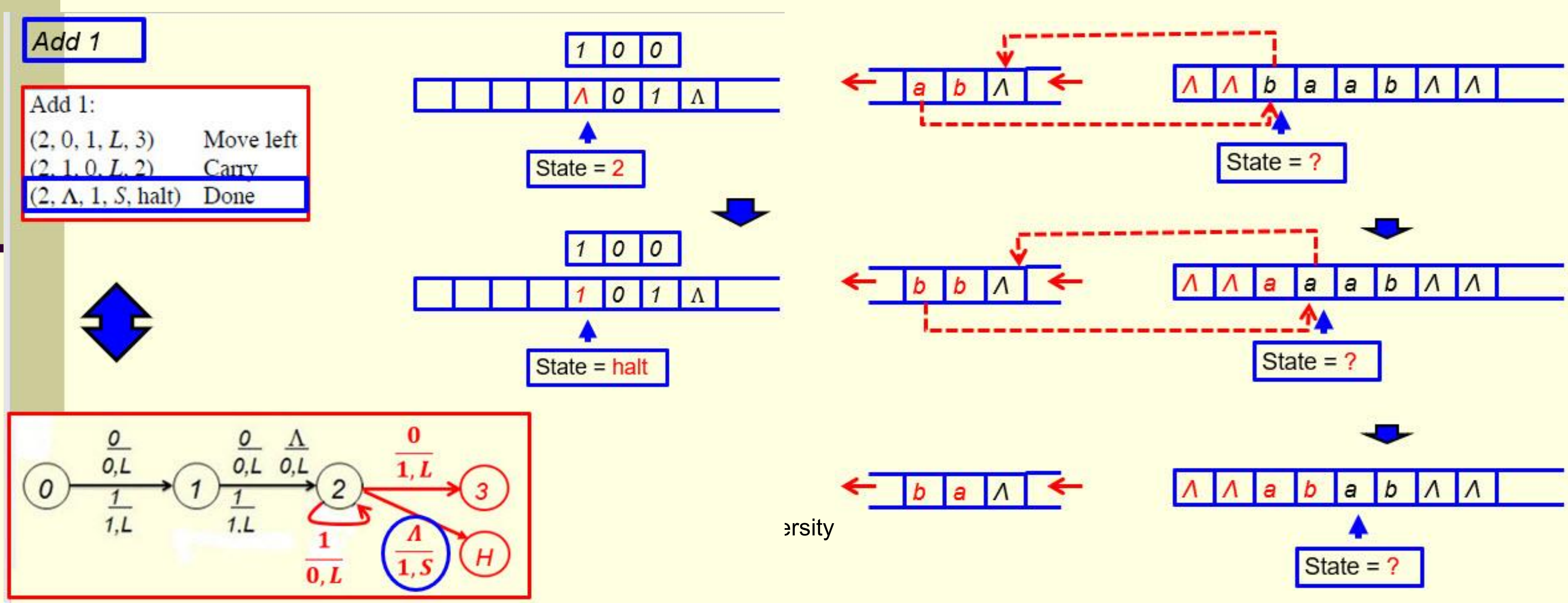
1: The foundation of modern day computers

i.e., Turing machines (1936)

- *a piece of work that changed the world/ human history*

1: The foundation of modern day computers

Such as: *how to design a machine that can perform arithmetic operations?*
or, how to design a machine that can process/parse strings?



Three things you will learn in this class:

2: The person who developed that theory

Dr. Alan Turing

- *The Nobel Prize equivalent in computer science area (Turing Award) was named after him*

Three things you will learn in this class:

3: A little history on the life of that person

Dr. Alan Turing (1912-1954)

- *including the reason why the logo of the Apple Computer company is a bitten apple*

The Arrangement:

Preliminaries

Regular Languages + Finite Automata

Context-Free Languages + Pushdown Automata

Turing machines + Church-Turing Thesis

Table of Contents:

- Week 1: Preliminaries (set algebra, relations, functions) (read Chapters 1-4)
- Weeks 2-5: Regular Languages, Finite Automata (Chapter 11)
- Weeks 6-9: Context-Free Languages, Pushdown Automata (Chapters 12)
- Weeks 10-12: Turing Machines (Chapter 13)

Table of Contents (conti):

- **Week 13 (if there is still time left after "Turing Machines"): Propositional Logic (Chapter 6), Predicate Logic (Chapter 7), Computational Logic (Chapter 9), Algebraic Structures (Chapter 10)**

1. Preliminaries – set algebra

- **Set** : **collection** of **things**

  (**order** not important; **repetition** not allowed)

- Notations: $x \in S$, $x \notin S$

  $S = \{x_1, x_2, x_3, \ldots, x_n\}$

  { }, Φ : **empty set**

  Basis element for the set of sets

  Z, N, Q, R

Notations:

A = B : two sets A and B are equal

$\{a,b,c\}=\{c,b,a\}?$

$\{a,a,b,c\}=\{a,b,c\}?$

Good way to represent an infinite set

$\{x \mid P\}$: *the set of all x that satisfies P*

e.g., the set of odd natural numbers = {1,3,5, … }

= $\{x \mid x = 2k + 1 \; for \; some \; k \in N\}$

Notations:

$A \subseteq B$: A is a subset of B

$N \subseteq Z \subseteq Q \subseteq R$

$\boxed{S \subseteq S}$ $\boxed{\Phi \subseteq S}$

Power(S) *= the set of all subsets of S*

power({a, b, c}) = ?

{Φ, {a}, {b}, {c}, {a,b}, {a,c}, {b,c}, {a,b,d} }

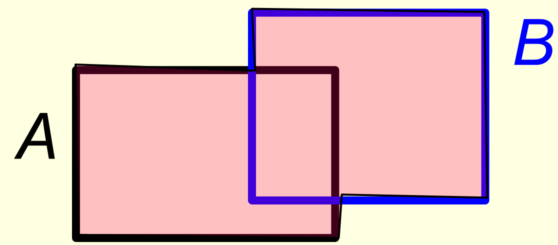
If |S| = n then |power(S)| = 2^n

Cardinality of S

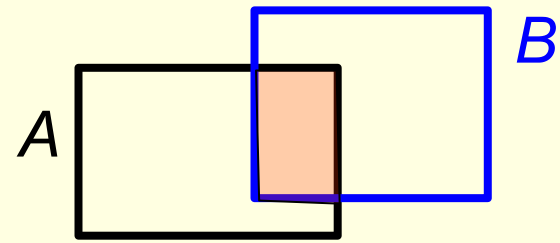
Venn diagram

Notations:

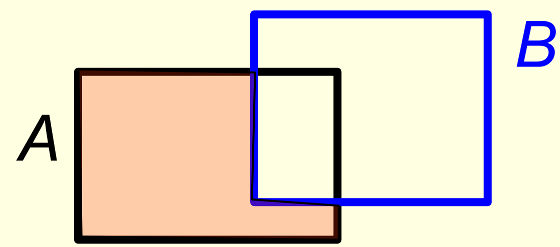
$A \cup B$: union



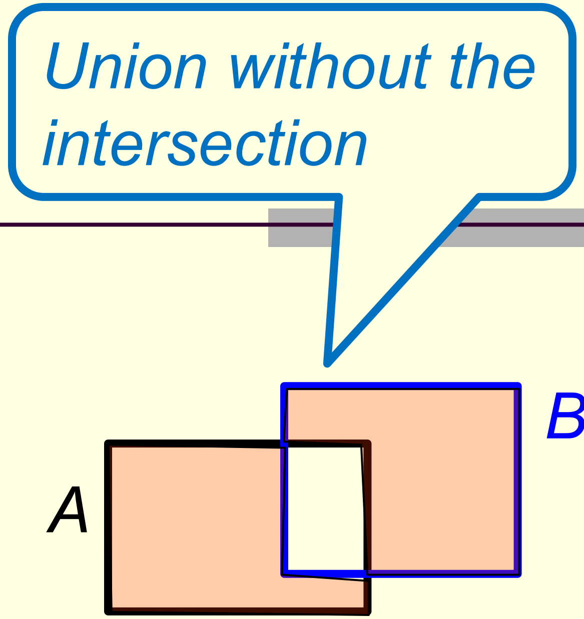
$A \cap B$: intersection



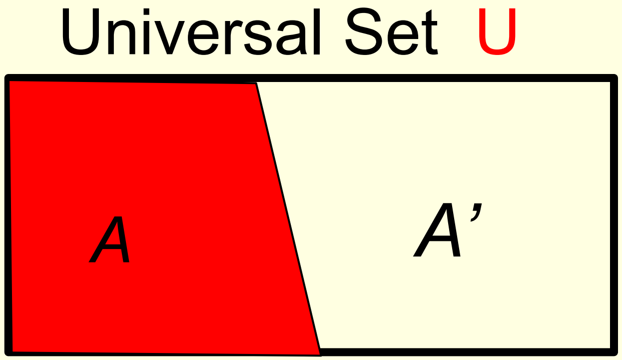
$A - B$: difference



Notations:



$A \oplus B$: *symmetric difference*

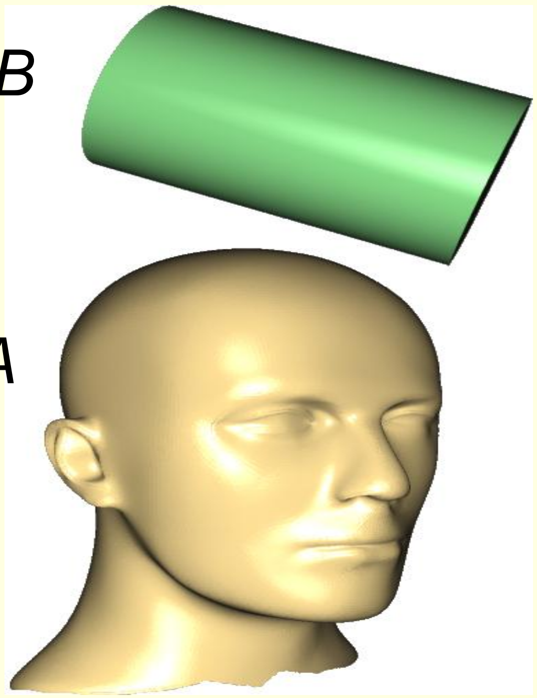


$A' = U - A$: *universal complement*

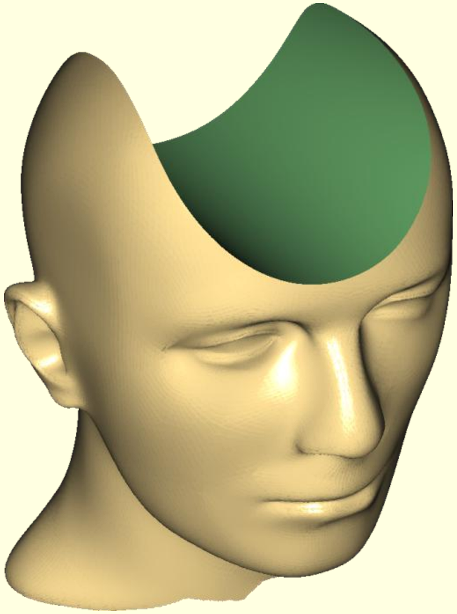
e.g.,

B

A



A - B
(depends on where B is)



Properties:

Union *and* ***intersection*** *are* ***commutative****,* ***associative****, and* ***distribute*** *over each other*

Absorption:

$$A \cup (A \cap B) = A$$

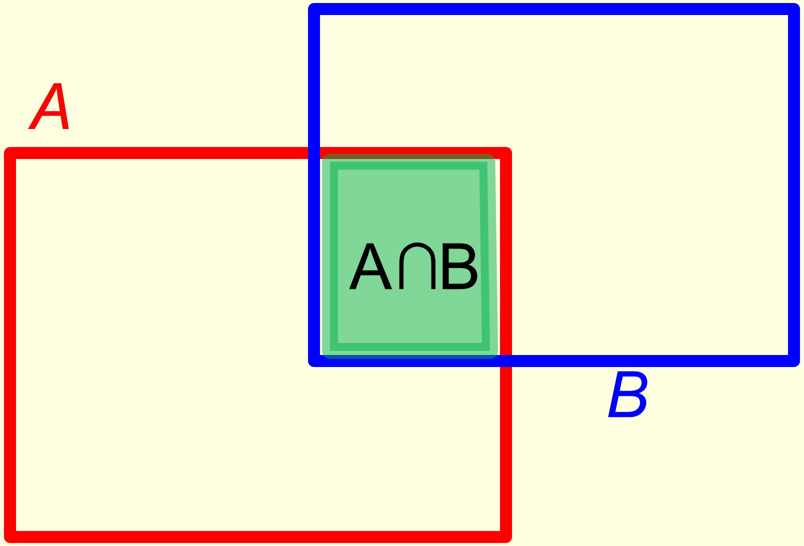
$$A \cap (A \cup B) = A$$

De Morgan's Laws:

$$(A \cup B)' = A' \cap B'$$

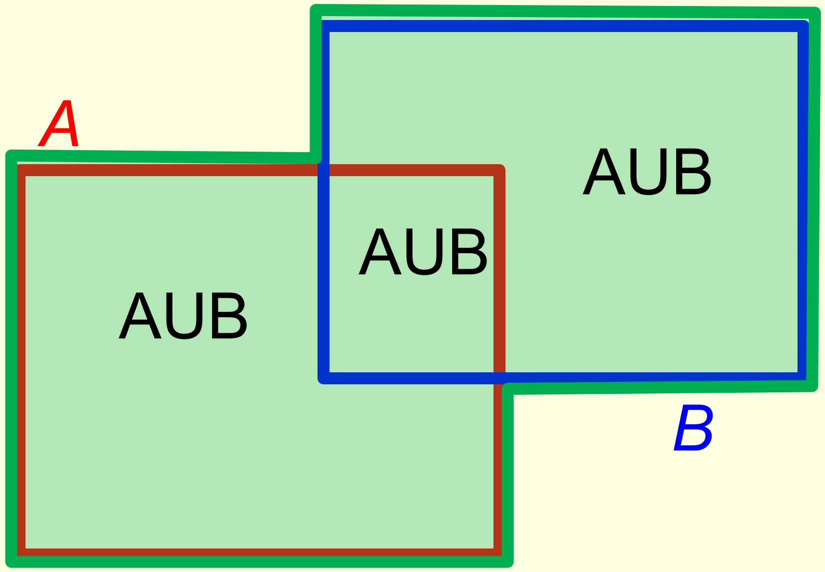
$$(A \cap B)' = A' \cup B'$$

Absorption 1: $A \cup (A \cap B) = A$



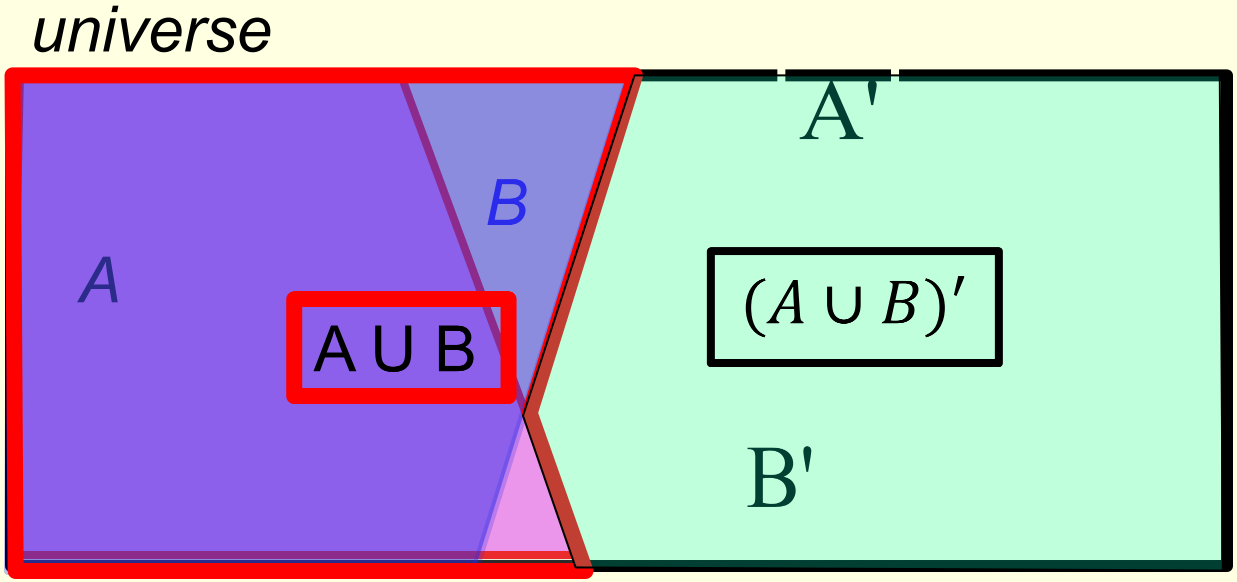
So $A \cup (A \cap B) = A$

Absorption 2 : $A \cap (A \cup B) = A$



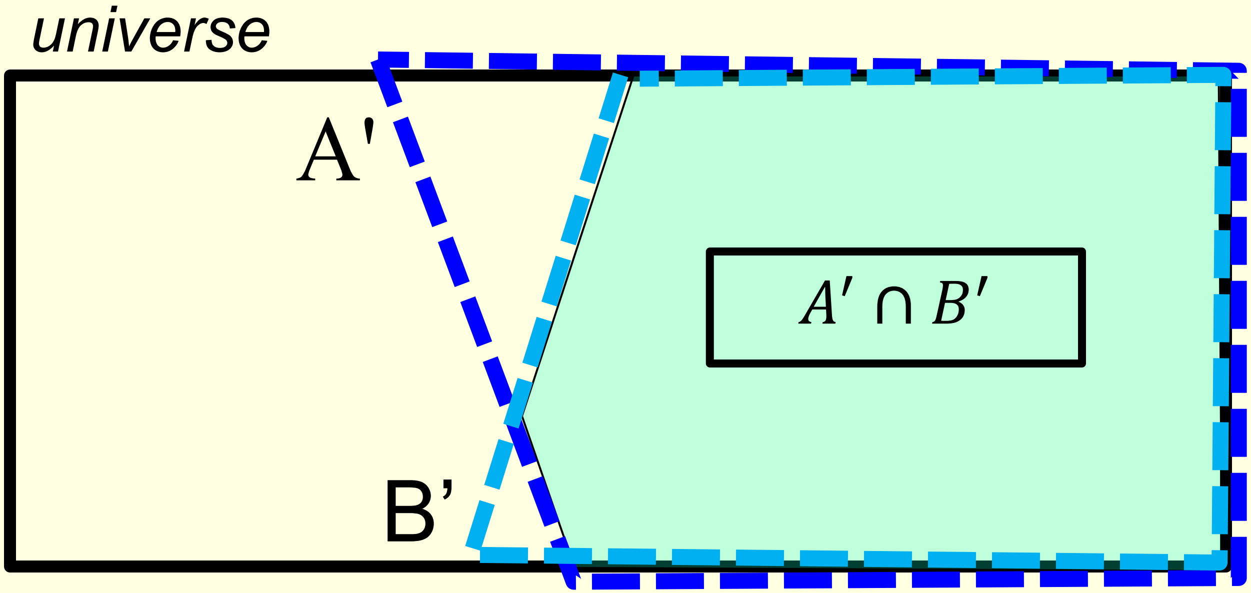
So $A \cap (A \cup B) = A$

De Morgan's Law 1: $(A \cup B)' = A' \cap B'$



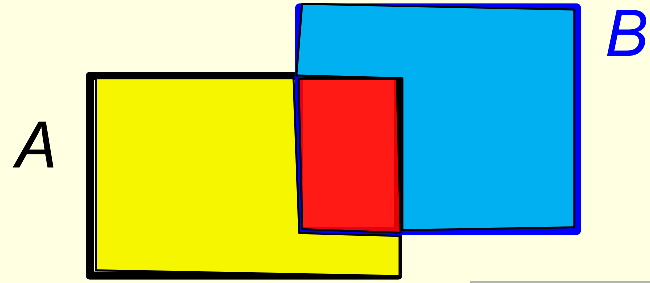
So $(A \cup B)' = A' \cap B'$

De Morgan's Law 1: $(A \cup B)' = A' \cap B'$



So $(A \cup B)' = A' \cap B'$

- Properties:



$|S|$: *cardinality of S*

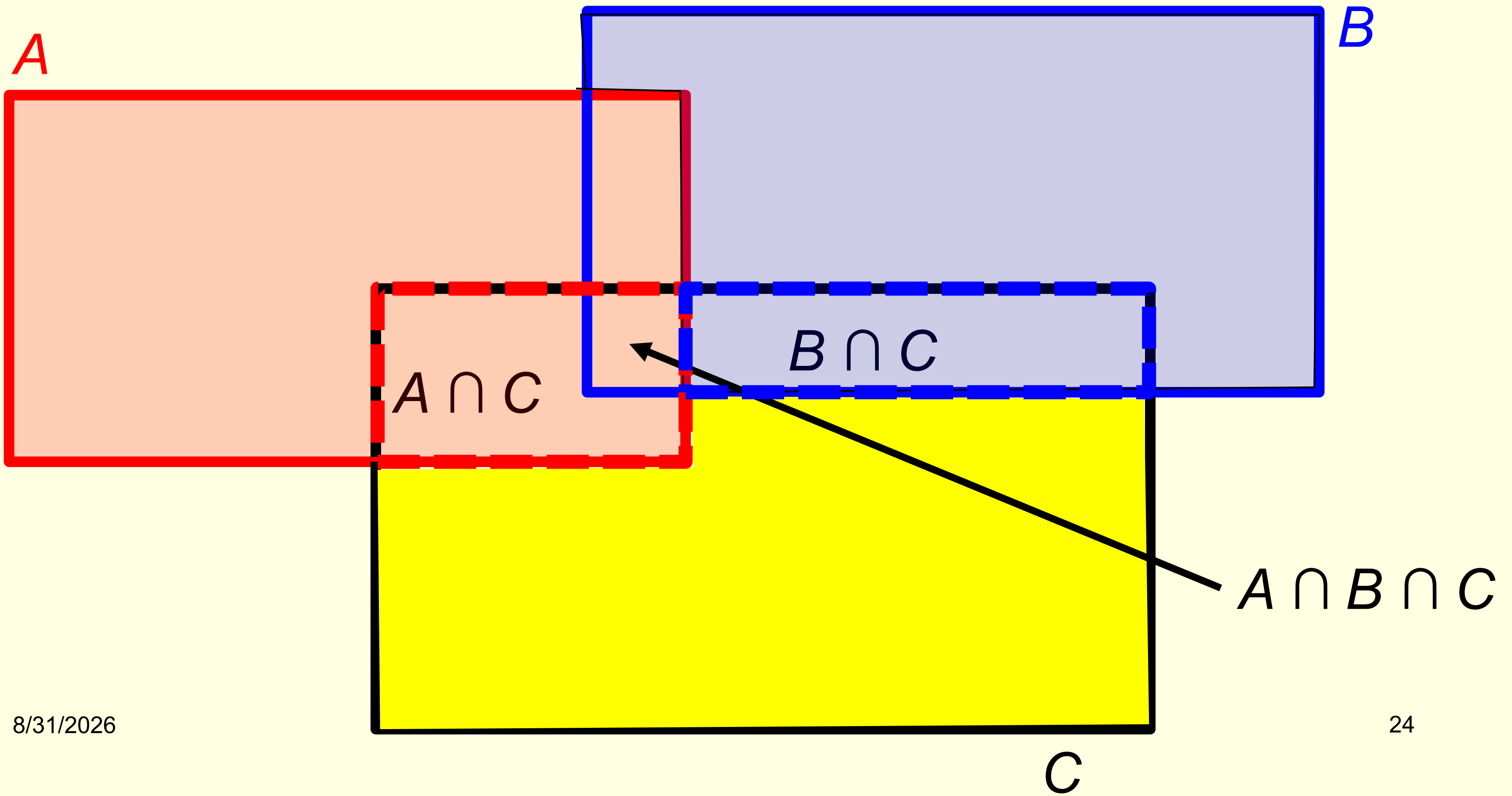
Union rule : $|A \cup B| = |A| + |B| - |A \cap B|$

Difference rule : $|A - B| = |A| - |A \cap B|$

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$

$$|A \cup B \cup C| = |A| + \underline{|B| - |A \cap B|}$$

$$+|C| - \underline{|A \cap C|} - \underline{(|B \cap C| - |A \cap B \cap C|)}$$



$$\text{Union rule}: \ |D \cup E| = |D| + |E| - |D \cap E|$$

$$|A \cup B \cup C| = |A \cup B| + |C| - |(A \cup B) \cap C|$$

$$= |A| + |B| - |A \cap B| + |C| - |(A \cap C) \cup (B \cap C)|$$

Why?

$$= |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$

$$\textit{Union rule}: \ |D \cup E| = |D| + |E| - |D \cap E|$$

Here is why:

$$|\underline{(A \cap C)} \cup \underline{(B \cap C)}|$$

$$= |A \cap C| + |B \cap C| - |\underline{(A \cap C) \cap (B \cap C)}|$$

$$= |A \cap C| + |B \cap C| - |\underline{A \cap B \cap C}|$$

- **Inductively defined sets**: used for sets with a linear order

- To define a set *S* *inductively is to do three things:*

Basis: *Specify one or more elements of S.*

Induction: *Specify one or more rules to construct elements of S from existing elements of S.*

Closure: *Specify that no other elements are in S (always assumed).*

(Basis elements and induction rules are called ***constructors***)

Example. Find an inductive definition for

$S = \{\Lambda, ac, aacc, aaaccc, \ldots\} = \{ a^n c^n \mid n \in \mathbf{N} \}$.

Solution:

Basis: $\Lambda \in S$.

Induction: If $x \in S$ then $axc \in S$.

$$\boxed{a^n c^n = a a^{n-1} c^{n-1} c}$$

Example. Find an inductive definition for

$S = \{ a^{n+1} b c^n \mid n \in \mathbf{N} \}$.

Solution:

Basis: $ab \in S$.

Induction: If $x \in S$ then $axc \in S$.

$$\boxed{a^{n+1} b c^n = a a^n b c^{n-1} c}$$

Functions

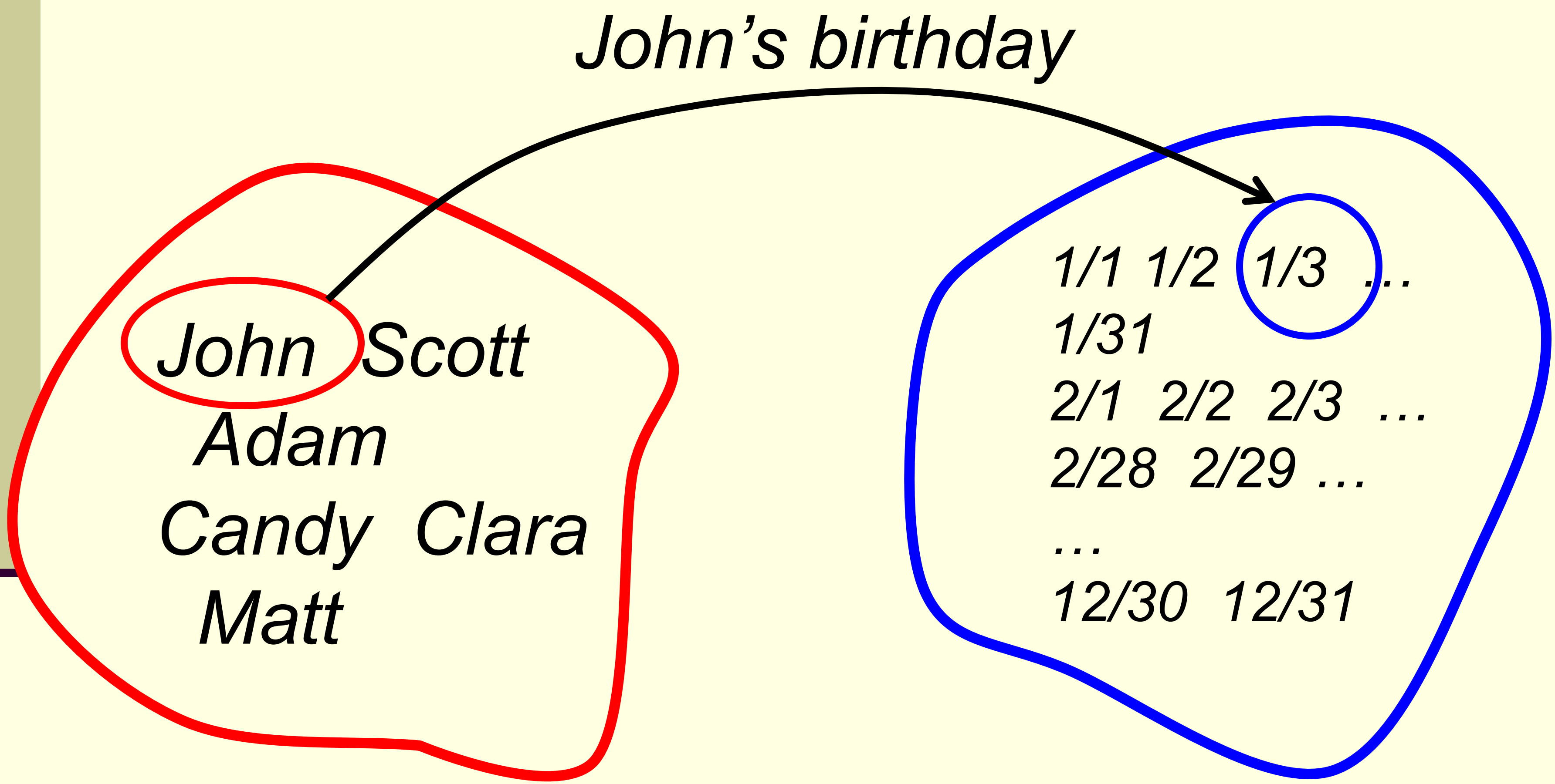
Must be precise and unique

What is a function?

a function is a way to classify/characterize things.

For instance, you can classify/characterize a group of people by their birthdays.

What is a function?



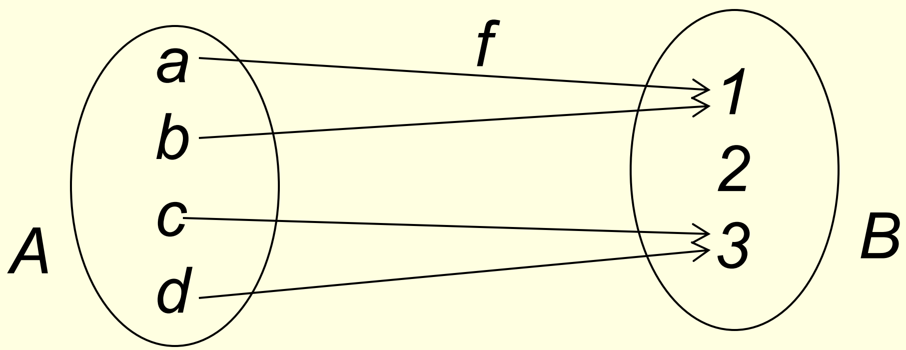
Functions

- A **function** f from A to B associates each element of A with **EXACTLY** one element of B.
- Notations: $\boxed{f : A \rightarrow B}$

domain *codomain* (*image*)

f(a)=f(b)=1

f(c)=f(d)=3



Function?

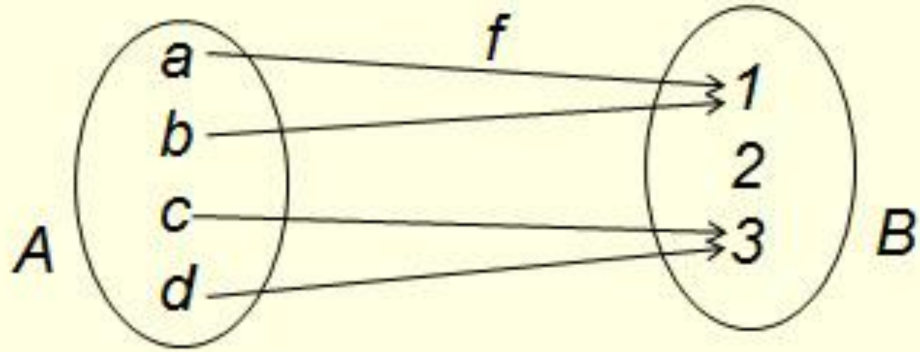
Yes

Functions

- Notations:

$f(a)=f(b)=1$

$f(c)=f(d)=3$



$range(f) = f(A) = \{f(x) \mid x \in A\} = \{1,3\}$

$f(\{a,b\}) = \{1\}$

$$f^{-1}(\{2\}) = \phi$$

$$f^{-1}(\{1,2,3\}) = \{a,b,c,d\}$$

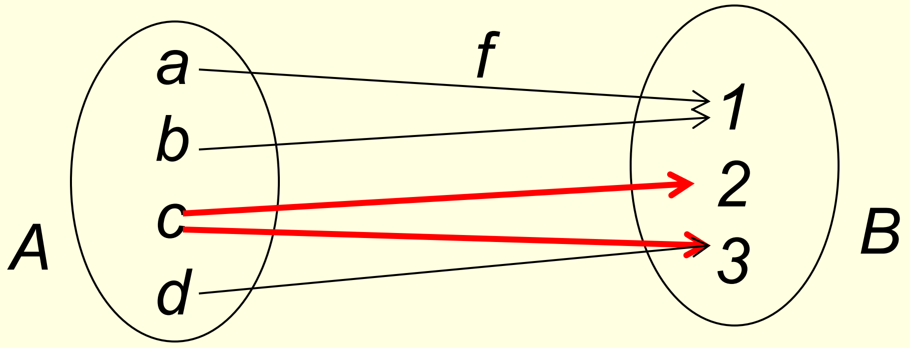
(*Inverse image of {2}*)

Functions

f(a)=f(b)=1

f(c)=f(d)=3

f(c) = 2



Function?

No

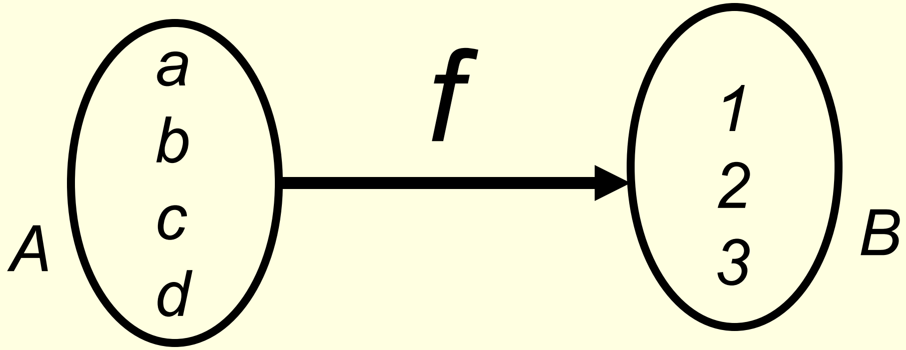
Floor and Ceiling functions:

$$\boxed{\lfloor x \rfloor = floor(x)}$$: *largest integer not greater than x*

$$\boxed{\lceil x \rceil = ceiling(x)}$$: *smallest integer not less than x*

are functions from R -> Z

Functions



How many different functions from A to B can be defined?

$$3 \times 3 \times 3 \times 3 = 3^4$$

Things you need to know about functions:

- *Composition of functions*
- *Inverse function*
- *GCD, Division algorithm, Euclid algorithm*
- *Mod functions and inverses*
- *Pigeon Hole Principle*
- *Hash functions*
- *Recursively defined functions*
- *Binary trees*

Functions

Greatest Common Divisor (gcd):

x, y : integers, not both zero

$gcd(x, y)$ = largest integer that divides x and y

Is $gcd(x, y)$ a function? **Yes**

$gcd(a,b) = gcd(b,a) = gcd(a, -b)$
$gcd(a,b) = gcd(b,a-bq)$ *for some integer q*
$gcd(a,b) = ma+nb$ *for some integers m and n*
If $d|ab$ *and* $gcd(d,a) = 1$, *then* $d|b$

$q = a / b$ $r = a \% b$

Functions

Division algorithm

a, b : integers, $b \mathrel{!}= 0$

there exist unique integers *q* and *r* such that

$a = bq + r$, $0 <= r < |b|$

Euclid's algorithm (for finding *gcd(a,b)*)

a, b: natural integers, not both zero

while *(b > 0)* do {

find *q, r* so that $a = bq+r$ and $0 <= r < b$;

a := b; b := r ;

}

Output(*a*);

Functions

Examples: find $gcd(189, 33)$

$a = q * b + r$

$189 = \boxed{5}\ 33 + \boxed{24}$

$33 = \boxed{1}\ 24 + \boxed{9}$

$24 = \boxed{2}\ 9 + \boxed{6}$

$9 = \boxed{1}\ 6 + \boxed{3}$

$9 = \boxed{1}\ 6 + \boxed{3}$

$6 = \boxed{2}\ 3 + \boxed{0}$

$3 = \boxed{}\ 0 + \boxed{}$

Since $b=0$, so output $a=3$

Functions

Any time you need to wrap around an index table, to handle cyclic data structures, …

mod function

a, *b* : integers with *b > 0*

a* mod *b* = *r if *a* = *bq* + *r* with *0 <= r < b*

Division algorithm

How to compute q and r?

Solve the equation for *q = a/b – r/b*

Since q is an integer and 0 <= r/b < 1

it follows that $\boxed{q = \lfloor a/b \rfloor}$

So we have $\boxed{r = a - bq = a - b \cdot \lfloor a/b \rfloor}$

Functions

Quotient of *a* divided by *b*

mod function

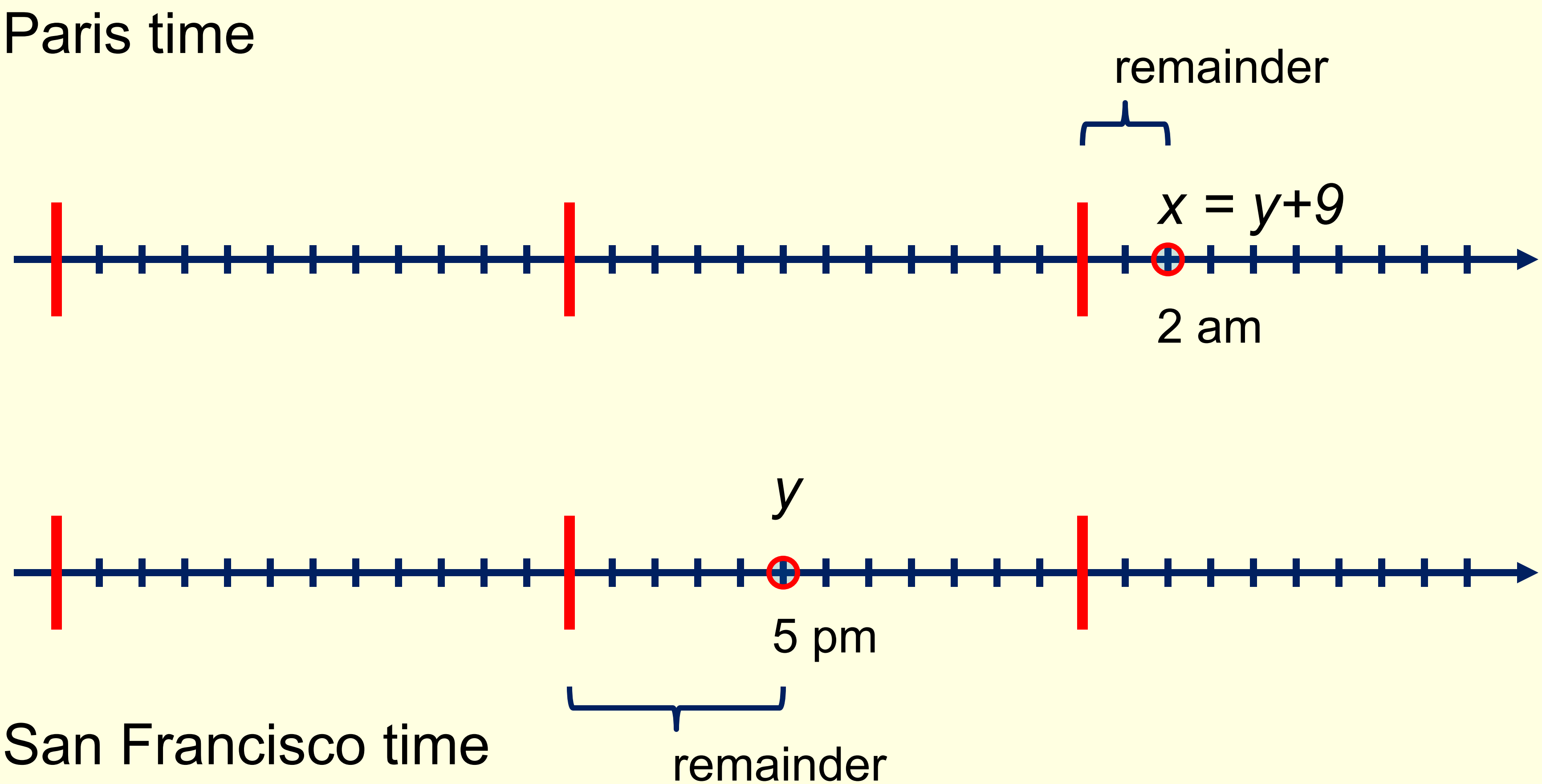
a, *b* : integers with *b>0*

$a \bmod b = a - b \cdot \lfloor a/b \rfloor$.

Example: It is *2am* in Paris. What time is it in San Francisco (9 hours difference)?

(12 hr clock): *(2-9) mod 12 = (-7) mod 12*

$= -7 - 12\lfloor -7/12 \rfloor = -7 - 12(-1) = 5$ pm


Paris time
remainder
x = y+9
2 am
y
5 pm
San Francisco time
remainder

Or, the following example

Example:

It takes me 53 hours non-stop to clean a building. If I start the cleaning job at 3pm this afternoon, what time would it be when I finish the job?

53 mod 24 = 5

3+5 = 8

8pm.

Constructing Functions

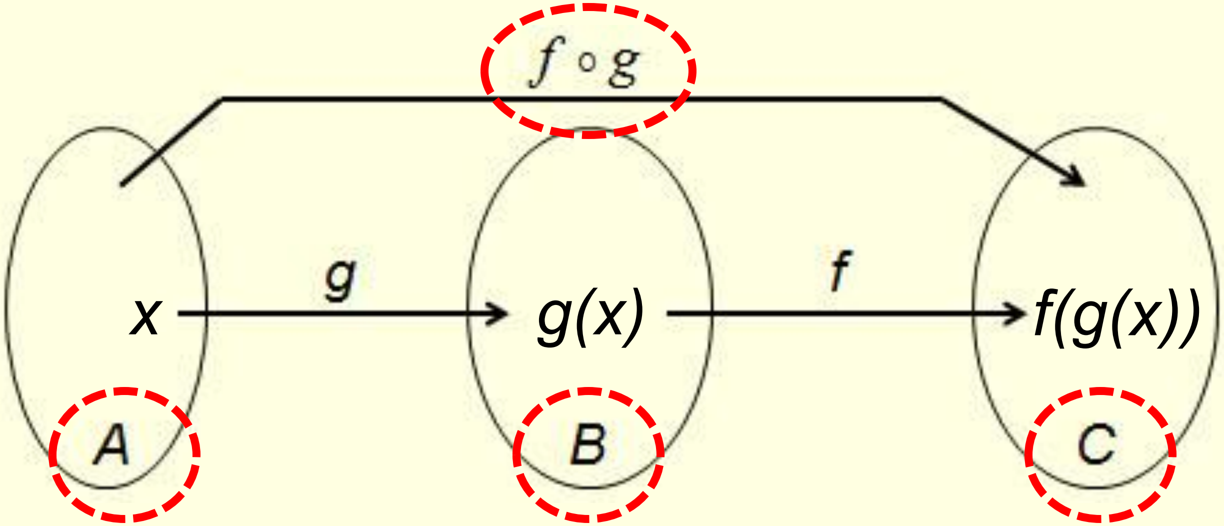
Composition:

g : A -> B and *f : B -> C*

Composition of f and g :

$$\boxed{(f \circ g): A \to C}$$

$$\boxed{(f \circ g)(x) = f(g(x))}$$



Constructing Functions

Composition:

g : A -> B and *f : B -> C*
Composition of *f* and *g* : $(f \circ g): A \to C$

$$(f \circ g)(x) = f(g(x))$$

Examples:

floor ($\log_2$ 20) = floor (4.xx) = 4

ceiling ($\log_2$ 20) = ceiling (4.xx) = 5

Properties of Functions

Given: $f : A \to B$

***Injective* (*one-to-one*) :** $x \neq y \Rightarrow f(x) \neq f(y)$

***Surjective* (*onto*):** $\forall b \in B \quad \exists a \in A \quad \text{such that} \quad b = f(a)$

***Bijective* (*one-to-one* & *onto*):** *injective + surjective*

Inverse: *If f is a bijection, then the inverse of f,* f^{-1} *,*
exists and is defined by $f^{-1}(b) = a$ *iff* $f(a)=b$

Properties of Functions

Injective (one-to-one) : $x \neq y \Rightarrow f(x) \neq f(y)$

Example: $f : N_8 \to N$ where $N_n = \{0,1,\ldots,n-1\}$

$f(x) = 2x \mod 8$

Is f injective? (f(0) =? f(4) = ?)

NO

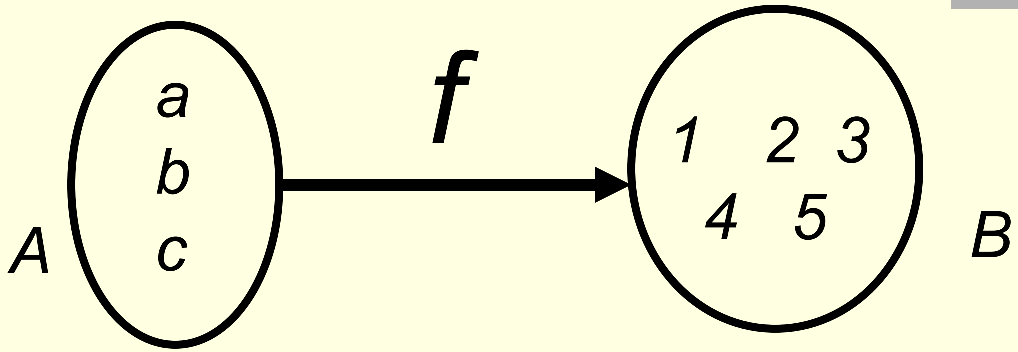
Example: f : Z -> N

$f(x) = x^2$

Is f injective? (f(2) = ? f(-2) = ?)

NO

One-to-one Functions



How many different one-to-one functions from A to B can be defined?

$$5 \times 4 \times 3 = \frac{5!}{(5-3)!} = \frac{5!}{2!}$$

Properties of Functions

Surjective (onto): $\forall b \in B \quad \exists a \in A \quad such\ that \quad b = f(a)$

Example: $f : Z \to N$

$$f(x) = x^2$$

Is f surjective?

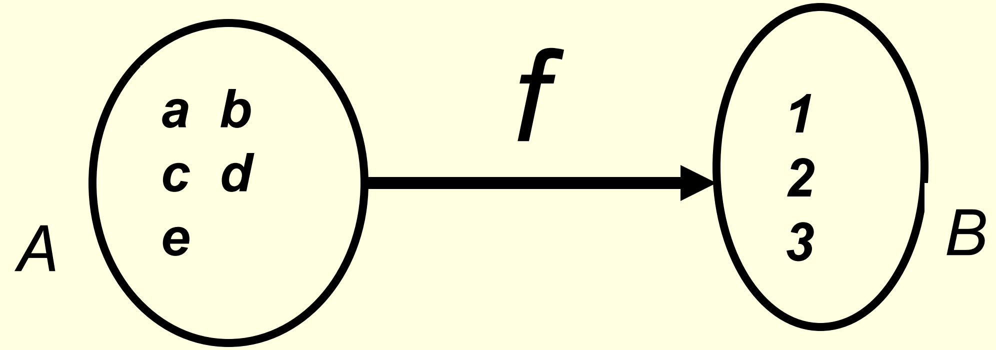
NO

Example: $f : Z \to N$

$$f(x) = |\,x\,|$$

f surjective but not injective

Onto Functions

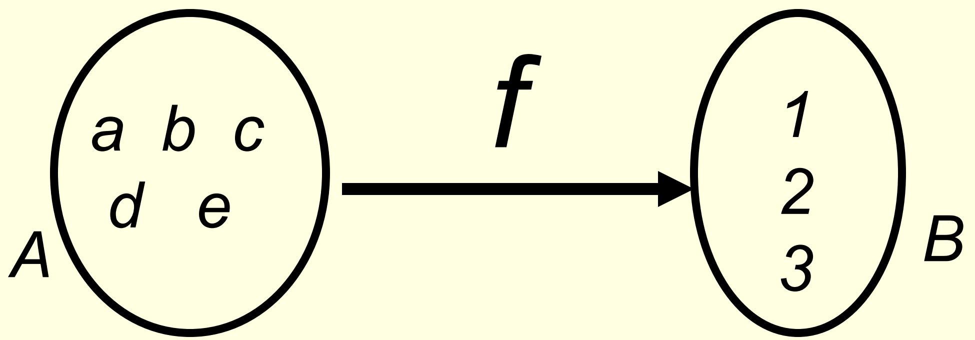


How many different onto functions from A to B can be defined?

$$3^5 - F_1 - F_2$$

where F_i $(i = 1, 2)$ is the number of different functions from A to B with each image set having i elements only.

Onto Functions



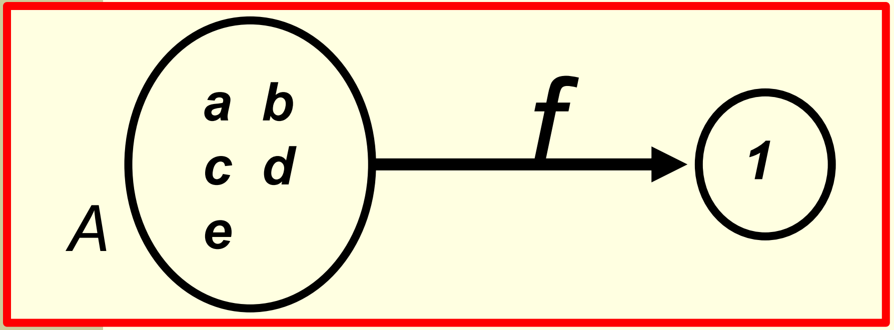
$$F_1 = 3; \qquad F_2 = (2^5 - 2)\binom{3}{2}$$

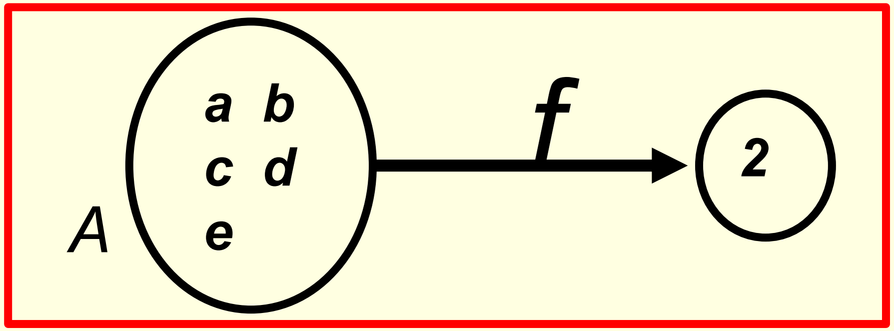
Hence, the number of different onto functions from A to B = $3^5 - 3 - (2^5 - 2)\binom{3}{2}$

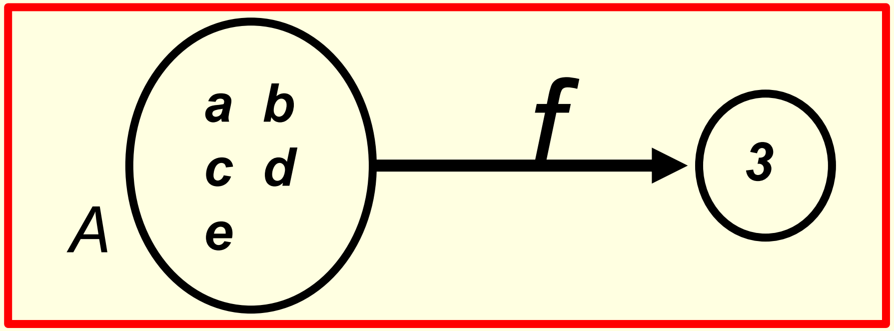
Onto Functions

$F_1 = 3$ *Why?*

(only ***three*** *functions whose image set has one element only)*



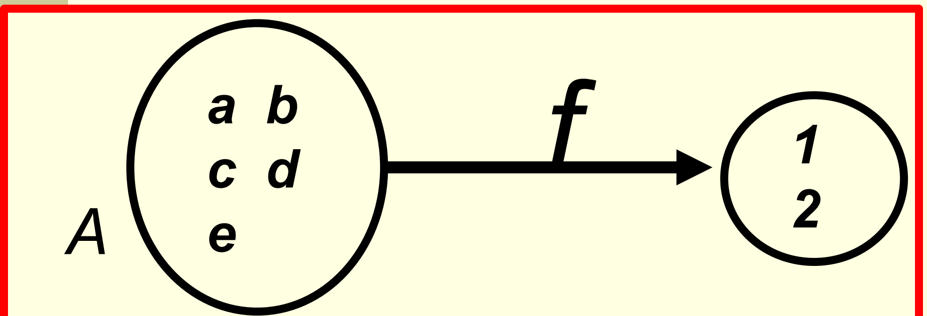




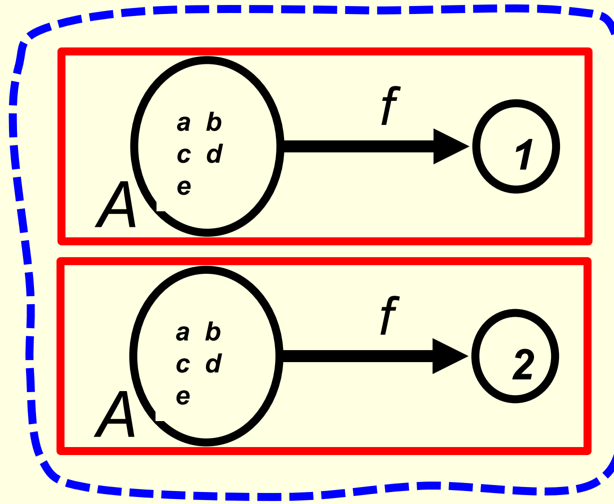
Onto Functions

$F_2 = (2^5 - 2)\binom{3}{2}$ *Why?*

For a 2-element subset of B, the number of onto functions from A to that subset is: $2^5 - 2$



excluding



And B has $\binom{3}{2}$ *2-element subsets.*

So $F_2 = (2^5 - 2)\binom{3}{2}$

Properties of Functions

Bijective (one-to-one & onto):

Example: $f : (0, 1) \rightarrow (2, 5)$ *defined by* $f(x) = 3x + 2$ *is a bijection*

Proof:

Example: $f : \mathbf{R} \rightarrow \mathbf{R}$ *defined by* $f(x) = x^3$

Is f bijective ? ***Yes***

Functions

$$\boxed{f : A \to B}$$

$|A| = m$ $|B| = n$

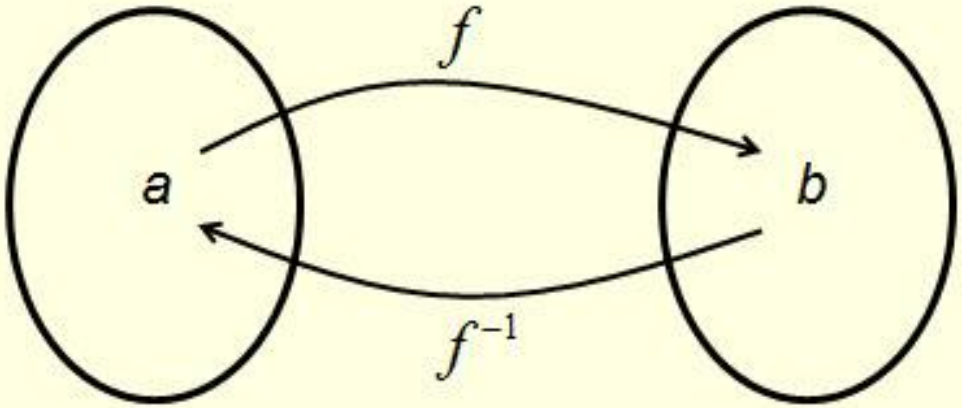
(1) How many different functions can be defined from A to B?

(2) If $m \leq n$ then how many different one-to-one functions can be defined from A to B?

(3) If $m \geq n$ then how many different onto functions can be defined from A to B?

Properties of Functions

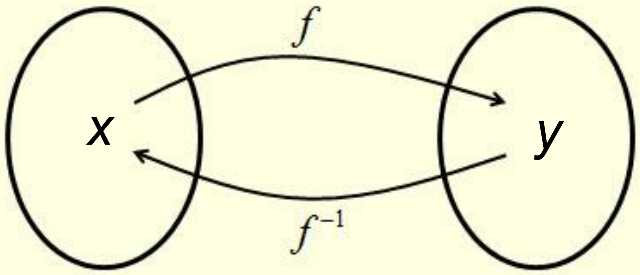
Inverse: If f is a bijection, then the inverse of f, f^{-1}, exists and is defined by $f^{-1}(b) = a$ iff $f(a) = b$



Hence, $f^{-1}(f(a)) = ?$ $\quad f(f^{-1}(b)) = ?$

$$\boxed{f^{-1}(f(a)) = a \qquad f(f^{-1}(b)) = b}$$

Properties of Functions



Example: $f : (0, 1) \to (2, 5)$ defined by $f(x) = 3x + 2$ is a bijection

$$f^{-1}(y) = ?$$

Note that $f(f^{-1}(y)) = y$

On the other hand, if we think of $f^{-1}(y)$ as an x, then by definition, we have

$$f(\boxed{f^{-1}(y)}) = 3\boxed{f^{-1}(y)} + 2 = y$$

So, $f^{-1}(y) = (y-2)/3$

$N_5 = \{0, 1, 2, 3, 4\}$

$f(N_5) = \{1, 0, 4, 3, 2\}$

Properties of Functions

Example: $f : N_5 \to N_5$ defined by f(x) = (4x+1) mod 5 is a bijection (gcd(4,5)=1)

$$f^{-1} = ?$$

Theorem (mod and inverses)

Let n > 1 and $f : N_n \to N_n$ be defined by f(x) = ($\boxed{a}$x+b) mod $\boxed{n}$. Then

- f is bijective iff gcd(a, n) =1
- If so, then $f^{-1}(x) = (\boxed{k}x + \boxed{c}) \mod n$ where $\boxed{f(c)=0}$ and $\boxed{1=ak+nm}$

Why should we take '*c*' and '*k*' such that $f(c) = 0$ and $1 = ak + nm$?

To ensure that $f\left(f^{-1}(y)\right) = y$

$$f\left(f^{-1}(y)\right) = f(ky + c) = a(ky + c) + b$$

$$= aky + ac + b$$

$$= aky + qn \quad for\ some\ q\ \epsilon Z$$

$$= (1 - nm)y + qn$$

$$= y - nmy + qn = y \bmod n$$

Properties of Functions

Example: $f : N_5 \rightarrow N_5$ defined by f(x) = (4 x +1) mod 5
is a bijection

$f^{-1} = ?$

Since gcd(4, 5) = 1, the theorem says that *f* is a bijection.

First, find a 'c' such that *f* (c) = 0. e.g., *f* (1) = 0.

Then use Euclid's algorithm to verify that 1 = gcd(4, 5) and work backwards through the equations to find that

1 = 4(–1) + 5(1). So k = –1.

Thus $f^{-1}(x)$ = (–x + 1) mod 5. = (4x + 1) mod 5

Properties of Functions

Find $gcd(5, 4)$

Find $1 = 4 \cdot (-1) + 5 \cdot 1$

$a = q * b + r$

$5 = \boxed{1}\ 4 + \boxed{1}$

$4 = \boxed{4}\ 1 + \boxed{0}$

$1 = \boxed{}\ 0 + \boxed{}$

Since $b=0$, so output $a=1$

$1 = 5 - 1 \cdot 4$

$1 = 5 \cdot (1) + 4 \cdot (-1)$

Properties of Functions

Find *gcd(23, 4)*

Find $1 = 4 \cdot (6) + 23 \cdot (-1)$

$$23 = \boxed{5}\ 4 + \boxed{3}$$

$$4 = \boxed{1}\ 3 + \boxed{1}$$

$$3 = \boxed{3}\ 1 + \boxed{0}$$

$$1 = \boxed{}\ 0 + \boxed{}$$

$$1 = 4 - 1 \cdot 3$$

$$1 = 4 + (23 - 5 \cdot 4) \cdot (-1)$$

$$1 = 4 \cdot 6 + 23 \cdot (-1)$$

Since b=0, so output a=1

Number of elements in the domain

Functions

Number of elements in the codomain

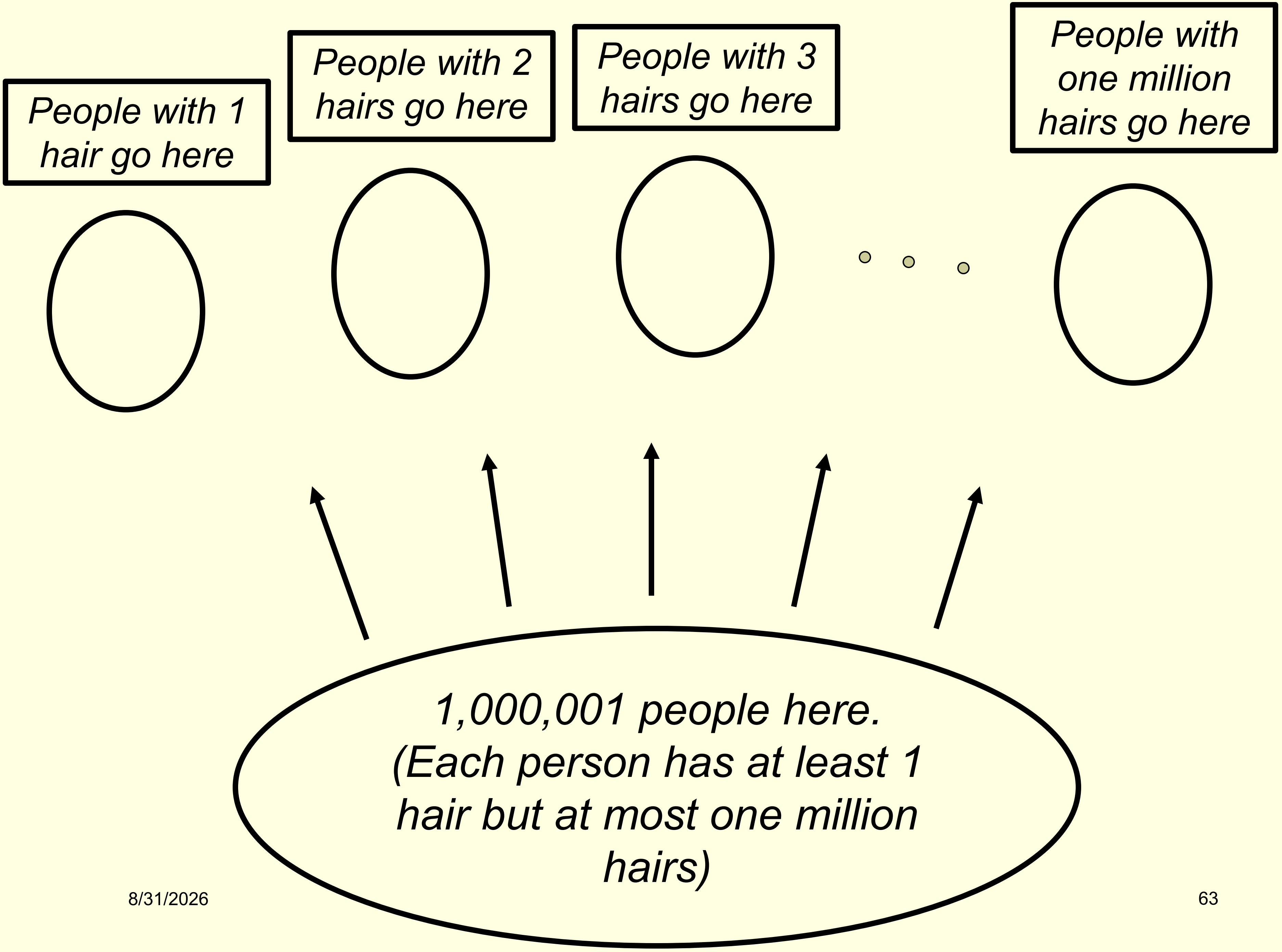
- **P on Hole Principle** If m pig s are put into n ho and m > n, then one hole has two ore pigeons. (or A and B are finite sets with | A | > |, then there are no ections from A to B)

Examp . In Mexico City there are two pe le with the same number f hairs on their heads (assumpti : everyone has less tha 10 million hairs on their head an the population of Mexico City is more than 10 million).

Number of pigeons=? Number of holes = ?

Population of Mexico City

Number of people with different number of hairs


People with 1 hair go here
People with 2 hairs go here
People with 3 hairs go here
People with one million hairs go here
1,000,001 people here. (Each person has at least 1 hair but at most one million hairs)

Properties of Functions

Example. *How many people are needed in a group to say that three were born on the same day of the week?*

Solution:

would 14 people work?

M	*T*	*W*	*R*	*F*	*ST*	*SU*
2	2	2	2	2	2	2

would 15 people work?

Properties of Functions: ***hashing***

Hash Functions use keys to look up information in a table, but without searching, so we can cut operation time from O(n) to O(1).

Items are stored into a table (called a hash table) based on the value of a search key (e.g., birthday).

If collisions occur, then a second key (e.g., last name) is used.

The idea is to use the keys to jump directly to the entry where the information is stored without any searching.

Properties of Functions: hashing

How to find T. Smith's information?

Get T. Smith's birthday : March 1

Compute the index: 31+28+1 = 60

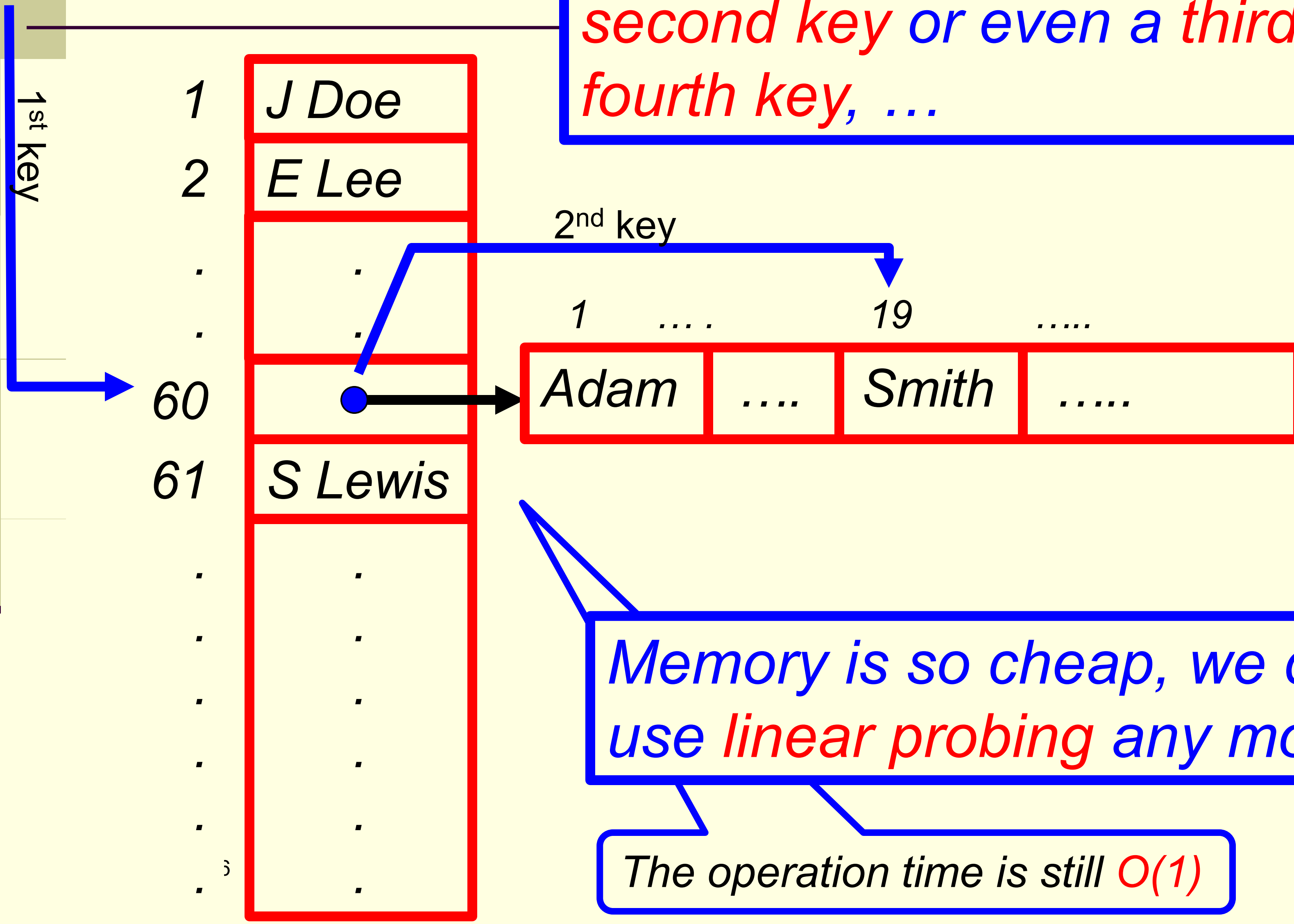
So go to entry 60 to get T. Smith's information

The operation time is O(1)

1	J Doe
2	E Lee
.	.
.	.
60	T Smith
61	S Lewis
.	.
.	.
.	.
.	.
.	.
.	.

Properties of Functions: hashing

To avoid collision problem, use a second key or even a third key, a fourth key, …



Memory is so cheap, we don't use linear probing any more.

The operation time is still O(1)

String Algebra

Given an alphabet A={a, b, c, d, e}

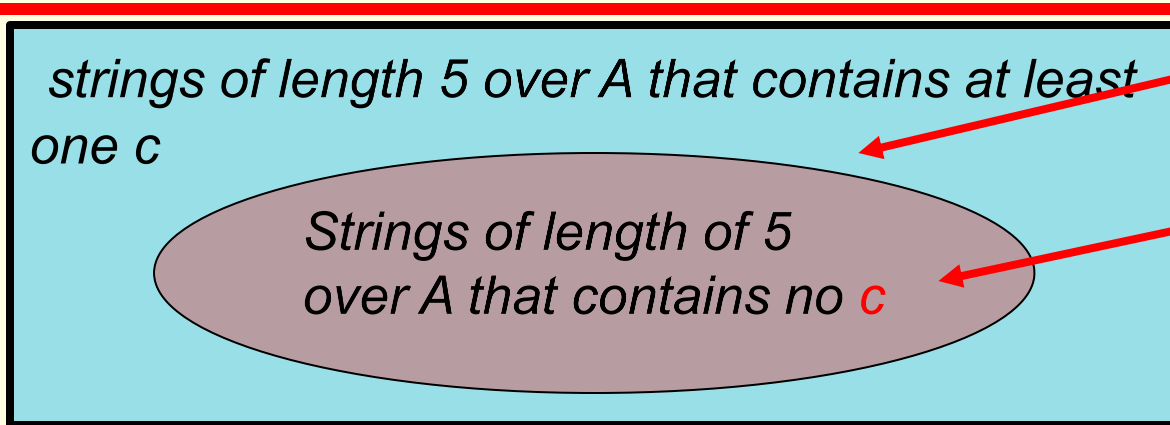
1. what is the number of strings of length 5 over A that ends in *a* or *b*?

5	*5*	*5*	*5*	*a/b*

$(5^4 * 2)$

2. what is the number of strings of length 5 over A that contains at least one c?

Strings of length 5 over A



$(5^5 - 4^5)$

4^5

Recursiveness does not save computation time, it only makes your code more compact

More on Functions

Recursively Defined Functions

Function f is recursively defined : at least one $f(x)$ is defined in terms of another $f(y)$, where $x \neq y$.

```
sum = 0;
for (i=1; i<=1000; i++)   sum = sum + i;
```

f(0) = 0;
f(n) = f(n-1) + rule(s) on n

Technique (when argument domain is inductively defined)

1. Specify a value $f(x)$ for each basis element x of S.

2. Specify rules that,

for each inductively defined element x in S,

define $f(x)$ in terms of previously defined values of f.

More on Functions

One way to write the code:

```
sum = 0;
sum = sum + 1;
sum = sum + 2;
sum = sum + 3;
sum = sum + 4;
sum = sum + 5;
      ⋮
sum = sum + 1,000,000;
```

Computation time is the same

A better way to write the code:

```
sum = 0;
for (i=1; i<=1000; i++)   sum = sum + i;
```

Binary Relations

Relation is a way to partition a set

A ***binary relation*** *R over a set A is a subset of A × A.*
If (x, y) ∈ R we also write xRy.

Example. *Binary relations over A = {0, 1} :*

∅, ***A × A***, ***eq*** *= {(0, 0), (1, 1)},* ***less*** *= {(0, 1)}.*

Definitions: Let *R be a binary relation over a set A.*

- *R is* ***reflexive*** *: xRx for all x ∈ A.*
- *R is* ***symmetric*** *: xRy ➡ yRx for all x, y ∈ A.*
- *R is* ***transitive*** *: xRy, yRz ➡ xRz for all x, y, z ∈ A.*

Binary Relations

Composition: If *R and S are binary relations, then composition of R and S is*

$R \circ S = \{(x, z) \mid xRy \text{ and } ySz \text{ for some } y\}.$

$(x, y) \circ (y, z) = (x, z)$

Example (digraph representations). *Let R = {(a, b), (b, a), (b, c)} over A = {a, b, c}. Then R,* $R^2 = R \circ R$, *and* $R^3 = R^2 \circ R$ *can be represented by directed graphs:*

$$R = \{ (a, b),\ (b, a),\ (b, c) \}$$

$$R^2 = \{ (a,\ a),\ (b,\ b),\ (a,\ c) \}$$

$$R^3 = \{ (a, b), (b, a), (b, c) \}$$

Equivalence Relations

Provides a way to partition the domain set

A binary relation is an ***equivalence relation*** *if it has the three properties:* **reflexive**, **symmetric**, and **transitive** (RST).

Examples. ***a.*** Equality on any set.

b. x ~ y iff | x | = | y | over the set of strings {a, b, c}*.

c. x ~ y iff x and y have the same birthday over the set of people.

Quiz. *Which of the relations are RST?*

a. xRy iff $x \le y$ or $x > y$ over Z.

b. xRy iff $| x - y | \le 2$ over Z. *(Not transitive)*

c. xRy iff x and y are both even over Z. *(Not reflexive)*

Answers. Yes, No, No.

Equivalence Relations

Equivalence Classes: If R is RST over A, then for each $a \in A$ the *equivalence class* of a, denoted [a], is the set [a] = {x | xRa}.

Property: For every pair $a, b \in A$ we have either [a] = [b] or [a] ∩ [b] = ∅.

Example. Suppose x ~ y iff x mod 3 = y mod 3 over **N.** Then the equivalence classes are,

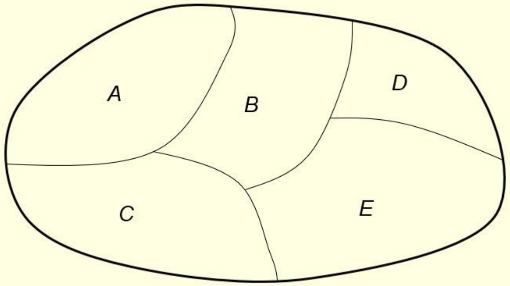
$[1] = \{1, 4, 7, \ldots\} = \{3k - 2 \mid k \in \mathbf{N}\}$

$[2] = \{2, 5, 8, \ldots\} = \{3k - 1 \mid k \in \mathbf{N}\}$

$[3] = \{3, 6, 9, \ldots\} = \{3k \mid k \in \mathbf{N}\}$

where $\mathbf{N} \equiv \{1, 2, 3, 4, \ldots\}$

Equivalence Relations



A **Partition** of a set is a collection of nonempty disjoint subsets whose union is the set.

Example. From the previous example, the sets [1], [2], [3] form a partition of ***N***.

Theorem (RSTs and Partitions). Let *A* be a set. Then the following statements are true.

1. Equivalence classes of any RST over *A* form a partition of *A*.
2. Any partition of *A* yields an RST over *A*, where the sets of the partition act as the equivalence classes.

Equivalence Relations

Example. *Let x ~ y iff x mod 2 = y mod 2 over* ***Z.*** Then ~ is an RST with equivalence classes *[0]*, the evens, and *[1]*, the odds. Also *{[0], [1]}* is a partition of **Z.**

Example*.* ***R*** can be partitioned into the set of half-open intervals *{(n, n + 1] | n ∈ Z}.* Then we have an RST ~ over ***R*,** where *x ~ y* iff *x, y ∈ (n, n + 1]* for some *n ∈ Z*.

Refinements of Partitions. If *P* and *Q* are partitions of a set *S*, then *P* is a refinement of *Q* if every *A ∈ P* is a subset of some *B ∈ Q*.

Equivalence Relations

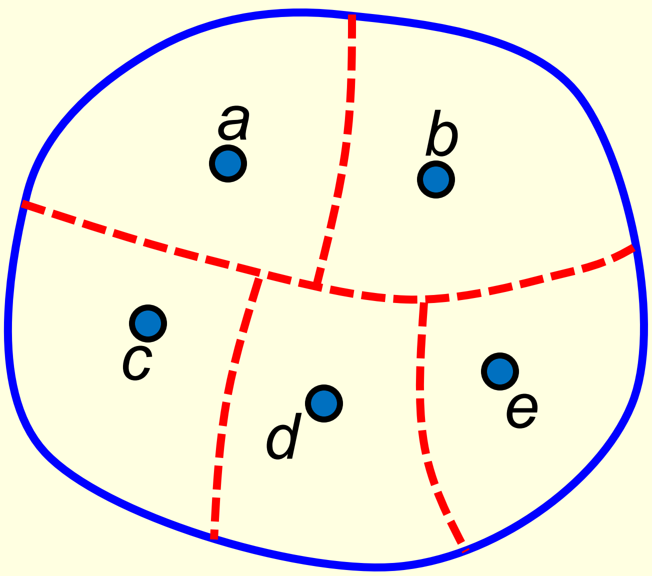
Example. Let $S = \{a, b, c, d, e\}$ and consider the following four partitions of S.

$P1 = \{\{a, b, c, d, e\}\},$

$P2 = \{\{a, b\}, \{c, d, e\}\},$

$P3 = \{\{a\}, \{b\}, \{c\}, \{d, e\}\},$

$P4 = \{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}\}.$



Each *Pi is a refinement of Pi–1.*

P1 is the "coarsest" and P4 is the "finest".

End of Preliminaries