

# A Math Review

- Mathematical notations
- Mathematical proofs
- Functions and predicates
- Graphs

# Sets

- order doesn't matter

# Sets

- order doesn't matter
- finite:  $\{2, 4, 6\}$

# Sets

- order doesn't matter
- finite:  $\{2, 4, 6\}$
- infinite:  $\{2, 4, 6, \dots\}$   $\{x \mid x \text{ is even}\}$

# Sets

- order doesn't matter
- finite:  $\{2, 4, 6\}$
- infinite:  $\{2, 4, 6, \dots\}$   $\{x \mid x \text{ is even}\}$
- subset:  $U \subset V, U \subseteq V$

# Sets

- order doesn't matter
- finite:  $\{2, 4, 6\}$
- infinite:  $\{2, 4, 6, \dots\}$   $\{x \mid x \text{ is even}\}$
- subset:  $U \subset V, U \subseteq V$
- union:  $U \cup V$

# Sets

- order doesn't matter
- finite:  $\{2, 4, 6\}$
- infinite:  $\{2, 4, 6, \dots\}$   $\{x \mid x \text{ is even}\}$
- subset:  $U \subset V, U \subseteq V$
- union:  $U \cup V$
- intersection:  $U \cap V$

# Sets

- order doesn't matter
- finite:  $\{2, 4, 6\}$
- infinite:  $\{2, 4, 6, \dots\}$   $\{x \mid x \text{ is even}\}$
- subset:  $U \subset V, U \subseteq V$
- union:  $U \cup V$
- intersection:  $U \cap V$
- power set of  $U$ :  $2^U$ , is set of all subsets

# Sets

- order doesn't matter
- finite:  $\{2, 4, 6\}$
- infinite:  $\{2, 4, 6, \dots\}$   $\{x \mid x \text{ is even}\}$
- subset:  $U \subset V, U \subseteq V$
- union:  $U \cup V$
- intersection:  $U \cap V$
- power set of  $U$ :  $2^U$ , is set of all subsets
- cardinalities:  $\aleph_0, \aleph_1, 2^{\aleph_0}$

# Sets

- order doesn't matter
- finite:  $\{2, 4, 6\}$
- infinite:  $\{2, 4, 6, \dots\}$   $\{x \mid x \text{ is even}\}$
- subset:  $U \subset V, U \subseteq V$
- union:  $U \cup V$
- intersection:  $U \cap V$
- power set of  $U$ :  $2^U$ , is set of all subsets
- cardinalities:  $\aleph_0, \aleph_1, 2^{\aleph_0}$
- the continuum hypothesis

# Sets

- order doesn't matter
- finite:  $\{2, 4, 6\}$
- infinite:  $\{2, 4, 6, \dots\}$   $\{x \mid x \text{ is even}\}$
- subset:  $U \subset V, U \subseteq V$
- union:  $U \cup V$
- intersection:  $U \cap V$
- power set of  $U$ :  $2^U$ , is set of all subsets
- cardinalities:  $\aleph_0, \aleph_1, 2^{\aleph_0}$
- the continuum hypothesis
- etc., etc.

# Sequences

- order matters

# Sequences

- order matters
- $(1, 3, 5, \dots)$

# Sequences

- order matters
- $(1, 3, 5, \dots)$
- finite sequence called a *tuple*

# Sequences

- order matters
- $(1, 3, 5, \dots)$
- finite sequence called a *tuple*
- 2 elements is a *pair*

# Sequences

- order matters
- $(1, 3, 5, \dots)$
- finite sequence called a *tuple*
- 2 elements is a *pair*
- $k$  elements is a  $k$ -tuple

# Functions and Predicates

A *function* or *mapping*

$$f : X \rightarrow Y$$

- $f : \textit{domain} \rightarrow \textit{range}$

# Functions and Predicates

A *function* or *mapping*

$$f : X \rightarrow Y$$

- $f : \textit{domain} \rightarrow \textit{range}$
- input values are called *arguments*.

# Functions and Predicates

A *function* or *mapping*

$$f : X \rightarrow Y$$

- $f : \textit{domain} \rightarrow \textit{range}$
- input values are called *arguments*.
- a  $k$ -ary function has  $k$  arguments.

# Functions and Predicates

A *function* or *mapping*

$$f : X \rightarrow Y$$

- $f : \textit{domain} \rightarrow \textit{range}$
- input values are called *arguments*.
- a  $k$ -ary function has  $k$  arguments.
- a *predicate* is a function with range  $\{\textit{true}, \textit{false}\}$ .

# Functions and Predicates

A *function* or *mapping*

$$f : X \rightarrow Y$$

- $f : \textit{domain} \rightarrow \textit{range}$
- input values are called *arguments*.
- a  $k$ -ary function has  $k$  arguments.
- a *predicate* is a function with range  $\{\textit{true}, \textit{false}\}$ .
- a *binary relation* is a predicate whose domain is a Cartesian product  $U \times V$ .

# A Popular Example

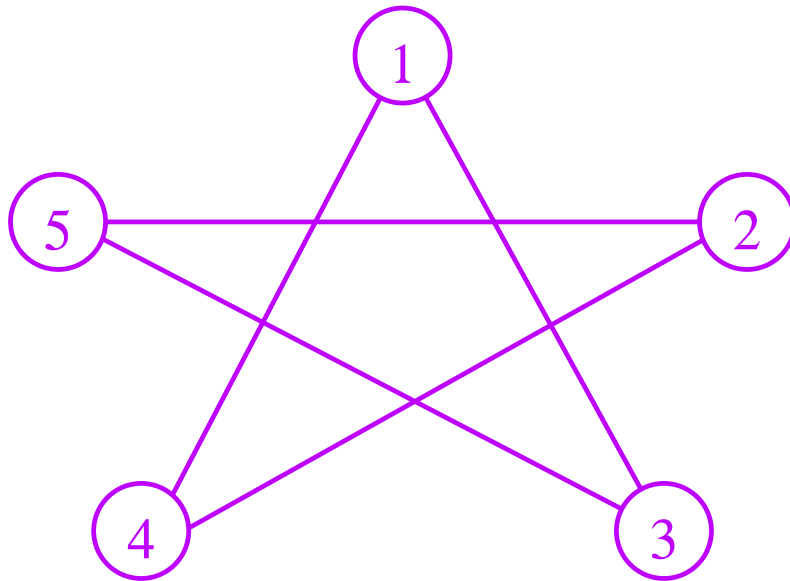
The “SCISSORS, PAPER, ROCK” game as a predicate (from the "row player" point of view):

	SCISSORS	PAPER	ROCK
SCISSORS	<i>false</i>	<i>true</i>	<i>false</i>
PAPER	<i>false</i>	<i>false</i>	<i>true</i>
ROCK	<i>true</i>	<i>false</i>	<i>false</i>

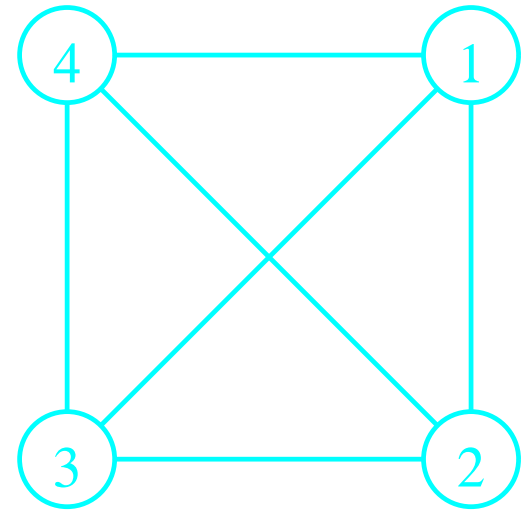
and as a relation:

$\{(\text{SCISSORS}, \text{PAPER}), (\text{PAPER}, \text{ROCK}), (\text{ROCK}, \text{SCISSORS})\}$ .

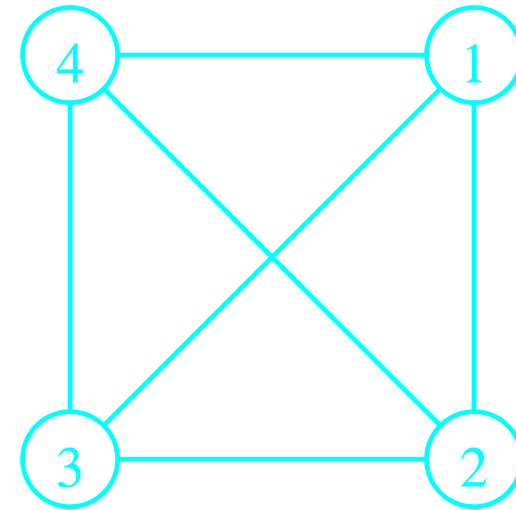
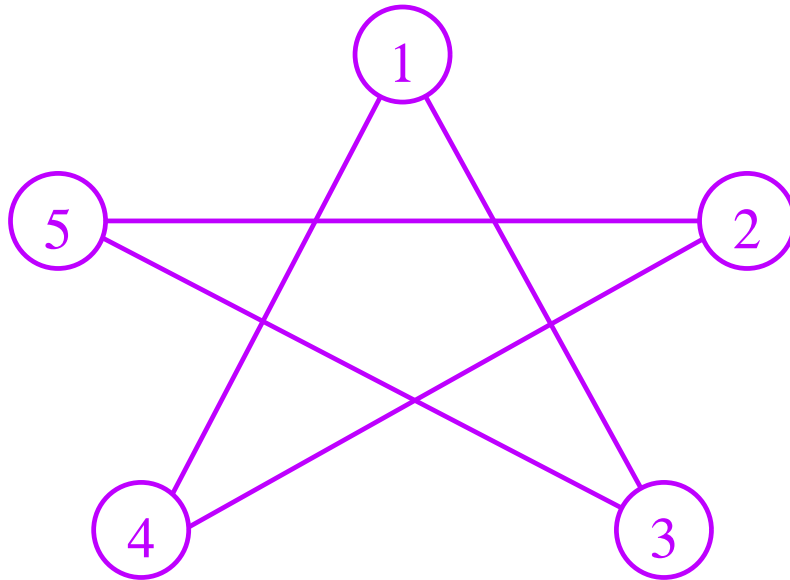
# Graphs



•  $G = (V, E)$ , where

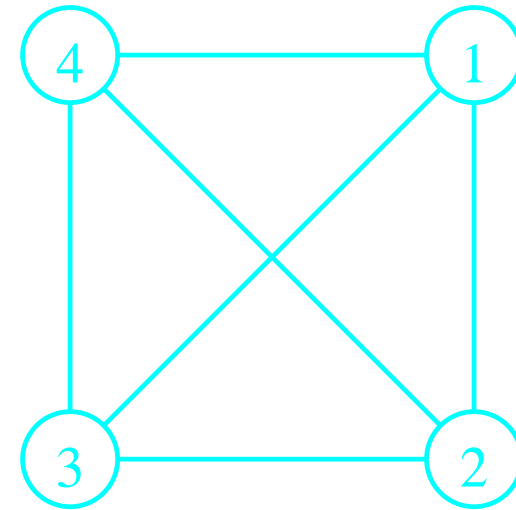
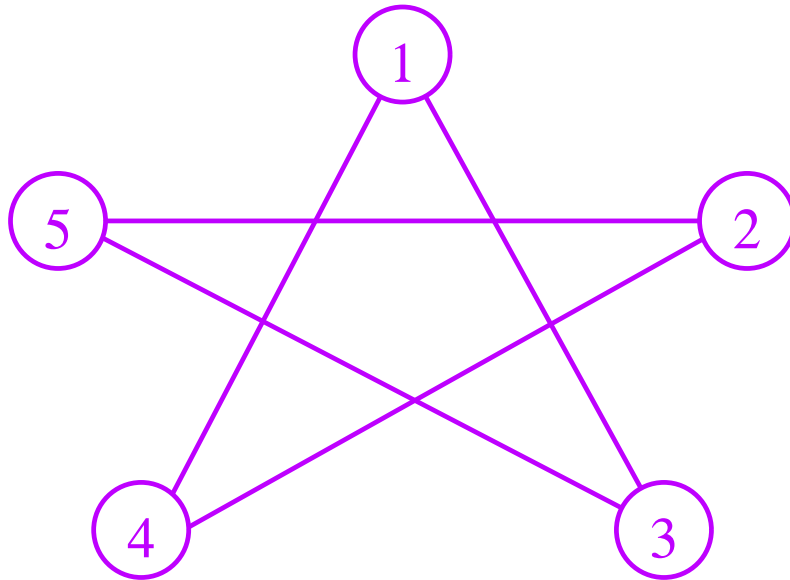


# Graphs



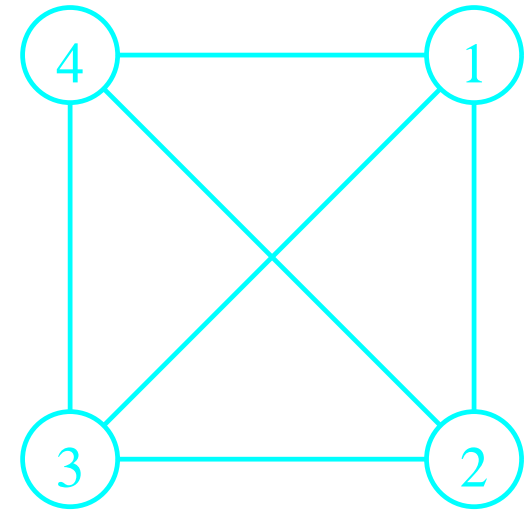
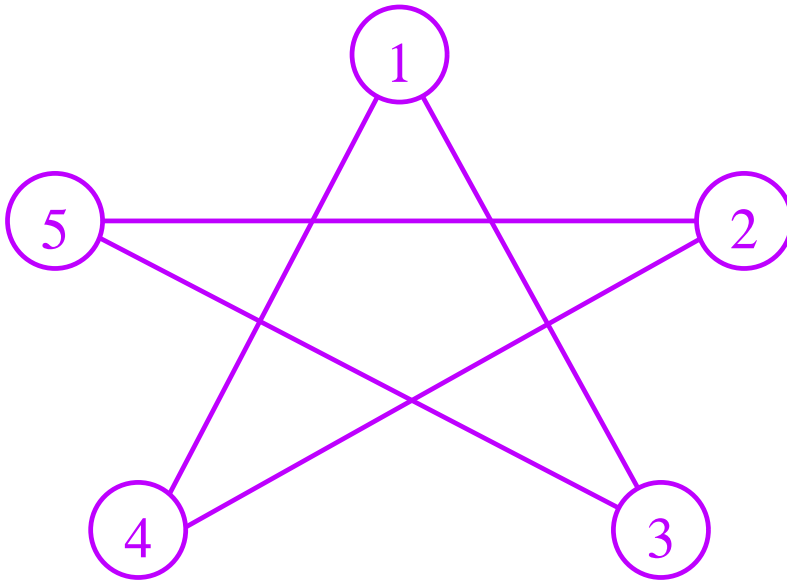
- $G = (V, E)$ , where
- $V$  is set of *nodes* or *vertices*, and

# Graphs



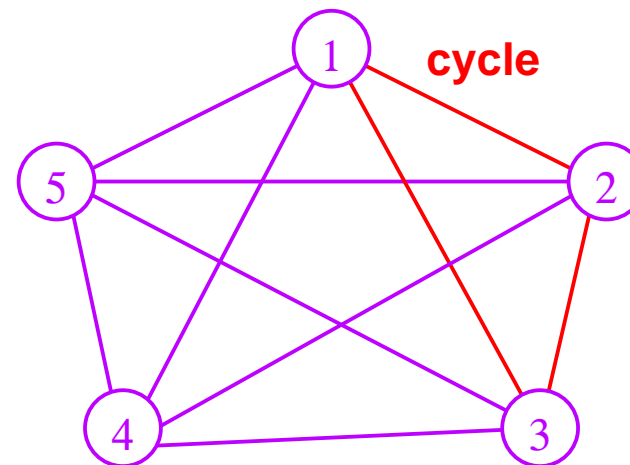
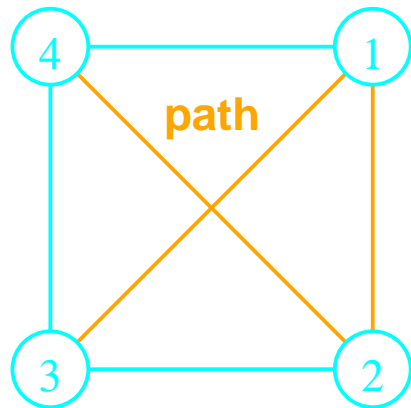
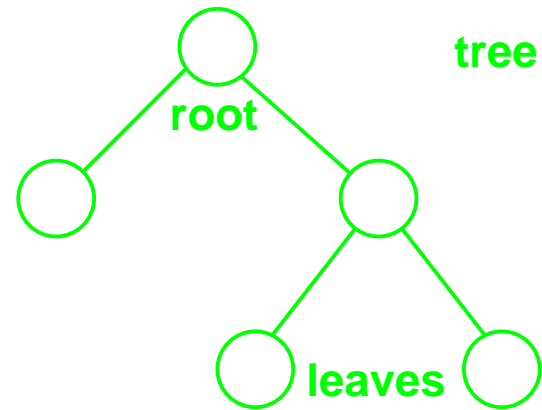
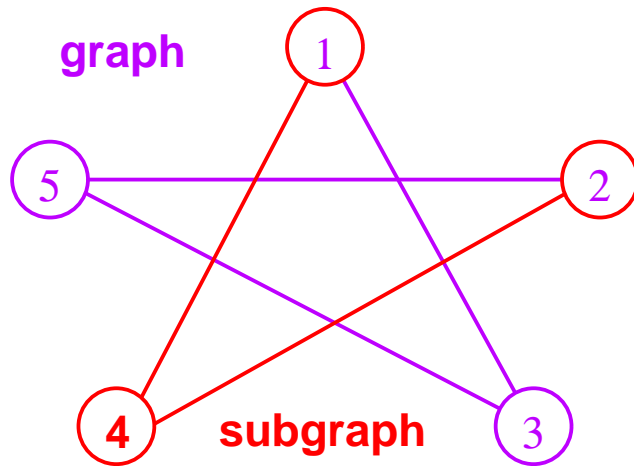
- $G = (V, E)$ , where
- $V$  is set of *nodes* or *vertices*, and
- $E$  is set of *edges*

# Graphs

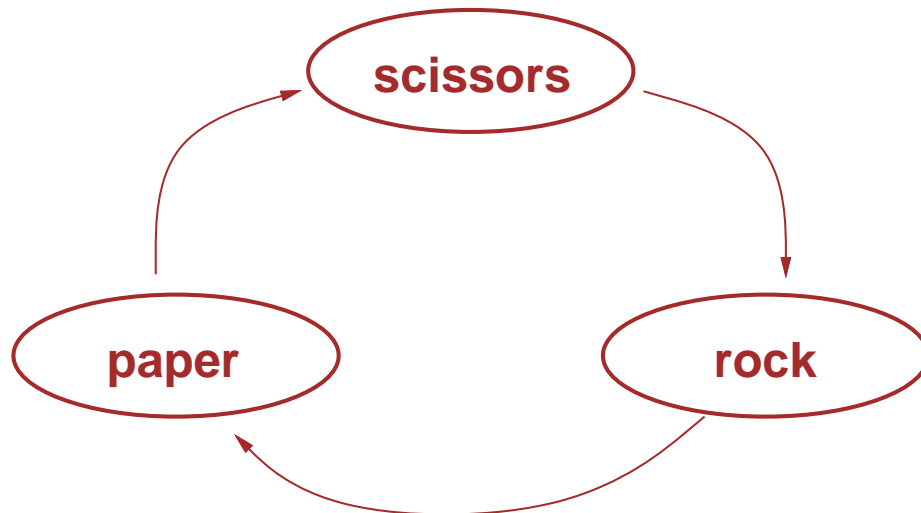
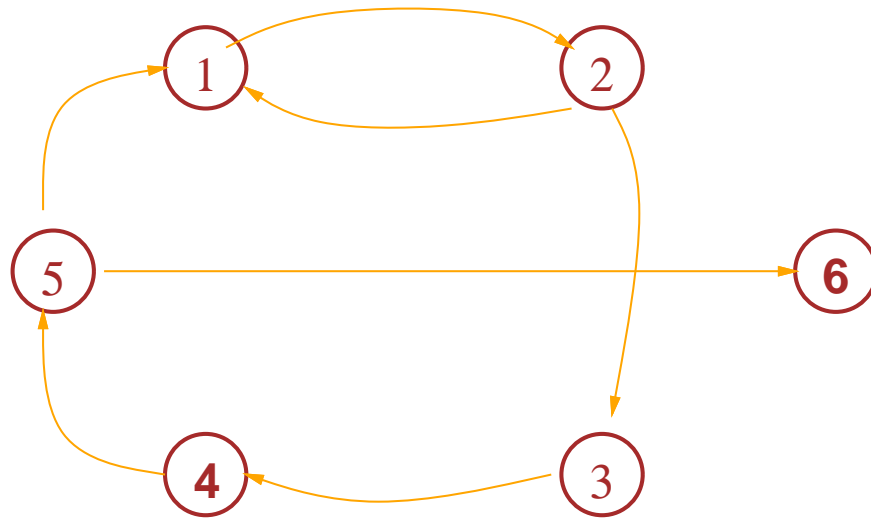


- $G = (V, E)$ , where
- $V$  is set of *nodes* or *vertices*, and
- $E$  is set of *edges*
- *degree* of a vertex is number of edges

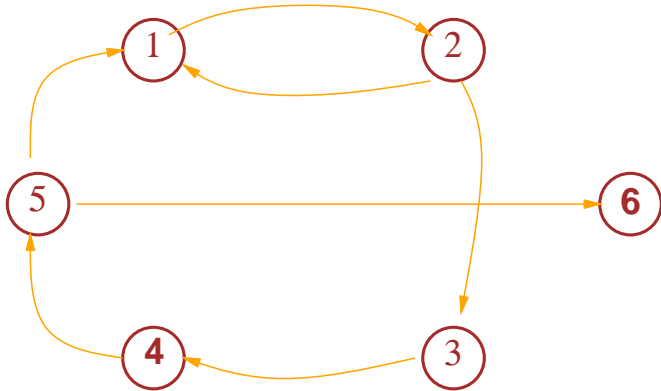
# Graphs



# Directed Graphs



# A Directed Graph and its Adjacency Matrix



$$\begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

# Strings and Languages

- an *alphabet* is a finite set of *symbols*

# Strings and Languages

- an *alphabet* is a finite set of *symbols*
- a *string over an alphabet* is a finite sequence of symbols from that alphabet.

# Strings and Languages

- an *alphabet* is a finite set of *symbols*
- a *string over an alphabet* is a finite sequence of symbols from that alphabet.
- the *length* of a string is the number of symbols

# Strings and Languages

- an *alphabet* is a finite set of *symbols*
- a *string over an alphabet* is a finite sequence of symbols from that alphabet.
- the *length* of a string is the number of symbols
- the *empty string*  $\varepsilon$

# Strings and Languages

- an *alphabet* is a finite set of *symbols*
- a *string over an alphabet* is a finite sequence of symbols from that alphabet.
- the *length* of a string is the number of symbols
- the *empty string*  $\varepsilon$
- *reverse*: *abcd* reversed is *dcba*.

# Strings and Languages

- an *alphabet* is a finite set of *symbols*
- a *string over an alphabet* is a finite sequence of symbols from that alphabet.
- the *length* of a string is the number of symbols
- the *empty string*  $\varepsilon$
- *reverse*: *abcd* reversed is *dcba*.
- *substring*: *xyz* in *xyzzy*.

# Strings and Languages

- an *alphabet* is a finite set of *symbols*
- a *string over an alphabet* is a finite sequence of symbols from that alphabet.
- the *length* of a string is the number of symbols
- the *empty string*  $\varepsilon$
- *reverse*: *abcd* reversed is *dcba*.
- *substring*: *xyz* in *xyzzy*.
- *concatenation* of *xyz* and *zy* is *xyzzy*.

# Strings and Languages

- an *alphabet* is a finite set of *symbols*
- a *string over an alphabet* is a finite sequence of symbols from that alphabet.
- the *length* of a string is the number of symbols
- the *empty string*  $\varepsilon$
- *reverse*: *abcd* reversed is *dcba*.
- *substring*: *xyz* in *xyzzy*.
- *concatenation* of *xyz* and *zy* is *xyzzy*.
- $x^k$  is  $x \cdots x$ ,  $k$  times.

# Strings and Languages

- an *alphabet* is a finite set of *symbols*
- a *string over an alphabet* is a finite sequence of symbols from that alphabet.
- the *length* of a string is the number of symbols
- the *empty string*  $\varepsilon$
- *reverse*:  $abcd$  reversed is  $dcba$ .
- *substring*:  $xyz$  in  $xyzzy$ .
- *concatenation* of  $xyz$  and  $zy$  is  $xyzzy$ .
- $x^k$  is  $x \cdots x$ ,  $k$  times.
- a *language*  $L$  is a set of strings.

# Proofs

We will use the following basic kinds of proofs.

- by construction

# Proofs

We will use the following basic kinds of proofs.

- by construction
- by contradiction

# Proofs

We will use the following basic kinds of proofs.

- by construction
- by contradiction
- by induction

# Proofs

We will use the following basic kinds of proofs.

- by construction
- by contradiction
- by induction
- by reduction

# Proofs

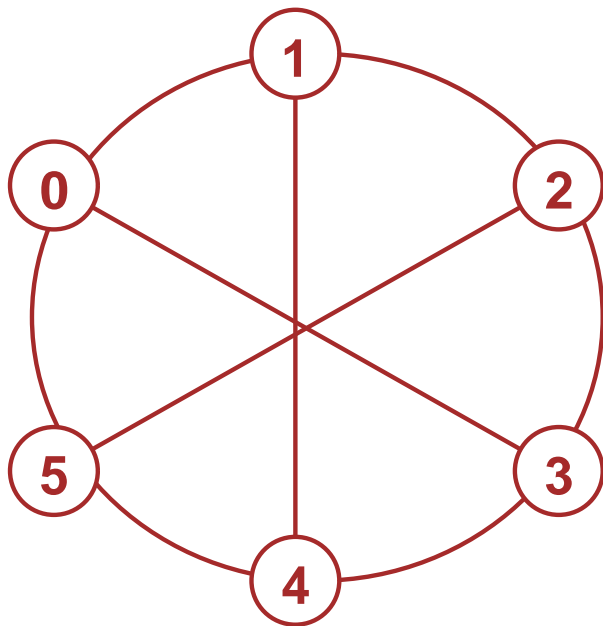
We will use the following basic kinds of proofs.

- by construction
- by contradiction
- by induction
- by reduction
- we will often mix them.

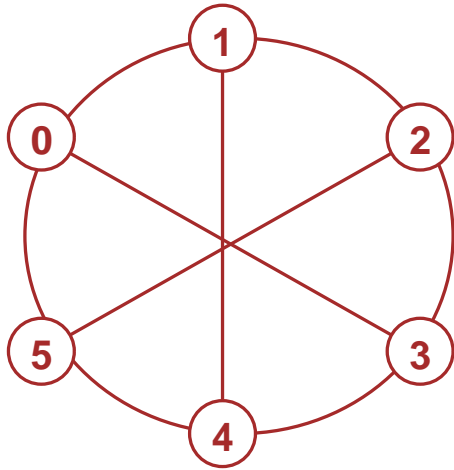
# Proof by Construction

A graph is  $k$ -regular if every node has degree  $k$ .

**Theorem:** For every even  $n > 2$ , there exists a 3-regular graph with  $n$  nodes.



# Proof by Construction



**Proof:** Construct  $G = (V, E)$ , where  
 $V = \{0, 1, \dots, n - 1\}$  and

$$E = \{\{i, i + 1\} \mid \text{for } 0 \leq i \leq n - 2\} \cup \{n - 1, 0\} \\ \cup \{\{i, i + n/2\} \mid \text{for } 0 \leq i \leq n/2 - 1\}.$$

*Note:* A picture is helpful, but it is *not* a proof!

# Proof by Contradiction

**Theorem:**  $\sqrt{2}$  is irrational.

**Proof:** Suppose not. Then  $\sqrt{2} = \frac{m}{n}$ , where  $m$  and  $n$  are relatively prime.

$$\begin{aligned}n\sqrt{2} &= m \\2n^2 &= m^2\end{aligned}$$

## Proof by Contradiction (cont.)

So  $m^2$  is even, and so is  $m = 2k$ .

$$\begin{aligned}2n^2 &= (2k)^2 \\ &= 4k^2 \\ n^2 &= 2k^2\end{aligned}$$

Thus  $n^2$  is even, and so is  $n$ .

Therefore both  $m$  and  $n$  are even, and not relatively prime!

*Reductio ad absurdum.*

# Proof by Induction

Prove properties of elements of an infinite set.

$$\mathcal{N} = \{1, 2, 3, \dots\}$$

To prove that  $\varphi$  holds for each element, show:

- *base step*: show that  $\varphi(1)$  is true.

# Proof by Induction

Prove properties of elements of an infinite set.

$$\mathcal{N} = \{1, 2, 3, \dots\}$$

To prove that  $\varphi$  holds for each element, show:

- *base step*: show that  $\varphi(1)$  is true.
- *induction step*: show that if  $\varphi(i)$  is true (the induction hypothesis), then so is  $\varphi(i + 1)$ .

# Induction Example

**Theorem:** All cows are the same color.

# Induction Example

**Theorem:** All cows are the same color.

*Base step:* A single-cow set is definitely the same color.

# Induction Example

**Theorem:** All cows are the same color.

*Base step:* A single-cow set is definitely the same color.

*Induction Step:* Assume all sets of  $i$  cows are the same color. Divide the set  $\{1, \dots, i + 1\}$  into  $U = \{1, \dots, i\}$ , and  $V = \{2, \dots, i + 1\}$ .

# Induction Example

**Theorem:** All cows are the same color.

*Base step:* A single-cow set is definitely the same color.

*Induction Step:* Assume all sets of  $i$  cows are the same color. Divide the set  $\{1, \dots, i + 1\}$  into

$U = \{1, \dots, i\}$ , and  $V = \{2, \dots, i + 1\}$ .

All cows in  $U$  are the same color by the induction hypothesis.

# Induction Example

**Theorem:** All cows are the same color.

*Base step:* A single-cow set is definitely the same color.

*Induction Step:* Assume all sets of  $i$  cows are the same color. Divide the set  $\{1, \dots, i + 1\}$  into  $U = \{1, \dots, i\}$ , and  $V = \{2, \dots, i + 1\}$ .

All cows in  $U$  are the same color by the induction hypothesis.

All cows in  $V$  are the same color by the induction hypothesis.

# Induction Example

**Theorem:** All cows are the same color.

*Base step:* A single-cow set is definitely the same color.

*Induction Step:* Assume all sets of  $i$  cows are the same color. Divide the set  $\{1, \dots, i + 1\}$  into  $U = \{1, \dots, i\}$ , and  $V = \{2, \dots, i + 1\}$ .

All cows in  $U$  are the same color by the induction hypothesis.

All cows in  $V$  are the same color by the induction hypothesis.

All cows in  $U \cap V$  are the same color by the induction hypothesis.

# Induction Example (cont.)

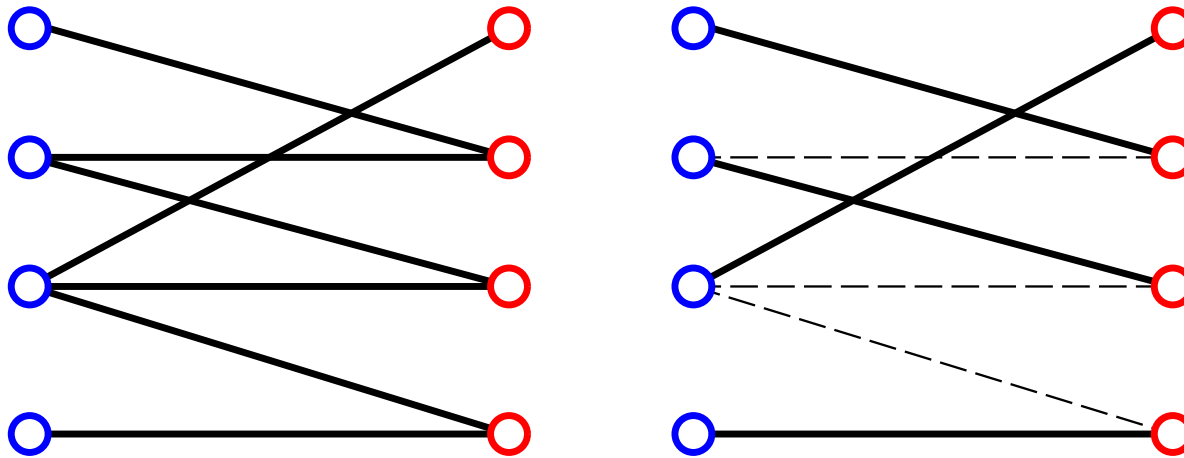
*Ergo*, all cows are the same color.

*Quod Erat Demonstrandum.*

# Proof by Reduction

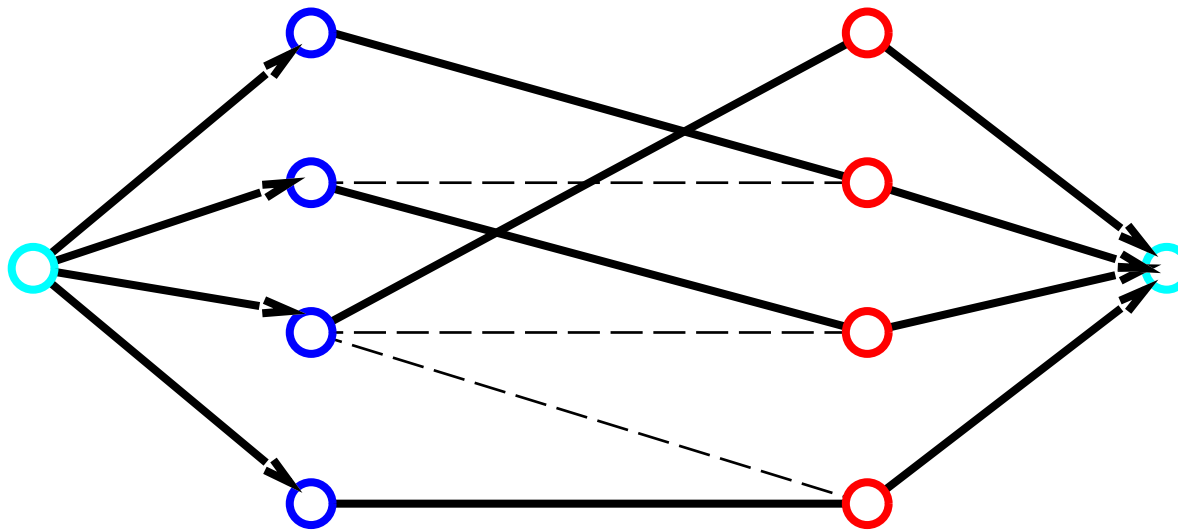
We can sometime solve problem **A** by **reducing** it to problem **B**, whose solution we already know.

Example: Maximal matching in bipartite graphs:



# Proof by Reduction

Reducing bipartite matching to MAX FLOW:



**Reduction:** Put capacity 1 on each edge.