# 1. Number theory

## ① even and odd integers

$n$ is odd $\iff \exists k \in \mathbb{Z}$ s.t. $n = 2k+1$

$n$ is even $\iff \exists k \in \mathbb{Z}$ s.t. $n = 2k$

↳ product of any two odd is odd (T1 Q9)

↳ $n^2$ is odd $\iff n$ is odd (T1 Q10)

## ② Divisibility of integers

$d \mid n \iff \exists k \in \mathbb{Z}$ s.t. $n = dk$

$d$ divides $n$, $n$ is divisible by $d$

↳ divisibility is transitive $a \mid b \wedge b \mid c \to a \mid c$ (L4.3.2)

↳ $a \mid b \to a \le b$ (L4.3.2)

## ③ Rational numbers

$r$ is rational $\iff \exists a, b \in \mathbb{Z}$ s.t. $r = \frac{a}{b}$, $b \neq 0$

↳ closure under $+$, $\times$ (multiply by 0)

↳ not closed under $\div$ (divide by 0)

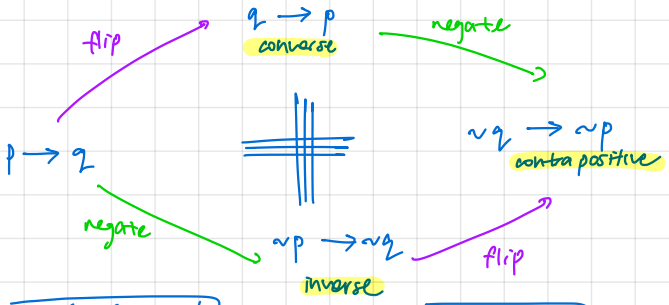
# 2. Logical statements

## ① operators and conditionals

$\sim \quad \wedge \quad \to \quad \vee \quad \leftrightarrow$

$p \oplus q \equiv (p \vee q) \wedge (\sim p \vee \sim q)$

### contrapositives, converse, inverse




### implication law

$p \to q \equiv \sim p \vee q$

$\sim(p \to q) \equiv p \wedge \sim q$

### biconditional

$p \leftrightarrow q \equiv (p \to q) \wedge (q \to p)$

### wording

"p only if q": $p \to q \equiv (\sim q \to \sim p)$
"p if and only if q": $p \leftrightarrow q \equiv (p \to q) \wedge (q \to p)$
"p is necessary for q": $q \to p$
"p is a sufficient condition for q": $p \to q$

## ② logical equivalences

| | | | |
|---|---|---|---|
| 1 | Commutative laws | $p \wedge q \equiv q \wedge p$ | $p \vee q \equiv q \vee p$ |
| 2 | Associative laws | $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$ | $(p \vee q) \vee r \equiv p \vee (q \vee r)$ |
| 3 | Distributive laws | $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$ | $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ |
| 4 | Identity laws | $p \wedge \mathbf{true} \equiv p$ | $p \vee \mathbf{false} \equiv p$ |
| 5 | Negation laws | $p \vee \sim p \equiv \mathbf{true}$ | $p \wedge \sim p \equiv \mathbf{false}$ |
| 6 | Double negative law | $\sim(\sim p) \equiv p$ | |
| 7 | Idempotent laws | $p \wedge p \equiv p$ | $p \vee p \equiv p$ |
| 8 | Universal bound laws | $p \vee \mathbf{true} \equiv \mathbf{true}$ | $p \wedge \mathbf{false} \equiv \mathbf{false}$ |
| 9 | De Morgan's laws | $\sim(p \wedge q) \equiv \sim p \vee \sim q$ | $\sim(p \vee q) \equiv \sim p \wedge \sim q$ |
| 10 | Absorption laws | $p \vee (p \wedge q) \equiv p$ | $p \wedge (p \vee q) \equiv p$ |
| 11 | Negation of **true** and **false** | $\sim\mathbf{true} \equiv \mathbf{false}$ | $\sim\mathbf{false} \equiv \mathbf{true}$ |

## ③ argument form

| Rule of inference | | |
|---|---|---|
| Modus Ponens | $p \to q$, $p$ • $q$ | |
| Modus Tollens | $p \to q$, $\sim q$ • $\sim p$ | |
| Generalization | $p$ • $p \vee q$ | $q$ • $p \vee q$ |
| Specialization | $p \wedge q$ • $p$ | $p \wedge q$ • $q$ |
| Conjunction | $p$, $q$ • $p \wedge q$ | |

| Rule of inference | | |
|---|---|---|
| Elimination | $p \vee q$, $\sim q$ • $p$ | $p \vee q$, $\sim p$ • $q$ |
| Transitivity | $p \to q$, $q \to r$ • $p \to r$ | |
| Proof by Division Into Cases | $p \vee q$, $p \to r$, $q \to r$ • $r$ | |
| Contradiction Rule | $\sim p \to \mathbf{false}$ • $p$ | |

## ④ fallacies and sound arguments

(sound argument: all premises true ∧ argument is valid)

1. ambiguous premise
2. circular reasoning
3. jumping to conclusion
4. converse error $p \to q$, $q$, $\therefore p$
5. inverse error $p \to q$, $\sim p$, $\therefore \sim q$

# 3. Quantified statements

## ① Negations

1. $\sim(\forall x \in D, P(x)) \equiv \exists x \in D, \sim P(x)$
2. $\sim(\exists x \in D, P(x)) \equiv \forall x \in D, \sim P(x)$
3. $\sim(\forall x \in D \ \exists y \in E$ s.t. $P(x,y))$
   $\equiv \exists x \in D \ \forall y \in E$ s.t. $\sim P(x,y)$

# 5. Sets

## ① notation

**roster notation**
↳ listing out all elements
$\{1, 2, 3 \dots\}$

**set builder notation**
↳ filtered subset
$\{x \in U \mid x > 2\}$
filter condition

**replacement notation**
↳ set of processed objects
$\{t(x) \mid x \in U\}$

## ② subsets and equality

**subset**

$A \subseteq B \iff \forall x\,(x \in A \to x \in B)$

$\sim(A \subseteq B) \iff \exists x\,(x \in A \to x \notin B)$

$\phi \subseteq A$, $A \subseteq A$

**proper subset**

$A \subsetneq B \iff A \subseteq B \wedge B \not\subseteq A$

**set equality**

$A = B \iff A \subseteq B \wedge B \subseteq A$
$\iff \forall z\,(z \in A \leftrightarrow z \in B)$

↳ proving $A = B$:

1. $x \in A$ … $x \in B$. $A \subseteq B$
2. $x \in B$ … $x \in A$. $B \subseteq A$.

## ③ operations on sets

**union**

$A \cup B = \{x \in U : x \in A \vee x \in B\}$

**intersection**

$A \cap B = \{x \in U : x \in A \wedge x \in B\}$

**difference**

$A \setminus B = \{x \in U \mid x \in A \wedge x \notin B\}$

**complement**

$A^C = U \setminus A = \{x \in U \mid x \notin A\}$

## ④ unique sets

**powerset** set of all subsets of $A$. $|P(A)| = 2^{|A|}$

**disjoint sets**

↳ pairwise disjoint: $A_i \cap A_j = \phi$, $i \neq j$
↳ mutually disjoint: $A_1 \cap A_2 \dots A_i = \phi$
↳ to prove: suppose not empty, take element from intersection, derive contradiction

**partitions**

A collection of non-empty sets is a partition iff:

1. $\forall S \in \mathcal{S},\ S \neq \phi \wedge S \subseteq A$ → all components are not empty subsets of $A$
2. $\forall x \in A, \exists S \in \mathcal{S}$ s.t. $x \in S$
$\wedge\ \forall x \in A, \forall S_1, S_2 \in \mathcal{S}(x \in S_1 \wedge x \in S_2 \to S_1 = S_2)$

↳ all elements of $A$ belong to exactly one component

**tuples and cartesian products**

$A \times B = \{(x, y) \mid x \in A \text{ and } y \in B\}$

## ⑤ properties of sets

For all set $A, B, C$ in a context where $U$ is the universal set, the following hold.

| | | |
|---|---|---|
| Identity Laws | $A \cup \varnothing = A$ | $A \cap U = A$ |
| Universal Bound Laws | $A \cup U = U$ | $A \cap \varnothing = \varnothing$ |
| Idempotent Laws | $A \cup A = A$ | $A \cap A = A$ |
| Double Complement Law | $\overline{(\overline{A})} = A$ | |
| Commutative Laws | $A \cup B = B \cup A$ | $A \cap B = B \cap A$ |
| Associative Laws | $(A \cup B) \cup C = A \cup (B \cup C)$ | $(A \cap B) \cap C = A \cap (B \cap C)$ |
| Distributive Laws | $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ | $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ |
| De Morgan's Laws | $\overline{A \cup B} = \overline{A} \cap \overline{B}$ | $\overline{A \cap B} = \overline{A} \cup \overline{B}$ |
| Absorption Laws | $A \cup (A \cap B) = A$ | $A \cap (A \cup B) = A$ |
| Complement Laws | $A \cup \overline{A} = U$ | $A \cap \overline{A} = \varnothing$ |
| Set Difference Law | $A \setminus B = A \cap \overline{B}$ | |
| | $\overline{\varnothing} = U$ | $\overline{U} = \varnothing$ |

Tutorial 3 Q9 $A \subseteq B \leftrightarrow A \cup B = B$

# 4. methods of proof

**direct**
↓
contrapositive
cases
↓
induction
direct

**indirect**
contradiction

example / counterexample

**uniqueness** → existis
→ $\geq 1, \leq 1$ or $x \in S_1 \wedge x \in S_2 \to S_1 = S_2$

# 6. Relations

## ① Relations and inverses

$xRy \iff$ some relationship

$R = \{(x,y) \in A \times B \mid xRy\}$ (A: domain, B: codomain)

$R^{-1} = \{(y,x) \in B \times A \mid yR^{-1}x\}$

## ② Equivalence relations

**Reflexivity**

$\forall x \in A, xRx$

for all points

**Symmetry**

$\forall x,y \in A\ (xRy \longrightarrow yRx)$

for all pairs

**Transitivity**

$\forall x,y,z \in A\ (xRy \wedge yRz \longrightarrow xRz)$

vacuously true if no or insufficient relation

↳ reflexivity : universal generalisation
↳ symmetry : universal generalisation
} direct proofs possible

↳ transitive : exhaustion (direct) or contradiction (indirect)

## ③ equivalence classes

$[x] = \{y \in A \mid xRy\}$

→ representative of its equivalence class, an element $\in [x]$ (Definition 7.1.1)

**Lemma 6.3.4**

$x \sim y \iff [x] = [y] \iff [x] \cap [y] \neq \emptyset$

**quotient as partition**

$A/\sim = \{[x] : x \in A\}$

---

## ④ congruence modulo n

$a \equiv b \pmod n$ ie. $a - b = nk \iff$ same remainder after being divided by $n$

$[x] = \{nk + q\}, \quad 0 \leq q < n$

↳ $[x] + [y] = [x+y]$
↳ $[x] \cdot [y] = [x \cdot y]$
} $\times$ and $+$ well defined on $\mathbb{Z}_n$ ($\mathbb{Z}$ \ congruence mod n)

## ⑤ antisymmetry & partial orders

**antisymmetry**

$\forall x,y\ (xRy \wedge yRx \longrightarrow y = x)$

**partial orders** $\preccurlyeq$

1. R is reflexive
2. R is transitive
3. R is antisymmetric

$(A, R)$ partially ordered set

**total order** $\preccurlyeq^*$

↳ partial order that is total

$\forall x,y\ xRy \vee yRx$

ie. everything $\in A$ is somehow reported

**min & max elements** (if $A \neq \emptyset$, minimal element must exist (proposition 7.4.6))

1. **minimal** : smallest along its chain

$\forall x \in A\ (x \preccurlyeq c \rightarrow c = x)$

prop 7.4.4

2. **minimum** : connected to everything & smaller

$\forall x \in A\ \ c \preccurlyeq x$

need not exist

3. **maximal** : largest along its chain

$\forall x \in A\ (x \succcurlyeq c \rightarrow x = c)$

4. **maximum** : connected to everything & larger

$\forall x \in A\ \ c \succcurlyeq x$

**Kahn's algorithm & linearisation**

Every partial order $\preccurlyeq$ has a linearisation $\preccurlyeq^*$ (Theorem 7.4.10)

↳ linearise $\preccurlyeq$ to get total order $\preccurlyeq^*$ s.t.

$\forall x,y \in A\ (x \preccurlyeq y \longrightarrow x \preccurlyeq^* y)$

order is preserved in linearisation for those that had an order

↳ linearisation may differ based on which you 'choose' at each 'level'

Proof.
2. Suppose the run produces $A_0, A_1, \ldots, A_n, c_0, c_1, \ldots, c_{n-1}$ and $\preccurlyeq^*$.
3. Note $A = \{c_0, c_1, \ldots, c_{n-1}\}$ because the removal of $c_0, c_1, \ldots, c_{n-1}$ from $A$ makes the set empty.
4. Note also that $\preccurlyeq^*$ is a total order on $A$ because it is by definition only a renaming of the total order $\leqslant$ on $\{0, 1, \ldots, n-1\}$.
5. 5.1. Take $x \in A$ and $c_j \in A$ such that $x \prec c_j$.
   5.2. Then $x \notin A_j$ as $c_j$ is minimal in $A_j$.
   5.3. So $x \in A \setminus A_j = \{c_0, c_1, \ldots, c_{j-1}\}$ by the choices of $A_0, A_1, \ldots, A_j, c_0, c_1, \ldots, c_{j-1}$.
   5.4. Let $i \in \{0, 1, \ldots, j-1\}$ such that $x = c_i$.
   5.5. Then $x = c_i \preccurlyeq^* c_j$ by the definition of $\preccurlyeq^*$, as $i \leqslant j - 1 < j$. □

Kahn's Algorithm (1962)
Input: a finite set $A$, a partial order $\preccurlyeq$ on $A$.
(1) Set $A_0 := A$ and $i := 0$.
(2) Repeat until $A_i = \varnothing$:
(2.1) find a minimal element $c_i$ of $A_i$ wrt $\preccurlyeq$;
(2.2) set $A_{i+1} := A_i \setminus \{c_i\}$;
(2.3) set $i := i + 1$.
Output: a linearization $\preccurlyeq^*$ of $\preccurlyeq$ defined by setting, for all indices $i, j$,

$$c_i \preccurlyeq^* c_j \quad \Leftrightarrow \quad i \leqslant j.$$

# 7. Functions

**well defined**

$f: A \to B \quad x \mapsto f(x)$

(domain, codomain)

↳ every input has a valid output
↳ every input has exactly one output

**identity**

$f: A \to A, f(x) = x$
↳ denoted $id_A$

**equality**

1. domains & codomains equal
2. $\forall x \in A \quad f(x) = g(x)$

---

## ① (setwise) image & preimage

$X \subseteq A \to f(X) = \{ f(x) : x \in X \}$

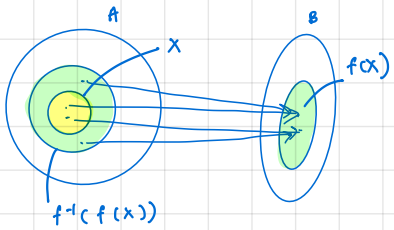
$Y \subseteq B \to f^{-1}(Y) = \{ x \in A : f(x) \in Y \}$

**properties**

1. preimage will always contain $X$, but converse is not true
↳ intuition: each $y \in B$ can have multiple $x$ mapping to it, including those outside $X \subseteq A$

$X \subseteq f^{-1}f(X)$ ✓

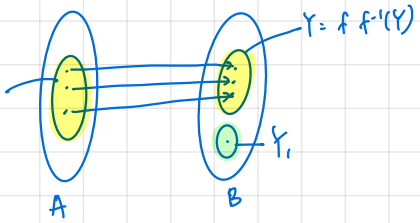
$f^{-1}f(X) \subseteq X$ ✗




2. image of $f^{-1}(Y)$ will always equal $Y$, but converse is false
↳ intuition: some elements in codomain may not be mapped to, preimage is empty set. image of that also empty.

$Y = ff^{-1}(X)$ ✓

$ff^{-1}(Y) \subseteq Y$ ✗




## ② composition

$f: A \to B \quad g \circ f = A \to C \quad g \circ f(x) = g(f(x))$
$g: B \to C$

} codomain of $f$ = domain of $g$

↳ **idempotency**: $f \circ f = f$, only defined if $f = id_A$
↳ **not commutative**: $g \circ f \neq f \circ g$
↳ **associative**: $h \circ (g \circ f) = (h \circ g) \circ f$

## ③ surjections & injections

**surjections**

$\forall y \in B, \exists x \in A$ s.t. $f(x) = y$

to prove: take $y \in B$, find $x \in A$ for each case. show $f(x) = y$.

↳ $f, g$ surjective → $g \circ f$ surjective
↳ $g \circ f$ surjective → $g$ (outer) surjective

**injections**

$\forall x_1, x_2 \; (f(x_1) = f(x_2) \to x_1 = x_2)$

To prove: take $x_1, x_2$ s.t. $f(x_1) = f(x_2)$. show $x_1 = x_2$

↳ $f, g$ injective → $g \circ f$ injective
↳ $g \circ f$ injective → $f$ (inner) injective

$f$ is injective ⟷ can left compose it w some $g$ to get an injective function $g \circ f$

## ④ bijections & inverses

**definition** $\forall y \in B \; \exists! x \in A \; y = f(x)$

$f$ is bijective ⟷ $f$ is surjective ∧ injective

**inverses**

$g = f^{-1}$ iff $\forall x \in A \; \forall y \in B \; (y = f(x) \leftrightarrow x = g(y))$

1. inverses are unique (proposition 9.3.17)
2. $f$ is bijective ⟷ $f^{-1}$ exists (theorem 9.3.19)
3. composition of inverses
$(g \circ f)^{-1} = f^{-1} \circ g^{-1}$
4. order of bijections: least $n \in \mathbb{Z}^+$ s.t. $\underbrace{f \circ f \circ \dots}_{n} = f$

## ⑤ objects as functions

**sequences**

$a_1, a_2 \dots = a: \mathbb{Z}_{\geq 0} \to Y \quad a: n \mapsto a_n$

↳ any function w domain $\mathbb{Z}_{\geq m}$ for $m \in \mathbb{Z}_{\geq 0}$ is equivalent to a sequence

**strings** order and repetition matter.

$a_0 a_1 \dots a_{\ell-1}$. $\ell \in \mathbb{Z}_{\geq 0}$ (length), $a_0 \dots \in A$.

$A$ = {set of characters} $A^*$ = {set of all possible strings}

$\varepsilon$ = empty string

$a_0 a_1 \dots a_{\ell-1} = a: \{0, 1, \dots \ell-1\} \to A, \; n \mapsto a_n$

↳ so all functions $a: \{m, m+1 \dots m+\ell-1\} \to A$ where $m \in \mathbb{Z}$ and $\ell \in \mathbb{Z}_{\geq 0}$ represent a string of length $\ell$.

# 8. induction & recursion

## ① sequences & sets

### sequences and functions

**sequence** $a_n$ $f(0)=a_0$ $f(1)=a_1$ ...

$\mathbb{Z}_{\geq 0}$ 0 1 ...

**subsequence** $a_m$ $f(m)=a_m$ $f(m+1)=a_{m+1}$ ...

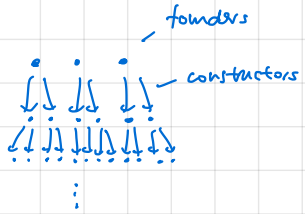
$\mathbb{Z}_{\geq 0}$ m m+1 ...

**recursively defined sequences**: definition of $a_n$ involves $a_0, a_1, a_{n-1}$ ... for all but finitely many $n \in \mathbb{Z}_{\geq 0}$

eg. $F_0, F_1, F_2$ ... $F_0 = 0$, $F_1 = 1$, $F_{n+2} = F_{n+1} + F_n$

↳

| $a_n$ | 0 | 1 | $F(0)+F(1)$ | $F(1)+F(2)$ |
|---|---|---|---|---|
| $\mathbb{Z}_{\geq 0}$ | 0 | 1 | 2 | 3 |




### recursively defined sets

(base clause) certain elements, founders are in $S$. $c \in S$.

(recursion clause) specify certain functions, called constructors, under which the set is closed.

if $x \in S \longrightarrow f(x) \in S$.

(minimality clause) membership for $S$ can always be demonstrated by finitely many applications of the base & recursion clauses.

## ④ useful manipulations in sequences

### summation

1. $\sum_{k=m}^{n} a_k = \sum_{k=m}^{n-1} a_k + a_n$

2. method of cancellation by compensation to create subtraction of successive terms

eg. $\sum_{k=1}^{n} \frac{1}{k(k+1)} = \sum_{1} \frac{k+1-k}{k(k+1)} = \sum_{1} \frac{1}{k} - \frac{1}{k+1}$

$= \frac{1}{1} - \cancel{\frac{1}{2}} + \cancel{\frac{1}{2}} - \cancel{\frac{1}{3}} + \cancel{\frac{1}{3}} - \frac{1}{4}$ ... $= 1 - \frac{1}{n+1}$

3. $c \cdot \sum_{k=m}^{n} a_k + d \sum_{k=m}^{n} b_k = \sum_{k=m}^{n} (c \cdot a_k + d \cdot b_k)$

↳ save in $\prod_{k=m}^{n}$

4. change of variable (just sub)

## ② mathematical induction on sequences

1) intuition: trying to prove that $\forall n \in \mathbb{Z}_{\geq m}$, $P(n)$ is true.

1. show that $P(m)$ is true.
2. show that if $P(n)$ is true, $P(n+1)$ is true.
3. then true, since cascading implication.

eg. $n \in \mathbb{Z}_{\geq 0}$.

1. $P(0)$
2. $P(n) \rightarrow P(n+1)$

so $P(0) \rightarrow P(1)$, $P(1) \rightarrow P(2)$, ⋮ } $P(n)\ \forall n \in \mathbb{Z}_{\geq 0}$

2) techniques

1. (base step) prove that base case is true.
2. (induction step) suppose $P(n)$ is true for all $n \in \mathbb{Z}_{\geq m}$.
   - ↳ express the $n+1^{th}$ case in terms of the $n^{th}$ case, such that it can apply to base.
   - ↳ use the truth of the $n^{th}$ case $P(n)$ to prove truth of $n+1^{th}$ case.

## ③ strong MI on recursively defined sequences

1) intuition: to prove that $\forall n \in \mathbb{Z}_{\geq m}$ $P(n)$. eg. 2 base cases

1. show that $P(m)$, $P(m+1)$ several base cases are true.
2. show that $P(n) \wedge P(n+1) \wedge P(n+2) \rightarrow P(n+3)$
3. then true, since cascading implication

eg. $n \in \mathbb{Z}_{\geq 0}$.

1. $P(0)$, $P(1)$, $P(2)$.
2. $P(n) \wedge P(n+1) \wedge P(n+2) \rightarrow P(n+3)$

so $P(0) \wedge P(1) \wedge P(2) \rightarrow P(3)$
$P(1) \wedge P(2) \wedge P(3) \rightarrow P(4)$
⋮

2) technique

1. (base step) prove that base cases are true.
2. (induction step) suppose $P(n)$, $P(n+1)$ ... are true.
   - ↳ express the $n+1^{th}$ case in terms of the prior cases.
   - ↳ use the truth of the prior cases to prove the truth of the $n+1^{th}$ case

## ④ structural induction on recursively defined sets

↳ MI & strong MI are particular cases of strong MI, where set is $\mathbb{Z}_{\geq 0}$. so founder element is 0 (or m) and constructor $f$ is $f: x \rightarrow x+1$

↳ we can generalize inductive proof technique to any recursively defined set

Technique

1. define some property $P(x)$ about an element $x$ in the set $S$.
2. prove $P(c)$ for all founders $c$.
3. prove that $P(x) \rightarrow P(f(x))$ for all cases.

# 9. Cardinality

## ① Cardinality

1) finite sets : sets with a finite number of elements.

2) infinite sets : a set B is infinite $\longleftrightarrow$ $\exists A$ s.t. $A \subset B$ and $|A| = |B|$ (T8 Q3)

intuition : $A = B \setminus \{b\}$. $|B \cup c|$

3) cardinality : the abstract notion of size → finite sets — number of elements
→ infinite sets — ?

## ② functions & cardinality

1) bijections & same cardinality

finite sets, pf by MI ↑ Cantor, infinite sets ↗

$|A| = |B| \longleftrightarrow$ there is a bijection $f : A \to B$ (Theorem 10.1.1, 10.2.1)

2) properties of equal cardinality → it is an equivalence relation

**reflexivity**

$\forall A, |A| = |A|$.

bijection : $id_A$

**symmetry**

$|A| = |B| \to |B| = |A|$

bijection exists. Find inverse. inverse also a bijection, shows that $|B| = |A|$.

**transitivity**

$|A| = |B|$ and $|B| = |c|$, $|A| = |c|$

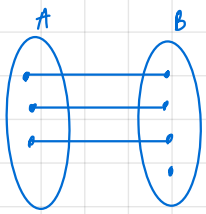
bijection $f: A \to B$, $g: B \to C$ exists. $(g \circ f)^{-1} = f^{-1} \circ g^{-1}$. $(g \circ f)^{-1}$ exists, so bijection $g \circ f$ exists. $|A| = |c|$.

3) pigeonhole & dual pigeonhole principle

**pigeonhole principle**
Theorem 10.1.2

A and B are finite.

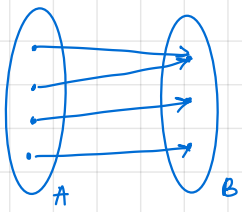
$|A| \leq |B| \longleftrightarrow$ injection $A \to B$.




**dual pigeonhole principle**
Theorem 10.1.3

A and B are finite.

Then $|A| \geq |B| \longleftrightarrow$ there is a surjection $A \to B$.




---

## ② Countability of sets

**Cantor's definition of countability** : a set is countable $\longleftrightarrow$ finite or same cardinality as $\mathbb{Z}_{\geq 0}$

**Lemma 10.3.7** : An infinite set B is countable $\longleftrightarrow$ there is a sequence $b_0, b_1 \ldots$ in which every element of B appears (w/ addtl stuff or repetition).

**Key properties of countable sets**

1. any subset A of a countable set B is countable (proposition 10.3.5)

↳ A → finite ⇒ countable
→ infinite → B has sequence → take away elements in B not A → left w/ sequence of A ⇒ countable

2. Every infinite set has a countable (infinite) subset (proposition 10.3.6)

↳ for every $n \in \mathbb{Z}_{\geq 0}$ choose $b_n \in B$. $b_n \neq b_j$ since set is infinite. So this subset is countable.

## ③ Cantor's diagonal argument

i) intuition — For sets that have elements that themselves have infinite elements, and their equality is judged by whether they 'contain' all the same elements.

1. proof by contradiction. suppose $A$ is countable. Then $A$ has a sequence that contains all elements of $A$.

2. enumerate them

$$\begin{matrix} a_{00} & a_{01} & a_{02} & \dots \\ a_{10} & a_{11} & a_{12} & \dots \\ a_{20} & a_{21} & a_{22} & \dots \end{matrix}$$

3. Find some $b$ s.t. $b_0 \neq a_{00}$, $b_1 \neq a_{11}$, $b_2 \neq a_{22}$ ... → then $b$ is not in the sequence since it can never equal any $a_{ij}$. This contradicts 1.

ii) countable & uncountable sets

$\mathbb{Z}$
$\mathbb{Q}$

$\mathbb{R}$
$P(A)$, where $A$ is countable infinite (Theorem 10.4.3)
$\mathbb{C}$

---

## ④ countability of operated sets

unions

1. countable infinite ∪ finite → finite sequence first, append after (T8Q1)
↳ $B$ is infinite, $C$ is finite. bijection $B \cup C \to B$ exists. so $|B| = |B \cup C|$ (T8Q3)
($|B \cup C|$: infinite ∪ finite)

2. countable infinite ∪ countable infinite → alternating sequence (proposition 10.4.1)

Let $A, B$ be countable infinite sets. Then $A \cup B$ is countable.

3. finite ∪ finite → finite

cartesian product: $\mathbb{Z}_{710} \times \mathbb{Z}_{710}$ is countable (Theorem 10.4.2)
↳ infinite unions

$\bigcup_{i \in \mathbb{Z}_0} S_i = S_0 \cup S_1 \cup S_2 \dots S_i$ is countable if $S_i$ is countable infinite. (T8Q4)

↳ can be written as

$$\begin{matrix} S_0 & a_{01} & a_{02} & \dots \\ S_1 & a_{11} & a_{12} & \dots \\ \vdots & a_{21} & a_{22} & \dots \\ S_i & \vdots & & \end{matrix}$$

, and each $a_{ij}$ can be mapped to $x \in \mathbb{Z}_{710} \times \mathbb{Z}_{710}$, as a bijection.

$|\mathbb{Z}_{710} \times \mathbb{Z}_{710}| = |\mathbb{Z}_{710}|$. countable.

# 10. Counting and probability

## ① Basics of counting

### i) no. of elements in a list & application

**Theorem 9.1.1 The Number of Elements in a List**

If $m$ and $n$ are integers and $m \le n$, then there are
$$n - m + 1$$
integers from $m$ to $n$ inclusive.

Q: from 100 to 999, how many divisible by 5?

100 101 102 103 104 105 106 107 108 109 110 ... 994 995 996 997 998 999

$5\times 20$ $5\times 21$ $5\times 22$ $5\times 199$

↳ 20, 21, 22 ... 199

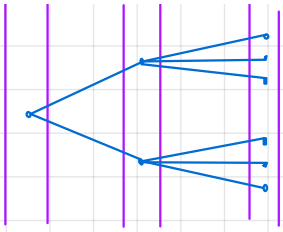
creating a subset ⇒ so there are $199 - 20 + 1 = 180$

### 2) rules of counting

(multiplication rule)

**Theorem 9.2.1 The Multiplication/Product Rule**

If an operation consists of $k$ steps and
the first step can be performed in $n_1$ ways,
the second step can be performed in $n_2$ ways
(regardless of how the first step was performed),
:
the $k^{th}$ step can be performed in $n_k$ ways
(regardless of how the preceding steps were performed),
Then the entire operation can be performed in
$n_1 \times n_2 \times n_3 \times \dots \times n_k$ ways.



only holds if each step on tree has same no. of ways for each branch

↳ be careful if there are restriction

(addition rule)

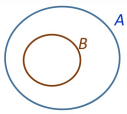
**Theorem 9.3.1 The Addition/Sum Rule**

Suppose a finite set $A$ equals the union of $k$ distinct mutually disjoint subsets $A_1, A_2, \dots, A_k$. Then
$$|A| = |A_1| + |A_2| + \dots + |A_k|.$$

(difference rule)

**Theorem 9.3.2 The Difference Rule**

If $A$ is a finite set and $B \subset A$, then
$$|A \setminus B| = |A| - |B|.$$




The difference rule holds for the following reason:
If $B \subseteq A$, then the two sets $B$ and $A \setminus B$ have no elements in common and $B \cup (A \setminus B) = A$.

Hence, by the addition rule,
$$|B| + |A \setminus B| = |A|.$$

Subtracting $|B|$ from both sides gives the equation
$$|A \setminus B| = |A| - |B|.$$

(inclusion / exclusion rule)

**Theorem 9.3.3 The Inclusion/Exclusion Rule for 2 or 3 Sets**

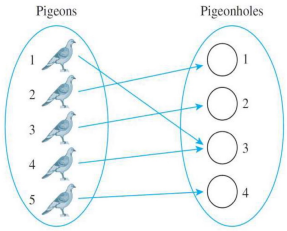
If $A$, $B$, and $C$ are any finite sets, then
$$|A \cup B| = |A| + |B| - |A \cap B|$$
and
$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$$

## ② pigeonhole principle

(PHP)

**Pigeonhole Principle (PHP)**

A function from one finite set to a smaller finite set cannot be one-to-one: There must be at least 2 elements in the domain that have the same image in the co-domain.

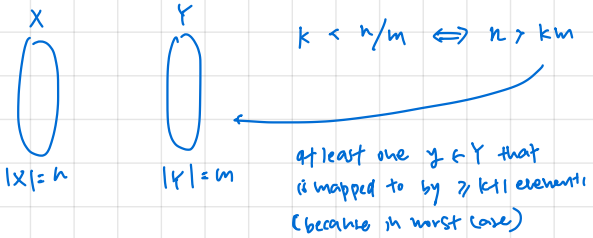



ie. not injective

(generalised PHP)

**Generalized Pigeonhole Principle**

For any function $f$ from a finite set $X$ with $n$ elements to a finite set $Y$ with $m$ elements and for any positive integer $k$, if $k < n/m$, then there is some $y \in Y$ such that $y$ is the image of at least $k + 1$ distinct elements of $X$.



## ③ counting subsets of a set : PnC

### 1) permutations of a set (no repetition, order matters)

**Theorem 9.2.2 Permutations**

The number of permutations of a set with $n$ ($n \geq 1$) elements is **$n!$**


### 2) r combinations & permutations

r permutations → remove permutations r! → r combinations

**Theorem 9.2.3 $r$-permutations from a set of $n$ elements**

If $n$ and $r$ are integers and $1 \leq r \leq n$, then the number of $r$-permutations of a set of $n$ elements is given by the formula

$$P(n, r) = n(n-1)(n-2)\ldots(n-r+1)$$ first version

or, equivalently,

$$P(n, r) = \frac{n!}{(n-r)!}$$ second version

but permuted

**Theorem 9.5.1 Formula for $\binom{n}{r}$**

The number of subsets of size $r$ (or $r$-combinations) that can be chosen from a set of $n$ elements, $\binom{n}{r}$, is given by the formula

$$\binom{n}{r} = \frac{P(n,r)}{r!}$$

Recall that $P(n,r) = \frac{n!}{(n-r)!}$

or, equivalently,

$$\binom{n}{r} = \frac{n!}{r!(n-r)!}$$

where $n$ and $r$ are non-negative integers with $r \leq n$.

### 3) permutation of a set w/ indistinguishable objects

**Theorem 9.5.2 Permutations with Sets of Indistinguishable Objects**

Suppose a collection consists of $n$ objects of which

$n_1$ are of type 1 and are indistinguishable from each other
$n_2$ are of type 2 and are indistinguishable from each other
:
$n_k$ are of type $k$ and are indistinguishable from each other

and suppose that $n_1 + n_2 + \ldots + n_k = n$. Then the number of distinguishable permutations of the $n$ objects is

$$\binom{n}{n_1}\binom{n-n_1}{n_2}\binom{n-n_1-n_2}{n_3}\cdots\binom{n-n_1-n_2-\cdots-n_{k-1}}{n_k} = \frac{n!}{n_1!n_2!n_3!\cdots n_k!}$$

intuition: choosing $n_1, n_2 \ldots n_k$ positions to put these indistinguishable objects in $n$ positions

another intuition: deprermuting double counts

### 4) multisets of size r as partitions

**Definition: Multiset**

An ***r*-combination with repetition allowed**, or **multiset of size *r***, chosen from a set $X$ of $n$ elements is an unordered selection of elements taken from $X$ with repetition allowed.

If $X = \{x_1, x_2, \cdots, x_n\}$, we write an $r$-combination with repetition allowed as $[x_{i_1}, x_{i_2}, \cdots, x_{i_r}]$ where each $x_{i_j}$ is in $X$ and some of the $x_{i_j}$ may equal each other.

**Theorem 9.6.1 Number of $r$-combinations with Repetition Allowed**

The number of $r$-combination with repetition allowed (multisets of size $r$) that can be selected from a set of $n$ elements is:

$$\binom{r+n-1}{r}$$

This equals the number of ways $r$ objects can be selected from $n$ categories of objects with repetitions allowed.



Intuition: r to be chosen w/ repetition allowed ⇒ categorising ⇒ need n-1 partitions

↳ total: r + (n-1) positions

choose r positions for objects

choose n-1 positions for partitions

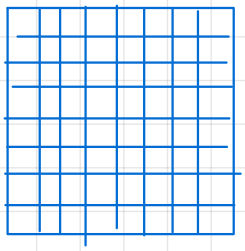
↳ = ↵

| | Category 1 | Category 2 | Category 3 | Category 4 |
|---|---|---|---|---|
| [1, 1, 1]: | xxx | \| | \| | \| |
| [1, 3, 4]: | x | \| | \| x | \| x |
| [2, 4, 4]: | | \| x | \| | \| xx |

## ④ The rook problem

in how many ways can one arrange $k$ rooks on an $m \times n$ board so that they do not attack each other?

$k \leq m, n \Rightarrow$ $k$ rooks must be different column AND different row



because there are $m \times n$ choices initially → place one, then $m \times n - 1$ ... permutation formula. But better to think of it as using multiplication rule.

total no. of ways $= m \times n \, P_k$

Application : modelling scenario where there are $m$ category 1, $n$ category 2, and any 2 objects cannot be same $m$ or $n$.

## ⑤ Theorems & combinatorial proofs

1) combinatorial proofs : "intuitive" proofs
   ↳ uses counting as the basis of proofs
   ↳ bijective proofs (eg. in cardinality)
   ↳ double counting : use two equivalent ways to count to derive an identity

2) identities

Pascal's formula

**Theorem 9.7.1 Pascal's Formula**

Let $n$ and $r$ be positive integers, $r \le n$. Then

$$\binom{n+1}{r} = \binom{n}{r-1} + \binom{n}{r}$$

($x$ chosen / $x$ not chosen)

intuition : mutually exclusive ways of choosing - chosen set either includes some element $x$. So you sum these outcomes.

Binomial theorem

$\binom{n}{r}$ is called binomial coefficient

**Theorem 9.7.2 Binomial Theorem**

Given any real numbers $a$ and $b$ and any non-negative integer $n$,

$$(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^{n-k} b^k$$
$$= a^n + \binom{n}{1}a^{n-1}b^1 + \binom{n}{2}a^{n-2}b^2 + \cdots + \binom{n}{n-1}a^1 b^{n-1} + b^n$$

34

intuition: in expanding $(a+b)^n$, there are essentially n 'groups' of $(a+b)$, and from these you choose k b's and n-k a's, to get the powers. Since there are diff ways to choose, you combinatorially sum them.

using to simplify a sum:

$$\sum_{k=0}^{n} \binom{n}{k} x^k = (1+x)^n$$

Sum of choices

$$\binom{n}{0} + \binom{n}{1} + \binom{n}{2} + \cdots + \binom{n}{n} = 2^n$$

(left: equivalent to choosing subsets in powerset; right: size of powerset)

Lecture 12 Example 8 symmetry

$$\binom{n}{r} = \binom{n}{n-r}$$

## ⑥ Events and probability

1) Relative frequency interpretation

**Equally Likely Probability Formula**

If $S$ is a finite sample space in which all outcomes are equally likely and $E$ is an event in $S$, then the **probability** of $E$, denoted $P(E)$, is

$$P(E) = \frac{\text{The number of outcomes in } E}{\text{The total number of outcomes in } S} = \frac{|E|}{|S|}$$

9

**Formula for the Probability of the Complement of an Event**

If $S$ is a finite sample space and $A$ is an event in $S$, then

$$P(\bar{A}) = 1 - P(A)$$

2) probability axioms

**Probability Axioms**

Let $S$ be a sample space. A probability function $P$ from the set of all events in $S$ to the set of real numbers satisfies the following axioms: For all events $A$ and $B$ in $S$,

1. $0 \le P(A) \le 1$
2. $P(\varnothing) = 0$ and $P(S) = 1$
3. If $A$ and $B$ are disjoint events ($A \cap B = \varnothing$), then $P(A \cup B) = P(A) + P(B)$

3) expected value

**Definition: Expected Value**

Suppose the possible outcomes of an experiment, or random process, are real numbers $a_1, a_2, a_3, \cdots, a_n$ which occur with probabilities $p_1, p_2, p_3, \cdots, p_n$ respectively. The **expected value** of the process is

$$\sum_{k=1}^{n} a_k p_k = a_1 p_1 + a_2 p_2 + a_3 p_3 + \cdots + a_n p_n$$

probability-weighted average

The expected value of the sum of random variables is equal to the sum of their individual expected values, regardless of whether they are independent.

For random variables $X$ and $Y$ (which may be dependent),

$$E[X + Y] = E[X] + E[Y]$$

More generally, for random variables $X_1, X_2, \cdots, X_n$ and constants $c_1, c_2, \cdots, c_n$,

$$E\left[\sum_{i=1}^{n} c_i \cdot X_i\right] = \sum_{i=1}^{n} (c_i \cdot E[X_i])$$

48

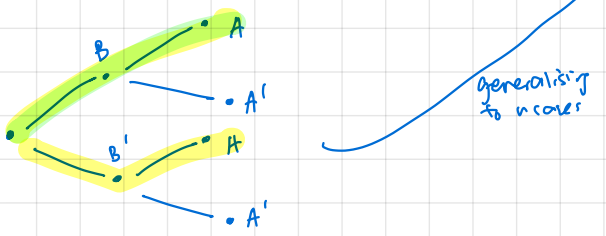
4) conditional probability

**Definition: Conditional Probability**

Let $A$ and $B$ be events in a sample space $S$. If $P(A) \neq 0$, then the **conditional probability of $B$ given $A$**, denoted $P(B|A)$, is

$$P(B|A) = \frac{P(A \cap B)}{P(A)} \qquad 9.9.1$$

intuition: shrinking sample space since A must have occured




generalising to n cases

5) Bayes' theorem

**Theorem 9.9.1 Bayes' Theorem**

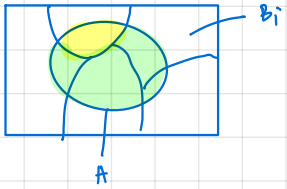
Suppose that a sample space $S$ is a union of mutually disjoint events $B_1, B_2, B_3, \ldots, B_n$.

Suppose $A$ is an event in $S$, and suppose $A$ and all the $B_i$ have non-zero probabilities.

If $k$ is an integer with $1 \leq k \leq n$, then

$$P(B_k|A) = \frac{P(A|B_k) \cdot P(B_k)}{P(A|B_1) \cdot P(B_1) + P(A|B_2) \cdot P(B_2) + \cdots + P(A|B_n) \cdot P(B_n)}$$

$= P(A \cap B_k)$, where B occurs first then A, given B has occured




intuition: given that A has occured and it can occur in k ways, what is the probability that it happened through that one way?

7) independent events

1) independence

**Definition: Independent Events**

If $A$ and $B$ are events in a sample space $S$, then $A$ and $B$ are **independent**, if and only if,

$$P(A \cap B) = P(A) \cdot P(B)$$

2) independence of sets of events

**Definition: Pairwise Independent and Mutually Independent**

Let $A$, $B$ and $C$ be events in a sample space $S$. $A$, $B$ and $C$ are **pairwise independent**, if and only if, they satisfy conditions 1 – 3 below. They are **mutually independent** if, and only if, they satisfy all four conditions below.

1. $P(A \cap B) = P(A) \cdot P(B)$
2. $P(A \cap C) = P(A) \cdot P(C)$
3. $P(B \cap C) = P(B) \cdot P(C)$
4. $P(A \cap B \cap C) = P(A) \cdot P(B) \cdot P(C)$

1–3: pairwise; 1–4: mutual

79

# 11 - Graphs

## ① Graphs

### 1) Basic terminology

↳ a graph is defined by its vertices and edges. $G = (V, E)$

$\{v_1, v_2, \dots\}$ set of vertices $\quad \{e_1, e_2 \dots\}$ set of edges

representation depends on direction

undirected graph

**Definition: Undirected Graph**

An undirected **graph** $G$ consists of 2 finite sets: a nonempty set $V$ of **vertices** and a set $E$ of **edges**, where each (undirected) edge is associated with a set consisting of either one or two vertices called its **endpoints**.

An edge is said to **connect** its endpoints; two vertices that are connected by an edge are called **adjacent vertices**; and a vertex that is an endpoint of a loop is said to be **adjacent to itself**.

An edge is said to be **incident on** each of its endpoints, and two edges incident on the same endpoint are called **adjacent edges**.

We write $e = \{v, w\}$ for an undirected edge $e$ incident on vertices $v$ and $w$.

directed graph

**Definition: Directed Graph**

A **directed graph**, or **digraph**, $G$, consists of 2 finite sets: a nonempty set $V$ of **vertices** and a set $E$ of **directed edges**, where each (directed) edge is associated with an ordered pair of vertices called its **endpoints**.

We write $e = (v, w)$ for a directed edge $e$ from vertex $v$ to vertex $w$.

### 2) Degree

Handshake theorem

↳ because edge is shared by two nodes → degree = 2 × no. of edges

**Definition: Degree of a Vertex and Total Degree of an Undirected Graph**

Let $G$ be a undirected graph and $v$ a vertex of $G$. The **degree** of $v$, denoted **deg($v$)**, equals the number of edges that are incident on $v$, with an edge that is a loop counted twice.

The **total degree of $G$** is the sum of the degrees of all the vertices of $G$.

**Theorem 10.1.1 The Handshake Theorem**

If $G$ is any graph, then the sum of the degrees of all the vertices of $G$ equals twice the number of edges of $G$. Specifically, if the vertices of $G$ are $v_1, v_2, \dots, v_n$, where $n \geq 0$, then

The total degree of $G = \deg(v_1) + \deg(v_2) + \dots + \deg(v_n)$

$= 2 \times$ (the number of edges of $G$).

Total degree is even

**Corollary 10.1.2**

The total degree of a graph is even.

**Proposition 10.1.3**

In any graph there are an even number of vertices of odd degree.

### 3) Types of graphs

$2^{\binom{n}{2}}$ simple graphs. ← intuition: $n$ vertices, $\binom{n}{2}$ edges to include or not.

Simple graph: no parallel edges or loops

**Definition: Simple Graph**

A **simple graph** is an undirected graph that does not have any loops or parallel edges. (That is, there is at most one edge between each pair of distinct vertices.)

Simple graph

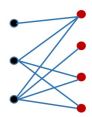
Non simple graph

Non simple graph

Bipartite graphs

**Definition: Bipartite Graph**

A **bipartite graph** (or bigraph) is a simple graph whose vertices can be divided into two disjoint sets $U$ and $V$ such that every edge connects a vertex in $U$ to one in $V$.

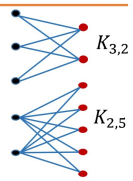


Bipartite graph

**Definition: Complete Bipartite Graph**

A **complete bipartite graph** is a bipartite graph on two disjoint sets $U$ and $V$ such that every vertex in $U$ connects to every vertex in $V$.

If $|U| = m$ and $|V| = n$, the complete bipartite graph is denoted as $K_{m,n}$.

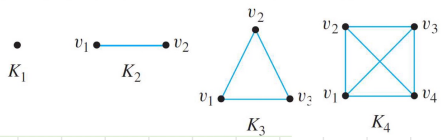



no. of edges = $m \times n$

Complete graph: fully connected

**Definition: Complete Graph**

A **complete graph** on ***n* vertices**, $n > 0$, denoted $K_n$, is a simple graph with $n$ vertices and exactly one edge connecting each pair of distinct vertices .




Draw $K_5$.

edges in complete graph = ${}^nC_2$

Subgraphs

**Definition: Subgraph of a Graph**

A graph $H$ is said to be a **subgraph** of graph $G$ if and only if every vertex in $H$ is also a vertex in $G$, every edge in $H$ is also an edge in $G$, and every edge in $H$ has the same endpoints as it has in $G$.

$H$ subgraph of $G \Leftrightarrow \forall v \in H, e \in H, v \in G$ & $e \in G$

↳ same idea as subset

## ② trails, paths and circuits

walk

A **walk from *v* to *w*** is a finite alternating sequence of adjacent vertices and edges of *G*. Thus a walk has the form

$$v_0 e_1 v_1 e_2 \dots v_{n-1} e_n v_n,$$

The **trivial walk** from *v* to *v* consists of the single vertex *v*.
A **closed walk** is a walk that starts and ends at the same vertex.

trail: no repeated edge

A **trail from *v* to *w*** is a walk from *v* to *w* that does not contain a repeated edge.

path: no repeated edge or vertices

A **path from *v* to *w*** is a trail that does not contain a repeated vertex.

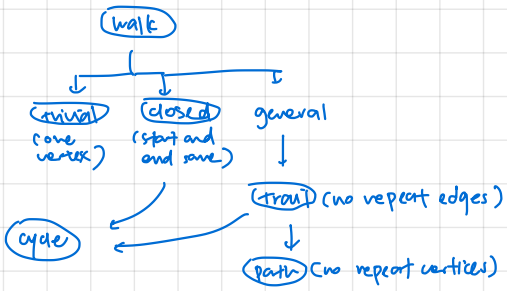
circuit/cycle: closed walk, no repeated edges (not necessarily all edges)

**Circuit** (or **cycle**): Let $n \in \mathbb{Z}_{\geq 3}$. An undirected graph $G(V, E)$ where $V = \{x_1, x_2, \cdots, x_n\}$ and $E = \{\{x_1, x_2\}, \{x_2, x_3\}, \cdots, \{x_{n-1}, x_n\}, \{x_n, x_1\}\}$ is called **a circuit/cycle**.

(A cycle is a closed walk that does not contain a repeated edge.)

A **simple circuit** (or **simple cycle**) is a circuit that does not have any other repeated vertex except the first and last.

An undirected graph is **cyclic** if it contains a loop or a cycle; otherwise, it is **acyclic**.




## ③ connectedness

A graph is connected if it is possible to travel from any vertex to any other vertex along a sequence of adjacent edges of the graph.

**Definition: Connectedness**

**Two vertices** *v* and *w* of a graph *G*=(*V*,*E*) are **connected** if and only if there is a walk from *v* to *w*.

**The graph *G* is connected** if and only if given *any* two vertices *v* and *w* in *G*, there is a walk from *v* to *w*. Symbolically,

*G* is connected iff ∀ vertices *v*, *w* ∈*V*, ∃ a walk from *v* to *w*.

proof: direct

show that any two vertices are connected or some share a common consin vertex

by contradiction

↳ suppose any two not connected

↳ (neighborhood) summed > n vertices directly connected to a vertex

by PHP

↳ total max no. of edges of vertices by definition

↳ total max no. possible (each can connect to n-1)

⇓

PHP

connected component: largest connected subgraph

**Definition: Connected Component**

A graph *H* is a **connected component** of a graph *G* if and only if

1. The graph *H* is a subgraph of *G*;
2. The graph *H* is connected; and
3. No connected subgraph of *G* has *H* as a subgraph and contains vertices or edges that are not in *H*.

circuits and connectedness

**Lemma 10.5.3**

If *G* is any connected graph, *C* is any circuit in *G*, and one of the edges of *C* is removed from *G*, then the graph that remains is still connected.

Essentially, the reason why Lemma 10.5.3 is true is that any two vertices in a circuit are connected by 2 distinct paths. It is possible to draw the graph so that one of these goes "clockwise" and the other goes "counter-clockwise" around the circuit.

## ④ Euler and hamiltonian circuits

1) Eulerian: traversing every edge exactly once, every vertex at least once

Eulerian circuits: every vertex at least once, every edge once, must return to same spot

**Definition: Euler Circuit**

Let *G* be a graph. An **Euler circuit** for *G* is a circuit that contains every vertex and traverses every edge of *G* exactly once.

→ **Theorem 10.2.4**

A graph *G* has an Euler circuit if and only if *G* is connected and every vertex of *G* has positive even degree.

Eulerian graph

**Definition: Eulerian Graph**

An **Eulerian graph** is a graph that contains an Euler circuit.

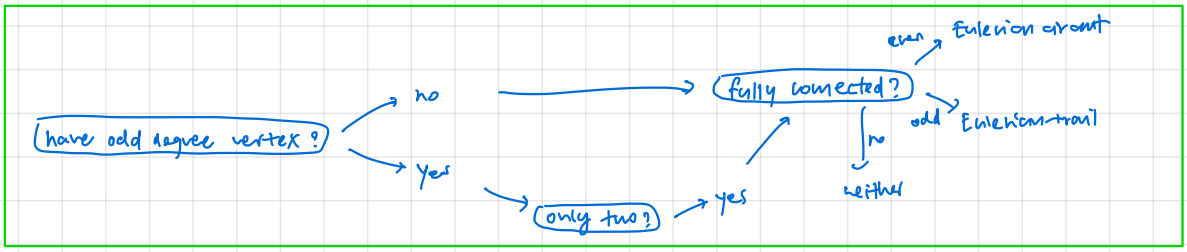
Eulerian trail: traverse every edge once, every vertex at least once

**Definition: Euler Trail**

Let *G* be a graph, and let *v* and *w* be two distinct vertices of *G*. An **Euler trail/path from *v* to *w*** is a sequence of adjacent edges and vertices that starts at *v*, ends at *w*, passes through every vertex of *G* at least once, and traverses every edge of *G* exactly once.

→ **Corollary 10.2.5**

Let *G* be a graph, and let *v* and *w* be two distinct vertices of *G*. There is an Euler trail from *v* to *w* if and only if *G* is connected, *v* and *w* have odd degree, and all other vertices of *G* have positive even degree.




2) Hamiltonian: travelling every vertex exactly once

no ways to prove that a graph contains a Hamiltonian circuit other than through trial and error

⇓

but can prove that it does not contain

### Definition: Hamiltonian Circuit

Given a graph $G$, a **Hamiltonian circuit** for $G$ is a simple circuit that includes every vertex of $G$. (That is, every vertex appears exactly once, except for the first and the last, which are the same.)

### Definition: Hamiltonian Graph

A **Hamiltonian graph** (also called **Hamilton graph**) is a graph that contains a Hamiltonian circuit.

### Proposition 10.2.6

If a graph $G$ has a Hamiltonian circuit, then $G$ has a subgraph $H$ with the following properties:

1. $H$ contains every vertex of $G$.
2. $H$ is connected.
3. $H$ has the same number of edges as vertices.
4. Every vertex of $H$ has degree 2.

↳ find all possible subgraphs & test

## 5) matrix representation of graphs

### 1) matrix definitions

### Definition: Matrix Multiplication

Let $\mathbf{A} = (a_{ij})$ be an $m \times k$ matrix and $\mathbf{B} = (b_{ij})$ an $k \times n$ matrix with real entries. The (matrix) product of **A** times **B**, denoted **AB**, is that matrix $(c_{ij})$ defined as follows:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1k} \\ a_{21} & a_{22} & \cdots & a_{2k} \\ \vdots & \vdots & & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{ik} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mk} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1j} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2j} & \cdots & b_{2n} \\ & & & \cdot & & \\ & & & \cdot & & \\ & & & \cdot & & \\ b_{k1} & b_{k2} & \cdots & b_{kj} & \cdots & b_{kn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1j} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2j} & \cdots & c_{2n} \\ \vdots & \vdots & & \vdots & & \vdots \\ c_{i1} & c_{i2} & \cdots & c_{ij} & \cdots & c_{in} \\ \vdots & \vdots & & \vdots & & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mj} & \cdots & c_{mn} \end{bmatrix}$$

where

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{ik}b_{kj} = \sum_{r=1}^{k} a_{ir}b_{rj}.$$

for all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$.

### Definition: Identity Matrix

For each positive integer $n$, the $\boldsymbol{n \times n}$ **identity matrix**, denoted $I_n = (\delta_{ij})$ or just **I** (if the size of the matrix is obvious from context), is the $n \times n$ matrix in which all the entries in the main diagonal are 1's and all other entries are 0's. In other words,

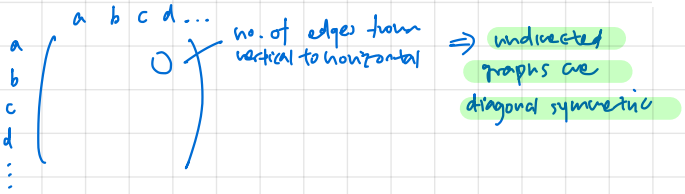
$$\delta_{ij} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases} \qquad \text{for all } i, j = 1, 2, \ldots, n.$$

### 2) representing graphs in matrices

### Definition: Adjacency Matrix of a Directed Graph

Let $G$ be a directed graph with ordered vertices $v_1, v_2, \ldots v_n$. The **adjacency matrix of *G*** is the $n \times n$ matrix $\mathbf{A} = (a_{ij})$ over the set of non-negative integers such that

$a_{ij}$ = the number of arrows from $v_i$ to $v_j$ for all $i, j = 1, 2, \ldots, n$.




### 3) counting walks of length N

### Theorem 10.3.2

If $G$ is a graph with vertices $v_1, v_2, \ldots, v_m$ and **A** is the adjacency matrix of $G$, then for each positive integer $n$ and for all integers $i, j = 1, 2, \ldots, m$,
the $ij$-th entry of $\mathbf{A}^n$ = the number of walks of length $n$ from $v_i$ to $v_j$.

## 6) isomorphic graphs and complements

### 1) isomorphic: same graph but orientated differently


checking for isomorphism

1. $|V_1| = |V_2|$ $|E_1| = |E_2|$ (bijection can exist)
2. can 'rearrange' orientation of nodes to get graph, without disconnecting / connecting anything

Two graphs $G$ and $G'$ that are the same except for the labeling of their vertices and edges are called *isomorphic*. In other words, there exists matching between the vertices such that two vertices are connected by an edge in $G$ if and only if corresponding vertices are connected by an edge in $G'$.

### Definition: Isomorphic Graph

Let $G = (V_G, E_G)$ and $G' = (V_{G'}, E_{G'})$ be two graphs.

***G* is isomorphic to *G'***, denoted $G \cong G'$, if and only if there exist bijections $g: V_G \to V_{G'}$ and $h: E_G \to E_{G'}$ that preserve the edge-endpoint functions of $G$ and $G'$ in the sense that for all $v \in V_G$ and $e \in E_G$,

$v$ is an endpoint of $e \Leftrightarrow g(v)$ is an endpoint of $h(e)$.

### Alternative definition

Let $G = (V_G, E_G)$ and $G' = (V_{G'}, E_{G'})$ be two graphs.

***G* is isomorphic to *G'*** if and only if there exists a permutation $\pi: V_G \to V_{G'}$ such that $\{u, v\} \in E_G \Leftrightarrow \{\pi(u), \pi(v)\} \in E_{G'}$.

### Theorem 10.4.1 Graph Isomorphism is an Equivalence Relation

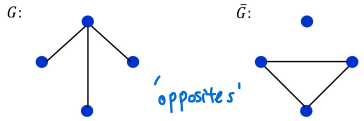
Let $S$ be a set of graphs and let $\cong$ be the relation of graph isomorphism on $S$. Then $\cong$ is an equivalence relation on $S$.

1. reflexive: identity function (bijection)
2. symmetric: bijection → inverse also bijection
3. transitive: composed bijection still bijection

## 2) Complements — union of graph & complement ⇒ complete graph

**Definition 1.** If $G$ is a simple graph, the *complement* of $G$, denoted $\bar{G}$, is obtained as follows: the vertex set of $\bar{G}$ is identical to the vertex set of $G$. However, two distinct vertices $v$ and $w$ of $\bar{G}$ are connected by an edge if and only if $v$ and $w$ are not connected by an edge in $G$.

The figure below shows a graph $G$ and its complement $\bar{G}$.




A graph $G$ and its complement $\bar{G}$.

**Definition 2.** A *self-complementary* graph is isomorphic with its complement.

## 7) planar graphs

**Definition: Planar Graph**

A **planar graph** is a graph that can be drawn on a (two-dimensional) plane without edges crossing.

**check if planar graph** — complete graphs > 4 non-planar

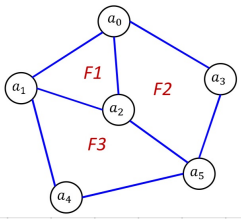
1. try to rearrange graph (find isomorphic) so that no edges cross

**Euler's formula for counting regions**

**Euler's Formula**

For a connected planar simple graph $G = (V, E)$ with $e = |E|$ and $v = |V|$, if we let $f$ be the number of faces, then

$$f = e - v + 2$$




outer face counts

$e = 8$
$v = 6$
$f = 8 - 6 + 2 = 4$

## 8) Modelling graph problems

1. nodes → meaning
2. edges → meaning
3. some algorithm/operation → meaning ⇒ impact on properties
   - (total) degree
   - (total) weight
   - (total) no. of edges

**colouring problems**

1. nodes → entity
2. edge → some r/s some cannot have in each other
3. algorithm → vertex colouring ⇒ four groups needed

## Notes:

1. Each vertex can be connected to 1 ... n-1 vertices - OR 0 ... n-2 vertices, because n-1 case connected to all others.
2. A vertex can have n-1 edges iff there is no vertex w/ 0 edges

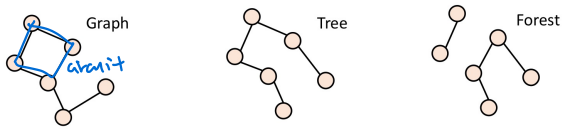
# 12. Trees

## ① Trees (special forms of undirected graphs)

### 1) trees

**Definition: Tree**

(The graph is assumed to be undirected here.)
A **graph** is said to be **circuit-free** if and only if it has no circuits.
A graph is called a **tree** if and only if it is circuit-free and connected.
A **trivial tree** is a graph that consists of a single vertex.
A graph is called a **forest** if and only if it is circuit-free and not connected.




4

### 2) vertices in a tree

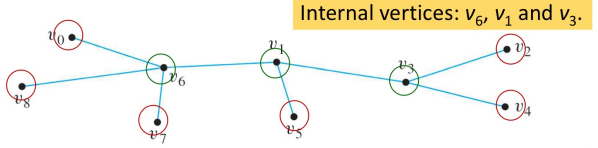
degree: $=1 \rightarrow$ leaf; $>1 \rightarrow$ internal

**Definitions: Terminal vertex (leaf) and internal vertex**

Let $T$ be a tree. If $T$ has only one or two vertices, then each is called a **terminal vertex** (or **leaf**). If $T$ has at least three vertices, then a vertex of degree 1 in $T$ is called a **terminal vertex** (or **leaf**), and a vertex of degree greater than 1 in $T$ is called an **internal vertex**.

Example: Find all terminal vertices (leaves) and all internal vertices in the following tree:

Leaves: $v_0$, $v_2$, $v_4$, $v_5$, $v_7$ and $v_8$.
Internal vertices: $v_6$, $v_1$ and $v_3$.




9

### 3) properties of trees

non-trivial trees have $\geq 1$ leaf

1. **Lemma 10.5.1**

Any non-trivial tree has at least one vertex of degree 1.

**Proof:** Let $T$ be a particular but arbitrarily chosen non-trivial tree.
Step 1: Pick a vertex $v$ of $T$ and let $e$ be an edge incident on $v$.
Step 2: While deg($v$) > 1, repeat steps 2a, 2b and 2c:
2a: Choose $e'$ to be an edge incident on $v$ such that $e' \neq e$.
2b: Let $v'$ be the vertex at the other end of $e'$ from $v$.
2c: Let $e = e'$ *and* $v = v'$.
The algorithm must eventually terminate because the set of vertices of the tree $T$ is finite and $T$ is circuit-free. When it does, a vertex $v$ of degree 1 will have been found.

7

3. Any non-trivial tree has at least two vertices of degree 1.

Proof:

1. Let T be a non-trivial tree.
2. Let S be the set of all paths in T.
3. Let P be the largest path in S (though not necessarily unique)
   3-1 Since T is non-trivial and trees are n-1 edges in T, a path can be of length 1 to n-1
   3-2 So there is a path of maximum length
4. The initial and final vertices in P must have degree 1
   4-1 suppose terminal vertex has degree 2.
   4-2 case 1: the other vertex joined by the other edge is not in P. Then adding it in would increase |P|. This contradicts that P is greatest.
   4-3 case 2: the other vertex is in P. Then it is cyclic. Then T is not a tree. This contradicts.

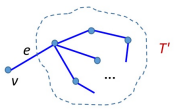
2. **Theorem 10.5.2**

Any tree with $n$ vertices ($n > 0$) has $n - 1$ edges.

(intuition) Because trees are no circuits, can be rearranged to form a straight line

**Proof:** By mathematical induction.
Let the property $P(n)$ be "any tree with $n$ vertices has $n - 1$ edges".
$P(1)$: Let $T$ be a tree with one vertex. Then $T$ has no edges.
So $P(1)$ is true.
Show that for all integers $k \geq 1$, if $P(k)$ is true then $P(k+1)$ is true.
Suppose $P(k)$ is true.

1. Let $T$ be a particular but arbitrarily chosen tree with $k + 1$ vertices.
2. Since $k$ is positive, $(k + 1) \geq 2$, and so $T$ has more than one vertex.
3. Hence, by Lemma 10.5.1, $T$ has a vertex $v$ of degree 1, and has at least another vertex in $T$ besides $v$.
4. Thus, there is an edge $e$ connecting $v$ to the rest of $T$.
5. Define a subgraph $T'$ of $T$ so that $V_{T'} = V_T - \{v\}$ and $E_{T'} = E_T - \{e\}$.
   5.1 The number of vertices of $T'$ is $(k + 1) - 1 = k$.
   5.2 $T'$ is circuit-free.
   5.3 $T'$ is connected.
6. Hence by definition, $T'$ is a tree.
7. Since $T'$ has $k$ vertices, by inductive hypothesis, number of edges of $T'$ = (number of vertices of $T'$) $- 1 = k - 1$.
8. But number of edges of $T$ = (number of edges of $T'$) $+ 1 = k$.
9. Hence $P(k+1)$ is true.




11

4) proving a graph is a tree ie. proving it is circuit free given connected

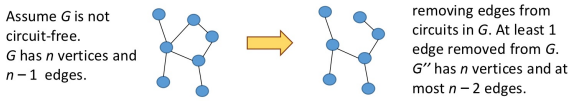
**Theorem 10.5.4**

If $G$ is a connected graph with $n$ vertices and $n-1$ edges, then $G$ is a tree.

no circuits

**Proof:**

1. Suppose $G$ is a particular but arbitrarily chosen graph that is connected and has $n$ vertices and $n-1$ edges.
2. Since $G$ is connected, it suffices to show that $G$ is circuit-free.
3. Suppose $G$ is not circuit free
   - 3.1 Let $C$ be the circuit in $G$.
   - 3.2 By Lemma 10.5.3, an edge of $C$ can be removed from $G$ to obtain a graph $G'$ that is connected.
   - 3.3 If $G'$ has a circuit, then repeat this process: Remove an edge of the circuit from $G'$ to form a new connected graph.
   - 3.4 Continue the process of removing edges from the circuits until eventually a graph $G''$ is obtained that is connected and is circuit-free.
   - 3.5 By definition, $G''$ is a tree.
   - 3.6 Since no vertices were removed from $G$ to form $G''$, $G''$ has $n$ vertices.
   - 3.7 Thus, by Theorem 10.5.2, $G''$ has $n-1$ edges.
   - 3.8 But the supposition that $G$ has a circuit implies that at least one edge of $G$ is removed to form $G''$.
   - 3.9 Hence $G''$ has no more than $(n-1)-1 = n-2$ edges, which contradicts its having $n-1$ edges.
   - 3.10 So the supposition is false.
4. Hence $G$ is circuit-free, and therefore $G$ is a tree.




16

5) Finding no. of non-isomorphic trees

1. $n \rightarrow n-1$ edges
2. total degree $= 2(n-1)$ by Handshake theorem
3. Find possible combinations of degrees

By Theorem 10.5.2, any tree with 4 vertices has 3 edges. And so by the Handshake Theorem, the tree has a total degree of 6.

**Theorem 10.1.1 The Handshake Theorem**

Given a graph $G=(V, E)$, the total degree of $G = 2|E|$.

Also, every non-trivial tree has at least two vertices of degree 1.

The only possible combinations of degrees for the 4 vertices are:

**1, 1, 1, 3** and **1, 1, 2, 2**

Therefore, there are two non-isomorphic trees with 4 vertices.

13

## (2) Rooted trees

**Definitions: Rooted Tree, Level, Height**

A **rooted tree** is a tree in which there is one vertex that is distinguished from the others and is called the **root**.

The **level** of a vertex is the number of edges along the unique path between it and the root.

The **height** of a rooted tree is the maximum level of any vertex of the tree.

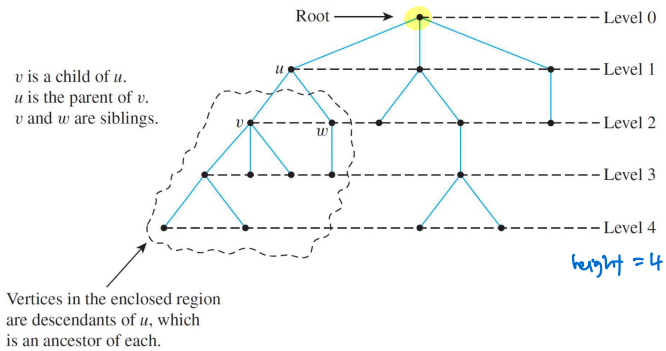
**Definitions: Child, Parent, Sibling, Ancestor, Descendant**

Given the root or any internal vertex $v$ of a rooted tree, the **children** of $v$ are all those vertices that are adjacent to $v$ and are one level farther away from the root than $v$.

If $w$ is a child of $v$, then $v$ is called the **parent** of $w$, and two distinct vertices that are both children of the same parent are called **siblings**.

Given two distinct vertices $v$ and $w$, if $v$ lies on the unique path between $w$ and the root, then $v$ is an **ancestor** of $w$, and $w$ is a **descendant** of $v$.

19




## (3) Binary trees

### 1) binary trees

this matters!

**Definitions: Binary Tree, Full Binary Tree**

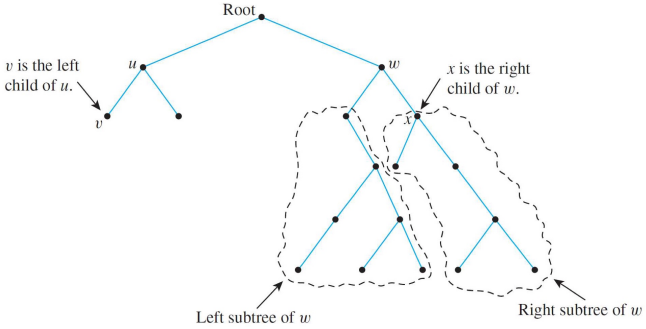
A **binary tree** is a rooted tree in which every parent has at most two children. Each child is designated either a **left child** or a **right child** (but not both), and every parent has at most one left child and one right child.

A **full binary tree** is a binary tree in which each parent has exactly two children.

**Definitions: Left Subtree, Right Subtree**

Given any parent $v$ in a binary tree $T$, if $v$ has a left child, then the **left subtree** of $v$ is the binary tree whose root is the left child of $v$, whose vertices consist of the left child of $v$ and all its descendants, and whose edges consist of all those edges of $T$ that connect the vertices of the left subtree.

The **right subtree** of $v$ is defined analogously.



## 2) properties of binary trees

1.

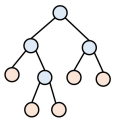
**Theorem 10.6.1: Full Binary Tree Theorem**

If $T$ is a full binary tree with $k$ internal vertices, then $T$ has a total of $2k + 1$ vertices and has $k + 1$ terminal vertices (leaves).

↳ k internal, k + 1 terminal

**Proof:**

1. Every vertex, except the root, has a parent.
2. Since every internal vertex of a full binary tree has exactly two children, the number of vertices that have a parent is twice the number of parents, or $2k$.
   #vertices of $T$ = #vertices that have a parent + #vertices that do not have a parent
   $= 2k + 1$
3. #terminal vertices = #vertices – #internal vertices
   $= 2k + 1 - k = k + 1$
4. Therefore $T$ has a total of **$2k + 1$ vertices** and has **$k + 1$ terminal vertices**.



27

2.

**Theorem 10.6.2**

For non-negative integers $h$, if $T$ is any binary tree with height $h$ and $t$ terminal vertices (leaves), then

$$t \le 2^h$$

Equivalently,

$$\log_2 t \le h$$

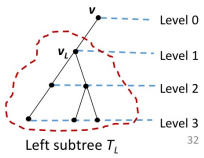
This theorem says that the maximum number of terminal vertices (leaves) of a binary tree of height $h$ is $2^h$. Alternatively, a binary tree with $t$ terminal vertices (leaves) has height of at least $\log_2 t$.

29

**Proof:** By strong mathematical induction

1. Let $P(h)$ be "If $T$ is any binary tree of height $h$, then the number of leaves of $T$ is at most $2^h$.
2. $P(0)$: $T$ consists of one vertex, which is a terminal vertex. Hence $t = 1 = 2^0$.
3. Show that for all integers $k \ge 0$, if $P(i)$ is true for all integers $i$ from 0 through $k$, then $P(k+1)$ is true.
4. Let $T$ be a binary tree of height $k + 1$, root $v$, and $t$ leaves.
5. Since $k \ge 0$, hence $k + 1 \ge 1$ and so $v$ has at least one child.
6. We consider two cases: If $v$ has only one child, or if $v$ has two children.

7. Case 1 ($v$ has only one child):
   - 7.1 Without loss of generality, assume that $v$'s child is a left child and denote it by $v_L$. Let $T_L$ be the left subtree of $v$.
   - 7.2 Because $v$ has only one child, $v$ has degree 1 (leaf), so the total number of leaves in $T$ equals the number of leaves in $T_L + 1$. Thus, if $t_L$ is the number of leaves in $T_L$, then $t = t_L + 1$.
   - 7.3 By inductive hypothesis, $t_L \le 2^k$ because the height of $T_L$ is $k$, one less than the height of $T$.
   - 7.4 Also, because $v$ has a child, $k+1 \ge 1$ and so $2^k \ge 2^0 = 1$.
   - 7.5 Therefore,
     $$t = t_L + 1 \le 2^k + 1 \le 2^k + 2^k = 2^{k+1}$$




32

8. Case 2 ($v$ has two children):
   - 8.1 Now $v$ has a left child $v_L$ and a right child $v_R$, and they are the roots of a left subtree $T_L$ and a right subtree $T_R$ respectively.
   - 8.2 Let $h_L$ and $h_R$ be the heights of $T_L$ and $T_R$ respectively.
   - 8.3 Then $h_L \le k$ and $h_R \le k$ since $T$ is obtained by joining $T_L$ and $T_R$ and adding a level.
   - 8.4 Let $t_L$ and $t_R$ be the number of leaves of $T_L$ and $T_R$ respectively.
   - 8.5 Then, since both $T_L$ and $T_R$ have heights less than $k + 1$, by inductive hypothesis, $t_L \le 2^{h_L}$ and $t_R \le 2^{h_R}$.
   - 8.6 Therefore,
     $$t = t_L + t_R \le 2^{h_L} + 2^{h_R} \le 2^k + 2^k \le 2^{k+1}$$
9. In both cases, $P(k+1)$ is true.
10. Hence if $T$ is any binary tree with height $h$ and $t$ terminal vertices (leaves), then $t \le 2^h$.

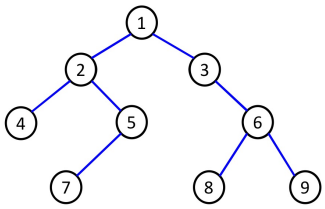
34

## A) Binary tree traversal

### 1) Breadth first search

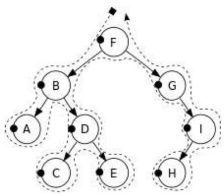
In breadth-first search (by E.F. Moore), it starts at the root and visits its adjacent vertices, and then moves to the next level.

The figure shows the order of the vertices visited.

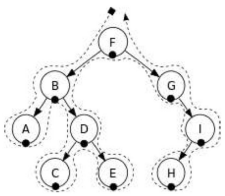



left → right per level, level by level

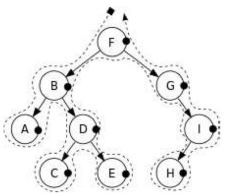
### 2) Depth first search




Pre-order:
F, B, A, D, C, E, G, I, H




In-order:
A, B, C, D, E, F, G, H, I




Post-order:
A, C, E, D, B, H, I, G, F

1. print current
2. left subtree recursively
3. right subtree recursively

1. go left, not, print
2. print root
3. go right, recursive

1. go left
2. go right
3. root

recursive

inferring tree structure from in order (+) left - centre - right

pre order: node - left - right →

post order: ← left - right - node

1. use pre/post order to get root
2. using adjacency in pre/post, get left/right subtree from in order → until terminal

## ⑤ spanning trees

### 1) spanning trees

**Definition: Spanning Tree**

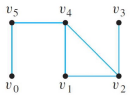
A **spanning tree** for a graph $G$ is a subgraph of $G$ that contains every vertex of $G$ and is a tree.

**Proposition 10.7.1**

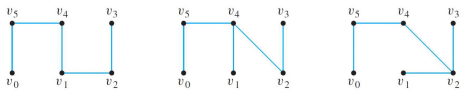
1. Every connected graph has a spanning tree.
2. Any two spanning trees for a graph have the same number of edges.

how to get spanning trees from graphs

1. if already tree, done
2. remove any combination of edges from circuits in the graph to give a tree




The graph $G$ has one circuit $v_2v_1v_4v_2$ and removal of any edge of the circuit gives a tree. Hence there are three spanning trees for $G$.




46

total no. of spanning trees in a complete graph = $n^{n-2}$

### 2) weighted graphs and minimum spanning trees

**Definitions: Weighted Graph, Minimum Spanning Tree**

A **weighted graph** is a graph for which each edge has an associated positive real number **weight** . The sum of the weights of all the edges is the **total weight** of the graph.

→ each edge has weight

→ total weight = $\sum$ weight of edge $i$

A **minimum spanning tree** for a connected weighted graph is a spanning tree that has the least possible total weight compared to all other spanning trees for the graph.

If $G$ is a weighted graph and $e$ is an edge of $G$, then $\boldsymbol{w(e)}$ denotes the weight of $e$ and $\boldsymbol{w(G)}$ denotes the total weight of $G$.

### 3) algorithms to find minimum spanning trees

Kruskal

Joseph B. Kruskal (1928 – 2010)

In Kruskal's algorithm, the edges of a connected weighted graph are examined one by one in order of increasing weight.

At each stage the edge being examined is added to what will become the minimum spanning tree, provided that this addition does not create a circuit.

After $n - 1$ edges have been added (where $n$ is the number of vertices of the graph), these edges, together with the vertices of the graph, form a minimum spanning tree for the graph.

49

**Algorithm 10.7.1 Kruskal**

Input: $G$ [a connected weighted graph with $n$ vertices]

Algorithm:

1. Initialize $T$ to have all the vertices of $G$ and no edges.
2. Let $E$ be the set of all edges of $G$, and let $m = 0$.
3. While $(m < n - 1)$
   - 3a. Find an edge $e$ in $E$ of least weight.
   - 3b. Delete $e$ from $E$.
   - 3c. If addition of $e$ to the edge set of $T$ does not produce a circuit, then add $e$ to the edge set of $T$ and set $m = m + 1$

   End while

Output: $T$ [$T$ is a minimum spanning tree for $G$]

Prim

Prim's algorithm works differently from Kruskal's. It builds a minimum spanning tree $T$ by expanding outward in connected links from some vertex.

One edge and one vertex are added at each stage. The edge added is the one of least weight that connects the vertices already in $T$ with those not in $T$, and the vertex is the endpoint of this edge that is not already in $T$.

**Algorithm 10.7.2 Prim**

Input: $G$ [a connected weighted graph with $n$ vertices]

Algorithm:

1. Pick a vertex $v$ of $G$ and let $T$ be the graph with this vertex only.
2. Let $V$ be the set of all vertices of $G$ except $v$.
3. For $i = 1$ to $n - 1$
   - 3a. Find an edge $e$ of $G$ such that (1) $e$ connects $T$ to one of the vertices in $V$, and (2) $e$ has the least weight of all edges connecting $T$ to a vertex in $V$. Let $w$ be the endpoint of $e$ that is in $V$.
   - 3b. Add $e$ and $w$ to the edge and vertex sets of $T$, and delete $w$ from $V$.

Output: $T$ [$T$ is a minimum spanning tree for $G$]

if tree is a unique minimum, algorithms will produce the same spanning tree