Lecture No.1 Logic

Course Objective:

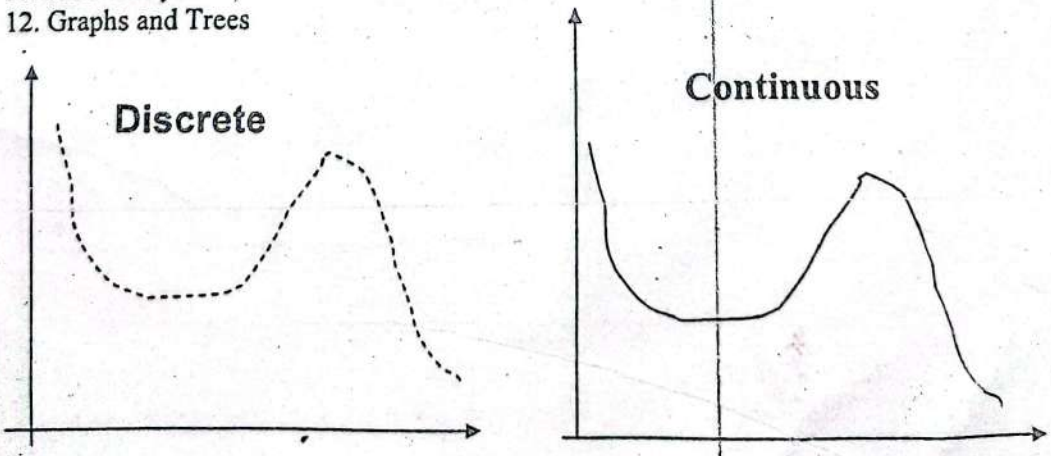
1.Express statements with the precision of formal logic
2.Analyze arguments to test their validity
3.Apply the basic properties and operations related to sets
4.Apply to sets the basic properties and operations related to relations and functions
5.Define terms recursively
6.Prove a formula using mathematical induction
7.Prove statements using direct and indirect methods
8.Compute probability of simple and conditional events
9.Identify and use the formulas of combinatorics in different problems
10.Illustrate the basic definitions of graph theory and properties of graphs
11.Relate each major topic in Discrete Mathematics to an application area in computing

1.Recommended Books:

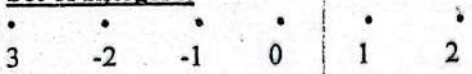
1.Discrete Mathematics with Applications (second edition) by Susanna S. Epp
2.Discrete Mathematics and Its Applications (fourth edition) by Kenneth H. Rosen
1.Discrete Mathematics by Ross and Wright

MAIN TOPICS:

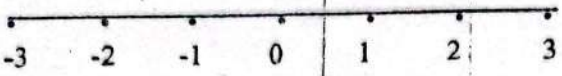
1. Logic
2. Sets & Operations on sets
3. Relations & Their Properties
4. Functions
5. Sequences & Series
6. Recurrence Relations
7. Mathematical Induction
8. Loop Invariants
9. Loop Invariants
10. Combinatorics
11. Probability
12. Graphs and Trees



Set of Integers:



Set of Real Numbers:



What is Discrete Mathematics?

Discrete Mathematics concerns processes that consist of a sequence of individual steps.

LOGIC:

Logic is the study of the principles and methods that distinguish between a valid and an invalid argument.

SIMPLE STATEMENT:

A statement is a declarative sentence that is either true or false but not both.
A statement is also referred to as a **proposition**

EXAMPLES:

a. 2+2 = 4,
b. It is Sunday today

If a proposition is true, we say that it has a truth value of "true".
If a proposition is false, its truth value is "false".
The truth values "true" and "false" are, respectively, denoted by the letters T and F.

EXAMPLES:

Propositions

1) Grass is green.
2) $4 + 2 = 6$
3) $4 + 2 = 7$
4) There are four fingers in a hand.

Not Propositions

1) Close the door.
2) x is greater than 2.
3) He is very rich

Rule:

If the sentence is preceded by other sentences that make the pronoun or variable reference clear, then the sentence is a statement.

Example:

$x = 1$
$x > 2$
"$x > 2$" is a statement with truth-value FALSE.

Example

Bill Gates is an American
He is very rich
"He is very rich" is a statement with truth-value TRUE.

(2)

UNDERSTANDING STATEMENTS

1) $x + 2$ is positive.	Not a statement
2) May I come in?	Not a statement
3) Logic is interesting.	A statement
4) It is hot today.	A statement
5) $-1 > 0$	A statement
6) $x + y = 12$	Not a statement

COMPOUND STATEMENT:
Simple statements could be used to build a compound statement.

LOGICAL CONNECTIVES

EXAMPLES:

1. "3 + 2 = 5" **and** "Lahore is a city in Pakistan"
2. "The grass is green" **or** " It is hot today"
3. "Discrete Mathematics is **not** difficult to me"

AND, OR, NOT are called LOGICAL CONNECTIVES.

SYMBOLIC REPRESENTATION

Statements are symbolically represented by letters such as p, q, r,...

EXAMPLES:

p = "Islamabad is the capital of Pakistan"
q = "17 is divisible by 3"

CONNECTIVE	MEANINGS	SYMBOLS	CALLED
Negation	not	$\sim$	Tilde
Conjunction	and	$\wedge$	Hat
Disjunction	or	$\vee$	Vel
Conditional	if...then...	$\rightarrow$	Arrow
Biconditional	if and only if	$\leftrightarrow$	Double arrow

EXAMPLES

p = "Islamabad is the capital of Pakistan"
q = "17 is divisible by 3"
$p \wedge q$ = "Islamabad is the capital of Pakistan and 17 is divisible by 3"
$p \vee q$ = "Islamabad is the capital of Pakistan or 17 is divisible by 3"
$\sim p$ = "It is not the case that Islamabad is the capital of Pakistan"
or simply "Islamabad is not the capital of Pakistan"

TRANSLATING FROM ENGLISH TO SYMBOLS

Let p = "It is hot", and q = " It is sunny"

SENTENCE	SYMBOLIC FORM
1.It is **not** hot.	$\sim p$
2.It is hot **and** sunny.	$p \wedge q$
3.It is hot **or** sunny.	$p \vee q$
4.It is **not** hot **but** sunny.	$\sim p \wedge q$
5.It is **neither** hot **nor** sunny.	$\sim p \wedge \sim q$

EXAMPLE

Let h = "Zia is healthy"
w = "Zia is wealthy"
s = "Zia is wise"
Translate the compound statements to symbolic form:

1) Zia is healthy and wealthy but not wise. $(h \wedge w) \wedge (\sim s)$
2) Zia is not wealthy but he is healthy and wise. $\sim w \wedge (h \wedge s)$
3) Zia is neither healthy, wealthy nor wise. $\sim h \wedge \sim w \wedge \sim s$

TRANSLATING FROM SYMBOLS TO ENGLISH:

Let m = "Ali is good in Mathematics"
c = "Ali is a Computer Science student"

Translate the following statement forms into plain English:

1) $\sim c$ Ali is **not** a Computer Science student
2) $c \vee m$ Ali is a Computer Science student **or** good in Maths.
3) $m \wedge \sim c$ Ali is good in Maths **but not** a Computer Science student

A convenient method for analyzing a compound statement is to make a truth table for it.

Truth Table
A **truth table** specifies the truth value of a compound proposition for all possible truth values of its constituent propositions.

NEGATION (~):
If p is a statement variable, then negation of p, *"not p"*, is denoted as "~p"
It has opposite truth value from p i.e., if p is true, then ~ p is false; if p is false, then ~ p is true.

TRUTH TABLE FOR ~ p

p	$\sim p$
T	F
F	T

CONJUNCTION (∧):
If p and q are statements, then the conjunction of p and q is *"p and q"*, denoted as *"$p \wedge q$"*.

Remarks
- $p \wedge q$ is true only when both p and q are true.
- If either p or q is false, or both are false, then $p \wedge q$ is false.

TRUTH TABLE FOR p ∧ q

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

DISJUNCTION (∨) or INCLUSIVE OR

If p & q are statements, then the disjunction of p and q is *"p or q"*, denoted as *"$p \vee q$"*.

Remarks:
- $p \vee q$ is true when at least one of p or q is true.
- $p \vee q$ is false only when both p and q are false.

(۲)

TRUTH TABLE FOR $p \vee q$

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

Note it that in the table F is only in that row where both p and q have F and all other values are T. Thus for finding out the truth values for the disjunction of two statements we will only first search out where the both statements are false and write down the F in the corresponding row in the column of $p \vee q$ and in all other rows we will write T in the column of $p \vee q$.

Remark:

Note that for Conjunction of two statements we find the T in both the statements, But in disjunction we find F in both the statements. In other words, we will fill T in the first row of conjunction and F in the last row of disjunction.

SUMMARY

1. What is a statement?
2. How a compound statement is formed.
3. Logical connectives (negation, conjunction, disjunction).
4. How to construct a truth table for a statement form.

(6)

Lecture No.2 Truth Tables

Truth Tables for:

1. $\sim p \wedge q$
2. $\sim p \wedge (q \vee \sim r)$
3. $(p \vee q) \wedge \sim (p \wedge q)$

Truth table for the statement form $\sim p \wedge q$

p	q	~p	~ p ∧ q
T	T	F	F
T	F	F	F
F	T	T	T
F	F	T	F

Truth table for $\sim p \wedge (q \vee \sim r)$

p	q	r	~r	q ∨ ~ r	~p	~ p ∧ (q ∨ ~ r)
T	T	T	F	T	F	F
T	T	F	T	T	F	F
T	F	T	F	F	F	F
T	F	F	T	T	F	F
F	T	T	F	T	T	T
F	T	F	T	T	T	T
F	F	T	F	F	T	F
F	F	F	T	T	T	T

Truth table for $(p \vee q) \wedge \sim (p \wedge q)$

p	q	p∨q	p∧q	~ (p∧q)		(p∨q) ∧ ~ (p∧q)
T	T	T	T	F		F
T	F	T	F	T		T
F	T	T	F	T		T
F	F	F	F	T		F

USAGE OF "OR" IN ENGLISH

In English language the word **OR** is sometimes used in an inclusive sense (p or q or both).

Example: I shall buy a pen or a book.
In the above statement, if you buy a pen or a book in both cases the statement is true and if you buy both pen and book, then statement is again true. Thus we say in the above statement we use or in inclusive sense.

The word **OR** is sometimes used in an exclusive sense (p or q but not both). As in the below statement

Example: Tomorrow at 9, I'll be in Lahore or Islamabad.
Now in above statement we are using **OR** in exclusive sense because if both the statements are true, then we have F for the statement.

While defining a disjunction the word **OR** is used in its inclusive sense. Therefore, the symbol $\vee$ means the "inclusive **OR**"

EXCLUSIVE OR:

When **OR** is used in its exclusive sense, The statement "p or q" means "p or q but not both" or "p or q and not p and q" which translates into symbols as $(p \vee q) \wedge \sim (p \wedge q)$
It is abbreviated as $p \oplus q$ or **p XOR q**

TRUTH TABLE FOR EXCLUSIVE OR:

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

TRUTH TABLE FOR $(p \vee q) \wedge \sim (p \wedge q)$

p	q	$p \vee q$	$p \wedge q$	$\sim (p \wedge q)$	$(p \vee q) \wedge \sim (p \wedge q)$
T	T	T	T	F	F
T	F	T	F	T	T
F	T	T	F	T	T
F	F	F	F	T	F

Note: Basically

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

LOGICAL EQUIVALENCE

If two logical expressions have the same logical values in the truth table, then we say that the two logical expressions are logically equivalent. In the following example, ~ (~ p) is logically equivalent p. So it is written as ~(~p) ≡ p

Double Negative Property ~(~p) ≡ p

p	~p	~(~p)
T	F	T
F	T	F

Example

Rewrite in a simpler form:

"It is not true that I am not happy."

Solution:

Let p = "I am happy"
then ~ p = "I am not happy"
and ~ (~ p) = "It is not true that I am not happy"
Since ~ (~ p) ≡ p
Hence the given statement is equivalent to "I am happy"

Example

Show that ~ (p∧q) and ~ p ∧ ~ q are not logically equivalent

Solution:

p	q	~p	~q	p∧q	~(p∧q)	~p ∧ ~q
T	T	F	F	T	F	F
T	F	F	T	F	T	F
F	T	T	F	F	T	F
F	F	T	T	F	T	T

Different truth values in row 2 and row 3

9

DE MORGAN'S LAWS

1) The negation of an **AND** statement is logically equivalent to the **OR** statement in which each component is negated.

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

2) The negation of an **OR** statement is logically equivalent to the **AND** statement in which each component is negated.

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

Truth Table of $\sim(p \vee q) \equiv \sim p \wedge \sim q$

p	q	~p	~q	p ∨ q	~(p ∨ q)	~p ∧ ~q
T	T	F	F	T	F	F
T	F	F	T	T	F	F
F	T	T	F	T	F	F
F	F	T	T	F	T	T

Same truth values

APPLICATION:

Give negations for each of the following statements:

a) The fan is slow **or** it is very hot.
b) Akram is unfit **and** Saleem is injured.

Solution:

a) The fan is **not** slow **and** it is **not** very hot.
b) Akram is **not** unfit **or** Saleem is **not** injured.

INEQUALITIES AND DEMORGAN'S LAWS:

Use DeMorgan's Laws to write the negation of

$-1 < x \leq 4$ for some particular real number x

Here, $-1 < x \leq 4$ means $x > -1$ **and** $x \leq 4$

The negation of $(x > -1$ **and** $x \leq 4)$ is $(x \leq -1$ **OR** $x > 4)$.

We can explain it as follows:

Suppose $p: x > -1$

$q: x \leq 4$

$\sim p: x \leq -1$

$\sim q: x > 4$

The negation of $x > -1$ **AND** $x \leq 4$

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

$\equiv \sim p \vee \sim q$ by DeMorgan's Law,
$\equiv x \leq -1$ **OR** $x > 4$

EXERCISE:
1. Show that $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$
2. Are the statements $(p \wedge q) \vee r$ and $p \wedge (q \vee r)$ logically equivalent?

TAUTOLOGY:
A tautology is a statement form that is always true regardless of the truth values of the statement variables. A tautology is represented by the symbol "t".

EXAMPLE: The statement form $p \vee \sim p$ is tautology

p	~p	p ∨ ~p
T	F	T
F	T	T

$p \vee \sim p \equiv t$

CONTRADICTION:
A contradiction is a statement form that is always false regardless of the truth values of the statement variables. A contradiction is represented by the symbol "c".

So if we have to prove that a given statement form is **CONTRADICTION,** we will make the truth table for the statement form and if in the column of the given statement form all the entries are F, then we say that statement form is contradiction.

EXAMPLE:
The statement form $p \wedge \sim p$ is a contradiction.

p	~p	p ∧ ~p
T	F	F
F	T	F

Since in the last column in the truth table we have F in all the entries, so it is a contradiction i.e. $p \wedge \sim p \equiv c$

REMARKS:
- Most statements are neither tautologies nor contradictions.
- The negation of a tautology is a contradiction and vice versa.
- In common usage we sometimes say that two statement are contradictory. By this we mean that their conjunction is a contradiction: they cannot both be true.

LOGICAL EQUIVALENCE INVOLVING TAUTOLOGY

1. Show that $p \wedge t \equiv p$

p	t	$p \wedge t$
T	T	T
F	T	F

Since in the above table the entries in the first and last columns are identical so we have the corresponding statement forms are Logically equivalent that is

$p \wedge t \equiv p$

LOGICAL EQUIVALENCE INVOLVING CONTRADICTION

Show that $p \wedge c \equiv c$

p	c	$p \wedge c$
T	F	F
F	F	F

There are same truth values in the indicated columns, so $p \wedge c \equiv c$

EXERCISE:

Use truth table to show that $(p \wedge q) \vee (\sim p \vee (p \wedge \sim q))$ is a tautology.

SOLUTION:

Since we have to show that the given statement form is Tautology, so the column of the above proposition in the truth table will have all entries as T. As clear from the table below

p	q	$p \wedge q$	$\sim p$	$\sim q$	$p \wedge \sim q$	$\sim p \vee (p \wedge \sim q)$	$(p \wedge q) \vee (\sim p \vee (p \wedge \sim q))$
T	T	T	F	F	F	F	T
T	F	F	F	T	T	T	T
F	T	F	T	F	F	T	T
F	F	F	T	T	F	T	T

Hence $(p \wedge q) \vee (\sim p \vee (p \wedge \sim q)) \equiv t$

EXERCISE:

Use truth table to show that $(p \wedge \sim q) \wedge (\sim p \vee q)$ is a contradiction.

SOLUTION:

Since we have to show that the given statement form is Contradiction, so its column in the truth table will have all entries as F. As clear from the table below.

p	q	~q	p∧~q	~p	~p∨q	(p∧~q)∧(~p∨q)
T	T	F	F	F	T	F
T	F	T	T	F	F	F
F	T	F	F	T	T	F
F	F	T	F	T	T	F

LAWS OF LOGIC

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**

$p \wedge t \equiv p$

$p \vee c \equiv p$

5) **Negation Laws**

$p \vee \sim p \equiv t$

$p \wedge \sim p \equiv c$

6) **Double Negation Law**

$\sim(\sim p) \equiv p$

7) **Idempotent Laws**

$p \wedge p \equiv p$

$p \vee p \equiv p$

8) **DeMorgan's Laws**

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

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

9) **Universal Bound Laws**

$p \vee t \equiv t$

$p \wedge c \equiv c$

10) Absorption Laws

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

11) Negation of t and c

$$\sim t \equiv c$$
$$\sim c \equiv t$$

(14)

Lecture No.3 Laws of Logic

APPLYING LAWS OF LOGIC

Using law of logic, simplify the statement form

$$p \vee [\sim(\sim p \wedge q)]$$

Solution:

$p \vee [\sim(\sim p \wedge q)] \equiv p \vee [\sim(\sim p) \vee (\sim q)]$	DeMorgan's Law
$\equiv p \vee [p \vee (\sim q)]$	Double Negative Law: $\sim(\sim p) \equiv p$
$\equiv [p \vee p] \vee (\sim q)$	Associative Law for $\vee$
$\equiv p \vee (\sim q)$	Idempotent Law: $p \vee p \equiv p$

That is the simplified statement form.

Example: Using Laws of Logic, verify the logical equivalence

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

Solution:

$\sim(\sim p \wedge q) \wedge (p \vee q) \equiv (\sim(\sim p) \vee \sim q) \wedge (p \vee q)$	DeMorgan's Law
$\equiv (p \vee \sim q) \wedge (p \vee q)$	Double Negative Law
$\equiv p \vee (\sim q \wedge q)$	Distributive Law
$\equiv p \vee c$	Negation Law
$\equiv p$	Identity Law

SIMPLIFYING A STATEMENT:

"You will get an A if you are hardworking and the sun shines, or you are hardworking and it rains." Rephrase the condition more simply.

Solution:

Let p = "You are hardworking'
q = "The sun shines"
r = "It rains" .

The condition is $(p \wedge q) \vee (p \wedge r)$

Using distributive law in reverse,

$$(p \wedge q) \vee (p \wedge r) \equiv p \wedge (q \vee r)$$

Putting $p \wedge (q \vee r)$ back into English, we can rephrase the given sentence as

"You will get an A if you are hardworking and the sun shines or it rains.

EXERCISE:

Use Logical Equivalence to rewrite each of the following sentences more simply.

1.It is not true that I am tired and you are smart.

{I am **not** tired **or** you are **not** smart.}

2.It is not true that I am tired or you are smart.

{I am **not** tired **and** you are **not** smart.}

3.I forgot my pen or my bag and I forgot my pen or my glasses.

{I forgot my pen **or** I forgot my bag **and** glasses.

4.It is raining and I have forgotten my umbrella, or it is raining and I have forgotten my hat.
{It is raining **and** I have forgotten my umbrella **or** my hat.}

CONDITIONAL STATEMENTS:

Introduction

Consider the statement:

"If you earn an A in Math, then I'll buy you a computer."

This statement is made up of two simpler statements:

p: "You earn an A in Math"
q: "I will buy you a computer."

The original statement is then saying :

if p is true, then q is true, or, more simply, **if p, then q.**

We can also phrase this as p **implies** q. It is denoted by $p \rightarrow q$.

CONDITIONAL STATEMENTS OR IMPLICATIONS:

If p and q are statement variables, the conditional of q by p is "If p then q" or "p implies q" and is denoted $p \rightarrow q$.

$p \rightarrow q$ is false when p is true and q is false; otherwise it is true.
The arrow "$\rightarrow$ " is the **conditional** operator.
In $p \rightarrow q$, the statement **p** is called **the hypothesis (or antecedent)** and q is called the **conclusion** (or **consequent**).

TRUTH TABLE:

p	q	$p \rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

PRACTICE WITH CONDITIONAL STATEMENTS:

Determine the truth value of each of the following conditional statements:

1. "If 1 = 1, then 3 = 3." **TRUE**
2. "If 1 = 1, then 2 = 3." **FALSE**
3. "If 1 = 0, then 3 = 3." **TRUE**
4. "If 1 = 2, then 2 = 3." **TRUE**
5. "If 1 = 1,then 1 = 2 and 2 = 3." **FALSE**
6. "If 1 = 3 or 1 = 2 then 3 = 3." **TRUE**

ALTERNATIVE WAYS OF EXPRESSING IMPLICATIONS:

The implication $p \rightarrow q$ could be expressed in many alternative ways as:

- "if p then q"
- "p implies q"
- "if p, q"
- "p only if q"
- "p is sufficient for q"
- "not p unless q"
- "q follows from p"
- "q if p"
- "q whenever p"
- "q is necessary for p"

EXERCISE:

Write the following statements in the form "if p, then q" in English.

a) Your guarantee is good only if you bought your CD less than 90 days ago.

If your guarantee is good, then you must have bought your CD player less than 90 days ago.

b) To get tenure as a professor, it is sufficient to be world-famous.

If you are world-famous, then you will get tenure as a professor.

c) That you get the job implies that you have the best credentials.

If you get the job, then you have the best credentials.

d) It is necessary to walk 8 miles to get to the top of the Peak.

If you get to the top of the peak, then you must have walked 8 miles.

TRANSLATING ENGLISH SENTENCES TO SYMBOLS:

Let p and q be propositions:

p = "you get an A on the final exam"
q = "you do every exercise in this book"
r = "you get an A in this class"

Write the following propositions using p, q, and r and logical connectives.

1.To get an A in this class it is necessary for you to get an A on the final.

SOLUTION $p \rightarrow r$

2.You do every exercise in this book; You get an A on the final, implies, you get an A in the class.

SOLUTION $p \wedge q \rightarrow r$

3. Getting an A on the final and doing every exercise in this book is sufficient For getting an A in this class.

SOLUTION $p \wedge q \rightarrow r$

TRANSLATING SYMBOLIC PROPOSITIONS TO ENGLISH:

Let p, q, and r be the propositions:

p = "you have the flu"
q = "you miss the final exam"
r = "you pass the course"

Express the following propositions as an English sentence.

1. $p \rightarrow q$

If you have flu, then you will miss the final exam.

2. $\sim q \rightarrow r$

If you don't miss the final exam, you will pass the course.

3. $\sim p \wedge \sim q \rightarrow r$

If you neither have flu nor miss the final exam, then you will pass the course.

HIERARCHY OF OPERATIONS FOR LOGICAL CONNECTIVES

- $\sim$ (negation)
- $\wedge$ (conjunction), $\vee$ (disjunction)
- $\rightarrow$ (conditional)

Example: Construct a truth table for the statement form $p \vee \sim q \rightarrow \sim p$

p	q	~ q	~p	p ∨ ~q	p ∨ ~q → ~p
T	T	F	F	T	F
T	F	T	F	T	F
F	T	F	T	F	T
F	F	T	T	T	T

Example: Construct a truth table for the statement form $(p \rightarrow q) \wedge (\sim p \rightarrow r)$

p	q	r	p→q	~p	~p→r	(p→q)∧(~ p →r)
T	T	T	T	F	T	T
T	T	F	T	F	T	T
T	F	T	F	F	T	F
T	F	F	F	F	T	F
F	T	T	T	T	T	T
F	T	F	T	T	F	F
F	F	T	T	T	T	T
F	F	F	T	T	F	F

LOGICAL EQUIVALENCE INVOLVING IMPLICATION

Use truth table to show $p \rightarrow q \equiv \sim q \rightarrow \sim p$

p	q	~q	~p	p→q	~q → ~p
T	T	F	F	T	T
T	F	T	F	F	F
F	T	F	T	T	T
F	F	T	T	T	T

same truth values

Hence the given two expressions are equivalent.

IMPLICATION LAW

$p \rightarrow q \equiv \sim p \vee q$

p	q	p→q	~p	~p∨q
T	T	T	F	T
T	F	F	F	F
F	T	T	T	T
F	F	T	T	T

same truth values

NEGATION OF A CONDITIONAL STATEMENT:

Since $p \rightarrow q \equiv \sim p \vee q$

So $\sim(p \rightarrow q) \equiv \sim(\sim p \vee q)$

$\equiv \sim(\sim p) \wedge (\sim q)$ by De Morgan's law

$\equiv p \wedge \sim q$ by the Double Negative law

Thus the negation of "**if p then q**" is logically equivalent to "**p and not q**".

Accordingly, the negation of an **if-then** statement does not start with the word **if**.

EXAMPLES

Write negations of each of the following statements:

1. If Ali lives in Pakistan then he lives in Lahore.
2. If my car is in the repair shop, then I cannot get to class.
3. If x is prime then x is odd **or** x is 2.
4. If n is divisible by 6, then n is divisible by 2 **and** n is divisible by 3.

SOLUTIONS:

1. Ali lives in Pakistan and he does not live in Lahore.
2. My car is in the repair shop and I can get to class.
3. x is prime but x is not odd **and** x is not 2.
4. n is divisible by 6 but n is not divisible by 2 **or** by 3.

INVERSE OF A CONDITIONAL STATEMENT:

The inverse of the conditional statement $p \rightarrow q$ is $\sim p \rightarrow \sim q$

A conditional and its inverse are not equivalent as could be seen from the truth table.

p	q	$p \rightarrow q$	$\sim p$	$\sim q$	$\sim p \rightarrow \sim q$
T	T	T	F	F	T
T	F	F	F	T	F
F	T	T	T	F	F
F	F	T	T	T	T

different truth values in rows 2 and 3

WRITING INVERSE:

1. ***If today is Friday, then 2 + 3 = 5.***
   If today is not Friday, then $2 + 3 \neq 5$.

2. ***If it snows today, I will ski tomorrow.***
   If it does not snow today I will not ski tomorrow.

3. ***If P is a square, then P is a rectangle.***
   If P is not a square then P is not a rectangle.

4. ***If my car is in the repair shop, then I cannot get to class.***
   If my car is not in the repair shop, then I shall get to the class.

CONVERSE OF A CONDITIONAL STATEMENT:

The converse of the conditional statement $p \rightarrow q$ is $q \rightarrow p$.

A conditional and its converse are not equivalent. i.e., $\rightarrow$ is not a commutative operator.

20

p	q	p→q	q→p
T	T	T	T
T	F	F	T
F	T	T	F
F	F	T	T

not the same

WRITING CONVERSE:

1.If today is Friday, then 2 + 3 = 5.
If 2 + 3 = 5, then today is Friday.

2.If it snows today, I will ski tomorrow.
I will ski tomorrow only if it snows today.

3. If P is a square, then P is a rectangle.
If P is a rectangle then P is a square.

4. If my car is in the repair shop, then I cannot get to class.
If I cannot get to the class, then my car is in the repair shop.

CONTRAPOSITIVE OF A CONDITIONAL STATEMENT:

The contra-positive of the conditional statement $p \rightarrow q$ is $\sim q \rightarrow \sim p$
A conditional and its contra-positive are equivalent.

Symbolically $p \rightarrow q \equiv \sim q \rightarrow \sim p$

1.If today is Friday, then 2 + 3 = 5.
If $2 + 3 \neq 5$, then today is not Friday.

2.If it snows today, I will ski tomorrow.
I will not ski tomorrow only if it does not snow today.

3. If P is a square, then P is a rectangle.
If P is not a rectangle then P is not a square.

4. If my car is in the repair shop, then I cannot get to class.
If I can get to the class, then my car is not in the repair shop.

EXERCISE:

1. Show that $p \rightarrow q \equiv \sim q \rightarrow \sim p$ (Use the truth table.)
2. Show that $q \rightarrow p \equiv \sim p \rightarrow \sim q$ (Use the truth table.)

Lecture No.4 Biconditional

BICONDITIONAL

If p and q are statement variables, the biconditional of p and q is "**p if and only if q**".
It is denoted **p↔q**. "*if and only if*" is abbreviated as **iff.**
The double headed arrow " ↔" is the **biconditional operator.**

TRUTH TABLE FOR p↔q.

p	q	p ↔q
T	T	T
T	F	F
F	T	F
F	F	T

Remark:

- p ↔ q is true only when p and q both are true or both are false.
- p ↔ q is false when either p or q is false.

EXAMPLES:
Identify which of the following are True or false?

1. "**1+1 = 3 if and only if earth is flat**"
   TRUE
2. "**Sky is blue iff 1 = 0**"
   FALSE
3. "**Milk is white iff birds lay eggs**"
   TRUE
4. "**33 is divisible by 4 if and only if horse has four legs**"
   FALSE
5. "**$x > 5$ iff $x^2 > 25$**"
   FALSE

REPHRASING BICONDITIONAL:

p↔q is also expressed as:

- "p is necessary and sufficient for q"
- "If p then q, and conversely"
- "p is equivalent to q"

Example: Show that $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$

p	q	$p \leftrightarrow q$	$p \rightarrow q$	$q \rightarrow p$	$(p \rightarrow q) \wedge (q \rightarrow p)$
T	T	T	T	T	T
T	F	F	F	T	F
F	T	F	T	F	F
F	F	T	T	T	T

same truth values

EXERCISE:

Rephrase the following propositions in the form "p if and only if q" in English.

1. If it is hot outside, you buy an ice cream cone, and if you buy an ice cream cone, it is hot outside.

Sol You buy an ice cream cone if and only if it is hot outside.

2. For you to win the contest it is necessary and sufficient that you have the only winning ticket.

Sol You win the contest if and only if you hold the only winning ticket.

3. If you read the news paper every day, you will be informed and conversely.

Sol You will be informed if and only if you read the news paper every day.

4. It rains if it is a weekend day, and it is a weekend day if it rains.

Sol It rains if and only if it is a weekend day.

5. The train runs late on exactly those days when I take it.

Sol The train runs late if and only if it is a day I take the train.

6. This number is divisible by 6 precisely when it is divisible by both 2 and 3.

Sol This number is divisible by 6 if and only if it is divisible by both 2 and 3.

TRUTH TABLE FOR $(p \rightarrow q) \leftrightarrow (\sim q \rightarrow \sim p)$

p	q	$p \rightarrow q$	$\sim q$	$\sim p$	$\sim q \rightarrow \sim p$	$(p \rightarrow q) \leftrightarrow (\sim q \rightarrow \sim p)$
T	T	T	F	F	T	T
T	F	F	T	F	F	T
F	T	T	F	T	T	T
F	F	T	T	T	T	T

TRUTH TABLE FOR (p↔q) ↔ (r ↔ q)

p	q	r	p ↔ q	r ↔ q	(p ↔q)↔(r↔ q)
T	T	T	T	T	T
T	T	F	T	F	F
T	F	T	F	F	T
T	F	F	F	T	F
F	T	T	F	T	F
F	T	F	F	F	T
F	F	T	T	F	F
F	F	F	T	T	T

TRUTH TABLE FOR p ∧ ~r ↔ q ∨ r

Here p ∧ ~ r ↔q ∨ r means (p ∧ (~ r)) ↔(q ∨ r)

p	q	r	~r	p∧~r	q∨r	p ∧ ~r ↔q∨r
T	T	T	F	F	T	F
T	T	F	T	T	T	T
T	F	T	F	F	T	F
T	F	F	T	T	F	F
F	T	T	F	F	T	F
F	T	F	T	F	T	F
F	F	T	F	F	T	F
F	F	F	T	F	F	T

LOGICAL EQUIVALENCE INVOLVING BICONDITIONAL

Example: Show that $\sim p \leftrightarrow q$ and $p \leftrightarrow \sim q$ are logically equivalent.

p	q	~p	~q	~p↔q	p ↔ ~q
T	T	F	F	F	F
T	F	F	T	T	T
F	T	T	F	T	T
F	F	T	T	F	F

same truth values

Hence $\sim p \leftrightarrow q \equiv p \leftrightarrow \sim q$

EXERCISE:

Show that $\sim(p \oplus q)$ and $p \leftrightarrow q$ are logically equivalent.

p	q	p⊕q	~(p⊕q)	p↔q
T	T	F	T	T
T	F	T	F	F
F	T	T	F	F
F	F	F	T	T

same truth values

Hence $\sim(p \oplus q) \equiv p \leftrightarrow q$

LAWS OF LOGIC:

1. Commutative Law: $p \leftrightarrow q \equiv q \leftrightarrow p$
2. Implication Laws: $p \rightarrow q \equiv \sim p \vee q$
   $\equiv \sim (p \wedge \sim q)$
3. Exportation Law: $(p \wedge q) \rightarrow r \equiv p \rightarrow (q \rightarrow r)$
4. Equivalence: $p \leftrightarrow q \equiv (p \rightarrow q) \wedge (q \rightarrow p)$
5. Reductio ad absurdum $p \rightarrow q \equiv (p \wedge \sim q) \rightarrow c$

APPLICATION:

Example: Rewrite the statement forms without using the symbols $\rightarrow$ or $\leftrightarrow$

1. $p \wedge \sim q \rightarrow r$
2. $(p \rightarrow r) \leftrightarrow (q \rightarrow r)$

Solution:

1. $p \wedge \sim q \rightarrow r \equiv (p \wedge \sim q) \rightarrow r$ Order of operations
   $\equiv \sim (p \wedge \sim q) \vee r$ Implication law

2. $(p \rightarrow r) \leftrightarrow (q \rightarrow r) \equiv (\sim p \vee r) \leftrightarrow (\sim q \vee r)$ Implication law
   $\equiv [(\sim p \vee r) \rightarrow (\sim q \vee r)] \wedge [(\sim q \vee r) \rightarrow (\sim p \vee r)]$ Equivalence of biconditional
   $\equiv [\sim(\sim p \vee r) \vee (\sim q \vee r)] \wedge [\sim(\sim q \vee r) \vee (\sim p \vee r)]$ Implication law

Example: Rewrite the statement form $\sim p \vee q \rightarrow r \vee \sim q$ to a logically equivalent form that uses only $\sim$ and $\wedge$.

Solution:

STATEMENT	REASON
$\sim p \vee q \rightarrow r \vee \sim q$	Given statement form
$\equiv (\sim p \vee q) \rightarrow (r \vee \sim q)$	Order of operations
$\equiv \sim[(\sim p \vee q) \wedge \sim (r \vee \sim q)]$	Implication law $p \rightarrow q \equiv \sim(p \wedge \sim q)$
$\equiv \sim[\sim(p \wedge \sim q) \wedge (\sim r \wedge q)]$	De Morgan's law

Example: Show that $\sim(p \rightarrow q) \rightarrow p$ is a tautology without using truth tables.

Solution:

STATEMENT	REASON
$\sim(p \rightarrow q) \rightarrow p$	Given statement form
$\equiv \sim[\sim(p \wedge \sim q)] \rightarrow p$	Implication law $p \rightarrow q \equiv \sim(p \wedge \sim q)$
$\equiv (p \wedge \sim q) \rightarrow p$	Double negation law
$\equiv \sim(p \wedge \sim q) \vee p$	Implication law $p \rightarrow q \equiv \sim p \vee q$
$\equiv (\sim p \vee q) \vee p$	De Morgan's law
$\equiv (q \vee \sim p) \vee p$	Commutative law of $\vee$
$\equiv q \vee (\sim p \vee p)$	Associative law of $\vee$
$\equiv q \vee t$	Negation law
$\equiv t$	Universal bound law

EXERCISE:

Suppose that p and q are statements so that $p \rightarrow q$ is false. Find the truth values of each of the following:

1. $\sim p \rightarrow q$
2. $p \vee q$
3. $q \leftrightarrow p$

SOLUTION

Hint: ($p \rightarrow q$ is false when p is true and q is false.)

1. TRUE
2. TRUE
3. FALSE

Lecture No.5 Argument

Before we discuss in detail about the argument, we first consider the following argument:

An interesting teacher keeps me awake. I stay awake in Discrete Mathematics class.
Therefore, my Discrete Mathematics teacher is interesting.

Is the above argument valid?

ARGUMENT:
An **argument** is a list of statements called **premises** (or **assumptions** or **hypotheses**) followed by a statement called the **conclusion.**

P_1 Premise
P_2 Premise
P_3 Premise
. . . . . .
P_n Premise

$\therefore C$ Conclusion

NOTE: The symbol $\therefore$ read "therefore" is normally placed just before the conclusion.

VALID AND INVALID ARGUMENT:
An argument is **valid** if the conclusion is true when all the premises are true.
Alternatively, an argument is valid if conjunction of its premises imply conclusion.
That is $(P_1 \wedge P_2 \wedge P_3 \wedge \ldots \wedge P_n) \rightarrow C$ is a tautology.
An argument is **invalid** if the conclusion is false when all the premises are true.
Alternatively, an argument is invalid if conjunction of its premises does not imply conclusion.
Critical Rows: The critical rows are those rows where the premises have truth value T.

EXAMPLE: Show that the following argument form is valid:

$p \rightarrow q$
p
$\therefore q$

SOLUTION

premises conclusion

p	q	p→q	p	q
T	T	T	T	T
T	F	F	T	F
F	T	T	F	T
F	F	T	F	F

← critical row

Since the conclusion q is true when the premises p→q and p are True. Therefore, it is a valid argument.

EXAMPLE Show that the following argument form is invalid:

$p \rightarrow q$
q
$\therefore p$

SOLUTION

premises conclusion

p	q	p→q	q	p
T	T	T	T	T
T	F	F	F	T
F	T	T	T	F
F	F	T	F	F

critical row

In the second critical row, the conclusion is false when the premises p→q and q are true. Therefore, the argument is invalid.

EXERCISE:

Use truth table to determine the argument form

$p \vee q$
$p \rightarrow \sim q$
$p \rightarrow r$
$\therefore r$

is valid or invalid.

premises conclusion

p	q	r	p∨q	p→~q	p→r	r
T	T	T	T	F	T	T
T	T	F	T	F	F	F
T	F	T	T	T	T	T
T	F	F	T	T	F	F
F	T	T	T	T	T	T
F	T	F	T	T	T	F
F	F	T	F	T	T	T
F	F	F	F	T	T	F

critical rows

28

In the third critical row, the conclusion is false when all the premises are true. Therefore, the argument is invalid.

The argument form is invalid

WORD PROBLEM

If Tariq is not on team A, then Hameed is on team B.
If Hameed is not on team B, then Tariq is on team A.
∴ Tariq is not on team A or Hameed is not on team B.

SOLUTION

Let

t = Tariq is on team A
h = Hameed is on team B

Then the argument is

$\sim t \rightarrow h$
$\sim h \rightarrow t$
$\therefore \sim t \vee \sim h$

t	h	$\sim t \rightarrow h$	$\sim h \rightarrow t$	$\sim t \vee \sim h$
T	T	T	T	F
T	F	T	T	T
F	T	T	T	T
F	F	F	F	T

Argument is invalid because there are three critical rows.

(Remember that the critical rows are those rows where the premises have truth value T) and in the first critical row conclusion has truth value F.

(Also remember that we say an argument is valid if in all critical rows conclusion has truth value T)

EXERCISE

If at least one of these two numbers is divisible by 6, then the product of these two numbers is divisible by 6.
Neither of these two numbers is divisible by 6.
∴ The product of these two numbers is not divisible by 6.

SOLUTION

Let d = at least one of these two numbers is divisible by 6.
p = product of these two numbers is divisible by 6.

Then the argument become in these symbols

$d \rightarrow p$
$\sim d$
$\therefore \sim p$

We will made the truth table for premises and conclusion as given below

d	p	d → p	~d	~p
T	T	T	F	F
T	F	F	F	T
F	T	T	T	F
F	F	T	T	T

In the first critical row, the conclusion is false when the premises are true. Therefore, the argument is invalid.

EXERCISE
If I got an Eid bonus, I'll buy a stereo.
If I sell my motorcycle, I'll buy a stereo.
∴ If I get an Eid bonus or I sell my motorcycle, then I'll buy a stereo.

SOLUTION:

Let

e = I got an Eid bonus
s = I'll buy a stereo
m = I sell my motorcycle

The argument is

$e \rightarrow s$
$m \rightarrow s$
$\therefore e \vee m \rightarrow s$

e	s	m	e → s	m → s	e∨m	e∨m → s
T	T	T	T	T	T	T
T	T	F	T	T	T	T
T	F	T	F	F	T	F
T	F	F	F	T	T	F
F	T	T	T	T	T	T
F	T	F	T	T	F	T
F	F	T	T	F	T	F
F	F	F	T	T	F	T

The argument is valid because in the five critical rows, the conclusion is true.

EXERCISE
An interesting teacher keeps me awake. I stay awake in Discrete Mathematics class. Therefore, my Discrete Mathematics teacher is interesting.

Solution:

t = My teacher is interesting
a = I stay awake
m = I am in Discrete Mathematics class

The argument to be tested is

$t \rightarrow a$,
$a \wedge m$
Therefore $m \wedge t$

t	a	m	$t \rightarrow a$	$a \wedge m$	$m \wedge t$
T	T	T	T	T	T
T	T	F	T	F	F
T	F	T	F	F	T
T	F	F	F	F	F
F	T	T	T	T	F
F	T	F	T	F	F
F	F	T	T	F	F
F	F	F	T	F	F

In the second critical row, the conclusion is false when the premises are true. Therefore, the argument is invalid.

Discret Structures 2015

Week -4

•Algorithms

Algorithms

•What is an algorithm?

•An algorithm is a finite set of precise instructions for performing a computation or for solving a problem.

•This is a rather vague definition. You will get to know a more precise and mathematically useful definition when you attend CS420 or CS620.

•But this one is good enough for now...

Algorithms

• Properties of algorithms:

•Input from a specified set,
•Output from a specified set (solution),
•Definiteness of every step in the computation,
•Correctness of output for every possible input,
•Finiteness of the number of calculation steps,
•Effectiveness of each calculation step and
•Generality for a class of problems.

Algorithm Examples

•We will use a pseudocode to specify algorithms, which slightly reminds us of Basic and Pascal.

•Example: an algorithm that finds the maximum element in a finite sequence

•procedure max($a_1, a_2, ..., a_n$: integers)
•max := a_1
•for i := 2 to n
• if max < a_i then max := a_i
•{max is the largest element}

Algorithm Examples

•Another example: a linear search algorithm, that is, an algorithm that linearly searches a sequence for a particular element.

•procedure linear_search(x: integer; $a_1, a_2, ..., a_n$: integers)
•i := 1
•while (i ≤ n and x ≠ a_i)
• i := i + 1
•if i ≤ n then location := i
•else location := 0
•{location is the subscript of the term that equals x, or is zero if x is not found}

Algorithm Examples

•If the terms in a sequence are ordered, a binary search algorithm is more efficient than linear search.

•The binary search algorithm iteratively restricts the relevant search interval until it closes in on the position of the element to be located.

Algorithm Examples

binary search for the letter 'j'

search interval

a c d f g h j l m o p r s u v x z

center element

Algorithm Examples

binary search for the letter 'j'

search interval

a c d f g h j l m o p r s u v x z

center element

Algorithm Examples

binary search for the letter 'j'

search interval

a c d f g h j l m o p r s u v x z

center element

Algorithm Examples

binary search for the letter 'j'

search interval

a c d f g h j l m o p r s u v x z

center element

Algorithm Examples

binary search for the letter 'j'

search interval

a c d f g h j l m o p r s u v x z

center element

found !

Algorithm Examples

- procedure binary_search(x: integer; $a_1, a_2, \ldots, a_n$: integers)
- $i := 1$ {i is left endpoint of search interval}
- $j := n$ {j is right endpoint of search interval}
- while $(i < j)$
- begin
  - $m := \lfloor (i + j)/2 \rfloor$
  - if $x > a_m$ then $i := m + 1$
  - else $j := m$
- end
- if $x = a_i$ then location := i
- else location := 0
- {location is the subscript of the term that equals x, or is zero if x is not found}

Algorithm Examples

- Obviously, on sorted sequences, binary search is more efficient than linear search.
- How can we analyze the efficiency of algorithms?
- We can measure the
  - time (number of elementary computations) and
  - space (number of memory cells) that the algorithm requires.
- These measures are called computational complexity and space complexity, respectively.

Complexity

- What is the time complexity of the linear search algorithm?
- We will determine the worst-case number of comparisons as a function of the number n of terms in the sequence.
- The worst case for the linear algorithm occurs when the element to be located is not included in the sequence.
- In that case, every item in the sequence is compared to the element to be located.

Algorithm Examples

- Here is the linear search algorithm again:
- procedure linear_search(x: integer; $a_1, a_2, \ldots, a_n$: integers)
- i := 1
- while ($i \leq n$ and $x \neq a_i$)
- i := i + 1
- if $i \leq n$ then location := i
- else location := 0
- {location is the subscript of the term that equals x, or is zero if x is not found}

Complexity

- For n elements, the loop
- while ($i \leq n$ and $x \neq a_i$)
  i := i + 1
- is processed n times, requiring 2n comparisons.
- When it is entered for the (n+1)th time, only the comparison $i \leq n$ is executed and terminates the loop.
- Finally, the comparison
  if $i \leq n$ then location := i
  is executed, so all in all we have a worst-case time complexity of 2n + 2.

Reminder: Binary Search Algorithm

- procedure binary_search(x: integer; $a_1, a_2, \ldots, a_n$: integers)
- i := 1 {i is left endpoint of search interval}
- j := n {j is right endpoint of search interval}
- while (i < j)
- begin
- m := $\lfloor (i + j)/2 \rfloor$
- if $x > a_m$ then i := m + 1
- else j := m
- end
- if $x = a_i$ then location := i
- else location := 0
- {location is the subscript of the term that equals x, or is zero if x is not found}

Complexity

- What is the time complexity of the binary search algorithm?
- Again, we will determine the worst-case number of comparisons as a function of the number n of terms in the sequence.
- Let us assume there are $n = 2^k$ elements in the list, which means that k = log n.
- If n is not a power of 2, it can be considered part of a larger list, where $2^k < n < 2^{k+1}$.

Complexity

- In the first cycle of the loop
- while (i < j)
- begin
  - m := ⌊(i + j)/2⌋
  - if x > a_m then i := m + 1
  - else j := m
- end
- the search interval is restricted to 2^{k-1} elements, using two comparisons.

Complexity

- In the second cycle, the search interval is restricted to 2^{k-2} elements, again using two comparisons.
- This is repeated until there is only one (2^0) element left in the search interval.
- At this point 2k comparisons have been conducted.

Complexity

- Then, the comparison
- while (i < j)
- exits the loop, and a final comparison
- if x = a_i then location := i
- determines whether the element was found.
- Therefore, the overall time complexity of the binary search algorithm is $2k + 2 = 2\lceil \log n \rceil + 2$.

Complexity

- In general, we are not so much interested in the time and space complexity for small inputs.
- For example, while the difference in time complexity between linear and binary search is meaningless for a sequence with n = 10, it is gigantic for $n = 2^{30}$.

Complexity

- For example, let us assume two algorithms A and B that solve the same class of problems.
- The time complexity of A is 5,000n, the one for B is $\lceil 1.1^n \rceil$ for an input with n elements.

Complexity

- Comparison: time complexity of algorithms A and B

Input Size	Algorithm A	Algorithm B
n	5,000n	$\lceil 1.1^n \rceil$
10	50,000	3
100	500,000	13,781
1,000	5,000,000	$2.5 \cdot 10^{41}$
1,000,000	$5 \cdot 10^9$	$4.8 \cdot 10^{41392}$

Complexity

- This means that algorithm B cannot be used for large inputs, while running algorithm A is still feasible.
- So what is important is the growth of the complexity functions.
- The growth of time and space complexity with increasing input size n is a suitable measure for the comparison of algorithms.

The Growth of Functions

- The growth of functions is usually described using the big-O notation.
- Definition: Let f and g be functions from the integers or the real numbers to the real numbers.
- We say that f(x) is O(g(x)) if there are constants C and k such that
- $|f(x)| \leq C|g(x)|$
- whenever $x > k$.

The Growth of Functions

- When we analyze the growth of complexity functions, f(x) and g(x) are always positive.
- Therefore, we can simplify the big-O requirement to
- $f(x) \leq C \cdot g(x)$ whenever $x > k$.
- If we want to show that f(x) is O(g(x)), we only need to find one pair (C, k) (which is never unique).

The Growth of Functions

- The idea behind the big-O notation is to establish an upper boundary for the growth of a function f(x) for large x.
- This boundary is specified by a function g(x) that is usually much simpler than f(x).
- We accept the constant C in the requirement
- $f(x) \leq C \cdot g(x)$ whenever $x > k$,
- because C does not grow with x.
- We are only interested in large x, so it is OK if $f(x) > C \cdot g(x)$ for $x \leq k$.

The Growth of Functions

- Example:
- Show that $f(x) = x^2 + 2x + 1$ is $O(x^2)$.
- For $x > 1$ we have:
- $x^2 + 2x + 1 \leq x^2 + 2x^2 + x^2$
- $\Rightarrow x^2 + 2x + 1 \leq 4x^2$
- Therefore, for $C = 4$ and $k = 1$:
- $f(x) \leq Cx^2$ whenever $x > k$.
- $\Rightarrow f(x)$ is $O(x^2)$.

Lecture No.27 Algorithm

PRE- AND POST-CONDITIONS OF AN ALGORITHM
LOOP INVARIANTS
LOOP INVARIANT THEOREM

ALGORITHM:

The word "algorithm" refers to a step-by-step method for performing some action. A computer program is, similarly, a set of instructions that are executed step-by-step for performing some specific task. Algorithm, however, is a more general term in that the term program refers to a particular programming language.

INFORMATION ABOUT ALGORITHM:

The following information is generally included when describing algorithms formally:

1.The name of the algorithm, together with a list of input and output variables.
2.A brief description of how the algorithm works.
3.The input variable names, labeled by data type.
4.The statements that make the body of the algorithm, with explanatory comments.
5.The output variable names, labeled by data type.
6.An end statement.

THE DIVISION ALGORITHM:

THEOREM (Quotient-Remainder Theorem):

Given any integer n and a positive integer d, there exist unique integers q and r such that $n = d \cdot q + r$ and $0 \leq r < d$.

Example:

a) $n = 54, d = 4$ $54 = 4 \cdot 13 + 2;$ hence $q = 13, r = 2$
b) $n = -54, d = 4$ $-54 = 4 \cdot (-14) + 2;$ hence $q = -14, r = 2$
c) $n = 54, d = 70$ $54 = 70 \cdot 0 + 54;$ hence $q = 0, r = 54$

ALGORITHM (DIVISION)

{Given a nonnegative integer a and a positive integer d, the aim of the algorithm is to find integers q and r that satisfy the conditions $a = d \cdot q + r$ and $0 \leq r < d$.
This is done by subtracting d repeatedly from a until the result is less than d but is still nonnegative.
The total number of d's that are subtracted is the quotient q. The quantity $a - d \cdot q$ equals the remainder r.}
Input: a {a nonnegative integer}, d {a positive integer}
Algorithm body: $r := a, q := 0$
{Repeatedly subtract d from r until a number less than d is obtained. Add 1 to d each time d is subtracted.}

while $(r \geq d)$
$r := r - d$ $q := q + 1$
end while
Output: q, r
end Algorithm (Division)

TRACING THE DIVISION ALGORITHM:

Example:

Trace the action of the Division Algorithm on the input variables $a = 54$ and $d = 11$

Solution

Iteration Number

Variable	0	1	2	3	4
a	54				
d	11				
r	54	43	32	21	10
q	0	1	2	3	4

PREDICATE:

Consider the sentence
"Aslam is a student at the Virtual University."
let P stand for the words
"is a student at the Virtual University"
and let Q stand for the words
"is a student at."
Then both P and Q are *predicate symbols*.
The sentences "x is a student at the Virtual University" and "x is a student at y" are symbolized as $P(x)$ and $Q(x, y)$, where x and y are predicate variables and take values in appropriate sets. When concrete values are substituted in place of predicate variables, a statement results.

DEFINITION:

A predicate is a sentence that contains a finite number of variables and becomes a statement when specific values are substituted for the variables.
The domain of a predicate variable is the set of all values that may be substituted in place of the variable.

PRE-CONDITIONS AND POST-CONDITIONS:

Consider an algorithm that is designed to produce a certain final state from a given state. Both the initial and final states can be expressed as predicates involving the input and output variables.

Often the predicate describing the initial state is called the **pre-condition of the algorithm** and the predicate describing the final state is called the **post-condition of the algorithm.**

EXAMPLE:

1. Algorithm to compute a product of two nonnegative integers
pre-condition: The input variables m and n are nonnegative integers.
pot-condition: The output variable p equals $m \cdot n$.
2. Algorithm to find the quotient and remainder of the division of one positive integer by another
pre-condition: The input variables a and b are positive integers.
pot-condition: The output variable q and r are positive integers such that
$a = b \cdot q + r$ and $0 \leq r < b$.

3. Algorithm to sort a one-dimensional array of real numbers
Pre-condition: The input variable $A[1], A[2], \ldots A[n]$ is a one-dimensional array of real numbers.
post-condition:The input variable $B[1], B[2], \ldots B[n]$ is a one-dimensional array of real numbers with same elements as $A[1], A[2], \ldots A[n]$ but with the property that $B[i] \leq B[j]$ whenever $i \leq j$.

THE DIVISION ALGORITHM:

[pre-condition: a is a nonnegative integer and
d is a positive integer, $r = a$, and $q = 0$]
while $(r \geq d)$
1. $r := r - d$
2. $q := q + 1$

end while
[post-condition: q and r are nonnegative integers
with the property that $a = q \cdot d + r$ and $0 \leq r < d$.]

LOOP INVARIANTS:

The method of loop invariants is used to prove correctness of a loop with respect to certain pre and post-conditions. It is based on the principle of mathematical induction.
[pre-condition for loop]
while (G)
[Statements in body of loop. None contain branching statements that lead outside the loop.]
end while[post-condition for loop]

DEFINITION:

A loop is defined as **correct with respect to its pre- and post-conditions** if, and only if, whenever the algorithm variables satisfy the pre-condition for the loop and the loop is executed, then the algorithm variables satisfy the post-condition of the loop.

THEOREM:

Let a **while** loop with guard G be given, together with pre- and post conditions that are predicates in the algorithm variables.

Also let a predicate $I(n)$, called the **loop invariant**, be given. If the following four properties are true, then the loop is correct with respect to its pre- and post-conditions.

I.Basis Property: The pre-condition for the loop implies that $I(0)$ is true before the first iteration of the loop.

II.Inductive property: If the guard G and the loop invariant $I(k)$ are both true for an integer $k \geq 0$ before an iteration of the loop, then $I(k+1)$ is true after iteration of the loop.

III.Eventual Falsity of Guard: After a finite number of iterations of the loop, the guard becomes false.

IV.Correctness of the Post-Condition: If N is the least number of iterations after which G is false and $I(N)$ is true, then the values of the algorithm variables will be as specified in the post-condition of the loop.

PROOF:

Let I(n) be a predicate that satisfies properties I-IV of the loop invariant theorem.

Properties I and II establish that:

For all integers $n \geq 0$, if the while loop iterates n times, then I(n) is true.

Property III indicates that the guard G becomes false after a finite number N of iterations.

Property IV concludes that the values of the algorithm variables are as specified by the post-condition of the loop.

Lecture No.28 Division algorithm

CORRECTNESS OF:
LOOP TO COMPUTE A PRODUCT
THE DIVISION ALGORITHM
THE EUCLIDEAN ALGORITHM

A LOOP TO COMPUTE A PRODUCT:

[pre-condition: m is a nonnegative integer, x is a real number, $i = 0$, and product = 0.]
while ($i \# m$)
1. product := product + x
2. $i := i + 1$

end while
[post-condition: product = $m \cdot x$]

PROOF:

Let the loop invariant be
$I(n)$: $i = n$ and product = $n \cdot x$
The guard condition G of the while loop is
G: $i \# m$

I.Basis Property:

[$I(0)$ is true before the first iteration of the loop.]
$I(0)$: $i = 0$ and product = $0 \cdot x = 0$
Which is true before the first iteration of the loop.

II.Inductive property:

[If the guard G and the loop invariant $I(k)$ are both true before a loop iteration (where $k \geq 0$), then $I(k + 1)$ is true after the loop iteration.]
Before execution of statement 1,

$$product_{old} = k \cdot x.$$

Thus the execution of statement 1 has the following effect:
$product_{new} = product_{old} + x = k \cdot x + x = (k + 1) \cdot x$
Similarly, before statement 2 is executed,
$i_{old} = k$,
So after execution of statement 2,
$i_{new} = i_{old} + 1 = k + 1$.
Hence after the loop iteration, the statement $I(k+1)$ (i.e., $i = k + 1$ and product = $(k + 1) \cdot x$) is true. This is what we needed to show.

III.Eventual Falsity of Guard:

[After a finite number of iterations of the loop, the guard becomes false.]

IV.Correctness of the Post-Condition:

[If N is the least number of iterations after which G is false and $I(N)$ is true, then the values of the algorithm variables will be as specified in the post-condition of the loop.]

THE DIVISION ALGORITHM:

[pre-condition: a is a nonnegative integer and d is a positive integer, $r = a$, and $q = 0$]

while $(r \geq d)$

1. $r := r - d$
2. $q := q + 1$

end while

[post-condition: q and r are nonnegative integers with the property that $a = q \cdot d + r$ and $0 \leq r < d$.]

PROOF:

Let the loop invariant be

$I(n)$: $r = a - n \cdot d$ and $n = q$.

The guard of the while loop is

G: $r \geq d$

I.Basis Property:

[$I(0)$ is true before the first iteration of the loop.]

$I(0)$: $r = a - 0 \cdot d = a$ and $0 = q$.

II.Inductive property:

[If the guard G and the loop invariant $I(k)$ are both true before a loop iteration (where $k \geq 0$), then $I(k+1)$ is true after the loop iteration.]

$I(k)$: $r = a - k \cdot d \geq 0$ and $k = q$

$I(k+1)$: $r = a - (k+1) \cdot d \geq 0$ and $k + 1 = q$

$$r_{new} = r - d$$
$$= a - k \cdot d - d$$
$$= a - (k+1) \cdot d$$
$$q = q + 1$$
$$= k + 1$$

also

$$r_{new} = r - d$$
$$\geq d - d = 0 \quad (\text{since } r \geq 0)$$

Hence $I(k+1)$ is true.

III.Eventual Falsity of Guard:

[After a finite number of iterations of the loop, the guard becomes false.]

IV.Correctness of the Post-Condition:

[If N is the least number of iterations after which G is false and $I(N)$ is true, then the values of the algorithm variables will be as specified in the post-condition of the loop.]

G is false and I(N) is true.

That is, $r \geq d$ and $r = a - N \cdot d \geq 0$ and $N = q$.

or $r = a - q \cdot d$

or $a = q \cdot d + r$

Also combining the two inequalities involving r we get

$0 \leq r < d$

THE EUCLIDEAN ALGORITHM:

The greatest common divisor (gcd) of two integers a and b is the largest integer that divides both a and b. For example, the gcd of 12 and 30 is 6.

The Euclidean algorithm takes integers A and B with $A > B \geq 0$ and compute their greatest common divisor.

HAND CALCULATION OF gcd:

Use the Euclidean algorithm to find gcd(330, 156)

SOLUTION:

$$\begin{array}{r} 2 \\ 156\overline{)\,330} \\ \underline{312} \\ 18 \end{array} \qquad \begin{array}{r} 8 \\ 18\overline{)\,156} \\ \underline{144} \\ 12 \end{array}$$

$$\begin{array}{r} 1 \\ 12\overline{)\,18} \\ \underline{12} \\ 6 \end{array} \qquad \begin{array}{r} 2 \\ 6\overline{)\,12} \\ \underline{12} \\ 0 \end{array}$$

Hence gcd(330, 156) = 6

EXAMPLE:

Use the Euclidean algorithm to find gcd(330, 156)

Solution:

1.Divide 330 by 156:

This gives $330 = 156 \cdot 2 + 18$

2.Divide 156 by 18:

This gives $156 = 18 \cdot 8 + 12$

3.Divide 18 by 12:

This gives $18 = 12 \cdot 1 + 6$

4.Divide 12 by 6:

This gives $12 = 6 \cdot 2 + 0$

Hence gcd(330, 156) = 6.

LEMMA:

If a and b are any integers with $b \neq 0$ and q and r are nonnegative integers such that $a = q \cdot d + r$

then
$\gcd(a, b) = \gcd(b, r)$
[pre-condition: A and B are integers with
$A > B \geq 0$, $a = A$, b = B, $r = B$.]

while ($b \neq 0$)
1. $r := a \bmod b$
2. $a := b$
3. b := r

end while[post-condition: $a = \gcd(A, B)$]

PROOF:

Let the **loop invariant** be
$I(n)$: $\gcd(a, b) = \gcd(A, B)$ and $0 \leq b < a$.
The guard of the **while** loop is
G: $b \neq 0$

I.Basis Property:

[$I(0)$ is true before the first iteration of the loop.]
$I(0)$: $\gcd(a, b) = \gcd(A, B)$ and $0 \leq b < a$.
According to the precondition,
a = A, b = B, r = B, and $0 \leq B < A$.
Hence $I(0)$ is true before the first iteration of the loop.

II.Inductive property:

[If the guard G and the loop invariant $I(k)$ are both true before a loop iteration (where $k \geq 0$), then $I(k + 1)$ is true after the loop iteration.]
Since $I(k)$ is true before execution of the loop we have,
$\gcd(a_{old}, b_{old}) = \gcd(A, B)$ and $0 \leq b_{old} < a_{old}$
After execution of statement 1,

$r_{new} = a_{old} \bmod b_{old}$ Thus,

$a_{old} = b_{old} \cdot q + r_{new}$ for some integer q

with,

$0 \leq r_{new} < b_{old}$.

But

$\gcd(a_{old}, b_{old}) = \gcd(b_{old}, r_{old})$

and we have,

$\gcd(b_{old}, r_{new}) = \gcd(A, B)$

When statements 2 and 3 are executed,

$a_{new} = b_{old}$ and $b_{new} = r_{new}$

It follows that

$\gcd(a_{new}, b_{new}) = \gcd(A, B)$

Also,

$0 \leq r_{new} < b_{old}$

becomes

$0 \leq b_{new} < a_{new}$

Hence $I(k + 1)$ is true.

III.Eventual Falsity of Guard:

[After a finite number of iterations of the loop, the guard becomes false.]

IV.Correctness of the Post-Condition:

[If N is the least number of iterations after which G is false and $I(N)$ is true, then the values of the algorithm variables will be as specified in the post-condition of the loop.]

Discrete Structures

11/11/2015

O.R
12/11/15

Number Theory

Introduction to Number Theory

Number theory is about **integers** and their properties.

We will start with the basic principles of

- divisibility,
- greatest common divisors,
- least common multiples, and
- modular arithmetic

and look at some relevant algorithms.

Division

If a and b are integers with $a \neq 0$, we say that a **divides** b if there is an integer c so that $b = ac$.

When a divides b we say that a is a **factor** of b and that b is a **multiple** of a.

The notation **a | b** means that a divides b.

We write **a X b** when a does not divide b. (see book for correct symbol).

Divisibility Theorems

For integers a, b, and c it is true that

- if a | b and a | c, then a | (b + c)
  Example: 3 | 6 and 3 | 9, so 3 | 15.
- if a | b, then a | bc for all integers c
  Example: 5 | 10, so 5 | 20, 5 | 30, 5 | 40, ...
- if a | b and b | c, then a | c
  Example: 4 | 8 and 8 | 24, so 4 | 24.

Primes

A positive integer p greater than 1 is called prime if the only positive factors of p are 1 and p.

A positive integer that is greater than 1 and is not prime is called composite.

The fundamental theorem of arithmetic:

Every positive integer can be written **uniquely** as the **product of primes**, where the prime factors are written in order of increasing size.

Primes

Examples:

$15 = 3 \cdot 5$

$48 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 3 = 2^4 \cdot 3$

$17 = 17$

$100 = 2 \cdot 2 \cdot 5 \cdot 5 = 2^2 \cdot 5^2$

$512 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 2^9$

$515 = 5 \cdot 103$

$28 = 2 \cdot 2 \cdot 7 = 2^2 \cdot 7$

(2) (1)

Primes

If n is a composite integer, then n has a prime divisor less than or equal . . .

This is easy to see: if n is a composite integer, it must have two divisors p_1 and p_2 such that $p_1 \cdot p_2 = n$ and $p_1 \geq 2$ and $p_2 \geq 2$.

p_1 and p_2 cannot both be greater than , because then $p_1 \cdot p_2$ would be greater than n.

If the smaller number of p_1 and p_2 is not a prime itself, then it can be broken up into prime factors that are smaller than itself but ≥ 2.

The Division Algorithm

Let **a** be an integer and **d** a positive integer. Then there are unique integers **q** and **r**, with $0 \leq r < d$, such that $a = dq + r$.

In the above equation,
- **d** is called the divisor,
- **a** is called the dividend,
- **q** is called the quotient, and
- **r** is called the remainder.

The Division Algorithm

Example:

When we divide 17 by 5, we have

$17 = 5 \cdot 3 + 2$.

- 17 is the dividend,
- 5 is the divisor,
- 3 is called the quotient, and
- 2 is called the remainder.

The Division Algorithm

Another example:

What happens when we divide -11 by 3 ?

Note that the remainder cannot be negative.

$-11 = 3 \cdot (-4) + 1$.

- -11 is the dividend,
- 3 is the divisor,
- -4 is called the quotient, and
- 1 is called the remainder.

Greatest Common Divisors

Let a and b be integers, not both zero. The largest integer d such that $d \mid a$ and $d \mid b$ is called the **greatest common divisor** of a and b.

The greatest common divisor of a and b is denoted by gcd(a, b).

Example 1: What is gcd(48, 72) ?

The positive common divisors of 48 and 72 are 1, 2, 3, 4, 6, 8, 12, 16, and 24, so gcd(48, 72) = 24.

Example 2: What is gcd(19, 72) ?

The only positive common divisor of 19 and 72 is 1, so gcd(19, 72) = 1.

Greatest Common Divisors

Using prime factorizations:

$a = p_1^{a_1} p_2^{a_2} \ldots p_n^{a_n}$, $b = p_1^{b_1} p_2^{b_2} \ldots p_n^{b_n}$,
where $p_1 < p_2 < \ldots < p_n$ and $a_i, b_i \in N$ for $1 \leq i \leq n$

$\gcd(a, b) = p_1^{\min(a_1, b_1)} p_2^{\min(a_2, b_2)} \ldots p_n^{\min(a_n, b_n)}$

Example:

$a = 60 = 2^2 3^1 5^1$

$b = 54 = 2^1 3^3 5^0$

$\gcd(a, b) = 2^1 3^1 5^0 = 6$

Relatively Prime Integers

Definition:

Two integers a and b are **relatively prime** if gcd(a, b) = 1.

Examples:

Are 15 and 28 relatively prime?
Yes, gcd(15, 28) = 1.
Are 55 and 28 relatively prime?
Yes, gcd(55, 28) = 1.
Are 35 and 28 relatively prime?
No, gcd(35, 28) = 7.

Relatively Prime Integers

Definition:

The integers $a_1, a_2, \ldots, a_n$ are **pairwise relatively prime** if $\gcd(a_i, a_j) = 1$ whenever $1 \le i < j \le n$.

Examples:

Are 15, 17, and 27 pairwise relatively prime?
No, because gcd(15, 27) = 3.

Are 15, 17, and 28 pairwise relatively prime?
Yes, because gcd(15, 17) = 1, gcd(15, 28) = 1 and gcd(17, 28) = 1.

Least Common Multiples

Definition:

The **least common multiple** of the positive integers a and b is the smallest positive integer that is divisible by both a and b.

We denote the least common multiple of a and b by lcm(a, b).

Examples:

lcm(3, 7) = 21
lcm(4, 6) = 12
lcm(5, 10) = 10

Least Common Multiples

Using prime factorizations:

$a = p_1^{a_1} p_2^{a_2} \ldots p_n^{a_n}$, $b = p_1^{b_1} p_2^{b_2} \ldots p_n^{b_n}$,
where $p_1 < p_2 < \ldots < p_n$ and $a_i, b_i \in \mathbb{N}$ for $1 \le i \le n$

$\text{lcm}(a, b) = p_1^{\max(a_1, b_1)} p_2^{\max(a_2, b_2)} \ldots p_n^{\max(a_n, b_n)}$

Example:

$a = 60 = 2^2 3^1 5^1$

$b = 54 = 2^1 3^3 5^0$

$\text{lcm}(a, b) = 2^2 3^3 5^1 = 4 \cdot 27 \cdot 5 = 540$

GCD and LCM

$a = 60 = (2^2)(3^1)(5^1)$

$b = 54 = (2^1)(3^3)(5^0)$

$\gcd(a, b) = (2^1 3^1 5^0) = 6$

$\text{lcm}(a, b) = (2^2 3^3 5^1) = 540$

Theorem: $a \cdot b = \gcd(a,b) \cdot \text{lcm}(a,b)$

Modular Arithmetic

Let a be an integer and m be a positive integer. We denote by **a mod m** the remainder when a is divided by m.

Examples:

9 mod 4 = 1
9 mod 3 = 0
9 mod 10 = 9
-13 mod 4 = 3

Week 6

Sequences & Summations

Sequences

Sequences represent **ordered lists** of elements.

A **sequence** is defined as a function from a subset of N to a set S. We use the notation a_n to denote the image of the integer n. We call a_n a term of the sequence.

Example:

subset of N:	1	2	3	4	5	...
	↓	↓	↓	↓	↓	
S:	2	4	6	8	10	...

Sequences

We use the notation $\{a_n\}$ to describe a sequence.

Important: Do not confuse this with the {} used in set notation.

It is convenient to describe a sequence with an **equation**.

For example, the sequence on the previous slide can be specified as $\{a_n\}$, where $a_n = 2n$.

The Equation Game

What are the equations that describe the following sequences $a_1, a_2, a_3, \ldots$?

1, 3, 5, 7, 9, ...	$a_n = 2n - 1$
-1, 1, -1, 1, -1, ...	$a_n = (-1)^n$
2, 5, 10, 17, 26, ...	$a_n = n^2 + 1$
0.25, 0.5, 0.75, 1, 1.25 ...	$a_n = 0.25n$
3, 9, 27, 81, 243, ...	$a_n = 3^n$

Strings

Finite sequences are also called **strings**, denoted by $a_1a_2a_3\ldots a_n$.

The **length** of a string S is the number of terms that it consists of.

The **empty string** contains no terms at all. It has length zero.

Summations

What does $\sum_{j=m}^{n} a_j$ stand for?

It represents the sum $a_m + a_{m+1} + a_{m+2} + \ldots + a_n$.

The variable j is called the **index of summation**, running from its **lower limit** m to its **upper limit** n. We could as well have used any other letter to denote this index.

Summations

How can we express the sum of the first 1000 terms of the sequence $\{a_n\}$ with $a_n=n^2$ for $n = 1, 2, 3, \ldots$?

We write it as $\sum_{j=1}^{1000} j^2$.

What is the value of $\sum_{j=1}^{6} j$?

It is $1 + 2 + 3 + 4 + 5 + 6 = 21$.

What is the value of $\sum_{j=1}^{100} j$?

It is so much work to calculate this...

Summations

It is said that Carl Friedrich Gauss came up with the following formula:

$$\sum_{j=1}^{n} j = \frac{n(n+1)}{2}$$

When you have such a formula, the result of any summation can be calculated much more easily, for example:

$$\sum_{j=1}^{100} j = \frac{100(100+1)}{2} = \frac{10100}{2} = 5050$$

Summations

Find the summation from 50 to 100

$$\sum_{k=50}^{100} k^2 = \sum_{k=1}^{100} k^2 - \sum_{k=1}^{49} k^2.$$

Using the formula $\sum_{k=1}^{n} k^2 = n(n+1)(2n+1)/6$ we see that

$$\sum_{k=50}^{100} k^2 = \frac{100 \cdot 101 \cdot 201}{6} - \frac{49 \cdot 50 \cdot 99}{6}$$

$$338,350 - 40,425 = 297,925.$$

Double Summations

Corresponding to nested loops in C or Java, there is also double (or triple etc.) summation:

Example:

$$\sum_{i=1}^{3} \sum_{j=1}^{3} ij$$

$$= \sum_{i=1}^{3} (i + 2i)$$

$$= \sum_{i=1}^{3} 3i$$

$$= 3+6+9+12+15 = 45$$

Double Summations

Table 2 in

4th Edition: Section 1.7

5th Edition: Section 3.2

6th and 7th Edition: Section 2.4

contains some very useful formulas for calculating sums.

In the same Section, Exercises 15 and 17 (7th Edition: Exercises 31 and 33) make a nice homework.

Some Summation Formulas:

(1) $$\sum_{k=0}^{n} ar^k = \frac{ar^{n+1} - a}{r-1} \qquad r \neq 0 \text{ and } r \neq 1$$

(2) $$\sum_{k=1}^{n} k^2 = \frac{n(n+1)(2n+1)}{6}$$

(3) $$\sum_{k=1}^{n} k^3 = \frac{n^2(n+1)^2}{4}$$

Matrices

Matrices

A **matrix** is a rectangular array of numbers.
A matrix with m rows and n columns is called an **m×n matrix.**

Example: $A = \begin{bmatrix} -1 & 1 \\ 2.5 & -0.3 \\ 8 & 0 \end{bmatrix}$ is a 3×2 matrix.

A matrix with the same number of rows and columns is called **square.**

Two matrices are **equal** if they have the same number of rows and columns and the corresponding entries in every position are equal.

Matrices

A general description of an m×n matrix A = $[a_{ij}]$:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

$\begin{bmatrix} a_{1j} \\ a_{2j} \\ \cdot \\ \cdot \\ \cdot \\ a_{mj} \end{bmatrix}$ j-th column of A

$[a_{i1}, a_{i2}, \ldots, a_{in}]$ i-th row of A

Matrix Addition

Let A = $[a_{ij}]$ and B = $[b_{ij}]$ be m×n matrices.
The sum of A and B, denoted by A+B, is the m×n matrix that has $a_{ij} + b_{ij}$ as its (i, j)th element.
In other words, A+B = $[a_{ij} + b_{ij}]$.

Example:

$$\begin{bmatrix} -2 & 1 \\ 4 & 8 \\ -3 & 0 \end{bmatrix} + \begin{bmatrix} 5 & 9 \\ -3 & 6 \\ -4 & 1 \end{bmatrix} = \begin{bmatrix} 3 & 10 \\ 1 & 14 \\ -7 & 1 \end{bmatrix}$$

Matrix Multiplication

Let A be an m×k matrix and B be a k×n matrix.
The **product** of A and B, denoted by AB, is the m×n matrix with (i, j)th entry equal to the sum of the products of the corresponding elements from the i-th row of A and the j-th column of B.

In other words, if AB = $[c_{ij}]$, then

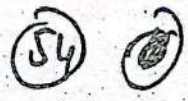
$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \ldots + a_{ik}b_{kj} = \sum_{t=1}^{k} a_{it}b_{tj}$$

Matrix Multiplication

A more intuitive description of calculating C = AB:

$$A = \begin{bmatrix} 3 & 0 & 1 \\ -2 & -1 & 4 \\ 0 & 0 & 5 \\ -1 & 1 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 2 & 1 \\ 0 & -1 \\ 3 & 4 \end{bmatrix}$$

- Take the first column of B
- Turn it counterclockwise by 90° and superimpose it on the first row of A
- Multiply corresponding entries in A and B and add the products: 3·2 + 0·0 + 1·3 = 9
- Enter the result in the upper-left corner of C



Matrix Multiplication

- Now superimpose the first column of B on the second, third, ..., m-th row of A to obtain the entries in the first column of C (same order).
- Then repeat this procedure with the second, third, ..., n-th column of B, to obtain to obtain the remaining columns in C (same order).
- After completing this algorithm, the new matrix C contains the product AB.

Matrix Multiplication

Let us calculate the complete matrix C:

$$A = \begin{bmatrix} 3 & 0 & 1 \\ -2 & -1 & 4 \\ 0 & 0 & 5 \\ -1 & 1 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 2 & 1 \\ 0 & -1 \\ 3 & 4 \end{bmatrix}$$

$$C = \begin{bmatrix} 9 & 7 \\ 8 & 15 \\ 15 & 20 \\ -2 & -2 \end{bmatrix}$$

Identity Matrices

The **identity matrix of order n** is the n×n matrix $I_n = [\delta_{ij}]$, where $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ if $i \neq j$:

$$A = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & 1 \end{bmatrix}$$

Multiplying an m×n matrix A by an identity matrix of appropriate size does not change this matrix:

$AI_n = I_mA = A$

Powers and Transposes of Matrices

The **power function** can be defined for **square** matrices. If A is an n×n matrix, we have:

$A^0 = I_n$,
$A^r = AAA...A$ (r times the letter A)

The **transpose** of an m×n matrix $A = [a_{ij}]$, denoted by A^t, is the n×m matrix obtained by interchanging the rows and columns of A.

In other words, if $A^t = [b_{ij}]$, then $b_{ij} = a_{ji}$ for $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, m$.

Powers and Transposes of Matrices

Example: $A = \begin{bmatrix} 2 & 1 \\ 0 & -1 \\ 3 & 4 \end{bmatrix} \quad A^t = \begin{bmatrix} 2 & 0 & 3 \\ 1 & -1 & 4 \end{bmatrix}$

A square matrix A is called **symmetric** if $A = A^t$. Thus $A = [a_{ij}]$ is symmetric if $a_{ij} = a_{ji}$ for all $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, n$.

$$A = \begin{bmatrix} 5 & 1 & 3 \\ 1 & 2 & -9 \\ 3 & -9 & 4 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 3 & 1 \\ 1 & 3 & 1 \\ 1 & 3 & 1 \end{bmatrix}$$

A is symmetric, B is not.

Zero-One Matrices

A matrix with entries that are either 0 or 1 is called a **zero-one matrix**. Zero-one matrices are often used like a "table" to represent discrete structures.

We can define Boolean operations on the entries in zero-one matrices:

a	b	a∧b
0	0	0
0	1	0
1	0	0
1	1	1

a	b	a∨b
0	0	0
0	1	1
1	0	1
1	1	1

Zero-One Matrices

Let $A = [a_{ij}]$ and $B = [b_{ij}]$ be m×n zero-one matrices.

Then the **join** of A and B is the zero-one matrix with (i, j)th entry $a_{ij} \vee b_{ij}$. The join of A and B is denoted by $A \vee B$.

The **meet** of A and B is the zero-one matrix with (i, j)th entry $a_{ij} \wedge b_{ij}$. The meet of A and B is denoted by $A \wedge B$.

Zero-One Matrices

Example: $A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 1 \\ 1 & 1 \\ 0 & 0 \end{bmatrix}$

Join: $A \vee B = \begin{bmatrix} 1 \vee 0 & 1 \vee 1 \\ 0 \vee 1 & 1 \vee 1 \\ 1 \vee 0 & 0 \vee 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 0 \end{bmatrix}$

Meet: $A \wedge B = \begin{bmatrix} 1 \wedge 0 & 1 \wedge 1 \\ 0 \wedge 1 & 1 \wedge 1 \\ 1 \wedge 0 & 0 \wedge 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$

Zero-One Matrices

Let $A = [a_{ij}]$ be an m×k zero-one matrix and $B = [b_{ij}]$ be a k×n zero-one matrix.

Then the **Boolean product** of A and B, denoted by $A \odot B$, is the m×n matrix with (i, j)th entry $[c_{ij}]$, where $c_{ij} = (a_{i1} \wedge b_{1j}) \vee (a_{i2} \wedge b_{2j}) \vee \ldots \vee (a_{ik} \wedge b_{kj})$.

Note that the actual Boolean product symbol has a dot in its center.

Basically, Boolean multiplication works like the multiplication of matrices, but with computing $\wedge$ instead of the product and $\vee$ instead of the sum.

Zero-One Matrices

Example:

$A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$

$A \odot B = \begin{bmatrix} (1 \wedge 0) \vee (0 \wedge 0) & (1 \wedge 1) \vee (0 \wedge 1) \\ (1 \wedge 0) \vee (1 \wedge 0) & (1 \wedge 1) \vee (1 \wedge 1) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$

Zero-One Matrices

Let A be a square zero-one matrix and r be a positive integer.

The **r-th Boolean power** of A is the Boolean product of r factors of A. The r-th Boolean power of A is denoted by $A^{[r]}$.

$A^{[0]} = I_n$,
$A^{[r]} = A \odot A \odot \ldots \odot A$ (r times the letter A)

Lecture No.25 Methods of proof

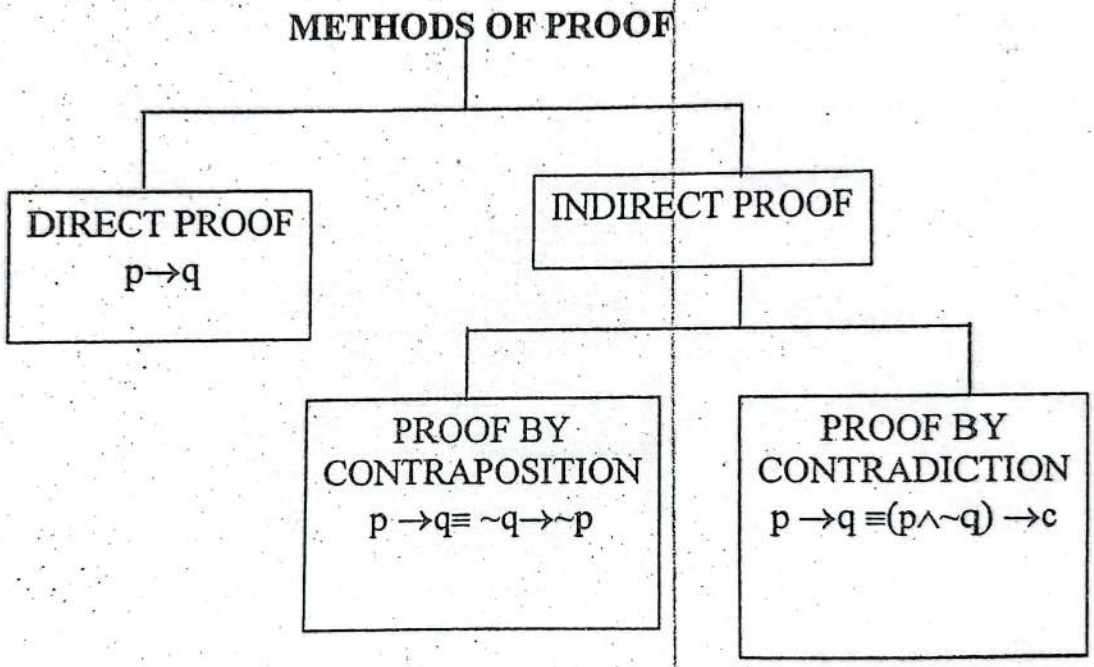
METHODS OF PROOF

-- **DIRECT PROOF**
-- **DISPROOF BYCOUNTER EXAMPLE**

INTRODUCTION:

To understand written mathematics, one must understand what makes up a correct mathematical argument, that is, a proof. This requires an under standing of the techniques used to build proofs. The methods we will study for building proofs are also used throughout computer science, such as the rules computers used to reason, the techniques used to verify that programs are correct, etc.

Many theorems in mathematics are implications, $p \rightarrow q$. The techniques of proving implications give rise to different methods of proofs.



DIRECT PROOF:

The implication $p \rightarrow q$ can be proved by showing that if p is true, the q must also be true. This shows that the combination p true and q false never occurs. A proof of this kind is called a direct proof.

p	q	p→q
T	T	T
T	F	F
F	T	T
F	F	T

SOME BASICS:

1. An integer n is even if, and only if, $n = 2k$ for some integer k.
2. An integer n is odd if, and only if, $n = 2k + 1$ for some integer k.
3. An integer n is prime if, and only if, $n > 1$ and for all positive integers r and s, if $n = r \cdot s$, then $r = 1$ or $s = 1$.
4. An integer $n > 1$ is composite if, and only if, $n = r \cdot s$ for some positive integers r and s with $r \neq 1$ and $s \neq 1$.
5. A real number r is rational if, and only if, $\frac{a}{b}$ for some integers a and b with $b \neq 0$.
6. If n and d are integers and $d \neq 0$, then d divides n, written $d \mid n$ if, and only if, $n = d.k$ for some integers k.
7. An integer n is called a perfect square if, and only if, $n = k^2$ for some integer k.

EXERCISE:

Prove that the sum of two odd integers is even.

SOLUTION:

Let m and n be two odd integers. Then by definition of odd numbers

$m = 2k + 1$ for some $k \in Z$

$n = 2l + 1$ for some $l \in Z$

Now $m + n = (2k + 1) + (2l + 1)$

$= 2k + 2l + 2$

$= 2(k + l + 1)$

$= 2r$ where $r = (k + l + 1) \in Z$

Hence m + n is even.

EXERCISE:

Prove that if n is any even integer, then $(-1)^n = 1$

SOLUTION:

Suppose n is an even integer. Then $n = 2k$ for some integer k.

Now

$(-1)^n = (-1)^{2k}$

$= [(-1)^2]^k$

$= (1)^k$

$= 1$ (proved)

EXERCISE:

Prove that the product of an even integer and an odd integer is even.

SOLUTION:

Suppose m is an even integer and n is an odd integer. Then

$m = 2k$ for some integer k

and $n = 2l + 1$ for some integer l

Now

$m \cdot n = 2k \cdot (2l + 1)$

$= 2 \cdot k(2l + 1)$

$= 2 \cdot r$ where $r = k(2l + 1)$ is an integer

Hence $m \cdot n$ is even. (Proved)

EXERCISE:

Prove that the square of an even integer is even.

SOLUTION:

Suppose n is an even integer. Then $n = 2k$

Now

$$\text{square of } n = n^2 = (2\cdot k)^2$$
$$= 4k^2$$
$$= 2\cdot(2k^2)$$
$$= 2\cdot p \quad \text{where } p = 2k^2 \in Z$$

Hence, n^2 is even. (proved)

EXERCISE:

Prove that if n is an odd integer, then $n^3 + n$ is even.

SOLUTION:

Let n be an odd integer, then $n = 2k + 1$ for some $k \in Z$

Now $n^3 + n = n(n^2 + 1)$
$= (2k + 1)((2k+1)^2 + 1)$
$= (2k + 1)(4k^2 + 4k + 1 + 1)$
$= (2k + 1)(4k^2 + 4k + 2)$
$= (2k + 1)\,2.\,(2k^2 + 2k + 1)$
$= 2\cdot(2k + 1)(2k^2 + 2k + 1)$ $\quad k \in Z$
= an even integer

EXERCISE:

Prove that, if the sum of any two integers is even, then so is their difference.

SOLUTION:

Suppose m and n are integers so that m + n is even. Then by definition of even numbers

$m + n = 2k$ for some integer k

$\Rightarrow m = 2k - n$ .....(1)

Now $m - n = (2k - n) - n$ using (1)
$= 2k - 2n$
$= 2(k - n) = 2r$ where $r = k - n$ is an integer

Hence m - n is even.

EXERCISE:

Prove that the sum of any two rational numbers is rational.

SOLUTION:

Suppose r and s are rational numbers.

Then by definition of rational

$$r = \frac{a}{b} \quad \text{and} \quad s = \frac{c}{d}$$

for some integers a, b, c, d with $b \neq 0$ and $d \neq 0$

Now $r+s = \frac{a}{b}+\frac{c}{d}$

$= \frac{ad+bc}{bd}$

$= \frac{p}{q}$

where $p = ad + bc \in Z$ and $q = bd \in Z$ and $q \neq 0$

Hence r + s is rational.

EXERCISE:

Given any two distinct rational numbers r and s with r < s. Prove that there is a rational number x such that r < x < s.

SOLUTION:

Given two distinct rational numbers r and s such that

$r < s$ .....(1)

Adding r to both sides of (1), we get

$r + r < r + s$

$2r < r + s$

$\Rightarrow \quad r < \frac{r+s}{2}$ .....(2)

Next adding s to both sides of (1), we get

$r + s < s + s$

$\Rightarrow \quad r + s < 2s$

$\Rightarrow \quad \frac{r+s}{2} < s$ .....(3)

Combining (2) and (3), we may write

$r < \frac{r+s}{2} < s$ .....(4)

Since the sum of two rationals is rational, therefore r + s is rational. Also the quotient of a rational by a non-zero rational, is rational, therefore $\frac{r+s}{2}$ is rational and by (4) it lies between r & s.

Hence, we have found a rational number such that r < x < s. (proved)

EXERCISE:

Prove that for all integers a, b and c, if a|b and b|c then a|c.

PROOF:

Suppose a|b and b|c where a, b, c ∈Z. Then by definition of divisibility b=a·r and c=b·s for some integers r and s.

Now $c = b \cdot s$

$= (a \cdot r) \cdot s$ (substituting value of b)

$= a \cdot (r \cdot s)$ (associative law)

$= a \cdot k$ where $k = r \cdot s \in Z$

$\Rightarrow \quad a \mid c$ by definition of divisibility

EXERCISE:

Prove that for all integers a, b and c if a|b and a|c then a|(b+c)

PROOF:

Suppose a|b and a|c where a, b, c $\in Z$

By definition of divides

$b = a \cdot r$ and $c = a \cdot s$ for some r, s $\in Z$

Now

$b + c = a \cdot r + a \cdot s$ (substituting values)

$= a \cdot (r+s)$ (by distributive law)

$= a \cdot k$ where $k = (r + s) \in Z$

Hence $a|(b + c)$ by definition of divides.

EXERCISE:

Prove that the sum of any three consecutive integers is divisible by 3.

PROOF:

Let n, n + 1 and n + 2 be three consecutive integers.

Now

$n + (n + 1) + (n + 2) = 3n + 3$

$= 3(n + 1)$

$= 3 \cdot k$ where $k=(n+1) \in Z$

Hence, the sum of three consecutive integers is divisible by 3.

EXERCISE:

Prove the statement:

There is an integer $n > 5$ such that $2^n - 1$ is prime

PROOF:

Here we are asked to show a single integer for which $2^n - 1$is prime. First of all we will check the integers from 1 and check whether the answer is prime or not by putting these values in $2^n - 1$.when we got the answer is prime then we will stop our process of checking the integers and we note that,

Let $n = 7$, then

$2^n - 1 = 2^7 - 1 = 128 - 1 = 127$

and we know that 127 is prime.

EXERCISE:

Prove the statement: There are real numbers a and b such that

$$\sqrt{a+b} = \sqrt{a} + \sqrt{b}$$

PROOF:

Let $\sqrt{a+b} = \sqrt{a} + \sqrt{b}$

Squaring, we get $a + b = a + b + 2\sqrt{a}\sqrt{b}$

$\Rightarrow \quad 0 = 2\sqrt{a}\sqrt{b}$ canceling a+b

$\Rightarrow \quad 0 = \sqrt{ab}$

$\Rightarrow \quad 0 = ab$ squaring

$\Rightarrow$ either a = 0 or b = 0

It means that if we want to find out the integers which satisfy the given condition then one of them must be zero.

Hence if we let a = 0 and b = 3 then

$$R.H.S = \sqrt{a+b} = \sqrt{0+3}$$

$$R.H.S = \sqrt{3}$$

$$Now\ L.H.S = \sqrt{a} + \sqrt{b}$$

By putting the values of a and b we get

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

$$= \sqrt{3}$$

From above it quite clear that the given condition is satisfied if we take a=0 and b=3.

PROOF BY COUNTER EXAMPLE:

Disprove the statement by giving a counter example.

For all real numbers a and b, if $a < b$ then $a^2 < b^2$.

SOLUTION:

Suppose a = -5 and b = -2

then clearly $-5 < -2$

But $a^2 = (-5)^2 = 25$ and $b^2 = (-2)^2 = 4$

But $25 > 4$

This disproves the given statement.

EXERCISE:

Prove or give counter example to disprove the statement.

For all integers n, $n^2 - n + 11$ is a prime number.

SOLUTION:

The statement is not true

For n = 11

we have , $n^2 - n + 11 = (11)^2 - 11 + 11$

$= (11)^2$

$= (11)(11)$

$= 121$

which is obviously not a prime number.

EXERCISE:

Prove or disprove that the product of any two irrational numbers is an irrational number.

SOLUTION:

We know that $\sqrt{2}$ is an irrational number. Now $(\sqrt{2})(\sqrt{2}) = (\sqrt{2})^2 = 2 = \frac{2}{1}$

which is a rational number. Hence the statement is disproved.

EXERCISE:

Find a counter example to the proposition:

For every prime number n, n + 2 is prime.

SOLUTION:

Let the prime number n be 7, then

$$n + 2 = 7 + 2 = 9$$

which is not prime.

Discrete Structures
week - 11

0.2
3/12/15

Graph Algorithms-1

Graphs
Definition

A *graph* $G=(V,E)$ is a set of vertices V and a set of edges E, where

$E \subseteq V \times V$ *(Subset of cross-product of vertices, called binary relation)*

Example: Let $G=(V, E)$ where
$V = \{a, b, c, d\}$
$E = \{(a, b), (a, d), (b, c), (b, a), (d, c), (d, b), (d, d)\}$

The number of vertices and edges are given by the cardinalities $|V|$ and $|E|$ of the corresponding sets. The sample graph consists of four vertices and seven edges. Thus,
$|V|= 4$ and $|E|=7$

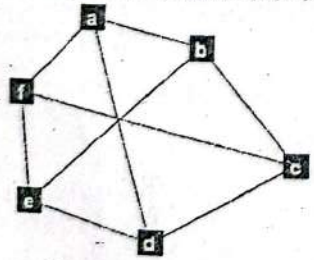
➢ Graphs are usually represented by pictorial diagrams. The vertices can be shown as labeled *circles* or *rectangles*. The edges are depicted as *arcs* or *lines*. Except for some applications, in which distances among vertices are important, the positions of the vertices are, in general, immaterial. Thus, graphs can be shown pictorially in several ways.

Graphs
Undirected Graph

A graph $G=(V,E)$ with vertex set $V=\{v1, v2, v3, \ldots\ldots vn\}$ is called ***undirected*** *if*
$(vi, vj) = (vj, vi)$ *for* $i \neq j$
An undirected graph is sometimes referred to as ***undigraph***.

In pictorial representation of undirected graph, the edges are not assigned any direction.

Example: Figure shows an undirected graph:
$G=(V,E)$, $V=\{a, b, c, d, e, f\}$, $E=\{(a,b), (a,d), (a,f), (b,c), (b,e), (c,f), (c,d), (d,e), (e,f)\}$

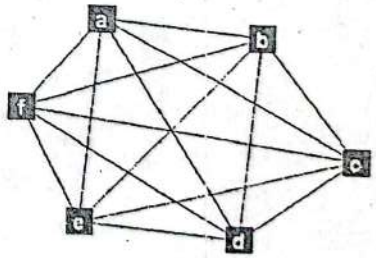


A sample undirected graph

Undirected Graphs
Complete Graph

A graph which has links among all of the vertices in a graph is referred to as ***complete.***

Example: The figure below shows a complete graph.



Sample undirected complete graph

➢ An ***undirected complete graph***, with n vertices, has $n(n-1)/2$ *edges.* The ***space complexity*** of complete graph is $O(n^2)$. A complete graph is ***dense***. By contrast, a graph with space complexity $O(n \lg n)$ is called ***sparse.***

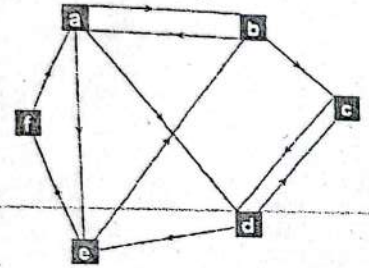
Graphs

Directed Graph

A graph $G=(V,E)$ with vertex set $V=\{v1, v2, v3, \ldots\ldots vn\}$ is called *directed* if $(vi, vj) \neq (vj, vi)$ for $i \neq j$

In other words, the edges (vi, vj) and (vj, vi), associated with any pair of vertices vi, vj, are considered *distinct*. In pictorial representation these are shown with arrows.

Example: The figure below shows a directed graph



A sample directed graph

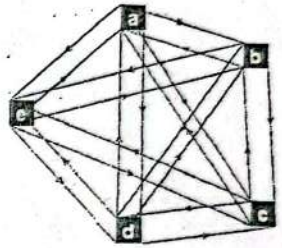
➢The directed graph is often referred to *digraph* or *network*.

Directed Graphs

Complete Graph

A *complete directed graph* has links among all of the vertices.

Example: The figure shows a directed complete graph



A sample directed complete graph

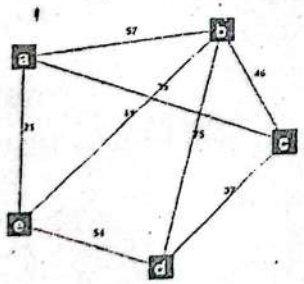
➢ A complete directed graph with n vertices has $n(n-1)$ edges

Weighted Graphs

Undirected Graph

A graph in which *labels* or *values* $w1, w2, w3, \ldots\ldots$ are associated with edges is called *weighted* graph. Weights are typically *costs* or *distances* in different applications of graphs.

Example: The figure shows an example of *weighted undirected* graph.



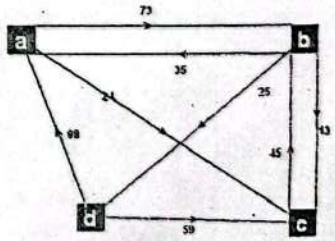
A sample undirected weighted graph

Weighted Graphs

Directed Graph

A weighted graph can also be *directed*. The weights attached to the edges between the same pair of vertices may ,in general, be different

Example: The figure below shows an examples of the *directed weighted* graph. Note that weight of edged from vertex a to vertex b is 73 and that from vertex b to vertex a is 35



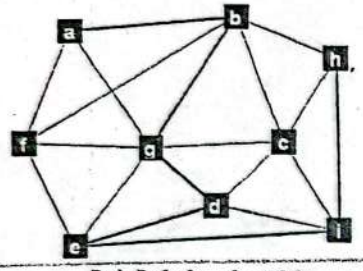
A sample directed weighted graph

Graph Paths

Definition

A ***path*** is a sequence of *adjacent vertices*. Two vertices are called adjacent if they have link or connecting edge. The path is denoted by enclosing the vertices in a pair of square brackets . In a path, the vertices may repeated

Example: The figure below depicts a path $P = [a, b, g, d, e, i, h]$ in a sample graph. The path is shown in bold red lines.



Path P=[a, b, g, d, e, i, h]

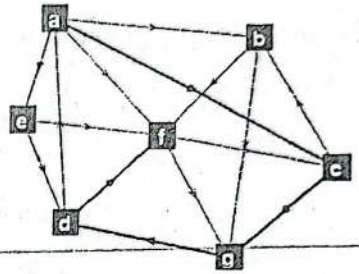
The number of edges connecting the vertices in a path is called *path length*. The path length in the above example is 6.

Graphs Paths

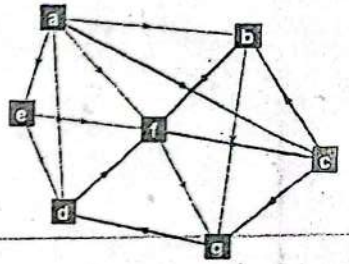
Simple Path

A path is called *simple* if no vertices are repeated; otherwise, the path is referred to as non-simple.

Example: The figures below show *simple* and *non-simple* paths in a graph. The path $P_1=[a, c, g, d, f]$ is simple. The path $P_2=[a, c, g, d, f, c, b, f]$ is non-simple, because it passes through vertices *c*, *f* twice



(a) A simple path P1=[a ,c ,g, d, f]



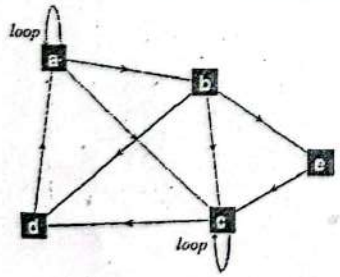
(b) A non- simple path P2=[a ,c ,g, d, f, c, b, f]

Graphs Paths

Loops

A ***loop*** is special path that originates and terminates at a single node, and does not pass through other vertices.

Example: The figure depicts two loops in a graph, at vertices *a* and *c*



A sample graph with two loops

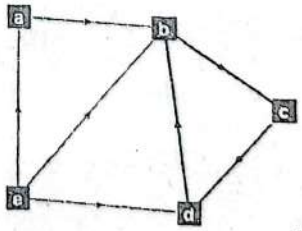
➢The loops are important in certain some applications. For example, loops represent certain states in Finite State Automata

Graphs Paths

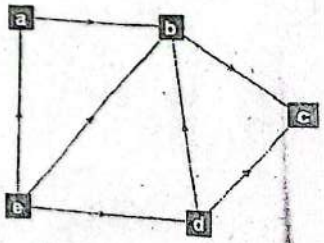
Cyclic and Acyclic Graphs

A path that originates and terminates at the ***same vertex***, and links ***two or more vertices***, is called ***cycle***. If a graph contains a cycle it is called ***cyclic***. By contrast, a graph which contains no cycles is known as ***acyclic***.

Example: In diagram *(i)*, the path $P=[b, c, d, a]$ is a cycle. The diagram *(ii)* represents a acyclic graph.



(i) A sample cyclic graph



*(ii) A sample **acyclic** graph*

Graphs Representation

Adjacency Matrix

One of the standard ways of representing a graph uses a matrix to denote links between pairs of vertices in a graph. The matrix is known as ***adjacency matrix.***

The adjacency matrix can be defined mathematically. Let $G=(V,E)$ be a graph, with $V=\{v1,v2,\ldots..vn\}$, and $E=\{(v1,v2),(v1,v3)\ldots.\}$
where $n=|V|$. The adjacency matrix $A=[a_{ij}]$ as

$$a_{ij} = \begin{cases} 1 \text{ if } (vi, vj) \in E \\ 0 \text{ otherwise} \end{cases}$$

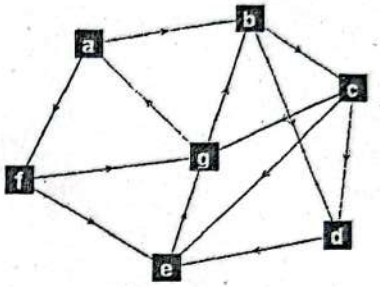
The definition implies that entry in the i^{th} row and j^{th} column of the matrix is 1 if a link exists between the i^{th} and j^{th} vertex; otherwise, it is 0.

➢ The size adjacency matrix is with vertex set V is $|V|x|V|$. Thus space complexity is $\theta(|V|^2)$

Graphs Representation

Adjacency Matrix

Example : Figure *(i)* shows a sample directed graph. Figure *(ii)* shows the adjacency matrix for the graph



	a	b	c	d	e	f	g
a	0	1	0	0	0	1	0
b	0	0	1	1	0	0	0
c	0	0	0	1	1	0	1
d	0	0	0	0	1	0	0
e	0	0	0	0	0	0	1
f	0	0	0	0	1	0	1
g	1	1	0	0	0	0	0

(i) A sample graph *(ii) Adjacency Matrix*

68

Graphs Representation

Adjacency Linked List

A graph can also be represented using ***linked lists***. The list representation consists of an ***array of linked lists***, each corresponding to one vertex. The list stores all of the vertices that are linked to a given vertex.

Let $G=(V,E)$ be a graph, and *Adj(u)* be the linked list corresponding to vertex *u*, then for all *v*,

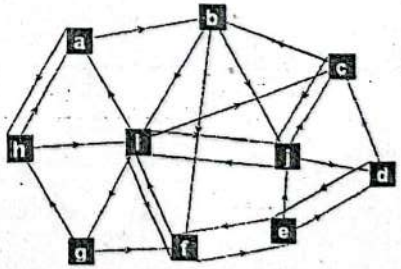
$$v \in Adj(u), \quad \text{if } (u,v) \in E.$$

➢ The vertices belonging to *Adj(u)* are called neighbors of vertex *u*, or ***adjacent vertices***.

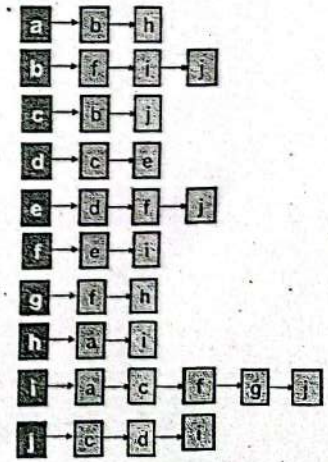
Graphs Representation

Linked List

Example: The diagrams below show a sample graph and its linked list representation



(i) Sample directed graph



(ii) Linked list Representation

Counting

Week 13

Basic Counting Principles

Counting problems are of the following kind:

"**How many** different 8-letter passwords are there?"

"**How many** possible ways are there to pick 11 soccer players out of a 20-player team?"

Most importantly, counting is the basis for computing **probabilities of discrete events**.

("What is the probability of winning the lottery?")

Basic Counting Principles

The sum rule:
If a task can be done in n_1 ways and a second task in n_2 ways, and if these two tasks cannot be done at the same time, then there are $n_1 + n_2$ ways to do either task.

Example:
The department will award a free computer to either a CS student or a CS professor.
How many different choices are there, if there are 530 students and 15 professors?

There are 530 + 15 = 545 choices.

Basic Counting Principles

Generalized sum rule:

If we have tasks T_1, T_2, ..., T_m that can be done in n_1, n_2, ..., n_m ways, respectively, and no two of these tasks can be done at the same time, then there are $n_1 + n_2 + \ldots + n_m$ ways to do one of these tasks.

Basic Counting Principles

The product rule:

Suppose that a procedure can be broken down into two successive tasks. If there are n_1 ways to do the first task and n_2 ways to do the second task after the first task has been done, then there are $n_1 n_2$ ways to do the procedure.

Generalized product rule:

If we have a procedure consisting of sequential tasks $T_1, T_2, \ldots, T_m$ that can be done in $n_1, n_2, \ldots, n_m$ ways, respectively, then there are $n_1 \cdot n_2 \cdot \ldots \cdot n_m$ ways to carry out the procedure.

Basic Counting Principles

Example:

How many different license plates are there that contain exactly three English letters ?

Solution:

There are 26 possibilities to pick the first letter, then 26 possibilities for the second one, and 26 for the last one.

So there are $26 \cdot 26 \cdot 26 = 17576$ different license plates.

Basic Counting Principles

The sum and product rules can also be phrased in terms of set theory.

Sum rule: Let $A_1, A_2, \ldots, A_m$ be disjoint sets. Then the number of ways to choose any element from one of these sets is $|A_1 \cup A_2 \cup \ldots \cup A_m| = |A_1| + |A_2| + \ldots + |A_m|$.

Product rule: Let $A_1, A_2, \ldots, A_m$ be finite sets. Then the number of ways to choose one element from each set in the order $A_1, A_2, \ldots, A_m$ is $|A_1 \times A_2 \times \ldots \times A_m| = |A_1| \cdot |A_2| \cdot \ldots \cdot |A_m|$.

Inclusion-Exclusion

How many bit strings of length 8 either start with a 1 or end with 00?

Task 1: Construct a string of length 8 that starts with a 1.

There is one way to pick the first bit (1),
two ways to pick the second bit (0 or 1),
two ways to pick the third bit (0 or 1),
.
.
.
two ways to pick the eighth bit (0 or 1).

Product rule: Task 1 can be done in $1 \cdot 2^7 = 128$ ways.

Inclusion-Exclusion

Task 2: Construct a string of length 8 that ends with 00.

There are two ways to pick the first bit (0 or 1),
two ways to pick the second bit (0 or 1),
.
.
.
two ways to pick the sixth bit (0 or 1),
one way to pick the seventh bit (0), and
one way to pick the eighth bit (0).

Product rule: Task 2 can be done in $2^6 = 64$ ways.

Inclusion-Exclusion

Since there are 128 ways to do Task 1 and 64 ways to do Task 2, does this mean that there are 192 bit strings either starting with 1 or ending with 00 ?

No, because here Task 1 and Task 2 can be done at **the same time.**

When we carry out Task 1 and create strings starting with 1, some of these strings end with 00.

Therefore, we sometimes do Tasks 1 and 2 at the same time, so **the sum rule does not apply.**

Inclusion-Exclusion

If we want to use the sum rule in such a case, we have to subtract the cases when Tasks 1 and 2 are done at the same time.

How many cases are there, that is, how many strings start with 1 **and** end with 00?

There is one way to pick the first bit (1),
two ways for the second, ..., sixth bit (0 or 1),
one way for the seventh, eighth bit (0).

Product rule: In $2^5 = 32$ cases, Tasks 1 and 2 are carried out at the same time.

Inclusion-Exclusion

Since there are 128 ways to complete Task 1 and 64 ways to complete Task 2, and in 32 of these cases Tasks 1 and 2 are completed at the same time, there are

$128 + 64 - 32 = 160$ ways to do either task.

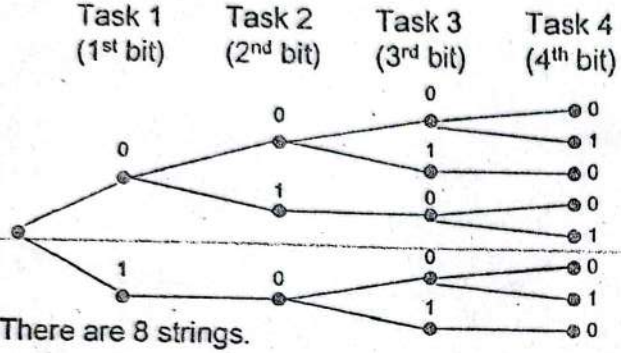
In set theory, this corresponds to sets A_1 and A_2 that are **not** disjoint. Then we have:

$|A_1 \cup A_2| = |A_1| + |A_2| - |A_1 \cap A_2|$

This is called the **principle of inclusion-exclusion.**

Tree Diagrams

How many bit strings of length four do not have two consecutive 1s?



There are 8 strings.

The Pigeonhole Principle

The pigeonhole principle: If (k + 1) or more objects are placed into k boxes, then there is at least one box containing two or more of the objects.

Example 1: If there are 11 players in a soccer team that wins 12-0, there must be at least one player in the team who scored at least twice.

Example 2: If you have 6 classes from Monday to Friday, there must be at least one day on which you have at least two classes.

The Pigeonhole Principle

The generalized pigeonhole principle: If N objects are placed into k boxes, then there is at least one box containing at least $\lceil N/k \rceil$ of the objects.

Example 1: In a 60-student class, at least 12 students will get the same letter grade (A, B, C, D, or F).

Example 2: In a 61-student class, at least 13 students will get the same letter grade.

The Pigeonhole Principle

Example 3: Assume you have a drawer containing a random distribution of a dozen brown socks and a dozen black socks. It is dark, so how many socks do you have to pick to be sure that among them there is a matching pair?

There are two types of socks, so if you pick at least 3 socks, there must be either at least two brown socks or at least two black socks.

Generalized pigeonhole principle: $\lceil 3/2 \rceil = 2$.

D.R.
7/12/15

Definition

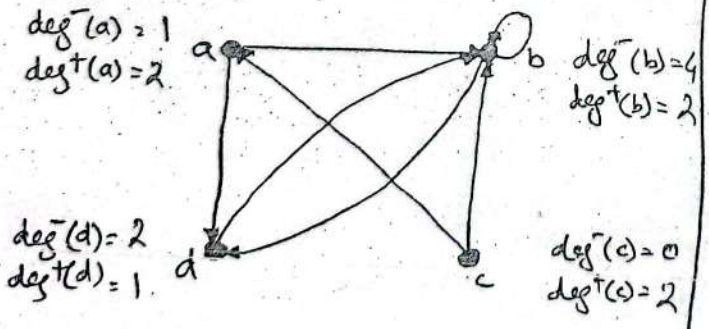
- when (u, v) is an edge of the graph G with directed edges, u is said to be adjacent to v, and v is said to be adjacent from u.
  The vertex u is called the initial vertex of (u, v), and v is called the terminal vertex of (u, v).
- The initial vertex and terminal vertex of a loop are the same.

Definition

Indegree: Denoted by $\deg^-(v)$ is the number of edges with v as their terminal vertex.

Outdegree: Denoted by $\deg^+(v)$ is the number of edges with v as their initial vertex.

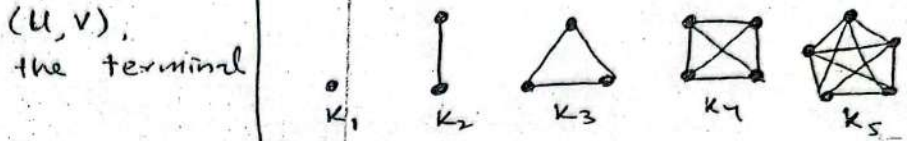
Exp: What are in-degree and out-degrees of the vertices a, b, c, d in this graph.



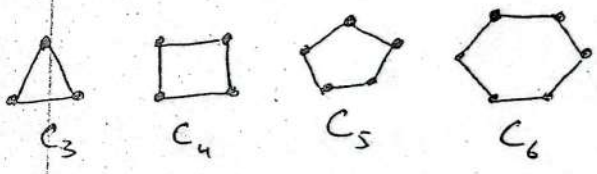
1-D of.

Special Graphs

Definition: The complete graph on n vertices, denoted by K_n, is the simple graph that contains exactly one edge between each pair of distinct vertices.



Def: The Cycle C_n, $n \geq 3$, consists of n vertices $v_1, v_2, v_3, \ldots, v_n$ and edges $\{v_1, v_2\}, \{v_2, v_3\}, \ldots, \{v_{n-1}, v_n\}, \{v_n, v_1\}$.

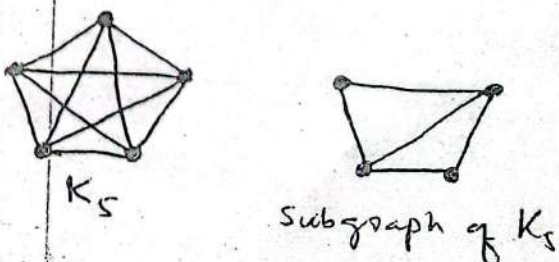


Operations on Graph

Subgraph: A subgraph of a graph $G = (V, E)$ is a graph $H = (W, F)$ where $W \subseteq V$ and $F \subseteq$

H is a valid graph, so we cannot remove any endpoints of remaining edges when creating it

Ex:

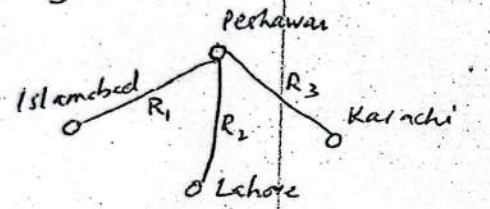


Directed Graph

- A directed graph $G=(V,E)$ consists of a set V of vertices and a set E of edges that are ordered pairs of elements in V.

For each $e \in E$, $e(u,v)$ where $u,v \in V$

- An edge e is a loop if $e = (u,u)$ for $u \in V$.
- A simple graph is just like a directed graph, but with no specified direction of its edges.



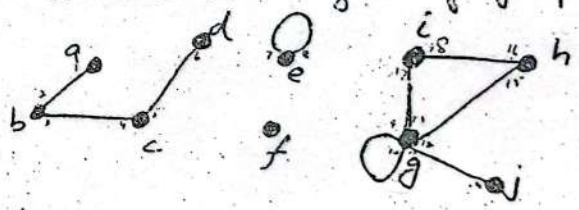
Graph Terminology

- Two vertices u and V is an undirected graph G are called adjacent (or neighbours) in G if $\{u,v\}$ is an edge in G.
- If $e = \{u,v\}$, the edge "e" is called incident with the vertices u and V. The edge "e" is also said to connect u and v. The vertices u and v are called endpoints of the edge u and v.

Degree of a Vertex

- It is the number of edges incident with it, except that a loop at a vertex contribute twice to the degree of that verti
- Degree of a vertex is display by counting the lines that touches that vertex.
- The degree of a vertex V is denoted by $\deg(V)$.
- A vertex of degree 0 is called Isolated, since it is not adjacent to any vertex.
- A vertex with a loop at it has atleast degree 2.
- A vertex ~~with~~ of degree 1 is called Pendant.

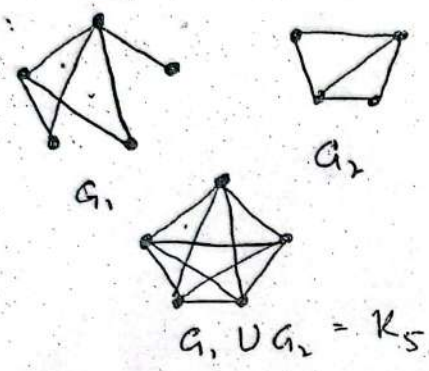
Example: Identify the vertices:
(1) Isolated (2) Pendant
(3) Maximum Degree of graph



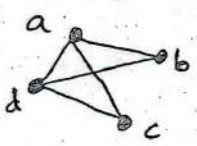
Soln: (1) f is isolated
(2) a, d, j are pendant
(3) Maximum degree = 5

Union: The Union of two simple graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is the simple graph with vertex set $V_1 \cup V_2$ and edge set $E_1 \cup E_2$.
The union of G_1 and G_2 is denoted by $G_1 \cup G_2$

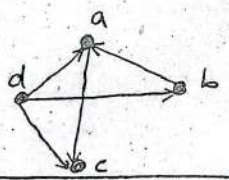
Exp:



Representing Graphs



Vertex	Adjacent Vertices
a	b, c, d
b	a, d
c	a, d
d	a, b, c

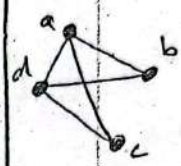


Initial Vertex	Terminal Vertex
a	c
b	a
c	
d	a, b, c

Representing Graphs

The Graph can also be represent through adjacency Matrix
For adjacency Matrix $A = [a_{ij}]$,
$a_{ij} = 1$ if $\{V_i, V_j\}$ is an edge of
$a_{ij} = 0$ otherwise

Exp: What is the Adjacency Matrix for the following graph G base on order of vertices a, b, c, d



Graph G

Soln

$$A_G = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

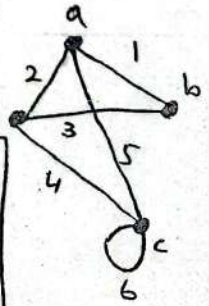
Representing Graphs

The Graph can also be represented through incidence Matrix of G. For an Incidence matrix $M = [m_{ij}]$,

$m_{ij} = 1$ if edge e_j is incident with V_i

$m_{ij} = 0$ otherwise.

Exp: What is the incidence Matrix M for the following graph G based on the order of vertices a, b, c, d and edges 1, 2, 3, 4, 5, 6?

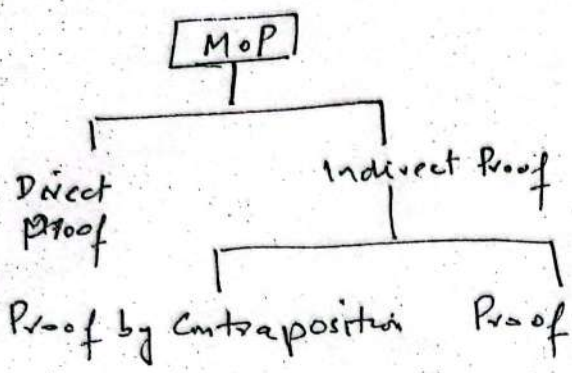


Soln

$$M = \begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

n x m
V E

Method of Proof



① Direct Proof

- The implication $P \rightarrow q$ can be proved by showing that if "P" is true, the "q" must also be true. This shows that the combination P true and q false never occurs.
- A proof of this form is called a directed proof.

② Proof by Contra Position

- The method of Proof by Contraposition may be summarised as:

① Express the statement in the form if "p" then "q"

② Rewrite this statement in the Contrapositive form if not q

③ Prove the contrapositive by direct proof.

③ Proof by Contradiction

- The method of Proof by Contradiction may be summerized as follows

① Suppose the statement to be proved is false

② Show that this supposition leads logically to a contradiction.

③ Conclude that the statement to be proved is true.

Remember:

Definition: An integer n;
- is even if there exist an integer $K \Rightarrow n = 2K$
- is odd if there exist an integer $K \Rightarrow n = 2K+1$

Q: Prove that the square of even integer is even.

Soln: Suppose that n is an even integer, then $n = 2K$ —— (1)

Now Taking the square of both sides of (1).

$$n^2 = (2K)^2$$
$$n^2 = 4K^2$$
$$n^2 = 2(2K^2)$$
$$\Rightarrow n^2 = 2p \text{ where } p = 2k$$

Hence n^2 is even (proved)

Q. Prove that if "n" is an integer and n^3+5 is odd, then "n" is even.

Soln: Suppose n is an odd integer since, a product of two odd integers is odd, therefore $n^2 = n \cdot n$ is odd and $n^3 = n^2 \cdot n$ is odd.

Since a sum of two odd integers is even therefore n^2+5 is even

Thus we have to prove that if n is odd then n^3+5 is even. since this is the contrapositive of the given Conditional Statement. So the given statement is True.

Q: Prove that if n is an integer and n^3+5 is odd, and then n is even using Contradiction method.

Soln, Suppose that n^3+5 is odd and n is not even (odd).

Since n is odd and the product of Two odd number is odd, it follows that n^2 is odd.

$n^3 = n^2 \cdot n$ is odd, Further, since the Difference of two odd Number is even, it follows that

$$5 = (n^3+5) - n^3 \text{ is even}$$

But this is a Contradiction, therefore the supposition that n^3+5 and n are both odd is wrong and so the given Statement is True.