**Chapter 1**

Table of Contents

[Chapter 1.1 4](#_Toc66759869)

[Propositions 4](#_Toc66759870)

[Logical Operators 5](#_Toc66759871)

[Conditionals 6](#_Toc66759872)

[Expressing Conditionals 7](#_Toc66759873)

[Bi-Conditionals 9](#_Toc66759874)

[Precedence of Logical Operators 10](#_Toc66759875)

[Bit-Strings 10](#_Toc66759876)

[Chapter 1.2 11](#_Toc66759877)

[Translating into Logical Expressions 11](#_Toc66759878)

[System Specifications 12](#_Toc66759879)

[Logic Puzzles 14](#_Toc66759880)

[Logical Gates 17](#_Toc66759881)

[Chapter 1.3 18](#_Toc66759882)

[Tautology 18](#_Toc66759883)

[Contradiction 18](#_Toc66759884)

[Contingency 18](#_Toc66759885)

[Logical Equivalence 19](#_Toc66759886)

[Logical Equivalence Rules 20](#_Toc66759887)

[Logical Equivalence Rules (Conditionals) 21](#_Toc66759888)

[Propositional Satisfiability 22](#_Toc66759889)

[Chapter 1.4 23](#_Toc66759890)

[Predicates 23](#_Toc66759891)

[Quantifiers 24](#_Toc66759892)

[Negation of Quantifiers 25](#_Toc66759893)

[Logical Expressions Using Quantifiers 26](#_Toc66759894)

[Chapter 1.6 27](#_Toc66759895)

[Rules of Inference 27](#_Toc66759896)

[Resolution 30](#_Toc66759897)

[Rules of Inference for Quantifiers 31](#_Toc66759898)

[Chapter 1.7 33](#_Toc66759899)

[Theorems 33](#_Toc66759900)

[Proofs 33](#_Toc66759901)

[Axioms 33](#_Toc66759902)

[Lemmas 33](#_Toc66759903)

[Corollaries 33](#_Toc66759904)

[Conjectures 34](#_Toc66759905)

[Methods of Proving Theorems 34](#_Toc66759906)

[Direct Proofs 34](#_Toc66759907)

[Proof by Contraposition 35](#_Toc66759908)

[Proof by Contradiction 35](#_Toc66759909)

[Mistakes in Proofs 36](#_Toc66759910)

## Chapter 1.1

### Propositions

A proposition declares a fact. It is a statement that has a truth value, meaning it can either be true or false, but not both. Truth values are denoted by T/F or 1/0.

Propositional Variables are variables used to represent propositions. For example

= “Donald is a good guy.”

Commonly, , , and are used as propositional variables, but any alphabet can be used. Logical comparisons can be made using the variables. The area of logic that deals with propositions is known as propositional logic or propositional calculus.

Sentences can be converted to propositions. For example,

is not a proposition.

However, if , and , then

is a proposition

### Logical Operators

There are 6 logical operators that deal with propositions.

1. Negation (­­­­­­­­­­) – This flips the truth value of a proposition.

p = “Thor is a good guy.”

p = “Thor is not a good guy.” / “Thor is a bad guy.”

2. Conjunction or AND () – This combines the truth value of two propositions.

q = “Thor is rich.”

pq = “Thor is a good guy and Thor is rich.”

The new proposition created, which is a combination of two propositions, is known as a compound proposition.

3. Disjunction or OR () – This is similar to a conjunction except it is only false when both propositions are false. Otherwise, it is true.

4. Exclusive Disjunction or XOR () – This returns a truth value of if both propositions have a truth value of or both propositions have a truth value of .

5. Conditionals or Implication () – Conditionals are written as , where is known as the hypothesis or premise, and is known as the conclusion or consequence. This essentially means that if a correct hypothesis is given, an incorrect conclusion cannot be reached ( implies ). So, the output is always , except in the scenario where and .

6. Bi-Conditionals () – Bi-Conditionals are written as . They represent the fact that if is a precondition for , then is also a precondition for . So, they return a value of when one of the two propositions has a value of , but the other does not.

Truth Table:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

Operators can be divided into two groups, binary operators and unary operators. Binary operators are operators that deal with 2 inputs. Unary operators are operators that deal with 1 input. Negation is the only unary operator.

### Conditionals

= “Richard learns discrete mathematics.”

= “Richard will get a good job.”

If , then .

If Richard learns discrete mathematics, then he will get a good job.

For , is sufficient.

For Richard to get a good job, it is sufficient for him to learn discrete mathematics.

unless .

Richard will get a good job, unless he does not learn discrete mathematics.

### Expressing Conditionals

* If , then
* implies
* If ,
* only if
* is sufficient for
* A sufficient condition for is
* if
* whenever
* when
* is necessary for
* A necessary condition for is
* follows from
* unless

Original Proposition:

Converse Proposition:

Inverse Proposition:

Contrapositive Proposition:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

Original Proposition Contrapositive Proposition

Converse Propositions Inverse Proposition

This is known as equivalence. Two compound propositions based on the same simple propositions and giving the same meaning, are equivalent.

The home team wins, whenever it is raining.

whenever .

= “It is raining.”

= “The home team wins.”

If , then .

Converse Proposition: If , then .

If the home team wins, then it is raining.

Inverse Proposition: If , then .

If is it not raining, then the home team does not win.

Contrapositive Proposition: If , then .

If the home team does not win, then it is not raining.

To change the proposition type, identify the hypothesis and conclusion from the original proposition, and assign them propositional values first. This will make the process easier.

### Bi-Conditionals

If is a precondition for , then is also a precondition for .

If the Pythagoras theorem is true, then the triangle is right-angled. If the triangle is right-angled, the Pythagoras theorem is true.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  | ✓ |  |
|  |  |  |  | ✕ |  |
|  |  |  |  | ✕ |  |
|  |  |  |  | ✓ |  |

= “You can take the flight.”

= “You bought a ticket.”

= “If you can take the flight, then you have bought a ticket.”

= “If you have bought a ticket, then you can take the flight.”

= “You can take the flight if and only if you have bought a ticket.”

if and only if .

### Precedence of Logical Operators

|  |  |
| --- | --- |
| Operator | Value |
|  | 1 |
|  | 2 |
|  | 3 |
|  | 4 |
|  | 5 |

Operators with lower values have higher precedence.

### Bit-Strings

A bit-string is a collection of bits.

-

\_\_\_\_\_\_\_

(NOT BINARY ADDITION)

For large bit-strings, dividing the string into blocks of 4 bits makes it visually easier to work with.

## Chapter 1.2

### Translating into Logical Expressions

1. “You can access the internet from campus only if you are a computer science major or you are not a freshman.”

= “You can access the internet from campus.”

= “You are a computer science major.”

= “You are not a freshman.”

2. “You cannot ride the roller coaster if you are under 4 feet tall, unless you are older than 16.”

= “You cannot ride the roller coaster.”

= “You are under 4 feet tall.”

= “You are older than 16.”

3. “The automated reply cannot be sent when the file system is full.”

= “The automated reply cannot be sent.”

= “The file system is full.”

### System Specifications

A consistent system should not contain conflicting requirements that could be used to derive a contradiction. A system cannot be developed using inconsistent specifications.

Example:

1. The message is stored in the buffer or retransmitted.

2. The message is not stored in the buffer.

3. If the message is stored in the buffer, then it is retransmitted.

= “The message is stored in the buffer.”

= “The message is retransmitted.”

1.

2.

3.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

The truth table shows that there is at least one circumstance under which all the specifications give a true result. This indicates that the system is consistent.

Another specification is added.

4. The message is not retransmitted. ()

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

The system is no longer consistent, since there are no circumstances under which all the specifications give a true result.

### Logic Puzzles

Example 1:

Two people, A and B, are either a knight or a knave. A knight will always tell the truth, and a knave will always lie. A says B is a knight. B says the two of them are different.

= “A is a knight.”

= “B is a knight.”

1. “B is a knight.” ()

2. “A and B are different.” ()

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

If A is a knight (), then the system is not consistent. If A is a knave () and B is a knight ), only then is the system consistent.

Example 2:

A boy and a girl are playing. Their father tells them “At least one of you has mud on their forehead.” He then asks both of them twice if they have mud on their own forehead. Each can see the other’s forehead, and they both answer at the same time, and answer truthfully.

= “Boy has a muddy forehead.”

= “Girl has a muddy forehead.”

1. “At least one of you has mud on their forehead.” ()

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Assuming the father is honest ():

If the girl has a clean forehead (), then the boy will immediately know that he must have a muddy forehead ().

So, the first time, the girl says she does not know and the boy says yes (If and , then ).

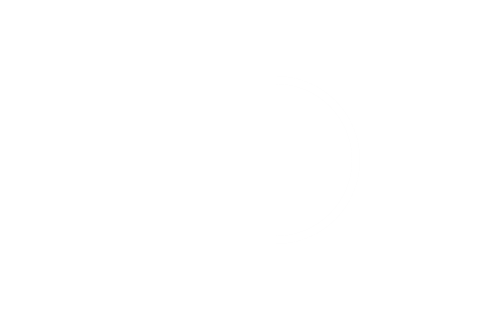
Since the boy said ‘Yes’ the first time, the girl can infer that her own forehead must be clean, since that is the only way the boy could have been sure his own forehead was not clean. So, the second time, the boy says ‘Yes’ and the girl says ‘No’.

The same thinking applies in the opposite scenario, where the girl has a muddy forehead but the boy does not.

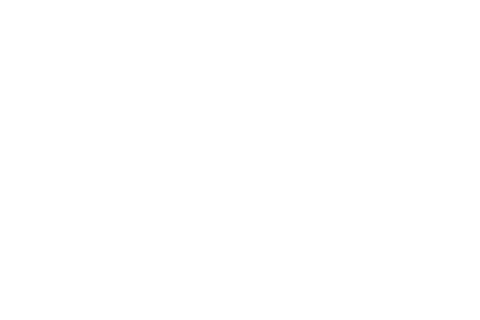
If however, they both have mud on their foreheads (), then the first time both will say they do not know, since they can see the other has mud on their forehead but cannot know about themselves (Since if , is still allowed to be and vice versa). However, since the other has answered that they do not know, they will find out that they both have mud on their foreheads. If one of them did but the other did not, then the one with the muddy forehead would immediately have known that they have a muddy forehead, as explained in the previous scenarios. So, the second time, both will answer yes.

Assuming that the father is dishonest, (), both children will answer ‘Yes’ the first time. Each will see the other has a clean forehead () and will assume that they must have a muddy forehead (since they are assuming ). However, since they both answered ‘Yes’ the first time, they will immediately know that the father was lying. As explained in the first scenario, the only way for one of them to know that they have a muddy forehead on the first guess, is if the other had a clean forehead. Since they both answered ‘Yes’ the first time, they must both have seen that the other had a clean forehead. The second time, they will both answer ‘No’ again, this time knowing correctly that they both have clean foreheads.

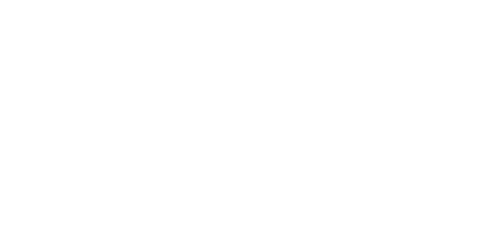
### Logical Gates

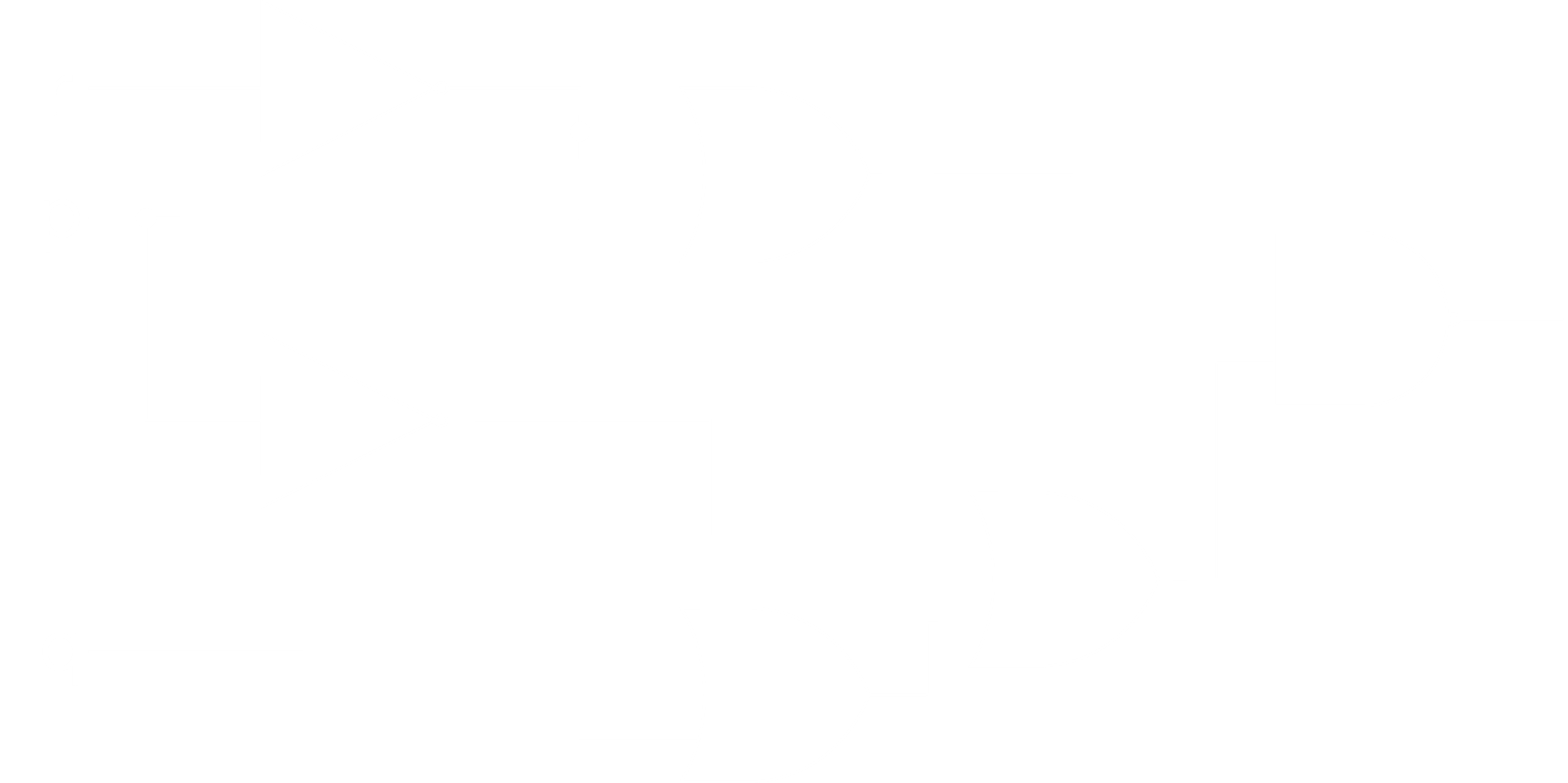
And Gate ():

Or Gate ():



Negation Gate ():





## Chapter 1.3

### Tautology

A compound proposition that is always true, no matter what the truth values of the propositional variables that occur in it. ()

## Contradiction

A compound proposition that is always false. ()

## Contingency

A compound proposition that is neither a tautology nor a contradiction.

### Logical Equivalence

2 compound propositions are logically equivalent if they have the same truth values for all of the truth values their propositional variables.

 ✓

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

✓

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Logical Equivalence Rules:

|  |  |
| --- | --- |
|  | Identity Laws |
|  |
|  | Domination Laws |
|  |
|  | Idempotent Laws |
|  |
|  | Double Negation Law |
|  | Commutative Laws |
|  |
|  | Associative Laws |
|  |
|  | Distributive Law |
|  |
|  | De Morgan’s Laws |
|  |
|  | Absorption Laws |
|  |
|  | Negation Laws |
|  |

Logical Equivalence Rules (Conditionals):

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

by 2 D

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

### Propositional Satisfiability

A compound proposition is ‘satisfiable’ when there is at least one scenario under which it is true. Otherwise, it is unsatisfiable.

## Chapter 1.4

### Predicates

Predicates help in converting non-propositional statements into propositions.

Computer is under attack.

Predicate - Computer is under attack.

Variable -

Once is given a value, has a truth value and becomes a proposition.

The statements that describe valid input are known as pre-conditions.

The conditions that the output should satisfy when the program has run are known as post-conditions.

This program interchanges the values of two variables and .

temp =

=

= temp

Both variables must have values before the program can be run. Thus, the precondition is:

The output should ensure that the values have been interchanged so:

### Quantifiers

Quantifiers express the extent to which a predicate is true. Quantifications include “all”, “some”, “many”, “none” and “few”.

Universal Quantification () is when a predicate is true for every element under consideration. An element for which the predicate is false is known as the counter example of the universal quantification of that predicate. “for all”, “for every”, “all of”, “for each”, “given any”, “for arbitrary”, “for each” and “for any” indicate universal quantification.

Existential Quantification () is when a predicate is true for at least one of the elements under consideration. “there exists”, “for some”, “for at least one” and “there is” indicate existential quantification.

Both and have higher precedence than all other logical operators.

Predicate Calculus is the area of logic that deals with predicates and quantifiers.

### Negation of Quantifiers

= There are honest politicians.

= There is at least one honest politician.

= There is not a single honest politician.

= Every politician is dishonest.

= Americans eat cheeseburgers.

= All Americans eat cheeseburgers.

= Not all Americans eat cheeseburgers.

= There is at least one American who does not eat cheeseburgers.

The rules for negations for quantifiers are called De Morgan’s laws of quantifiers.

### Logical Expressions Using Quantifiers

= “For every person, if he is a student of this class, then he has studied calculus.”

= “ is a student of this class”.

= “ has studied calculus.”

= “Some student in this class has visited Mexico.”

= “Student in this class has visited Mexico.”

When considering the local domain (students of this class only),

When considering the universal domain (all people),

Let = is a student in this class.

makes no sense. This statement is true even if is false.

Thus, .

= has visited Canada.

= has visited Mexico.

Local Domain

Universal Domain

In general,

Local Domain

Universal Domain

Local Domain

Universal Domain

## Chapter 1.6

### Rules of Inference

Infer means to deduce or conclude something from evidence and reasoning rather than from explicit statements.

Proofs are valid arguments that establish the truth of mathematical statements. Arguments refer to a sequence of statements that lead to a conclusion. By valid, we mean that the conclusion (final statement) of the argument must follow from the truth of the premises (preceding statements) of the argument. Thus, an argument is valid if the truth of all its premises implies that the conclusion is true, i.e., the conclusion is true if all the premises are true.

An argument is valid if and only if it is impossible for all the premises to be true and the conclusion to be false ( is a tautology).

Rules of Inference:

|  |  |  |  |
| --- | --- | --- | --- |
| Premises | | Conclusion | Rule Name |
|  |  |  | Modus Ponens |
|  |  |  | Modus Tollens |
|  |  |  | Hypothetical Syllogism |
|  |  |  | Disjunctive Syllogism |
|  | |  | Addition |
|  | |  | Simplification |
|  |  |  | Conjunction |
|  |  |  | Resolution |

Example 1:

“If you have a current password, then you can log onto the network.”

= “You have a current password.”

= “You can log onto the network.”

AND

(Modus Ponens)

If one of the premises were false however, then the conclusion would be false.

Example 2:

Premises:

1. “It is not sunny this afternoon, and it is colder than yesterday.”

2. “We will go swimming only if it is sunny.”

3. “If we do not go swimming, then we will take a canoe trip.”

4. “If we take a canoe trip, then we will be home by sunset.”

Conclusion:

“We will be home by sunset.”

= “It is sunny this afternoon.”

= “It is colder than yesterday.”

= “We will go swimming.”

= “We will take a canoe trip.”

= “We will be home by sunset.”

1.

(Addition)

2.

AND

(Modus Tollens)

3.

AND

(Modus Ponens)

4.

AND

(Modus Ponens)

### Resolution

Resolution is a rule of inference based on the tautology,

Based on this, if is false,

This is just disjunctive syllogism.

### Rules of Inference for Quantifiers

Universal Instantiation - is true, where is a member of the domain, given that the premise is true.

Universal Generalization – Given that is true for all random elements within the domain, must be true.

Existential Instantiation – If is true, there must be some value of within the domain, that is not arbitrary, for which is true.

Existential Generalization – If is true for a known, non-arbitrary element within the domain, then must be true.

Example 1:

1. “Everyone in this class has taken a course in computer science.”

2. “Marla is a student in this class.”

= “ is in this class.”

= “ has taken a course in computer science.”

1.

Universal Instantiation

2.

and

Modus Ponens

Thus, conclusion = “Marla has taken a course in computer science.”

Example 2:

“A student in this class has not read the book.”

“Everyone in this class passed the exam.”

= “ is in this class.”

= “ has read the book.”

= “ passed the exam.”

1.

Existential Instantiation

Simplification

Simplification

2.

Universal Instantiation

and

Modus Ponens

and

Conjunction

Existential Generalization

## Chapter 1.7

### Theorems

These are statements that can be shown to be true. The term is usually reserved for statements that are considered important. Less important ‘theorems’ are called propositions.

### Proofs

These are valid arguments that establish the truth of a theorem.

### Axioms

These are statements that are assumed to be true. They may be stated using simple terms that make it evidently true, i.e. it does not need to be separately proven.

### Lemmas

These are less important theorems that are helpful in the proof of other results.

### Corollaries

These are theorems that can be established directly from other theorems that have already been proven, i.e. they do no need their own proofs. For example, if , then a corollary is that .

### Conjectures

These are statements that are being proposed to be true. This is usually done on the basis of partial evidence, expert intuition or logical argument. If a conjecture is proven true, it becomes a theorem. If it is proven false, if stops being a conjecture.

### Methods of Proving Theorems

1. Direct Proofs

2. Proof by Contraposition

3. Proof by Contradiction

4. Vacuous and Trivial Proofs

(Vacuous proofs are proofs for statements that are incorrect in themselves, but can still be proven – all unicorns have 3 horns; well, there aren’t any that DON’T so…. Trivial proofs are proofs that are stupidly simple – all unicorns have 1 horn; well, duh. These are not separately included under this topic.)

#### Direct Proofs

Statement: If is an odd integer, is also an odd integer.

If is an odd integer, , where is some integer.

where is some integer.

Thus is also odd.

Statement: If and are prefect squares, then is also a perfect square.

Let and , where and are some integers.

where is some integer.

Thus, the statement is true.

#### Proof by Contraposition

This is based on the fact that a statement and its contraposition are logically equivalent. Thus, can be proven by proving , i.e. if the opposite conclusion leads to the opposite hypothesis, the original statement must be true.

Statement: If is odd, then is odd.

The contrapositive would be if is even, is also even.

Let .

is even when is even, thus proving by contraposition that the original statement was true.

#### Proof by Contradiction

This is based on the fact that if the conclusion is assumed to be false, and a contradiction is found, then the conclusion must be true.

Statement: If is odd, then is odd.

If is even, .

Thus, is even, which contradicts the hypothesis, meaning cannot be even. It is proven by contradiction that is odd.

### Mistakes in Proofs

A statement alongside with ‘proof’ that it is true is given. Incorrect assumptions or contradictions within the proof must be found in order to show that the proof is wrong. The statement itself may or may not be correct.

Statement: is an even integer, whenever is an even integer.

Proof: Suppose is even. Then for some integer . Let for some integer . This shows that is even.

Mistake: No argument was presented as to why can be written as . This is the same as simply saying is even, with no proof given.