Discrete Mathematics and its Applications

Ngày 8 tháng 9 năm 2012

Logic

(Introduction)

“The universe cannot be read until we have learnt the language and become familiar with the characters in which it is written. It is written in mathematical language... without which means it is humanly impossible to comprehend a single word"

Galileo Galiley

Logic is the bridge netween the language the universe is written in and mathematics.

In this lecture we shall learn the basic entities of logic:

Logic

(Introduction)

"The universe cannot be read until we have learnt the language and become familiar with the characters in which it is written. It is written in mathematical language... without which means it is humanly impossible to comprehend a single word"

Galileo Galiley

Logic is the bridge netween the language the universe is written in and mathematics.

In this lecture we shall learn the basic entities of logic:

1. *Propositions*

Logic

(Introduction)

“The universe cannot be read until we have learnt the language and become familiar with the characters in which it is written. It is written in mathematical language... without which means it is humanly impossible to comprehend a single word"

Galileo Galiley

Logic is the bridge netween the language the universe is written in and mathematics.

In this lecture we shall learn the basic entities of logic:

1. *Propositions*
2. *Boolean Variables*

Logic

(Introduction)

“The universe cannot be read until we have learnt the language and become familiar with the characters in which it is written. It is written in mathematical language... without which means it is humanly impossible to comprehend a single word"

Galileo Galiley

Logic is the bridge netween the language the universe is written in and mathematics.

In this lecture we shall learn the basic entities of logic:

1. *Propositions*
2. *Boolean Variables*
3. *Logical (boolean) operators.*

Logic

(Introduction)

"The universe cannot be read until we have learnt the language and become familiar with the characters in which it is written. It is written in mathematical language... without which means it is humanly impossible to comprehend a single word"

Galileo Galiley

Logic is the bridge netween the language the universe is written in and mathematics.

In this lecture we shall learn the basic entities of logic:

1. *Propositions*
2. *Boolean Variables*
3. *Logical (boolean) operators.*
4. *Truth tables.*

Propositions

Definition

A **Proposition** *is a statement of a fact which is either true or false but not both.*

Propositions

Definition

A **Proposition** *is a statement of a fact which is either true or false but not both.*

(**Examples)**

Propositions

Definition

A **Proposition** *is a statement of a fact which is either true or false but not both.*

(Examples)

1. *Today is Saturday*

Propositions

Definition

A **Proposition** *is a statement of a fact which is either true or false but not both.*

(Examples)

1. *Today is Saturday*
2. *It is raining today.*

Propositions

Definition

A **Proposition** *is a statement of a fact which is either true or false but not both.*

(Examples)

1. *Today is Saturday*
2. *It is raining today.*
3. *if n is an integer then* $(2n+1)^2 \bmod 8 = 1$.

Propositions

Definition

A **Proposition** *is a statement of a fact which is either true or false but not both.*

(Examples)

1. *Today is Saturday*
2. *It is raining today.*
3. *if n is an integer then* $(2n+1)^2 \bmod 8 = 1$.
4. *if n is an odd prime number then* $2^{n-1} \bmod n = 1$

Propositions

Definition

A **Proposition** *is a statement of a fact which is either true or false but not both.*

(Examples)

1. *Today is Saturday*
2. *It is raining today.*
3. *if n is an integer then* $(2n+1)^2 \bmod 8 = 1$.
4. *if n is an odd prime number then* $2^{n-1} \bmod n = 1$
5. *There are no positive integers* x, y, z *satisfying the equality* $x^5 + y^5 = z^5$

Propositions

Definition

A **Proposition** *is a statement of a fact which is either true or false but not both.*

(Examples)

1. *Today is Saturday*
2. *It is raining today.*
3. *if n is an integer then* $(2n+1)^2 \bmod 8 = 1$.
4. *if n is an odd prime number then* $2^{n-1} \bmod n = 1$
5. *There are no positive integers* x, y, z *satisfying the equality* $x^5 + y^5 = z^5$
6. *There are infinitely many prime numbers q such that* $q = 4p + 1$ *where p is prime.*

Non-propositions

Here are some examples of sentences which are not propositions.

Non-propositions

Here are some examples of sentences which are not propositions.

1. *Do not drive over the speed limit.*

Non-propositions

Here are some examples of sentences which are not propositions.

1. *Do not drive over the speed limit.*
2. *Do not use your cell phone while driving.*

Non-propositions

Here are some examples of sentences which are not propositions.

1. *Do not drive over the speed limit.*
2. *Do not use your cell phone while driving.*
3. $a^2 + b^2 = c^2$

Non-propositions

Here are some examples of sentences which are not propositions.

1. *Do not drive over the speed limit.*
2. *Do not use your cell phone while driving.*
3. $a^2 + b^2 = c^2$
4. *Hoang lives in Hanoi. He owns a xe may. Hoang fixes all xe mays in Hanoi belonging to people that do not fix their own xe may.*

Non-propositions

Here are some examples of sentences which are not propositions.

1. *Do not drive over the speed limit.*
2. *Do not use your cell phone while driving.*
3. $a^2 + b^2 = c^2$
4. *Hoang lives in Hanoi. He owns a xe may. Hoang fixes all xe mays in Hanoi belonging to people that do not fix their own xe may.*

Non-propositions

Here are some examples of sentences which are not propositions.

1. *Do not drive over the speed limit.*
2. *Do not use your cell phone while driving.*
3. $a^2 + b^2 = c^2$
4. *Hoang lives in Hanoi. He owns a xe may. Hoang fixes all xe mays in Hanoi belonging to people that do not fix their own xe may.*

- *1 and 2 are not propositions as they do not state a fact.*

Non-propositions

Here are some examples of sentences which are not propositions.

1. *Do not drive over the speed limit.*
2. *Do not use your cell phone while driving.*
3. $a^2 + b^2 = c^2$
4. *Hoang lives in Hanoi. He owns a xe may. Hoang fixes all xe mays in Hanoi belonging to people that do not fix their own xe may.*

- *1 and 2 are not propositions as they do not state a fact.*
- *3 can be both true and false, depending on the values of* a, b, c

Non-propositions

Here are some examples of sentences which are not propositions.

1. *Do not drive over the speed limit.*
2. *Do not use your cell phone while driving.*
3. $a^2 + b^2 = c^2$
4. *Hoang lives in Hanoi. He owns a xe may. Hoang fixes all xe mays in Hanoi belonging to people that do not fix their own xe may.*

- *1 and 2 are not propositions as they do not state a fact.*
- *3 can be both true and false, depending on the values of a, b, c*
- *4 is a bit more intricate. Hoang cannot fix his own xe may since he fixes only those belonging to people that do not fix their own xe may but if he does not fix his own xe may then he is fixing it.*

Logic operation

Logic operations and compound propositions were introduced by the English mathematician George Boole in 1848. This lay the foundation of developing the digital computer 100 years later.

Logic operation

Logic operations and compound propositions were introduced by the English mathematician George Boole in 1848. This lay the foundation of developing the digital computer 100 years later.

We shall denote propositions by letters: $p, q, r, s, \ldots$.

Definition

A **Boolean Variable** *is a variable whose domain is propositions and range just two values, commonly denoted by* **TRUE, FALSE**. *Sometimes we also use* $\{1, 0\}$.

Logic operation

Logic operations and compound propositions were introduced by the English mathematician George Boole in 1848. This lay the foundation of developing the digital computer 100 years later.

We shall denote propositions by letters: $p, q, r, s, \ldots$.

Definition

A **Boolean Variable** *is a variable whose domain is propositions and range just two values, commonly denoted by* **TRUE, FALSE**. *Sometimes we also use* $\{1, 0\}$.

Comment

Almost all programming languages include boolean variables.

Logic operations

Question

What can be done with a single boolean variable that has only two values?

Logic operations

Question

What can be done with a single boolean variable that has only two values?

Answer

Not much more than a light switch, it can be off or on. But combining an array of boolean variables, like 32 *in common processors yields* 2^{32} *different patterns, That is more than* $4,000,000,000$ *patterns!*

Logic operations

Question

What can be done with a single boolean variable that has only two values?

Answer

Not much more than a light switch, it can be off or on. But combining an array of boolean variables, like 32 *in common processors yields* 2^{32} *different patterns, That is more than* $4,000,000,000$ *patterns!*

Combining boolean variable is done with logic or boolean operators.

Example (**Compound Propositions)**

Example (**Compound Propositions)**

1. *Phuong's PC does* **not** *run UNIX.*

Example (**Compound Propositions)**

1. *Phuong's PC does* **not** *run UNIX.*
2. *The speed limit in Hanoi for xe may is 40 km/hour* **and** *for trucks 30 km/hour.*

Example (**Compound Propositions)**

1. *Phuong's PC does* **not** *run UNIX.*
2. *The speed limit in Hanoi for xe may is 40 km/hour* **and** *for trucks 30 km/hour.*
3. **If** *n is prime* **then if** $a < n$ **then** $a^{n-1} \bmod n = 1$

Example (**Compound Propositions)**

1. *Phuong's PC does* **not** *run UNIX.*
2. *The speed limit in Hanoi for xe may is 40 km/hour* **and** *for trucks 30 km/hour.*
3. **If** *n is prime* **then if** $a < n$ **then** $a^{n-1} \bmod n = 1$
4. **if** *n is prime* **or** $n = p^k$, *p prime* **then** *there is a finite field of order n.*

Example (**Compound Propositions**)

1. *Phuong's PC does* **not** *run UNIX.*
2. *The speed limit in Hanoi for xe may is 40 km/hour* **and** *for trucks 30 km/hour.*
3. **If** *n is prime* **then if** $a < n$ **then** $a^{n-1} \bmod n = 1$
4. **if** *n is prime* **or** $n = p^k$, *p prime* **then** *there is a finite field of order n.*

Discussion

Each proposition is composed of one or more propositions connected by key words:

Example (**Compound Propositions**)

1. *Phuong's PC does* **not** *run UNIX.*
2. *The speed limit in Hanoi for xe may is 40 km/hour* **and** *for trucks 30 km/hour.*
3. **If** *n is prime* **then if** $a < n$ **then** $a^{n-1} \bmod n = 1$
4. **if** *n is prime* **or** $n = p^k$*, p prime* **then** *there is a finite field of order n.*

Discussion

Each proposition is composed of one or more propositions connected by key words:

1. **not***: number 1 has one proposition.*

Example (**Compound Propositions**)

1. *Phuong's PC does* **not** *run UNIX.*
2. *The speed limit in Hanoi for xe may is 40 km/hour* **and** *for trucks 30 km/hour.*
3. **If** *n is prime* **then if** $a < n$ **then** $a^{n-1} \bmod n = 1$
4. **if** *n is prime* **or** $n = p^k$, *p prime* **then** *there is a finite field of order n.*

Discussion

Each proposition is composed of one or more propositions connected by key words:

1. **not**: *number 1 has one proposition.*
2. **and**: *number 2 has two propositions*

Example (**Compound Propositions)**

1. *Phuong's PC does* **not** *run UNIX.*
2. *The speed limit in Hanoi for xe may is 40 km/hour* **and** *for trucks 30 km/hour.*
3. **If** *n is prime* **then if** $a < n$ **then** $a^{n-1} \bmod n = 1$
4. **if** *n is prime* **or** $n = p^k$, *p prime* **then** *there is a finite field of order n.*

Discussion

Each proposition is composed of one or more propositions connected by key words:

1. **not**: *number 1 has one proposition.*
2. **and**: *number 2 has two propositions*
3. **or:** *number 4 has 3 propositions*

Example (**Compound Propositions**)

1. *Phuong's PC does* **not** *run UNIX.*
2. *The speed limit in Hanoi for xe may is 40 km/hour* **and** *for trucks 30 km/hour.*
3. **If** *n is prime* **then if** $a < n$ **then** $a^{n-1} \bmod n = 1$
4. **if** *n is prime* **or** $n = p^k$, *p prime* **then** *there is a finite field of order n.*

Discussion

Each proposition is composed of one or more propositions connected by key words:

1. **not**: *number 1 has one proposition.*
2. **and**: *number 2 has two propositions*
3. **or:** *number 4 has 3 propositions*
4. **if then** *number 3 has 3 propositions.*

1. **not** is a unary operator, $\neg p$ inverts the truth value of p.

1. **not** is a unary operator, $\neg p$ inverts the truth value of p.
2. **and** is a binary operator, $p \wedge q$ is true only when both p and q are true.

1. **not** is a unary operator, $\neg p$ inverts the truth value of p.
2. **and** is a binary operator, $p \wedge q$ is true only when both p and q are true.
3. **or** is a binary operator, $p \vee q$ is false only when both are false.

1. **not** is a unary operator, $\neg p$ inverts the truth value of p.
2. **and** is a binary operator, $p \wedge q$ is true only when both p and q are true.
3. **or** is a binary operator, $p \vee q$ is false only when both are false.
4. **implies** (if then) is a binary operator $p \rightarrow q$ is false only when p is true and q is false.

1. **not** is a unary operator, $\neg p$ inverts the truth value of p.
2. **and** is a binary operator, $p \wedge q$ is true only when both p and q are true.
3. **or** is a binary operator, $p \vee q$ is false only when both are false.
4. **implies** (if then) is a binary operator $p \rightarrow q$ is false only when p is true and q is false.

Truth tables *are used to describe these and also the truth values of compound propositions.*

1. **not** is a unary operator, $\neg p$ inverts the truth value of p.
2. **and** is a binary operator, $p \wedge q$ is true only when both p and q are true.
3. **or** is a binary operator, $p \vee q$ is false only when both are false.
4. **implies** (if then) is a binary operator $p \rightarrow q$ is false only when p is true and q is false.

Truth tables *are used to describe these and also the truth values of compound propositions.*

1. All these operators can be physically implemented electronically.

1. **not** is a unary operator, $\neg p$ inverts the truth value of p.
2. **and** is a binary operator, $p \wedge q$ is true only when both p and q are true.
3. **or** is a binary operator, $p \vee q$ is false only when both are false.
4. **implies** (if then) is a binary operator $p \rightarrow q$ is false only when p is true and q is false.

Truth tables *are used to describe these and also the truth values of compound propositions.*

1. All these operators can be physically implemented electronically.
2. These are the building blocks of micro-processors and many other systems.

1. **not** is a unary operator, $\neg p$ inverts the truth value of p.
2. **and** is a binary operator, $p \wedge q$ is true only when both p and q are true.
3. **or** is a binary operator, $p \vee q$ is false only when both are false.
4. **implies** (if then) is a binary operator $p \rightarrow q$ is false only when p is true and q is false.

Truth tables *are used to describe these and also the truth values of compound propositions.*

1. All these operators can be physically implemented electronically.
2. These are the building blocks of micro-processors and many other systems.

1. **not** is a unary operator, $\neg p$ inverts the truth value of p.
2. **and** is a binary operator, $p \wedge q$ is true only when both p and q are true.
3. **or** is a binary operator, $p \vee q$ is false only when both are false.
4. **implies** (if then) is a binary operator $p \to q$ is false only when p is true and q is false.

Truth tables *are used to describe these and also the truth values of compound propositions.*

1. All these operators can be physically implemented electronically.
2. These are the building blocks of micro-processors and many other systems.

There are other binary operators. Truth tables will help us understand how to construct them.

Truth Tables

Truth table for the unary operator **not**:

p	$\neg p$
T	F
F	T

Truth Tables

Truth table for the unary operator **not**:

p	$\neg p$
T	F
F	T

Truth tables for the binary operators $\wedge$ $\vee$ $\rightarrow$:

p	q	$p \wedge q$		$p \vee q$		$p \rightarrow q$
T	T	T		T		T
F	T	F		T		T
T	F	F		T		F
F	F	F		F		T

Example

We wish to build the truth table for the compound proposition:
$(p \rightarrow q) \wedge (\neg p \rightarrow q)$

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

Question

Question

1. *How many possible binary operators are there?*

Question

1. *How many possible binary operators are there?*
2. *How many non-trivial binary operators are there?*

Question

1. *How many possible binary operators are there?*
2. *How many non-trivial binary operators are there?*
3. *How many rows will be needed in the truth table of a compound proposition with 5 different boolean variables?*

Question

1. *How many possible binary operators are there?*
2. *How many non-trivial binary operators are there?*
3. *How many rows will be needed in the truth table of a compound proposition with* 5 *different boolean variables?*

Comment

Here is a list of commonly used binary operators, their names and description:

Question

1. *How many possible binary operators are there?*
2. *How many non-trivial binary operators are there?*
3. *How many rows will be needed in the truth table of a compound proposition with* 5 *different boolean variables?*

Comment

Here is a list of commonly used binary operators, their names and description:

1. **nor**, *the reverse of* **or**, $p \downarrow q$ *is true only when both p and q are false.*

Question

1. *How many possible binary operators are there?*
2. *How many non-trivial binary operators are there?*
3. *How many rows will be needed in the truth table of a compound proposition with 5 different boolean variables?*

Comment

Here is a list of commonly used binary operators, their names and description:

1. **nor**, *the reverse of* **or**, $p \downarrow q$ *is true only when both p and q are false.*
2. **nand**, *the reverse of* **and**. $p \mid q$ *is false only when both p and q are true.*

Question

1. *How many possible binary operators are there?*
2. *How many non-trivial binary operators are there?*
3. *How many rows will be needed in the truth table of a compound proposition with 5 different boolean variables?*

Comment

Here is a list of commonly used binary operators, their names and description:

1. **nor**, *the reverse of* **or**, $p \downarrow q$ *is true only when both p and q are false.*
2. **nand**, *the reverse of* **and**. $p \mid q$ *is false only when both p and q are true.*
3. **xor** *(exclusive or)* $p \oplus q$ *is true only when they are different (one is true and the other is false).*

Question

1. *How many possible binary operators are there?*
2. *How many non-trivial binary operators are there?*
3. *How many rows will be needed in the truth table of a compound proposition with 5 different boolean variables?*

Comment

Here is a list of commonly used binary operators, their names and description:

1. **nor**, *the reverse of* **or**, $p \downarrow q$ *is true only when both p and q are false.*
2. **nand**, *the reverse of* **and**. $p \mid q$ *is false only when both p and q are true.*
3. **xor** *(exclusive or)* $p \oplus q$ *is true only when they are different (one is true and the other is false).*
4. **implies** $p \rightarrow q$ *is false only when p = true and q = false.*

Question

1. *How many possible binary operators are there?*
2. *How many non-trivial binary operators are there?*
3. *How many rows will be needed in the truth table of a compound proposition with 5 different boolean variables?*

Comment

Here is a list of commonly used binary operators, their names and description:

1. **nor**, *the reverse of* **or**, $p \downarrow q$ *is true only when both p and q are false.*
2. **nand**, *the reverse of* **and**. $p \mid q$ *is false only when both p and q are true.*
3. **xor** *(exclusive or)* $p \oplus q$ *is true only when they are different (one is true and the other is false).*
4. **implies** $p \to q$ *is false only when p = true and q = false.*

Question

1. *How many possible binary operators are there?*
2. *How many non-trivial binary operators are there?*
3. *How many rows will be needed in the truth table of a compound proposition with 5 different boolean variables?*

Comment

Here is a list of commonly used binary operators, their names and description:

1. **nor**, *the reverse of* **or**, *$p \downarrow q$ is true only when both p and q are false.*
2. **nand**, *the reverse of* **and**. *$p \mid q$ is false only when both p and q are true.*
3. **xor** *(exclusive or) $p \oplus q$ is true only when they are different (one is true and the other is false).*
4. **implies** *$p \rightarrow q$ is false only when p = true and q = false.*

Question

1. *How many possible binary operators are there?*
2. *How many non-trivial binary operators are there?*
3. *How many rows will be needed in the truth table of a compound proposition with 5 different boolean variables?*

Comment

Here is a list of commonly used binary operators, their names and description:

1. **nor**, *the reverse of* **or**, $p \downarrow q$ *is true only when both p and q are false.*
2. **nand**, *the reverse of* **and**. $p \mid q$ *is false only when both p and q are true.*
3. **xor** *(exclusive or)* $p \oplus q$ *is true only when they are different (one is true and the other is false).*
4. **implies** $p \rightarrow q$ *is false only when p = true and q = false.*

Logic calculations

Applying multiple logic operations is similar to using arithmetic operations. We need precedence rules. To understand why consider the expressions $p \vee q \wedge s$.

Logic calculations

Applying multiple logic operations is similar to using arithmetic operations. We need precedence rules. To understand why consider the expressions $p \vee q \wedge s$.
Now suppose both p and q are true and s is false. The truth value of this expression will be true if we first evaluate $q \wedge s$. But if we first calculate $p \vee q$ the result is false. So we need precedences. Here they are:

Logic calculations

Applying multiple logic operations is similar to using arithmetic operations. We need precedence rules. To understand why consider the expressions $p \vee q \wedge s$.
Now suppose both p and q are true and s is false. The truth value of this expression will be true if we first evaluate $q \wedge s$. But if we first calculate $p \vee q$ the result is false. So we need precedences. Here they are:

Operator	Precedence
$\neg$	1
$\wedge$	2
$\vee$	3
$\rightarrow$	4

Logic Computations Rules

Equivalence	Name
$p \vee F \equiv p; \quad p \wedge T \equiv p$	Identity
$p \vee T \equiv T; \quad p \wedge F \equiv F$	Domination
$p \vee p \equiv p; \quad p \wedge p \equiv p$	Idempotent
$p \vee q \equiv q \vee p; \quad p \wedge q \equiv q \wedge p$	commutative
$p \vee (q \vee r) \equiv (p \vee q) \vee r$ $p \wedge (q \wedge r) \equiv (p \wedge (q \wedge r)$	Associative
$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r$	Distributive
$\neg(p \wedge q) \equiv \neg p \vee \neg q$ $\neg(p \vee q) \equiv \neg p \wedge \neg q$	De Morgan

Bảng: Basic computation laws

summary

In this lecture we studied:

1. propositions

summary

In this lecture we studied:

1. propositions
2. compound propositions

summary

In this lecture we studied:

1. propositions
2. compound propositions
3. logical operators

In this lecture we studied:

1. propositions
2. compound propositions
3. logical operators
4. truth tables

summary

In this lecture we studied:

1. propositions
2. compound propositions
3. logical operators
4. truth tables
5. computation rules

summary

In this lecture we studied:

1. propositions
2. compound propositions
3. logical operators
4. truth tables
5. computation rules
6. we learned how to use truth tables to evaluate compound propositions

summary

In this lecture we studied:

1. propositions
2. compound propositions
3. logical operators
4. truth tables
5. computation rules
6. we learned how to use truth tables to evaluate compound propositions
7. we conclude with two entertaining puzzles.

A simple puzzle

Trung, Hóa and Tuán had the same Pho for lunch at the same restaurant. They asked Hà to guess what they ate and where. To challenge Há they decided that each will tell Há two facts and at least one of the facts will be true..

- Trung: We ate Pho bò tái at Pho-24

A simple puzzle

Trung, Hóa and Tuán had the same Pho for lunch at the same restaurant. They asked Hà to guess what they ate and where. To challenge Há they decided that each will tell Há two facts and at least one of the facts will be true..

- Trung: We ate Pho bò tái at Pho-24
- Hóa: We ate Pho gà at Quàn ăn Ngon.

A simple puzzle

Trung, Hóa and Tuán had the same Pho for lunch at the same restaurant. They asked Hà to guess what they ate and where. To challenge Há they decided that each will tell Há two facts and at least one of the facts will be true..

- Trung: We ate Pho bò tái at Pho-24
- Hóa: We ate Pho gà at Quàn ăn Ngon.
- Tuán: We ate at Cha Cá but definitely not Pho bò tái

A simple puzzle

Trung, Hóa and Tuán had the same Pho for lunch at the same restaurant. They asked Hà to guess what they ate and where. To challenge Há they decided that each will tell Há two facts and at least one of the facts will be true..

- Trung: We ate Pho bò tái at Pho-24
- Hóa: We ate Pho gà at Quàn ǎn Ngon.
- Tuán: We ate at Cha Cá but definitely not Pho bò tái

A simple puzzle

Trung, Hóa and Tuán had the same Pho for lunch at the same restaurant. They asked Hà to guess what they ate and where. To challenge Há they decided that each will tell Há two facts and at least one of the facts will be true..

- Trung: We ate Pho bò tái at Pho-24
- Hóa: We ate Pho gà at Quàn ăn Ngon.
- Tuán: We ate at Cha Cá but definitely not Pho bò tái

What did they eat and where?

To solve this puzzle we introduce five propositions:

- a: they ate Pho bò tái

To solve this puzzle we introduce five propositions:

- a: they ate Pho bò tái
- b: they ate at Pho-24

To solve this puzzle we introduce five propositions:

- a: they ate Pho bò tái
- b: they ate at Pho-24
- c: they ate Pho gà

To solve this puzzle we introduce five propositions:

- a: they ate Pho bò tái
- b: they ate at Pho-24
- c: they ate Pho gà
- d: they ate at Quàn ăn Ngon

To solve this puzzle we introduce five propositions:

- a: they ate Pho bò tái
- b: they ate at Pho-24
- c: they ate Pho gà
- d: they ate at Quàn ăn Ngon
- e: they ate at Cha Cá

To solve this puzzle we introduce five propositions:

- a: they ate Pho bò tái
- b: they ate at Pho-24
- c: they ate Pho gà
- d: they ate at Quàn ăn Ngon
- e: they ate at Cha Cá

The compound proposition describing their three claims is:
$(a \vee b) \wedge (c \vee d) \wedge (\neg a \vee e) =$ **true** *which when expanded yields:*

$(a \wedge c \wedge \neg a) \vee (a \wedge c \wedge e) \vee (a \wedge d \wedge \neg a) \vee (a \wedge d \wedge e) \vee (b \wedge c \wedge \neg a) \vee (b \wedge c \wedge e) \vee (b \wedge d \wedge \neg a) \vee (b \wedge d \wedge e) =$ **true**.

To solve this puzzle we introduce five propositions:

- a: they ate Pho bò tái
- b: they ate at Pho-24
- c: they ate Pho gà
- d: they ate at Quàn ăn Ngon
- e: they ate at Cha Cá

The compound proposition describing their three claims is:
$(a \vee b) \wedge (c \vee d) \wedge (\neg a \vee e)$ = **true** *which when expanded yields:*

$(a \wedge c \wedge \neg a) \vee (a \wedge c \wedge e) \vee (a \wedge d \wedge \neg a) \vee (a \wedge d \wedge e) \vee (b \wedge c \wedge \neg a) \vee (b \wedge c \wedge e) \vee (b \wedge d \wedge \neg a) \vee (b \wedge d \wedge e)$ = **true**.

To solve this puzzle we introduce five propositions:

- a: they ate Pho bò tái
- b: they ate at Pho-24
- c: they ate Pho gà
- d: they ate at Quàn ăn Ngon
- e: they ate at Cha Cá

The compound proposition describing their three claims is:
$(a \vee b) \wedge (c \vee d) \wedge (\neg a \vee e) =$ **true** *which when expanded yields:*

$(a \wedge c \wedge \neg a) \vee (a \wedge c \wedge e) \vee (a \wedge d \wedge \neg a) \vee (a \wedge d \wedge e) \vee (b \wedge c \wedge \neg a) \vee (b \wedge c \wedge e) \vee (b \wedge d \wedge \neg a) \vee (b \wedge d \wedge e) =$ **true**.

Among the eight triples there must be at least one which is true. For instance, $(b \wedge c \wedge e)$ would mean that they ate at two different restaurants which is false.

To solve this puzzle we introduce five propositions:

- a: they ate Pho bò tái
- b: they ate at Pho-24
- c: they ate Pho gà
- d: they ate at Quàn ăn Ngon
- e: they ate at Cha Cá

The compound proposition describing their three claims is:
$(a \vee b) \wedge (c \vee d) \wedge (\neg a \vee e) =$ **true** *which when expanded yields:*

$(a \wedge c \wedge \neg a) \vee (a \wedge c \wedge e) \vee (a \wedge d \wedge \neg a) \vee (a \wedge d \wedge e) \vee (b \wedge c \wedge \neg a) \vee (b \wedge c \wedge e) \vee (b \wedge d \wedge \neg a) \vee (b \wedge d \wedge e) =$ **true**.

Among the eight triples there must be at least one which is true. For instance, $(b \wedge c \wedge e)$ would mean that they ate at two different restaurants which is false.

The only triple which is true is $(b \wedge c \wedge \neg a)$ which says that they ate Pho gà at Pho-24.

A classic logic puzzle

A logician is lost in a jungle in Africa. This jungle has two tribes of cannibals. One tribe always answer every question truthfully while the other tribe always lies.

A classic logic puzzle

A logician is lost in a jungle in Africa. This jungle has two tribes of cannibals. One tribe always answer every question truthfully while the other tribe always lies.

The logician meets a cannibal in the jungle. The cannibal grants our logician his final wish: to ask him one question. If as a result of the answer the logician can determine to which tribe the cannibal belongs his life will be spared.

A classic logic puzzle

A logician is lost in a jungle in Africa. This jungle has two tribes of cannibals. One tribe always answer every question truthfully while the other tribe always lies.

The logician meets a cannibal in the jungle. The cannibal grants our logician his final wish: to ask him one question. If as a result of the answer the logician can determine to which tribe the cannibal belongs his life will be spared.

Design a question that will guarantee to save the logician's life.