Notes for Discrete Mathematics (2. Ed) by Rosen: Chapter 1: Logic and Proofs

Andreas Hansen

May 2025

Chapter 1.1: Propositional Logic (pg. 1-13)

Propositions: (pg. 2-3)

Definition: A proposition is a declarative sentence that is either true or false, but not both.

- **Examples of propositions:**
  - *"Washington, D.C., is the capital of the United States of America."*, which is true.
  - *"Toronto is the capital of Canada."*, which is false.
  - $1 + 1 = 2$, which is true.
  - $2 + 2 = 3$, which is false.
- **Examples of sentence that are NOT proposition:**
  - *"What time is it?"*, which is a question.
  - *"Read this carefully."*, which is a command.
  - $x + 1 = 2$, which is neither true or false because we have not assigned a value or values to x
  - $x + y = z$, which is neither true or false because we have not assigned a value or values to x, y or z.
- Use letters to denote propositional variables(or sentential variables), that is, variables that represent propositions, just as letters are used to denote numerical variables.
- The conventional letters used for propositional variables are p, q, r, s, ...
- **Truth values:** The truth value of a proposition is true, denoted by T, if it is a true proposition, and the truth value of a proposition is false, denoted by F, if it is a false proposition.
- Propositions that cannot be expressed in terms of simpler propositions are called atomic propositions.
- New propositions, called compound propositions, are formed from existing propositions using logical operators.

Logic operator: Negation (pg. 3-4)

Definition: Negation

Let p be a proposition. The negation of p, denoted by $\neg p$ (also written as $\overline{p}$), is the statement:

"It is not the case that p."

The truth value of $\neg p$ is the opposite of the truth value of p.

Common Notations for Negation and Their Contexts

Notation	Read as	Commonly Used In
$\neg p$	not p	Mathematical logic, philosophy, discrete math
$\overline{p}$	not p	Boolean algebra, electrical engineering
$\sim p$	not p	Older logic texts, set theory
$-p$	not p (context-dependent)	Some logic systems (rare), can cause ambiguity
p'	p prime / not p	Boolean functions, circuit notation
Np	not p	Some formal logic systems (rare)
$!p$	not p	Programming languages (C, Python, JavaScript)

Examples of Propositions and Their Negations

Original Proposition	Formal Logical Negation	Natural English Negation
Michael's PC runs Linux.	It is not the case that Michael's PC runs Linux.	Michael's PC does not run Linux.
The number 5 is even.	It is not the case that the number 5 is even.	The number 5 is odd.
$2 + 2 = 4$.	It is not the case that $2 + 2 = 4$.	$2 + 2$ does not equal 4.
Cats are mammals.	It is not the case that cats are mammals.	Cats are not mammals.
Vandana's smartphone has at least 32 GB of memory	It is not the case that Vandana's smartphone has at least 32 GB of memory.	Vandana's smartphone has less than 32 GB of memory.

Truth Table for the Negation of a Proposition

p	$\neg p$
T	F
F	T

Remark 1

The truth value of $\neg p$ is always the opposite of the truth value of p. If p is true, then $\neg p$ is false; if p is false, then $\neg p$ is true.

Remark 2

The negation of a proposition can also be considered the result of the operation of the negation operator on a proposition. The negation operator constructs a new proposition from a single existing proposition.

Logic operator (Connective): Conjuction (pg. 4)

Definition: Conjunction

Let p and q be propositions. The conjunction of p and q, denoted by $p \wedge q$, is the proposition:

"p and q."

The conjunction $p \wedge q$ is true when both p and q are true; it is false otherwise.

Common Notations for Conjunction and Their Contexts

Notation	Read as	Commonly Used In
$p \wedge q$	p and q	Mathematical logic, discrete math, philosophy
$p \cdot q$	p and q	Boolean algebra (uses multiplication symbol)
pq	p and q	Boolean expressions (abbreviated multiplication)
$p * q$	p and q	Some algebraic logic, circuit simplification tools
$p\&\&q$	p and q	Programming languages (C, Java, JavaScript, Python)

Examples of Propositions and Their Conjunctions

Proposition p	**Proposition** q	**Conjunction** $p \wedge q$ **(Plain English)**
It is raining.	It is cold.	It is raining and it is cold.
5 is an odd number.	2 is an even number.	5 is an odd number and 2 is an even number.
Rebecca's PC has more than 16 GB free hard disk space	The processor in Rebecca's PC runs faster than 1 GHz.	Rebecca's PC has more than 16 GB free hard disk space, and its processor runs faster than 1 GHz.

Truth Table for the Conjunction $p \wedge q$

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

Remark 1

The conjunction $p \wedge q$ is true only when both p and q are true. In all other cases, the conjunction is false.

Remark 2

Note that in logic the word "*but*" sometimes is used instead of "*and*" in a conjunction. For example, the statement "*The sun is shining, but it is raining*" is another way of saying "*The sun is shining and it is raining.*"

Logic operator (Connective): Disjunction (pg. 4-5)

Definition: Disjunction

Let p and q be propositions. The disjunction of p and q, denoted by $p \vee q$, is the proposition:

"p or q."

The disjunction $p \vee q$ is false when both p and q are false and is true otherwise.

Common Notations for Disjunction and Their Contexts

Notation	Read as	Commonly Used In
$p \vee q$	p or q	Mathematical logic, discrete math, philosophy
$p + q$	p or q	Boolean algebra (addition-like behavior)
$p \mid q$	p or q	Sometimes used in older logic notation or vertical bar systems
p|q	p or q	Programming (e.g., Python, JavaScript: bitwise OR)
p||q	p or q	Programming (C, Java, etc.: logical OR)

Examples of Propositions and Their Disjunctions

Proposition p	**Proposition** q	**Disjunction** $p \vee q$ **(Plain English)**
A student has taken calculus.	A student has taken introductory computer science.	A student who has taken calculus or introductory computer science can take this class.
Rebecca's PC has at least 16 GB of free hard disk space.	Rebecca's PC's processor runs faster than 1 GHz.	Rebecca's PC has at least 16 GB free hard disk space, or its processor runs faster than 1 GHz.

Truth Table for the Disjunction $p \vee q$

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

Remark 1

The disjunction $p \vee q$ is an inclusive "or," meaning it is true when either p, q, or both are true. It is only false when both are false.

Logic operator (Connective): Exclusive or (XOR) (pg. 5-6)

Definition: Exclusive Or (XOR)

Let p and q be propositions. The exclusive or of p and q, denoted by $p \oplus q$, is the proposition:

"p or q, but not both."

The exclusive or $p \oplus q$ is true when exactly one of p or q is true, and false otherwise.

Common Notations for Exclusive Or (XOR) and Their Contexts

Notation	Read as	Commonly Used In
$p \oplus q$	p XOR q	Mathematical logic, discrete math, CS theory
$p \veebar q$	p exclusive or q	Set theory, logic (less common)
$p \neq q$	p not equal to q	Logic circuits (when p, q are Boolean)
$p\hat{}\ q$	p XOR q	Programming (Python, C++, JavaScript — bitwise XOR)
`xor`(p, q)	XOR function of p and q	Pseudocode, some programming languages and logic textbooks

Examples of Propositions and Their Exclusive Or (XOR)

Proposition p	Proposition q	Exclusive Or $p \oplus q$ (Plain English)
A student can have a salad with dinner.	A student can have soup with dinner.	A student can have soup or salad, but not both, with dinner.
I will use all my savings to travel to Europe.	I will use all my savings to buy an electric car.	I will use all my savings to travel to Europe or to buy an electric car, but not both.

Truth Table for the Exclusive Or $p \oplus q$

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

Remark 1

The exclusive or $p \oplus q$ is true only when exactly one of p or q is true. It is false when both are true or both are false.

Remark 2

In Boolean logic and programming, XOR is often used to represent binary decisions where two conditions are mutually exclusive.

Conditional Statements: (pg. 6-9)

Definition: Conditional Statement (Implication)

Let p and q be propositions. The conditional statement $p \to q$ is the proposition:

"if p, then q."

The conditional statement $p \to q$ is false when p is true and q is false, and true otherwise.

In the conditional statement $p \to q$, p is called the **hypothesis** (or antecedent or premise) and q is called the **conclusion** (or consequence)

Remark 1

The statement $p \to q$ is called a *conditional statement* because $p \to q$ asserts that q is true on the condition that p holds.

Remark 2

A conditional statement is also called an **implication**.

Common Notations for Conditional Statements (Implications)

Notation	Read as	Commonly Used In
$p \to q$	if p, then q	Standard logic, discrete mathematics
$p \Rightarrow q$	p implies q	Proofs, math writing, theoretical CS
$p \supset q$	p implies q	Older logic notation, set-theoretic logic
$if\ p,\ then\ q$	verbal form	Natural language, programming explanations
q if p	reverse phrasing	Common in math and spoken logic

Common Ways to Express a Conditional Statement ($p \to q$)

Alternate Wording	Meaning or Context
If p, then q	Standard form of a conditional
p implies q	Common in mathematics and logic
If p, q	Shortened form, sometimes used in prose
p only if q	Equivalent to: if not q, then not p
p is sufficient for q	p guarantees q (sufficient condition)
A sufficient condition for q is p	Same as above
q if p	Equivalent to "if p, then q" (reversed order)
q whenever p	Conditional triggered by p
q when p	Same as "if p, then q"
q is necessary for p	If not q, then not p (i.e., $p \to q$)
A necessary condition for p is q	Same as above
q follows from p	Logical consequence
q unless $\neg p$	Equivalent to "if p, then q"
q provided that p	Again, equivalent to "if p, then q"

Examples and Interpretations of Conditional Statements ($p \to q$)

Statement (English)	Interpretation
If I am elected, then I will lower taxes.	If the politician is elected, voters expect them to lower taxes. If not elected, there is no expectation either way. The only breach occurs if elected and fails to lower taxes — this is the case where p is true and q is false.
If you get 100% on the final, then you will get an A.	Scoring 100% should guarantee an A. If you don't score 100%, the outcome is uncertain. The only contradiction is if you get 100% and don't receive an A.
You can receive an A only if your score on the final is at least 90%.	You must score at least 90% to get an A. That is, receiving an A implies you scored $\geq$ 90%. This expresses a necessary condition: $A \to$ (score $\geq$ 90)
If Maria learns discrete mathematics, then she will find a good job.	This expresses a natural relationship: learning discrete math is seen as a path to employment. The implication is only false if she learns it and still does not find a job. Otherwise, it's considered true.
If it is sunny, then we will go to the beach.	This conditional is understood to mean that sunshine causes the beach trip. It is true unless it's sunny and we don't go — that's the only logically false case.
If Juan has a smartphone, then $2 + 3 = 5$.	This is true from the definition of a conditional statement, because its conclusion is true. (The truth value of the hypothesis does not matter then.)
If Juan has a smartphone, then $2 + 3 = 6$.	The conditional statement is true if Juan does not have a smartphone, even though $2+3 = 6$ is false
If $2 + 2 = 4$, then $x := x + 1$.	Because $2 + 2 = 4$ is true, the assignment statement $x := x + 1$ is executed. Hence, x has the value $0+1 = 1$ after this statement is encountered

Truth Table for the Conditional Statement $p \to q$

p	q	$p \to q$
T	T	T
T	F	F
F	T	T
F	F	T

Remark 3

A useful way to understand the truth value of a conditional statement is to think of an obligation or a contract.

Remark 4

To remember that "*p only if q*" expresses the same thing as "*if p, then q,*" note that "*p only if q*" says that p cannot be true when q is not true.

That is, the statement is false if p is true, but q is false. When p is false, q may be either true or false, because the statement says nothing about the truth value of q.

Remark 5

Be careful not to use "*q only if p*" to express $p \to q$ because this is incorrect. The word "**only**" plays an essential role here. To see this, note that the truth values of *"q only if p"* and $p \to q$ are different when p and q have different truth values

Remark 6

To see why "*q is necessary for p*" is equivalent to "*if p, then q,*" observe that "*q is necessary for p*" means that p cannot be true unless q is true, or that if q is false, then p is false. This is the same as saying that if p is true, then q is true.

Remark 7

To see why "*p is sufficient for q*" is equivalent to "*if p, then q,*" note that "*p is sufficient for q*" means if p is true, it must be the case that q is also true. This is the same as saying that if p is true, then q is also true.

Remark 8

To remember that "*q unless* $\neg p$" expresses the same conditional statement as "*if p, then q,*" note that "*q unless* $\neg p$" means that if $\neg p$ is false, then q must be true. That is, the statement "*q unless* $\neg p$" is false when p is true but q is false, but it is true otherwise. Consequently, "*q unless* $\neg p$" and $p \to q$ always have the same truth value.

Remark 9

We would not use these two conditional statements:
"*If Juan has a smartphone, then* $2 + 3 = 5$" or
"*If Juan has a smartphone, then* $2 + 3 = 6$"
in natural language (except perhaps in sarcasm), because there is no relationship between the hypothesis and the conclusion in either statement

Remark 10

In mathematical reasoning, we consider conditional statements of a more general sort than we use in English. The mathematical concept of a conditional statement is independent of a cause-and-effect relationship between hypothesis and conclusion. Our definition of a conditional statement specifies its truth values; it is not based on English usage. Propositional language is an artificial language; we only parallel English usage to make it easy to use and remember.

Remark 11

The if-then construction used in many programming languages is different from that used in logic. Most programming languages contain statements such as **if** p **then** S, where p is a proposition and S is a program segment (one or more statements to be executed).
(Although this looks as if it might be a conditional statement, S is not a proposition, but rather is a set of executable instructions.) When execution of a program encounters such a statement, S is executed if p is true, but S is not executed if p is false

Converse, Contrapositive, and Inverse (pg. 9)

Definitions

Given a conditional statement $p \to q$, we define:

Converse: $q \to p$

Inverse: $\neg p \to \neg q$

Contrapositive: $\neg q \to \neg p$

Only the contrapositive is logically equivalent to the original conditional statement.

Truth Values of Conditional Forms

p	q	$p \to q$	$q \to p$ **(Converse)**	$\neg p \to \neg q$ **(Inverse)**	$\neg q \to \neg p$ **(Contrapositive)**
T	T	T	T	T	T
T	F	F	T	T	F
F	T	T	F	F	T
F	F	T	T	T	T

Example: Rewriting the Conditional Statement

Original Statement: "The home team wins whenever it is raining." This translates to:

If it is raining, then the home team wins. $(p \to q)$

- Converse:

If the home team wins, then it is raining. $(q \to p)$

- Inverse:

If it is not raining, then the home team does not win. $(\neg p \to \neg q)$

- Contrapositive:

If the home team does not win, then it is not raining. $(\neg q \to \neg p)$

Only the contrapositive is logically equivalent to the original.

Remark 1

A conditional statement and its contrapositive always have the same truth value — they are logically equivalent

Remark 2

The converse and inverse are equivalent to each other, but not to the original statement.

Remark 3

Assuming the converse is equivalent to the original is a common logical fallacy.

Biconditional Statements: (pg. 10)

Definition: Biconditional Statement

Let p and q be propositions. The biconditional statement $p \leftrightarrow q$ is the proposition:

"*p if and only if q.*”"

The biconditional statement $p \leftrightarrow q$ is true when p and q have the same truth values, and is false otherwise.

Biconditional statements are also called bi-implications.

Common Notations for Biconditional Statements

Notation	Read as	Used In
$p \leftrightarrow q$	p if and only if q	Logic, mathematics
$p \Leftrightarrow q$	p is logically equivalent to q	Formal proofs, equivalences
iff(p, q) or "iff"	iff = if and only if	Mathematical shorthand

Common Ways to Express a Biconditional Statement $p \leftrightarrow q$

Expression	Notes
"p if and only if q"	Standard form in logic and math
"p is necessary and sufficient for q"	Emphasizes that each implies the other
"If p then q, and conversely"	States both directions explicitly
"p iff q"	Abbreviation: “if and only if”
"p exactly when q"	Informal but equivalent phrasing

Truth Table for the Biconditional $p \leftrightarrow q$

p	q	$p \leftrightarrow q$
T	T	T
T	F	F
F	T	F
F	F	T

Remark 1

The statement $p \leftrightarrow q$ is true when both the conditional statements:

$$p \to q$$

$$q \to p$$

are true and is false otherwise

Remark 2

That is why we use the words “*if and only if*” to express this logical connective and why it is symbolically written by combining the symbols $\to$ and $\leftarrow$.

Remark 3

Note that $p \leftrightarrow q$ has exactly the same truth value as

$$(p \rightarrow q) \wedge (q \rightarrow p)$$

Example of biconditional statement

Statements:

$$p: \textit{"You can take the flight"} \text{ and } q: \textit{"You buy a ticket"}$$

Biconditional statement:

You can take the flight if and only if you buy a ticket ($p \leftrightarrow q$)

When true?

This statement is **true** if p and q are *either both true* or *both false*, that is, if you buy a ticket and can take the flight or if you do not buy a ticket and you cannot take the flight

When false?

It is **false** when p and q have *opposite truth values*, that is, when you do not buy a ticket, but you can take the flight (such as when you get a free trip) and when you buy a ticket but you cannot take the flight (such as when the airline bumps you).

Remark 4

You should be aware that biconditionals are not always explicit in natural language. In particular, the "*if and only if*" construction used in biconditionals is rarely used in common language.

Instead, biconditionals are often expressed using an "*if, then*" or an "*only if*" construction. The other part of the "*if and only if*" is implicit. That is, the converse is implied, but not stated.

Example of biconditional statement 2

Statement:

If you finish your meal, then you can have dessert

Really meant:

You can have dessert if and only if you finish your meal

Logically equivalent to:

If you finish your meal, then you can have dessert
You can have dessert only if you finish your meal

Remark 5

Because of this imprecision in natural language, we need to make an assumption whether a conditional statement in natural language implicitly includes its converse.
Because precision is essential in mathematics and in logic, we will always distinguish between the conditional statement $p \rightarrow q$ and the biconditional statement $p \leftrightarrow q$.

Truth Tables of Compound Propositions (pg. 11)

- **Compound propositions** We can use these connectives to build up complicated compound propositions involving any number of propositional variables
- We can use truth tables to determine the truth values of these compound propositions

Example of compound propositions

Compound proposition:

$$(p \vee \neg q) \rightarrow (p \wedge q)$$

Translated into and read as

If either p is true or q is not true, then both p and q are true.

How to solve (through truth table)?
Because this truth table involves two propositional variables p and q, there are four rows in this truth table, one for each of the pairs of truth values TT, TF, FT, and FF
Coloumn 1 and 2: Used for the truth values of p and q, respectively.
Coloumn 3: We find the truth value of $\neg q$
Coloum 4: We find the truth value of $p \vee \neg q$
Coloumn 5: We find the truth value of $p \wedge q$.
Coloumn 6: We find the truth value of the original compound proposition $(p \vee \neg q) \rightarrow (p \wedge q)$

Truth Table for $(p \vee \neg q) \rightarrow (p \wedge q)$

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

Precedence of Logical Operators (pg. 11)

Remark 1

We will generally use parentheses to specify the order in which logical operators in a compound proposition are to be applied.

For instance:

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

is the conjunction of $p \vee q$ and $\neg r$.

Translation:

p or q and not r (logic)
Either p is true or q is true, and r is not true (plain English)

Remark 2

However, to reduce the number of parentheses, we specify that the negation operator is applied before all other logical operators.

Example:

$$\neg p \wedge q \text{ can be seen as } (\neg p) \wedge q$$

Remark 3

Another general rule of precedence is that the conjunction operator takes precedence over the disjunction operator

Example:

$$p \vee q \wedge r \text{ means } p \vee (q \wedge r)$$
$$p \wedge q \vee r \text{ means } (p \wedge q) \vee r$$

Remark 4

Finally, it is an accepted rule that the conditional and biconditional operators, $\rightarrow$ and $\leftrightarrow$, have lower precedence than the conjunction and disjunction operators

Example:

$$p \rightarrow q \vee r \text{ means } p \rightarrow (q \vee r)$$
$$p \vee q \rightarrow r \text{ means } (p \vee q) \rightarrow r$$

Remark 5

We will use parentheses when the order of the conditional operator and biconditional operator is at issue, although the conditional operator has precedence over the biconditional operator

Precedence of Logical Operators (from highest to lowest)

Precedence	Symbol	Operator Name
1 (highest)	$\neg$	Negation (Not)
2	$\wedge$	Conjunction (And)
3	$\vee$	Disjunction (Or)
4	$\oplus$	Exclusive Or (XOR)
5	$\rightarrow$	Conditional (Implication)
6	$\leftrightarrow$	Biconditional (If and only if)

Logic and Bit Operations (pg. 12)

Definition: A **bit** is a symbol with two possible values, namely, 0 (zero) and 1 (one).

Remark 1

Computers represent information using bits.

This meaning of the word bit comes from binary digit, because zeros and ones are the digits used in binary representations of numbers

Remark 2

A bit can be used to represent a truth value, because there are two truth values, namely, true and false.
As is customarily done, we will use a 1 bit to represent true and a 0 bit to represent false. That is, 1 represents T (true), 0 represents F (false).

Definition: A variable is called a **Boolean variabl**e if its value is either true or false.

Remark 3

Consequently, a Boolean variable can be represented using a bit.

Remark 4

Computer bit operations correspond to the logical connectives. By replacing true by a one and false by a zero in the truth tables for the operators: $\wedge, \vee$ and $\oplus$.

We will also use the notation **OR**, **AND**, and **XOR** for the operators as is done in various programming languages

Bitwise Table (0/1)

p	q	$p \vee q$	$p \wedge q$	$p \oplus q$
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	0

Truth Table (T/F)

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

Definition: A **bit string** is a sequence of zero or more bits. The length of this string is the number of bits in the string.

Remark 5

Information is often represented using bit strings, which are lists of zeros and ones.
When this is done, operations on the bit strings can be used to manipulate this information.

Remark 6

We define the **bitwise OR**, **bitwise AND**, and **bitwise XOR** of two strings of the same length to be the strings that have as their bits the OR, AND, and XOR of the corresponding bits in the two strings, respectively

Example of bitwise operations

Find the **bitwise OR**, **bitwise AND**, and b**itwise XOR** of the bit strings

01 1011 0110 and
11 0001 1101

(Here, and throughout this book, bit strings will be split into blocks of four bits to make them easier to read.)

Solution:
The bitwise OR, bitwise AND, and bitwise XOR of these strings are obtained by taking the OR, AND, and XOR of the corresponding bits, respectively

Bitwise OR:
To compute the **bitwise OR**, we compare each corresponding pair of bits from A and B. The OR operation results in 1 if *either* of the bits is 1, and 0 only if both are 0.

$$\begin{array}{ccccccccccccc} A & = & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ B & = & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ \hline A \text{ OR } B & = & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}$$

Bitwise AND:
To compute the **bitwise AND**, we compare each corresponding pair of bits from A and B. The AND operation results in 1 only if *both* bits are 1; otherwise, it is 0.

$$\begin{array}{ccccccccccccc} A & = & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ B & = & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ \hline A \text{ AND } B & = & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{array}$$

Bitwise XOR:
To compute the **bitwise XOR**, we compare each corresponding pair of bits from A and B. The XOR operation results in 1 if the bits are *different*, and 0 if they are the same.

$$\begin{array}{ccccccccccccc} A & = & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ B & = & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ \hline A \text{ XOR } B & = & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \end{array}$$

Full solution:

Bit String A:	01 1011 0110
Bit String B:	11 0001 1101
Bitwise OR:	11 1011 1111
Bitwise AND:	01 0001 0100
Bitwise XOR:	10 1010 1011