

# Intro

28 March 2021

08:37 AM

# Propositional Logic

28 March 2021 08:26 AM

## Proposition

A statement that is either true or false but not both.

These statements are denoted by small case variable names

Eg: p: roses are red

A proposition that cannot be expressed in terms of simpler propositions are called atomic propositions.

The area of logic that deals with propositions is called the propositional calculus or propositional logic

## Compound Propositions

compound propositions, are formed from existing propositions using logical operators

Some operators combining proposition

Operator	Name	Logical Function	Translation	Remarks
$\sim$ $\neg$ — , <b>N</b> !	tilde logical not	negation	not, it is not the case that	
<b>&amp;</b> $\wedge$ .	ampersand logical and dot	conjunction	and, also, moreover, but	
$\vee$	wedge logical or	disjunction	or, unless, nor	
$\supset$ $\rightarrow$	horseshoe	material conditional	if . . . then . . . , only if	
$\equiv$ $\leftrightarrow$	triple bar double arrow	material bi-conditional (equivalence)	if and only if, just in case	
$p \oplus q$		exclusive or XOR		

Truth tables

Basic Setup	Negation	Conjunction	Disjunction	Material Conditional	Material Bi-conditional	Exclusive or
<b>P Q</b>	$\neg P \neg Q$	<b>P &amp; Q</b>	<b>P <math>\vee</math> Q</b>	<b>P <math>\supset</math> Q</b>	<b>P <math>\equiv</math> Q</b>	<b>p <math>\oplus</math> q</b>
<b>T T</b>	F F	T	T	T	T	F
<b>T F</b>	F T	F	T	F	F	T
<b>F T</b>	T F	F	T	T	F	T
<b>F F</b>	T T	F	F	T	T	F

## Other Statements:

- $p \rightarrow q$      -implication
- $q \rightarrow p$      -converse of  $p \rightarrow q$
- $\neg p \rightarrow \neg q$    -inverse of  $p \rightarrow q$
- $\neg q \rightarrow \neg p$    -contrapositive of  $p \rightarrow q$

## Truth tables:

Basic Setup	Negation	Conjunction	Disjunction	Material Conditional	Material Bi-conditional	Exclusive or
$P \quad Q$	$\neg P \quad \neg Q$	$P \ \& \ Q$	$P \vee Q$	$P \supset Q$	$P \equiv Q$	$p \oplus q$
<b>T   T</b>	<b>F   F</b>	<b>T</b>	<b>T</b>	<b>T</b>	<b>T</b>	<b>F</b>
<b>T   F</b>	<b>F   T</b>	<b>F</b>	<b>T</b>	<b>F</b>	<b>F</b>	<b>T</b>
<b>F   T</b>	<b>T   F</b>	<b>F</b>	<b>T</b>	<b>T</b>	<b>F</b>	<b>T</b>
<b>F   F</b>	<b>T   T</b>	<b>F</b>	<b>F</b>	<b>T</b>	<b>T</b>	<b>F</b>

## In general..

- $P \supset Q$  is only false when P is true and Q is false.
- $P \equiv Q$  is true when P and Q have the same truth value—either they are both true or they are both false

## An example of bitwise operation between two bit strings...

01 1011 0110

11 0001 1101

-----

11 1011 1111 bitwise OR

01 0001 0100 bitwise AND

10 1010 1011 bitwise XOR

## Logical Equivalences

**Basic definition:** A compound proposition that is always true, no matter what the truth values of the propositional variables that occur in it, is called a tautology. A compound proposition that is always false is called a contradiction. A compound proposition that is neither a tautology nor a contradiction is called a contingency.

**Satisfiability:** A compound proposition is satisfiable if there is an assignment of truth values to its variables that makes it true otherwise unsatisfiable.

## Predicates

Predicates are the non variable part of a statement involving a variable.

Eg: a computer x is connected to network n  $\Rightarrow P(x,n)$  where p represents the predicate of statement leaving the subjects x and n.

# Conditional statements

28 March 2021 08:57 AM

These are other important ways of combining proposition to make new propositions using conditionals (if this then that)

Implication ( $p \rightarrow q$ )

If  $p$  then  $q$

$p$  is called the hypothesis (or antecedent or premise) and

$q$  is called the conclusion (or consequence).

Truth table

<b>TABLE 5 The Truth Table for the Conditional Statement</b> $p \rightarrow q$ .		
$p$	$q$	$p \rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

Converse contrapositive and Inverse

The proposition  $q \rightarrow p$  is called the converse of  $p \rightarrow q$ .

The proposition  $\neg q \rightarrow \neg p$  is called the contrapositive of  $p \rightarrow q$ .

The proposition  $\neg p \rightarrow \neg q$  is called the inverse of  $p \rightarrow q$ .

When two compound propositions always have the same truth values, regardless of the truth values of its propositional variables, we call them equivalent.

The contrapositive, but neither the converse or inverse, of a conditional statement is equivalent to it.

The converse and the inverse of a conditional statement are also equivalent to each other.

Biconditionals ( $p \leftrightarrow q$ )

$P$  iff (if and only if)  $q$

the statement  $p \leftrightarrow q$  is true when both the conditional statements  $p \rightarrow q$  and  $q \rightarrow p$  are true and is false otherwise.

Truth table

**TABLE 6** The 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

# Precedence of Logical Operators

02 April 2021 11:09

<b>TABLE 8</b> <b>Precedence of Logical Operators.</b>	
<i>Operator</i>	<i>Precedence</i>
$\neg$	1
$\wedge$	2
$\vee$	3
$\rightarrow$	4
$\leftrightarrow$	5

# Applications of propositional logic

02 April 2021 11:20

Omitted 1.2 of rossen



# Propositional equivalences

04 April 2021 08:34

Tautology – a compound proposition that is always true

Contradiction – a compound proposition that is always false

Contingency – a compound proposition that is neither tautology nor contradiction

Logical equivalence – compound proposition that have the same truth values in all possible cases

Denoted by  $p \equiv q$

Some examples of logical equivalences

TABLE 6 Logical Equivalences.	
Equivalence	Name
$p \wedge \mathbf{T} \equiv p$ $p \vee \mathbf{F} \equiv p$	Identity laws
$p \vee \mathbf{T} \equiv \mathbf{T}$ $p \wedge \mathbf{F} \equiv \mathbf{F}$	Domination laws
$p \vee p \equiv p$ $p \wedge p \equiv p$	Idempotent laws
$\neg(\neg p) \equiv p$	Double negation law
$p \vee q \equiv q \vee p$ $p \wedge q \equiv q \wedge p$	Commutative laws
$(p \vee q) \vee r \equiv p \vee (q \vee r)$ $(p \wedge q) \wedge r \equiv p \wedge (q \wedge r)$	Associative laws
$p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$ $p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$	Distributive laws
$\neg(p \wedge q) \equiv \neg p \vee \neg q$ $\neg(p \vee q) \equiv \neg p \wedge \neg q$	De Morgan's laws
$p \vee (p \wedge q) \equiv p$ $p \wedge (p \vee q) \equiv p$	Absorption laws
$p \vee \neg p \equiv \mathbf{T}$ $p \wedge \neg p \equiv \mathbf{F}$	Negation laws

TABLE 7 Logical Equivalences Involving Conditional Statements.

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

TABLE 8 Logical Equivalences Involving Biconditional Statements.

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

# Satisfiability

06 April 2021 07:24 AM

A compound statement is called satisfiable if there exist an assignment of truth values if that assignment makes the statement true.

**Predicate** - predicate is functional proposition that takes a subject applies logic to the subject and gives a truth values.

$P(x)$  is computer  $x$  is under attack.

When checking the correctness of a computer program The statements that describe valid input are known as preconditions and the conditions that the output should satisfy when the program has run are known as postconditions.

**Quantifiers** - to quantify a predicate (to tell whether it is T or F) we need to assign a value to subject however there are other methods of quantification,

universal quantification, which tells us that a predicate is true for every element under consideration.

The universal quantification of  $P(x)$  is the statement " $P(x)$  for all values of  $x$  in the domain."

The notation  $\forall xP(x)$  denotes the universal quantification of  $P(x)$ . Here  $\forall$  is called the universal quantifier.

We read  $\forall xP(x)$  as "for all  $xP(x)$ " or "for every  $xP(x)$ ." An element for which  $P(x)$  is false is called a counterexample to  $\forall xP(x)$ .

existential quantification, which tells us that there is one or more element under consideration for which the predicate is true.

The existential quantification of  $P(x)$  is the proposition "There exists an element  $x$  in the domain such that  $P(x)$ ." We use the notation  $\exists xP(x)$  for the existential quantification of  $P(x)$ . Here  $\exists$  is called the existential quantifier.

quantification  $\exists xP(x)$  is read as "There is an  $x$  such that  $P(x)$ ," "There is at least one  $x$  such that  $P(x)$ ,"

uniqueness quantifier, denoted by  $\exists!$  or  $\exists 1$ . The notation  $\exists! xP(x)$  [or  $\exists 1xP(x)$ ] states "There exists a unique  $x$  such that  $P(x)$  is true."

### Precedence of quantifiers

The quantifiers  $\forall$  and  $\exists$  have higher precedence than all logical operators from propositional calculus.

# Binding variables

06 April 2021 07:53 AM

When a quantifier is used on the variable  $x$ , we say that this occurrence of the variable is bound. An occurrence of a variable that is not bound by a quantifier or set equal to a particular value is said to be free.

The part of a logical expression to which a quantifier is applied is called the scope of this quantifier. Consequently, a variable is free if it is outside the scope of all quantifiers in the formula that specify this variable.

# Equivalence

06 April 2021 07:58 AM

Statements involving predicates and quantifiers are logically equivalent if and only if they have the same truth value no matter which predicates are substituted into these statements and which domain of discourse is used for the variables in these propositional functions. We use the notation  $S \equiv T$  to indicate that two statements  $S$  and  $T$  involving predicates and quantifiers are logically equivalent.

**TABLE 2 De Morgan's Laws for Quantifiers.**

<i>Negation</i>	<i>Equivalent Statement</i>	<i>When Is Negation True?</i>	<i>When False?</i>
$\neg \exists x P(x)$	$\forall x \neg P(x)$	For every $x$ , $P(x)$ is false.	There is an $x$ for which $P(x)$ is true.
$\neg \forall x P(x)$	$\exists x \neg P(x)$	There is an $x$ for which $P(x)$ is false.	$P(x)$ is true for every $x$ .

# Nested quantifiers

06 April 2021 08:07 AM

TABLE 1 Quantifications of Two Variables.		
Statement	When True?	When False?
$\forall x \forall y P(x, y)$ $\forall y \forall x P(x, y)$	$P(x, y)$ is true for every pair $x, y$ .	There is a pair $x, y$ for which $P(x, y)$ is false.
$\forall x \exists y P(x, y)$	For every $x$ there is a $y$ for which $P(x, y)$ is true.	There is an $x$ such that $P(x, y)$ is false for every $y$ .
$\exists x \forall y P(x, y)$	There is an $x$ for which $P(x, y)$ is true for every $y$ .	For every $x$ there is a $y$ for which $P(x, y)$ is false.
$\exists x \exists y P(x, y)$ $\exists y \exists x P(x, y)$	There is a pair $x, y$ for which $P(x, y)$ is true.	$P(x, y)$ is false for every pair $x, y$ .

# Rules of inference

06 April 2021 08:12 AM

An argument in propositional logic is a sequence of propositions. All but the final proposition in the argument are called premises and the final proposition is called the conclusion. An argument is valid if the truth of all its premises implies that the conclusion is true.

An argument form in propositional logic is a sequence of compound propositions involving propositional variables. An argument form is valid if no matter which particular propositions are substituted for the propositional variables in its premises, the conclusion is true if the premises are all True

TABLE 1 Rules of Inference.		
Rule of Inference	Tautology	Name
$\begin{array}{l} p \\ p \rightarrow q \\ \hline \therefore q \end{array}$	$(p \wedge (p \rightarrow q)) \rightarrow q$	Modus ponens
$\begin{array}{l} \neg q \\ p \rightarrow q \\ \hline \therefore \neg p \end{array}$	$(\neg q \wedge (p \rightarrow q)) \rightarrow \neg p$	Modus tollens
$\begin{array}{l} p \rightarrow q \\ q \rightarrow r \\ \hline \therefore p \rightarrow r \end{array}$	$((p \rightarrow q) \wedge (q \rightarrow r)) \rightarrow (p \rightarrow r)$	Hypothetical syllogism
$\begin{array}{l} p \vee q \\ \neg p \\ \hline \therefore q \end{array}$	$((p \vee q) \wedge \neg p) \rightarrow q$	Disjunctive syllogism
$\begin{array}{l} p \\ \hline \therefore p \vee q \end{array}$	$p \rightarrow (p \vee q)$	Addition
$\begin{array}{l} p \wedge q \\ \hline \therefore p \end{array}$	$(p \wedge q) \rightarrow p$	Simplification
$\begin{array}{l} p \\ q \\ \hline \therefore p \wedge q \end{array}$	$((p) \wedge (q)) \rightarrow (p \wedge q)$	Conjunction
$\begin{array}{l} p \vee q \\ \neg p \vee r \\ \hline \therefore q \vee r \end{array}$	$((p \vee q) \wedge (\neg p \vee r)) \rightarrow (q \vee r)$	Resolution

**TABLE 2 Rules of Inference for Quantified Statements.**

<i>Rule of Inference</i>	<i>Name</i>
$\frac{\forall xP(x)}{\therefore P(c)}$	Universal instantiation
$\frac{P(c) \text{ for an arbitrary } c}{\therefore \forall xP(x)}$	Universal generalization
$\frac{\exists xP(x)}{\therefore P(c) \text{ for some element } c}$	Existential instantiation
$\frac{P(c) \text{ for some element } c}{\therefore \exists xP(x)}$	Existential generalization



# Proofs

06 April 2021 10:37 AM

A **theorem** is a statement that can be shown to be true. In mathematical writing, the term theorem is usually reserved for a statement that is considered at least somewhat important. Less important theorems sometimes are called **propositions**. (Theorems can also be referred to as facts or results.) A theorem may be the universal quantification of a conditional statement. Links with one or more premises and a conclusion. However, it may be some other type of logical statement, as the examples later in this chapter will show. We demonstrate that a theorem is true with a proof. A **proof** is a valid argument that establishes the truth of a theorem. The statements used in a proof can include axioms (or postulates), which are statements we assume to be true.

A less important theorem that is helpful in the proof of other results is called a **lemma** (plural lemmas or lemmata). Complicated proofs are usually easier to understand when they are proved using a series of lemmas, where each lemma is proved individually. A **corollary** is a theorem that can be established directly from a theorem that has been proved. A **conjecture** is a statement that is being proposed to be a true statement, usually on the basis of some partial evidence, a heuristic argument, or the intuition of an expert. When a proof of a conjecture is found, the conjecture becomes a theorem. Many times conjectures are shown to be false, so they are not theorems.

# Methods of proving

06 April 2021 10:42 AM

## Direct proof

A direct proof of a conditional statement  $p \rightarrow q$  is constructed when the first step is the assumption that  $p$  is true; subsequent steps are constructed using rules of inference, with the final step showing that  $q$  must also be true.

## Indirect proof

Proofs of theorems of the form  $\forall x(P(x) \rightarrow Q(x))$  are generally not possible by direct proofs, that is, that do not start with the premises and end with the conclusion, are called indirect proofs.

## Proofs by contraposition

Proofs by contraposition make use of the fact that the conditional statement  $p \rightarrow q$  is equivalent to its contrapositive,  $\neg q \rightarrow \neg p$ . This means that the conditional statement  $p \rightarrow q$  can be proved by showing that its contrapositive,  $\neg q \rightarrow \neg p$ , is true. In a proof by contraposition of  $p \rightarrow q$ , we take  $\neg q$  as a premise, and using axioms, definitions, and previously proven theorems, together with rules of inference, we show that  $\neg p$  must follow.

## VACUOUS AND TRIVIAL PROOFS

We can quickly prove that a conditional statement  $p \rightarrow q$  is true when we know that  $p$  is false, because  $p \rightarrow q$  must be true when  $p$  is false. Consequently, if we can show that  $p$  is false, then we have a proof, called a vacuous proof, of the conditional statement  $p \rightarrow q$ . Vacuous proofs are often used to establish special cases of theorems that state that a conditional statement is true for all positive integers [i.e., a theorem of the kind  $\forall n P(n)$ , where  $P(n)$  is a propositional function]

## proofs by contradiction

Suppose we want to prove that a statement  $p$  is true. Furthermore, suppose that we can find a contradiction  $q$  such that  $\neg p \rightarrow q$  is true. Because  $q$  is false, but  $\neg p \rightarrow q$  is true, we can conclude that  $\neg p$  is false, which means that  $p$  is true. How can we find a contradiction  $q$  that might help us prove that  $p$  is true in this way? Because the statement  $r \wedge \neg r$  is a contradiction whenever  $r$  is a proposition, we can prove that  $p$  is true if we can show that  $\neg p \rightarrow (r \wedge \neg r)$  is true for some proposition  $r$ . Proofs of this type are called proofs by contradiction.

## Proof by cases

The tautology  $[(p_1 \vee p_2 \vee \dots \vee p_n) \rightarrow q] \leftrightarrow [(p_1 \rightarrow q) \wedge (p_2 \rightarrow q) \wedge \dots \wedge (p_n \rightarrow q)]$  can be used as a rule of inference. This shows that the original conditional statement with a hypothesis made up of a disjunction of the propositions  $p_1, p_2, \dots, p_n$  can be proved by proving each of the  $n$  conditional statements  $p_i \rightarrow q$ ,  $i = 1, 2, \dots, n$ , individually. Such an argument is called a proof by cases. Sometimes to prove that a conditional statement  $p \rightarrow q$  is true, it is convenient to use a disjunction  $p_1 \vee p_2 \vee \dots \vee p_n$  instead of  $p$  as the hypothesis of the conditional statement, where  $p$  and  $p_1 \vee p_2 \vee \dots \vee p_n$  are equivalent.

## EXHAUSTIVE PROOF

Some theorems can be proved by examining a relatively small number of examples. Such proofs are called exhaustive proofs, or proofs by exhaustion because these proofs proceed by exhausting all possibilities. An exhaustive proof is a special type of proof by cases where each case involves checking a single example.

## Existence proof

Many theorems are assertions that objects of a particular type exist. A theorem of this type is a proposition of the form  $\exists x P(x)$ , where  $P$  is a predicate. A proof of a proposition

of the form  $\exists xP(x)$  is called an existence proof. There are several ways to prove a theorem of this type. Sometimes an existence proof of  $\exists xP(x)$  can be given by finding an element  $a$ , called a witness, such that  $P(a)$  is true. This type of existence proof is called constructive. It is also possible to give an existence proof that is nonconstructive

#### Uniqueness proof

Some theorems assert the existence of a unique element with a particular property. In other words, these theorems assert that there is exactly one element with this property. To prove a statement of this type we need to show that an element with this property exists and that no other element has this property. The two parts of a uniqueness proof are:

Existence: We show that an element  $x$  with the desired property exists.

Uniqueness: We show that if  $x$  and  $y$  both have the desired property, then  $x = y$ .

# Intro

06 April 2021 11:29 AM

The part of mathematics devoted to the study of the set of integers and their properties is known as number theory

## Topics

- Divisibility

- Modular Arithmetic

## Definition

If  $a$  and  $b$  are integers with  $a \neq 0$ , we say that  $a$  divides  $b$  if there is an integer  $c$  such that  $b = ac$  (or equivalently, if  $b/a$  is an integer). When  $a$  divides  $b$  we say that  $a$  is a factor or divisor of  $b$ , and that  $b$  is a multiple of  $a$ . The notation  $a \mid b$  denotes that  $a$  divides  $b$ . We write  $a \nmid b$  when  $a$  does not divide  $b$ .

## Theorem

Let  $a$ ,  $b$ , and  $c$  be integers, where  $a \neq 0$ . Then

- (i) if  $a \mid b$  and  $a \mid c$ , then  $a \mid (b + c)$ ;
- (ii) if  $a \mid b$ , then  $a \mid bc$  for all integers  $c$ ;
- (iii) if  $a \mid b$  and  $b \mid c$ , then  $a \mid c$ .

## 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  $a = dq + r$ .

Here

$A$  - dividend

$D$  - divisor

$Q$  - quotient

$R$  - remainder

In integer operations

$Q = a/d$

$R = a \% d$

## Congruence

If  $a$  and  $b$  are integers and  $m$  is a positive integer, then  $a$  is congruent to  $b$  modulo  $m$  if  $m$  divides  $a - b$  or equivalently if  $a \bmod m = b \bmod m$ . We use the notation  $a \equiv b \pmod{m}$  to indicate that  $a$  is congruent to  $b$  modulo  $m$ . We say that  $a \equiv b \pmod{m}$  is a congruence and that  $m$  is its modulus (plural moduli). If  $a$  and  $b$  are not congruent modulo  $m$ , we write  $a \not\equiv b \pmod{m}$ .

## Theoream

Let  $a$  and  $b$  be integers, and let  $m$  be a positive integer. Then  $a \equiv b \pmod{m}$  if and only if  $a \bmod m = b \bmod m$ .

Let  $m$  be a positive integer. The integers  $a$  and  $b$  are congruent modulo  $m$  if and only if there is an integer  $k$  such that  $a = b + km$ .

Let  $m$  be a positive integer. If  $a \equiv b \pmod{m}$  and  $c \equiv d \pmod{m}$ , then  $a + c \equiv b + d \pmod{m}$  and  $ac \equiv bd \pmod{m}$

Let  $m$  be a positive integer and let  $a$  and  $b$  be integers. Then  $(a + b) \bmod m = ((a \bmod m) + (b \bmod m)) \bmod m$  and  $ab \bmod m = ((a \bmod m)(b \bmod m)) \bmod m$ .

We can define arithmetic operations on  $\mathbb{Z}_m$ , the set of nonnegative integers less than  $m$ , that is, the set  $\{0, 1, \dots, m-1\}$ . In particular,

$$a +_m b = (a + b) \bmod m,$$

$$a \cdot_m b = (a \cdot b) \bmod m,$$

**Distributivity** If  $a$ ,  $b$ , and  $c$  belong to  $\mathbb{Z}_m$ , then  $a \cdot_m (b +_m c) = (a \cdot_m b) +_m (a \cdot_m c)$  and  $(a +_m b) \cdot_m c = (a \cdot_m c) +_m (b \cdot_m c)$ .

The operation described above are closed, associative, commutative, have additive inverses and identity element.

We in our daily usage use integer with base 10  
however integers can be expressed in any base greater  
than 1 particularly computer uses base 2 for integer  
representation.

#### ALGORITHM 1 Constructing Base $b$ Expansions.

```

procedure base  $b$  expansion( $n, b$ : positive integers with  $b > 1$ )
 $q := n$ 
 $k := 0$ 
while  $q \neq 0$ 
     $a_k := q \bmod b$ 
     $q := q \div b$ 
     $k := k + 1$ 
return  $(a_{k-1}, \dots, a_1, a_0)$   $\{(a_{k-1} \dots a_1 a_0)_b$  is the base  $b$  expansion of  $n\}$ 

```

#### CONVERSION BETWEEN BINARY, OCTAL, AND HEXADECIMAL EXPANSIONS

Conversion between binary and octal and between binary and hexadecimal expansions is extremely easy because each octal digit corresponds to a block of three binary digits and each hexadecimal digit corresponds to a block of four binary digits, with these correspondences shown in Table 1 without initial 0s shown. (We leave it as Exercises 13–16 to show that this is the case.) This conversion is illustrated in Example 7.

**TABLE 1** Hexadecimal, Octal, and Binary Representation of the Integers 0 through 15.

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Octal	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17
Binary	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111

#### ALGORITHM 2 Addition of Integers.

```

procedure add( $a, b$ : positive integers)
{the binary expansions of  $a$  and  $b$  are  $(a_{n-1}a_{n-2} \dots a_1a_0)_2$ 
 and  $(b_{n-1}b_{n-2} \dots b_1b_0)_2$ , respectively}
 $c := 0$ 
for  $j := 0$  to  $n - 1$ 
     $d := \lfloor (a_j + b_j + c)/2 \rfloor$ 
     $s_j := a_j + b_j + c - 2d$ 
     $c := d$ 
 $s_n := c$ 
return  $(s_0, s_1, \dots, s_n)$  {the binary expansion of the sum is  $(s_ns_{n-1} \dots s_0)_2$ }

```



### ALGORITHM 3 Multiplication of Integers.

```
procedure multiply( $a, b$ : positive integers)
{ the binary expansions of  $a$  and  $b$  are  $(a_{n-1}a_{n-2} \dots a_1a_0)_2$ 
  and  $(b_{n-1}b_{n-2} \dots b_1b_0)_2$ , respectively }
for  $j := 0$  to  $n - 1$ 
    if  $b_j = 1$  then  $c_j := a$  shifted  $j$  places
    else  $c_j := 0$ 
{  $c_0, c_1, \dots, c_{n-1}$  are the partial products }
 $p := 0$ 
for  $j := 0$  to  $n - 1$ 
     $p := \text{add}(p, c_j)$ 
return  $p$  {  $p$  is the value of  $ab$  }
```

### ALGORITHM 4 Computing div and mod.

```
procedure division algorithm( $a$ : integer,  $d$ : positive integer)
 $q := 0$ 
 $r := |a|$ 
while  $r \geq d$ 
     $r := r - d$ 
     $q := q + 1$ 
if  $a < 0$  and  $r > 0$  then
     $r := d - r$ 
     $q := -(q + 1)$ 
return  $(q, r)$  {  $q = a \text{ div } d$  is the quotient,  $r = a \text{ mod } d$  is the remainder }
```

The problem of finding  $b^n \bmod m$  is called modular exponentiation

The problem is very significant in cryptography

To motivate the fast modular exponentiation algorithm, we illustrate its basic idea. We will explain how to use the binary expansion of  $n$ , say  $n = (a_{k-1} \dots a_1 a_0)_2$ , to compute  $b^n$ . First, note that

$$b^n = b^{a_{k-1} \cdot 2^{k-1} + \dots + a_1 \cdot 2 + a_0} = b^{a_{k-1} \cdot 2^{k-1}} \dots b^{a_1 \cdot 2} \cdot b^{a_0}.$$

This shows that to compute  $b^n$ , we need only compute the values of  $b, b^2, (b^2)^2 = b^4, (b^4)^2 = b^8, \dots, b^{2^k}$ . Once we have these values, we multiply the terms  $b^{2^j}$  in this list, where  $a_j = 1$ . (For efficiency and to reduce space requirements, after multiplying by each term, we reduce the result modulo  $m$ .)

#### ALGORITHM 5 Fast Modular Exponentiation.

```

procedure modular_exponentiation( $b$ : integer,  $n = (a_{k-1}a_{k-2} \dots a_1a_0)_2$ ,
     $m$ : positive integers)
 $x := 1$ 
 $power := b \bmod m$ 
for  $i := 0$  to  $k - 1$ 
    if  $a_i = 1$  then  $x := (x \cdot power) \bmod m$ 
     $power := (power \cdot power) \bmod m$ 
return  $x$  { $x$  equals  $b^n \bmod m$ }
  
```

$O((\log m)^2 \log n)$

## Definition

An 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 integer greater than 1 can be written uniquely as a prime or as the product of two or more primes, where the prime factors are written in order of nondecreasing size.

## Theorem 1

If  $n$  is a composite integer, then  $n$  has a prime divisor less than or equal to  $\sqrt{n}$ .

## Problem of factoring an integer into primes

### Trail and error

Start by checking from lowest prime and continue till  $\sqrt{n}$  whether it divides the given number

If it does we get two factors  $p$  and  $n/p$  so we continue the same until the second factor is itself prime

## Problem of finding all primes less than $n$

Fact that follows from Theorem 1 : a composite number  $c$  must have a prime factor less than or equal to  $\sqrt{n}$ .

### sieve of Eratosthenes

List all the numbers up to  $n$

Cancel 1

Then start with 2 and cancel all the multiples of it except two itself

Then see the next uncanceled no. it must be prime and cancel all the multiples of it except the prime itself

Repeat until a prime greater than  $\sqrt{n}$  is encountered at which point stop the process

All uncanceled numbers are prime

## Theorem 2

there are infinitely many primes.

## Additional points

The primes of the form  $2^p - 1$  is called Mersenne primes

The primality is tested with lucas-lehmar test

### Distribution of primes

**THE PRIME NUMBER THEOREM** The ratio of  $\pi(x)$ , the number of primes not exceeding  $x$ , and  $x / \ln x$  approaches 1 as  $x$  grows without bound. (Here  $\ln x$  is the natural logarithm of  $x$ .)

There are a lot of conjectures relating to prime an interesting field of research.

## GCD

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  $\gcd(a, b)$ .

The integers  $a$  and  $b$  are relatively prime if their greatest common divisor is 1.

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

## LCM

The least common multiple of the positive integers  $a$  and  $b$  is the smallest positive integer that is divisible by both  $a$  and  $b$ . The least common multiple of  $a$  and  $b$  is denoted by  $\text{lcm}(a, b)$

$$a = p_1^{a_1} p_2^{a_2} \cdots p_n^{a_n}, \quad b = p_1^{b_1} p_2^{b_2} \cdots p_n^{b_n},$$

Where  $p$  means prime

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

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

## Theorem

Let  $a$  and  $b$  be positive integers. Then  $ab = \gcd(a, b) \cdot \text{lcm}(a, b)$ .

## The Euclidean algorithm

## Lemma

Let  $a = bq + r$ , where  $a, b, q$ , and  $r$  are integers. Then  $\gcd(a, b) = \gcd(b, r)$ .

So  $\gcd(a, b) = \gcd(b, a-b)$  if  $a > b$

**ALGORITHM 1 The Euclidean Algorithm.**

**procedure**  $\gcd(a, b)$ : positive integers)

$x := a$

$y := b$

**while**  $y \neq 0$

$r := x \bmod y$

$x := y$

$y := r$

**return**  $x$  {  $\gcd(a, b)$  is  $x$  }

## BEZOUT'S THEOREM

If  $a$  and  $b$  are positive integers, then there exist integers  $s$  and  $t$  such that  $\gcd(a, b) = sa + tb$ .

The above equation is called Bezout's identity  
 $s$  and  $t$  are called Bezout coefficients

Finding bezout's coefficient

Solving linear congruences, which have the form  $ax \equiv b \pmod{m}$ , is an essential task in the study of number theory and its applications, just as solving linear equations plays an important role in calculus and linear algebra.

we solve the congruence  $ax \equiv b \pmod{m}$  by multiplying both sides of the congruence by the inverse of  $a \pmod{m}$ .

Inverse of  $a \pmod{m}$  is a number  $a^{-1}$  if  $aa^{-1} \equiv 1 \pmod{m}$

By Theorem 6 of Section 4.3, because  $\gcd(a, m) = 1$ , there are integers  $s$  and  $t$  such that  $sa + tm = 1$ .

This implies that  $sa + tm \equiv 1 \pmod{m}$ .

Because  $tm \equiv 0 \pmod{m}$ ,

it follows that  $sa \equiv 1 \pmod{m}$ .

Consequently,  $s$  is an inverse of  $a$  modulo  $m$

## Brute force

Using inspection to find an inverse of a modulo  $m$  is easy when  $m$  is small.

To find this inverse, we look for a multiple of  $a$  that exceeds a multiple of  $m$  by 1.

For example, to find an inverse of 3 modulo 7, we can find  $j \cdot 3$  for  $j = 1, 2, \dots, 6$ , stopping when we find a multiple of 3 that is one more than a multiple of 7. We can speed this approach up if we note that  $2 \cdot 3 \equiv -1 \pmod{7}$ . This means that  $(-2) \cdot 3 \equiv 1 \pmod{7}$ . Hence,  $5 \cdot 3 \equiv 1 \pmod{7}$ , so 5 is an inverse of 3 modulo 7.

## More efficient algorithm

We can design a more efficient algorithm than brute force to find an inverse of  $a$  modulo  $m$  when  $\gcd(a, m) = 1$  using the steps of the Euclidean algorithm. By reversing these steps as in Example 17 of Section 4.3, we can find a linear combination  $sa + tm = 1$ , where  $s$  and  $t$  are integers. Reducing both sides of this equation modulo  $m$  tells us that  $s$  is an inverse of  $a$  modulo  $m$ .

**EXAMPLE 3** What are the solutions of the linear congruence  $3x \equiv 4 \pmod{7}$ ?


**Solution:** By Example 1 we know that  $-2$  is an inverse of 3 modulo 7. Multiplying both sides of the congruence by  $-2$  shows that

$$-2 \cdot 3x \equiv -2 \cdot 4 \pmod{7}.$$

Because  $-6 \equiv 1 \pmod{7}$  and  $-8 \equiv 6 \pmod{7}$ , it follows that if  $x$  is a solution, then  $x \equiv -8 \equiv 6 \pmod{7}$ .

We need to determine whether every  $x$  with  $x \equiv 6 \pmod{7}$  is a solution. Assume that  $x \equiv 6 \pmod{7}$ . Then, by Theorem 5 of Section 4.1, it follows that

$$3x \equiv 3 \cdot 6 = 18 \equiv 4 \pmod{7},$$

which shows that all such  $x$  satisfy the congruence. We conclude that the solutions to the congruence are the integers  $x$  such that  $x \equiv 6 \pmod{7}$ , namely, 6, 13, 20, ... and  $-1, -8, -15, \dots$  



If

$$x \equiv 2 \pmod{3},$$

$$x \equiv 3 \pmod{5},$$

$$x \equiv 2 \pmod{7}?$$

The find x

The solution to the problem is given by chinese remainder theorem

**THE CHINESE REMAINDER THEOREM** Let  $m_1, m_2, \dots, m_n$  be pairwise relatively prime positive integers greater than one and  $a_1, a_2, \dots, a_n$  arbitrary integers. Then the system

$$x \equiv a_1 \pmod{m_1},$$

$$x \equiv a_2 \pmod{m_2},$$

.

.

.

$$x \equiv a_n \pmod{m_n}$$

has a unique solution modulo  $m = m_1 m_2 \cdots m_n$ . (That is, there is a solution  $x$  with  $0 \leq x < m$ , and all other solutions are congruent modulo  $m$  to this solution.)

See examples for how to find x..

To perform arithmetic with large integers, we select moduli  $m_1, m_2, \dots, m_n$ , where each  $m_i$  is an integer greater than 2,  $\gcd(m_i, m_j) = 1$  whenever  $i \neq j$ , and  $m = m_1 m_2 \cdots m_n$  is greater than the results of the arithmetic operations we want to carry out.

Once we have selected our moduli, we carry out arithmetic operations with large integers by performing componentwise operations on the  $n$ -tuples representing these integers using their remainders upon division by  $m_i$ ,  $i = 1, 2, \dots, n$ . Once we have computed the value of each component in the result, we recover its value by solving a system of  $n$  congruences modulo  $m_i$ ,  $i = 1, 2, \dots, n$ . This method of performing arithmetic with large integers has several valuable features. First, it can be used to perform arithmetic with integers larger than can ordinarily be carried out on a computer. Second, computations with respect to the different moduli can be done in parallel, speeding up the arithmetic.

$p$  divides  $a^p - a$  whenever  $p$  is prime and  $a$  is an integer not divisible by  $p$ .

**FERMAT'S LITTLE THEOREM** If  $p$  is prime and  $a$  is an integer not divisible by  $p$ , then

$$a^{p-1} \equiv 1 \pmod{p}.$$

Furthermore, for every integer  $a$  we have

$$a^p \equiv a \pmod{p}.$$

**EXAMPLE 9** Find  $7^{222} \bmod 11$ .

*Solution:* We can use Fermat's little theorem to evaluate  $7^{222} \bmod 11$  rather than using the fast modular exponentiation algorithm. By Fermat's little theorem we know that  $7^{10} \equiv 1 \pmod{11}$ , so  $(7^{10})^k \equiv 1 \pmod{11}$  for every positive integer  $k$ . To take advantage of this last congruence, we divide the exponent 222 by 10, finding that  $222 = 22 \cdot 10 + 2$ . We now see that

$$7^{222} = 7^{22 \cdot 10 + 2} = (7^{10})^{22} 7^2 \equiv (1)^{22} \cdot 49 \equiv 5 \pmod{11}.$$

It follows that  $7^{222} \bmod 11 = 5$ .



Let  $b$  be a positive integer. If  $n$  is a composite positive integer, and  $b^{n-1} \equiv 1 \pmod{n}$ , then  $n$  is called a *pseudoprime to the base  $b$* .

A composite integer  $n$  that satisfies the congruence  $b^{n-1} \equiv 1 \pmod{n}$  for all positive integers  $b$  with  $\gcd(b, n) = 1$  is called a *Carmichael number*. (These numbers are named after Robert Carmichael, who studied them in the early twentieth century.)

A *primitive root* modulo a prime  $p$  is an integer  $r$  in  $\mathbf{Z}_p$  such that every nonzero element of  $\mathbf{Z}_p$  is a power of  $r$ .

Suppose that  $p$  is a prime,  $r$  is a primitive root modulo  $p$ , and  $a$  is an integer between 1 and  $p - 1$  inclusive. If  $r^e \bmod p = a$  and  $0 \leq e \leq p - 1$ , we say that  $e$  is the *discrete logarithm* of  $a$  modulo  $p$  to the base  $r$  and we write  $\log_r a = e$  (where the prime  $p$  is understood).

# Omissions

16 May 2021 10:38 AM

Euclidian algorithms --to be done while doing sums  
Application of linear congruences  
cryptography