… and the following mathematical appetizer is about…

Functions

Functions

A function f from a set A to a set B is an assignment of exactly one element of B to each element of A.

We write

$f(a) = b$

if b is the unique element of B assigned by the function f to the element a of A.

If f is a function from A to B, we write

$f: A \rightarrow B$

(note: Here, "$\rightarrow$" has nothing to do with if… then)

Functions

If $f:A \to B$, we say that A is the domain of f and B is the codomain of f.

If $f(a) = b$, we say that b is the image of a and a is the pre-image of b.

The range of $f:A \to B$ is the set of all images of all elements of A.

We say that $f:A \to B$ maps A to B.

Functions

Let us take a look at the function $f: P \rightarrow C$ with

$P = \{Linda, Max, Kathy, Peter\}$

$C = \{Boston, New York, Hong Kong, Moscow\}$

$f(Linda) = Moscow$

$f(Max) = Boston$

$f(Kathy) = Hong Kong$

$f(Peter) = New York$

Here, the range of f is C.

Functions

Let us re-specify f as follows:

f(Linda) = Moscow

f(Max) = Boston

f(Kathy) = Hong Kong

f(Peter) = Boston

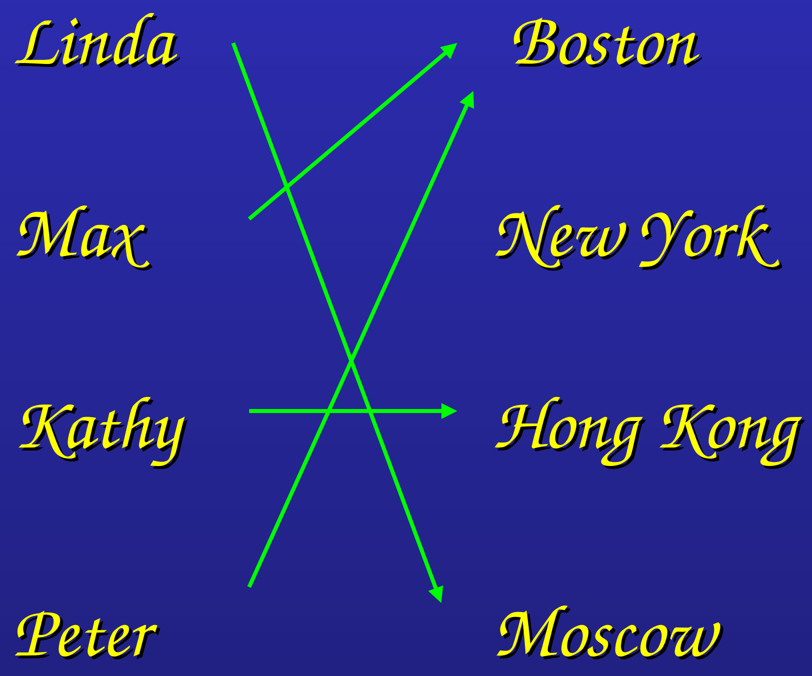
Is f still a function? yes

What is its range? {Moscow, Boston, Hong Kong}

Functions

Other ways to represent *f*:

x	*f*(*x*)
Linda	Moscow
Max	Boston
Kathy	Hong Kong
Peter	Boston



Functions

If the domain of our function f is large, it is convenient to specify f with a formula, e.g.:

$f:\mathbf{R}\rightarrow\mathbf{R}$

$f(x) = 2x$

This leads to:

$f(1) = 2$

$f(3) = 6$

$f(-3) = -6$

…

Functions

Let f_1 and f_2 be functions from A to $\mathbf{R}$.

Then the sum and the product of f_1 and f_2 are also functions from A to $\mathbf{R}$ defined by:

$(f_1 + f_2)(x) = f_1(x) + f_2(x)$

$(f_1f_2)(x) = f_1(x) f_2(x)$

Example:

$f_1(x) = 3x,\ f_2(x) = x + 5$

$(f_1 + f_2)(x) = f_1(x) + f_2(x) = 3x + x + 5 = 4x + 5$

$(f_1f_2)(x) = f_1(x) f_2(x) = 3x (x + 5) = 3x^2 + 15x$

Functions

We already know that the range of a function $f: A \to B$ is the set of all images of elements $a \in A$.

If we only regard a subset $S \subseteq A$, the set of all images of elements $s \in S$ is called the image of S.

We denote the image of S by $f(S)$:

$$f(S) = \{f(s) \mid s \in S\}$$

Functions

Let us look at the following well-known function:

f(Linda) = Moscow

f(Max) = Boston

f(Kathy) = Hong Kong

f(Peter) = Boston

What is the image of S = {Linda, Max} ?

f(S) = {Moscow, Boston}

What is the image of S = {Max, Peter} ?

f(S) = {Boston}

Properties of Functions

A function $f: A \to B$ is said to be one-to-one (or injective), if and only if

$$\forall x, y \in A \ (f(x) = f(y) \to x = y)$$

In other words: f is one-to-one if and only if it does not map two distinct elements of A onto the same element of B.

Properties of Functions

And again…

f(Linda) = Moscow	g(Linda) = Moscow
f(Max) = Boston	g(Max) = Boston
f(Kathy) = Hong Kong	g(Kathy) = Hong Kong
f(Peter) = Boston	g(Peter) = New York
Is f one-to-one?	Is g one-to-one?
No, Max and Peter are mapped onto the same element of the image.	Yes, each element is assigned a unique element of the image.

Properties of Functions

How can we prove that a function f is one-to-one?

Whenever you want to prove something, first take a look at the relevant definition(s):

$\forall x, y \in A\ (f(x) = f(y) \rightarrow x = y)$

Example:

$f: \mathbf{R} \rightarrow \mathbf{R}$

$f(x) = x^2$

Disproof by counterexample:

$f(3) = f(-3)$, but $3 \neq -3$, so f is not one-to-one.

Properties of Functions

… and yet another example:

$f: \mathbf{R} \to \mathbf{R}$

$f(x) = 3x$

One-to-one: $\forall x, y \in A\ (f(x) = f(y) \to x = y)$

To show: $f(x) \neq f(y)$ whenever $x \neq y$ (indirect proof)

$x \neq y$

$\Leftrightarrow 3x \neq 3y$

$\Leftrightarrow f(x) \neq f(y)$,

so if $x \neq y$, then $f(x) \neq f(y)$, that is, f is one-to-one.

Properties of Functions

A function $f:A \to B$ is called onto, or surjective, if and only if for every element $b \in B$ there is an element $a \in A$ with $f(a) = b$.

In other words, f is onto if and only if its range is its entire codomain.

A function $f: A \to B$ is a one-to-one correspondence, or a bijection, if and only if it is both one-to-one and onto.

Obviously, if f is a bijection and A and B are finite sets, then $|A| = |B|$.

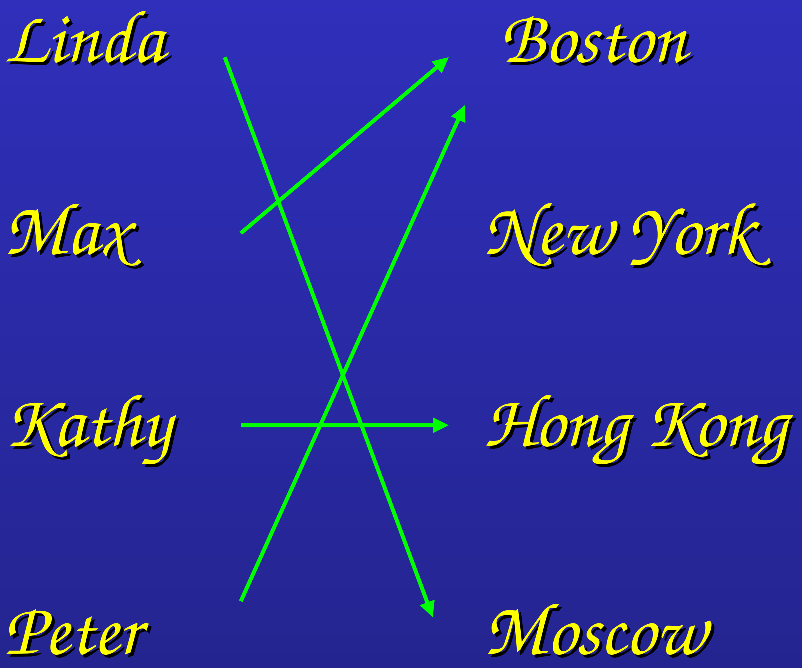
Properties of Functions

Examples:

In the following examples, we use the arrow representation to illustrate functions $f:A \rightarrow B$.

In each example, the complete sets A and B are shown.

Properties of Functions



Is f injective?

No.

Is f surjective?

No.

Is f bijective?

No.

Properties of Functions

Is f injective?

No.

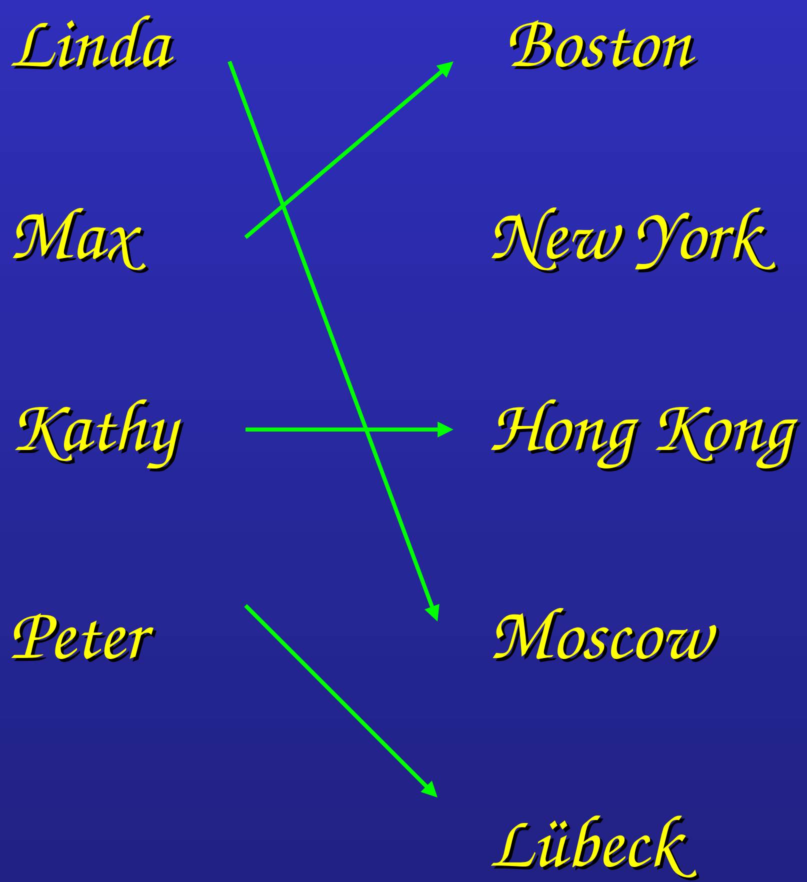
Is f surjective?

Yes.

Is f bijective?

No.

Properties of Functions



Is f injective?

Yes.

Is f surjective?

No.

Is f bijective?

No.

Properties of Functions

Is f injective?

No! f is not even a function!

Properties of Functions

Is f injective?

Yes.

Is f surjective?

Yes.

Is f bijective?

Yes.

Inversion

An interesting property of bijections is that they have an ***inverse function****.*

The ***inverse function*** *of the bijection* $f:A \to B$ *is the function* $f^{-1}:B \to A$ *with*

$f^{-1}(b) = a$ *whenever* $f(a) = b$.

Inversion

Example:

$f(Linda) = Moscow$

$f(Max) = Boston$

$f(Kathy) = Hong\ Kong$

$f(Peter) = Lübeck$

$f(Helena) = New\ York$

Clearly, f is bijective.

The inverse function f^{-1} is given by:

$f^{-1}(Moscow) = Linda$

$f^{-1}(Boston) = Max$

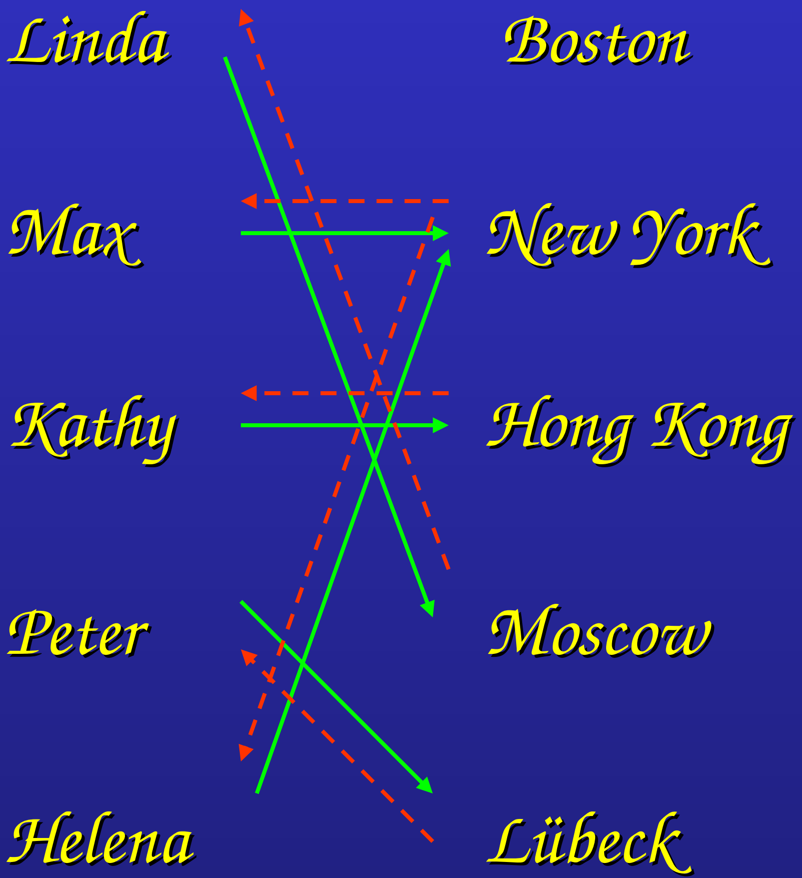
$f^{-1}(Hong\ Kong) = Kathy$

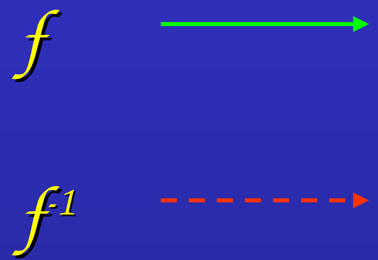
$f^{-1}(Lübeck) = Peter$

$f^{-1}(New\ York) = Helena$

Inversion is only possible for bijections
(= invertible functions)

Inversion





$f^{-1}: C \rightarrow P$ is no function, because it is not defined for all elements of C and assigns two images to the pre-image New York.

Composition

The **composition** of two functions $g: A \to B$ and $f: B \to C$, denoted by $f \circ g$, is defined by

$$(f \circ g)(a) = f(g(a))$$

This means that

- **first**, function g is applied to element $a \in A$, mapping it onto an element of B,
- **then**, function f is applied to this element of B, mapping it onto an element of C.
- **Therefore**, the composite function maps from A to C.

Composition

Example:

$f(x) = 7x - 4$, $g(x) = 3x$,

$f: \mathbb{R} \to \mathbb{R}$, $g: \mathbb{R} \to \mathbb{R}$

$(f \circ g)(5) = f(g(5)) = f(15) = 105 - 4 = 101$

$(f \circ g)(x) = f(g(x)) = f(3x) = 21x - 4$

Composition

Composition of a function and its inverse:

$(f^{-1} \circ f)(x) = f^{-1}(f(x)) = x$

The composition of a function and its inverse is the ***identity function*** $i(x) = x$.

Floor and Ceiling Functions

The **floor** and **ceiling** functions map the real numbers onto the integers ($\mathbf{R} \rightarrow \mathbf{Z}$).

The **floor** function assigns to $r \in \mathbf{R}$ the largest $z \in \mathbf{Z}$ with $z \leq r$, denoted by $\lfloor r \rfloor$.

Examples: $\lfloor 2.3 \rfloor = 2$, $\lfloor 2 \rfloor = 2$, $\lfloor 0.5 \rfloor = 0$, $\lfloor -3.5 \rfloor = -4$

The **ceiling** function assigns to $r \in \mathbf{R}$ the smallest $z \in \mathbf{Z}$ with $z \geq r$, denoted by $\lceil r \rceil$.

Examples: $\lceil 2.3 \rceil = 3$, $\lceil 2 \rceil = 2$, $\lceil 0.5 \rceil = 1$, $\lceil -3.5 \rceil = -3$

Now, something about

BooleanAlge
bra

Boolean Algebra

Boolean algebra provides the operations and the rules for working with the set **{0, 1}**.

These are the rules that underlie **electronic circuits**, and the methods we will discuss are fundamental to **VLSI design**.

We are going to focus on three operations:

- Boolean complementation,
- Boolean sum, and
- Boolean product

Boolean Operations

The ***complement*** *is denoted by a bar (on the slides, we will use a minus sign). It is defined by*

$-0 = 1$ *and* $-1 = 0$.

The ***Boolean sum****, denoted by + or by OR, has the following values:*

$1 + 1 = 1, \quad 1 + 0 = 1, \quad 0 + 1 = 1, \quad 0 + 0 = 0$

The ***Boolean product****, denoted by · or by AND, has the following values:*

$1 \cdot 1 = 1, \quad 1 \cdot 0 = 0, \quad 0 \cdot 1 = 0, \quad 0 \cdot 0 = 0$

Boolean Functions and Expressions

Definition: Let $B = \{0, 1\}$. The variable x is called a **Boolean variable** if it assumes values only from B.

A function from B^n, the set $\{(x_1, x_2, \ldots, x_n) \mid x_i \in B, 1 \leq i \leq n\}$, to B is called a **Boolean function of degree n.**

Boolean functions can be represented using expressions made up from the variables and Boolean operations.

Boolean Functions and Expressions

The **Boolean expressions** in the variables $x_1, x_2, \ldots, x_n$ are defined recursively as follows:

- $0, 1, x_1, x_2, \ldots, x_n$ are Boolean expressions.
- If E_1 and E_2 are Boolean expressions, then $(-E_1)$, (E_1E_2), and $(E_1 + E_2)$ are Boolean expressions.

Each Boolean expression represents a Boolean function. The values of this function are obtained by substituting 0 and 1 for the variables in the expression.

Boolean Functions and Expressions

For example, we can create Boolean expression in the variables x, y, and z using the "building blocks"
0, 1, x, y, and z, and the construction rules:

Since x and y are Boolean expressions, so is xy.

Since z is a Boolean expression, so is (-z).

Since xy and (-z) are expressions, so is xy + (-z).

… and so on…

Boolean Functions and Expressions

Example: *Give a Boolean expression for the Boolean function $F(x, y)$ as defined by the following table:*

x	y	$F(x, y)$
0	0	0
0	1	1
1	0	0
1	1	0

Possible solution: $F(x, y) = (-x) \cdot y$

Boolean Functions and Expressions

Another Example:

x	y	z	$F(x, y, z)$
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

Possible solution I:

$F(x, y, z) = -(xz + y)$

Possible solution II:

$F(x, y, z) = (-(xz))(-y)$

Boolean Functions and Expressions

There is a simple method for deriving a Boolean expression for a function that is defined by a table. This method is based on **minterms**.

Definition: A **literal** is a Boolean variable or its complement. A **minterm** of the Boolean variables $x_1, x_2, \ldots, x_n$ is a Boolean product $y_1 y_2 \ldots y_n$, where $y_i = x_i$ or $y_i = -x_i$.

Hence, a minterm is a product of n literals, with one literal for each variable.

Boolean Functions and Expressions

Consider $F(x,y,z)$ again:

x	y	z	$F(x, y, z)$
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

$F(x, y, z) = 1$ if and only if:

$x = y = z = 0$ or

$x = y = 0, z = 1$ or

$x = 1, y = z = 0$

Therefore,

$F(x, y, z) =$

$(-x)(-y)(-z) +$

$(-x)(-y)z +$

$x(-y)(-z)$

Boolean Functions and Expressions

Definition: The Boolean functions F and G of n variables are **equal** if and only if $F(b_1, b_2, \ldots, b_n) = G(b_1, b_2, \ldots, b_n)$ whenever $b_1, b_2, \ldots, b_n$ belong to B.

Two different Boolean expressions that represent the same function are called **equivalent**.

For example, the Boolean expressions xy, $xy + 0$, and $xy \cdot 1$ are equivalent.

Boolean Functions and Expressions

The ***complement*** *of the Boolean function F is the function $-F$, where*
$-F(b_1, b_2, \ldots, b_n) =$
$-(F(b_1, b_2, \ldots, b_n))$.

Let F and G be Boolean functions of degree n. The ***Boolean sum $F+G$*** *and* ***Boolean product FG*** *are then defined by*

$$(F + G)(b_1, b_2, \ldots, b_n) = F(b_1, b_2, \ldots, b_n) + G(b_1, b_2, \ldots, b_n)$$

$$(FG)(b_1, b_2, \ldots, b_n) = F(b_1, b_2, \ldots, b_n)\, G(b_1, b_2, \ldots, b_n)$$

Boolean Functions and Expressions

Question: How many different Boolean functions of degree 1 are there?

Solution: There are four of them, F_1, F_2, F_3, and F_4:

x	F_1	F_2	F_3	F_4
0	0	0	1	1
1	0	1	0	1

Boolean Functions and Expressions

Question: How many different Boolean functions of degree 2 are there?

Solution: There are 16 of them, F_1, F_2, …, F_{16}:

x	y	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}	F_{16}
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Boolean Functions and Expressions

Question: How many different Boolean functions of degree n are there?

Solution:

There are 2^n different n-tuples of 0s and 1s.

A Boolean function is an assignment of 0 or 1 to each of these 2^n different n-tuples.

Therefore, there are 2^{2^n} different Boolean functions.

Duality

There are useful identities of Boolean expressions that can help us to transform an expression A into an equivalent expression B

We can derive additional identities with the help of the **dual** of a Boolean expression.

The dual of a Boolean expression is obtained by interchanging Boolean sums and Boolean products and interchanging 0s and 1s.

Duality

Examples:

The dual of $x(y + z)$ is $x + yz$.

The dual of $-x \cdot 1 + (-y + z)$ is $(-x + 0)((-y)z)$.

The **dual of a Boolean function F** represented by a Boolean expression is the function represented by the dual of this expression.

This dual function, denoted by F^d, **does not depend** on the particular Boolean expression used to represent F.

Duality

Therefore, an identity between functions represented by Boolean expressions **remains valid** when the duals of both sides of the identity are taken.

We can use this fact, called the **duality principle**, to derive new identities.

For example, consider the absorption law
$x(x + y) = x$.

By taking the duals of both sides of this identity, we obtain the equation $x + xy = x$, which is also an identity (and also called an absorption law).

Definition of a Boolean Algebra

All the properties of Boolean functions and expressions that we have discovered also apply to **other mathematical structures** such as propositions and sets and the operations defined on them.

If we can show that a particular structure is a Boolean algebra, then we know that all results established about Boolean algebras apply to this structure.

For this purpose, we need an **abstract definition** of a Boolean algebra.

Definition of a Boolean Algebra

Definition: *A Boolean algebra is a set B with two binary operations $\vee$ and $\wedge$, elements 0 and 1, and a unary operation – such that the following properties hold for all x, y, and z in B:*

$x \vee 0 = x$ and $x \wedge 1 = x$ (identity laws)

$x \vee (-x) = 1$ and $x \wedge (-x) = 0$ (domination laws)

$(x \vee y) \vee z = x \vee (y \vee z)$ and
$(x \wedge y) \wedge z = x \wedge (y \wedge z)$ and (associative laws)

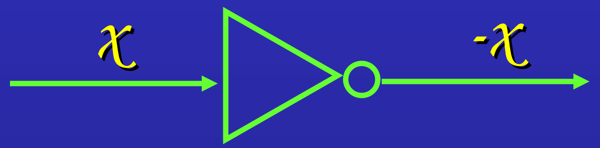
$x \vee y = y \vee x$ and $x \wedge y = y \wedge x$ (commutative laws)

$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$ and
$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ (distributive laws)

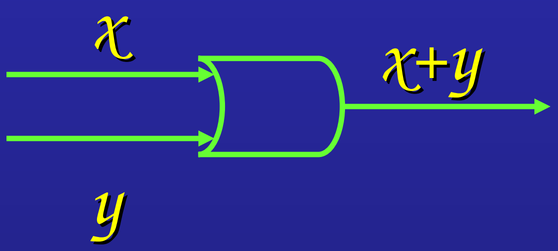
Logic Gates

Electronic circuits consist of so-called gates.

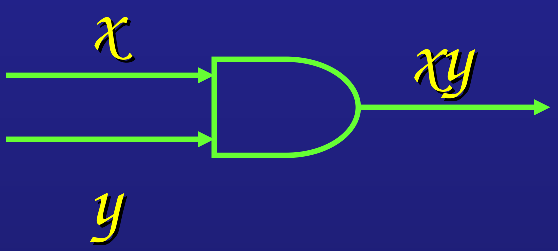
There are three basic types of gates:



inverter



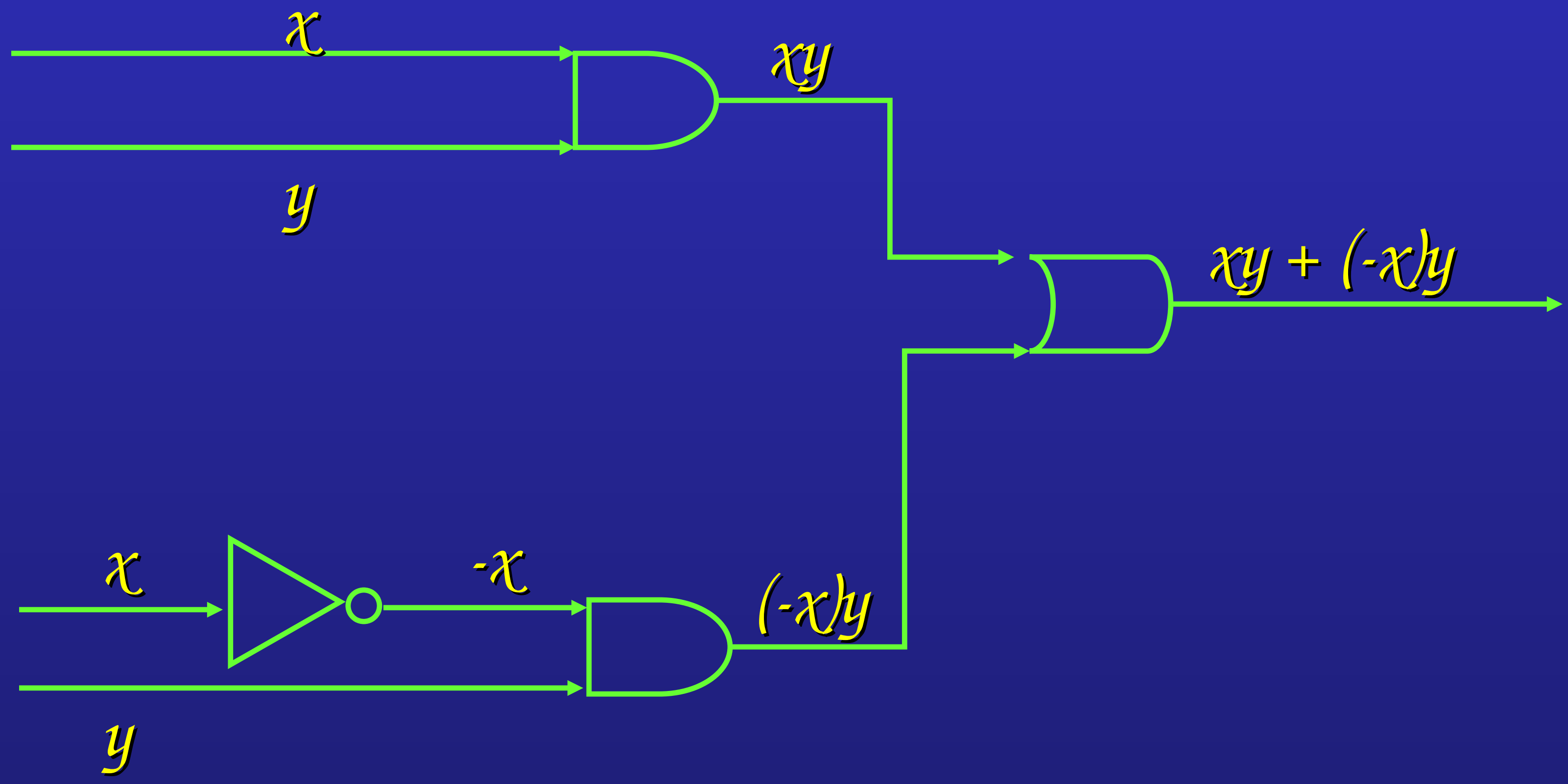
OR gate



AND gate

Logic Gates

Example: *How can we build a circuit that computes the function xy + (-x)y ?*



Logic, Sets, and Boolean Algebra

Logic	Set	Boolean Algebra
False	$\varnothing$	0
True	U	1
$A \wedge B$	$A \cap B$	$A \cdot B$
$A \vee B$	$A \cup B$	$A + B$
$\neg A$	A^C	$\overline{A}$

Compare the equivalence laws of them