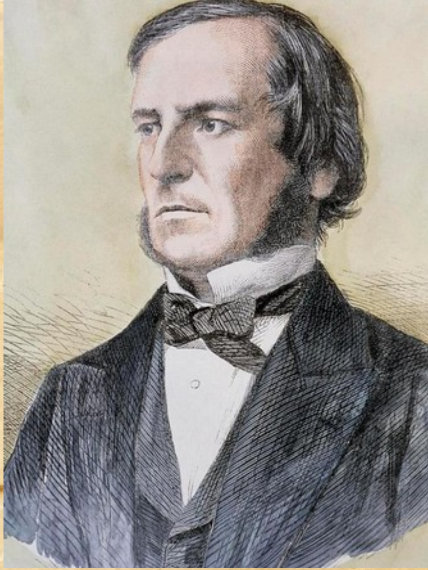
Boolean Logic

Introduction

Lecture Contents

- George Boole and Boolean Algebra
- Logic Gates
  - NOT, AND, OR, XOR
  - NAND, NOR, XNOR
- Algebraic Properties
  - Associative, Commutative, Distributive
- De Morgan’s Law
- Bitwise Operators

George Boole (1815-1864)

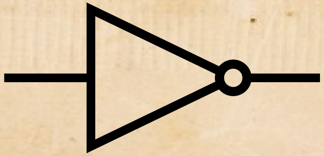


- Largely self-taught son of a shoemaker
- Professor of Mathematics, Queen's College, Cork, Ireland
- Founder of the system of binary algebra called ***Boolean algebra*** or *Boolean logic*
  - Elementary algebra describes numerical operations; *Boolean algebra* describes ***logical operations***
  - Relevant works:
    - *The Mathematical Analysis of Logic* (1847)
    - *An Investigation of the Laws of Thought* (1854)

Boolean Algebra

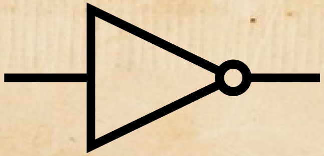
- Fundamental in the development of digital electronics
  - Operations provided for in all modern programming languages
- Used in set theory and statistics

Negation: **`NOT`**

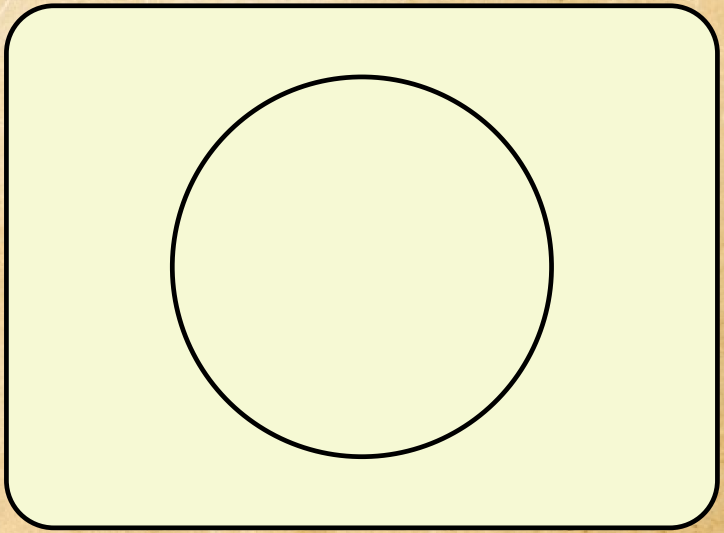


- Example English statement:
  - It is ***not*** Monday

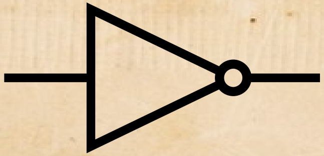
- Symbols:
  - Logic: $\neg$ Programming: ~ Set notation: $'$

Negation: **NOT** 

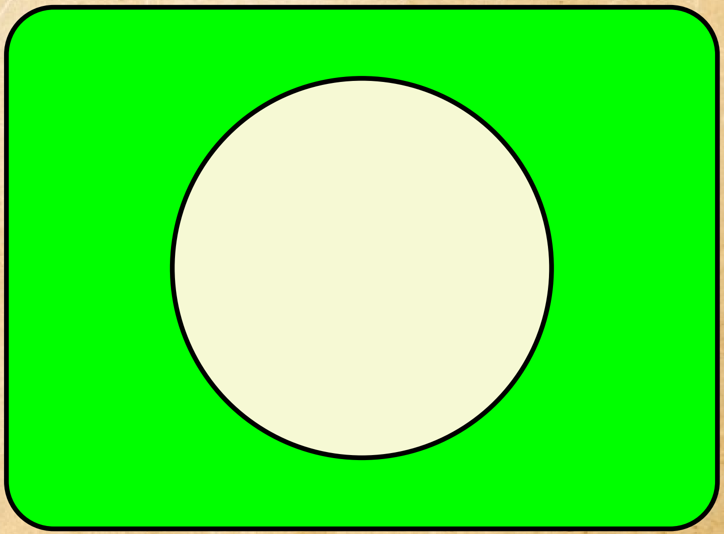
- Example English statement:
  - It is ***not*** Monday



- Symbols:
  - Logic: **¬**   Programming: **~**   Set notation: **′**

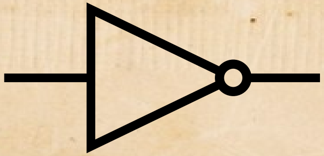
Negation: **NOT** 

- Example English statement:
  - It is ***not*** Monday



- Symbols:
  - Logic: ¬   Programming: ~   Set notation: ′

Negation: **NOT**

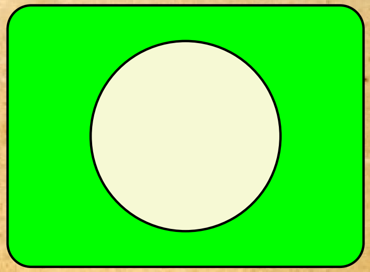


- Example English statement:
  - It is ***not*** Monday

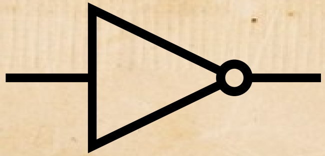
Truth Table for NOT

a	¬a
0	
1	

- Symbols:
  - Logic: ¬ Programming: ~ Set notation: ′



Negation: **NOT**

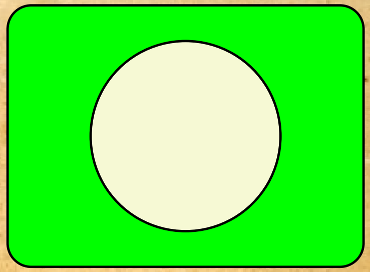


- Example English statement:
  - It is ***not*** Monday

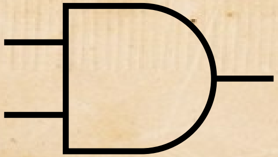
Truth Table for NOT

a	¬a
0	1
1	0

- Symbols:
  - Logic: ¬ Programming: ~ Set notation: ′



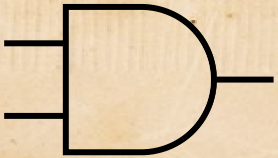
Conjunction: **AND**



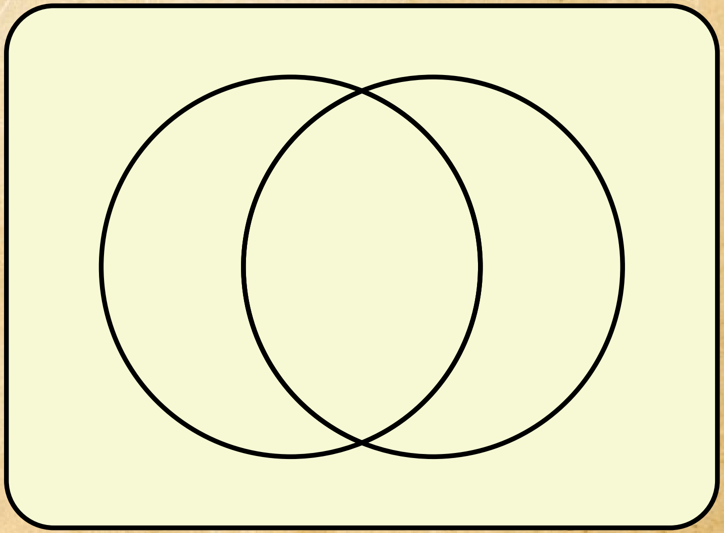
- Example English statement:
  - It is Monday ***and*** it is raining

- Switch equivalent

- Symbols:
  - Logic: $\wedge$ Programming: **&** Set notation: $\cap$

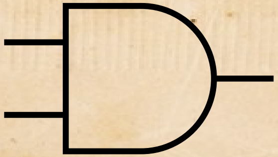
Conjunction: **AND** 

- Example English statement:
  - It is Monday ***and*** it is raining



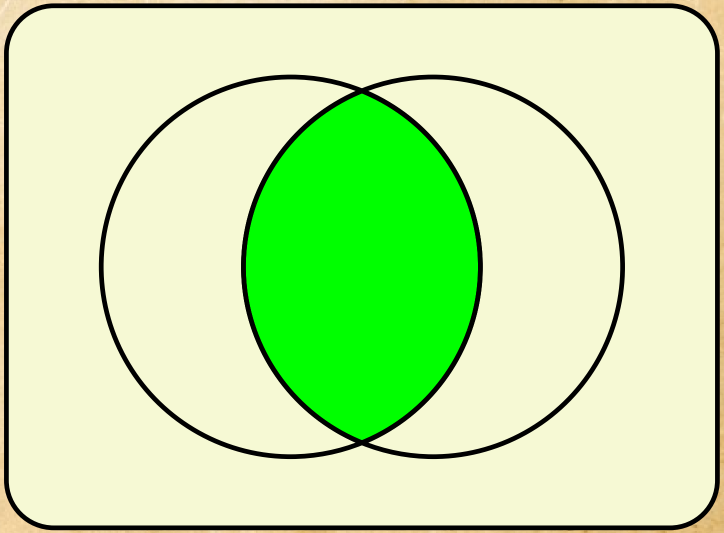
- Switch equivalent

- Symbols:
  - Logic: ∧ Programming: **&** Set notation: ∩

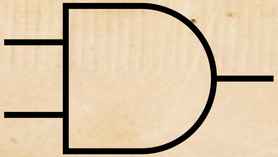
Conjunction: **AND** 

- Example English statement:
  - It is Monday ***and*** it is raining

- Switch equivalent



- Symbols:
  - Logic: ∧ Programming: **&** Set notation: ∩

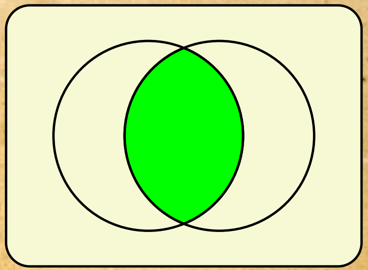
Conjunction: **AND** 

- Example English statement:
  - It is Monday ***and*** it is raining
- Switch equivalent

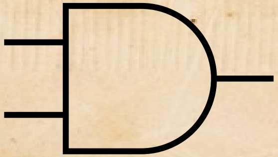
- Symbols:
  - Logic: $\wedge$ Programming: **&** Set notation: $\cap$

Truth Table for AND

a	b	a ∧ b
0	0	
0	1	
1	0	
1	1	



Conjunction: **AND**



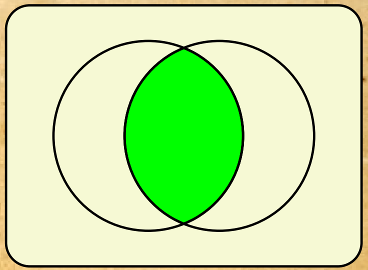
- Example English statement:
  - It is Monday ***and*** it is raining

- Switch equivalent

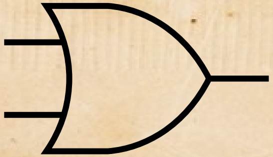
Truth Table for AND

a	b	a ∧ b
0	0	0
0	1	0
1	0	0
1	1	1

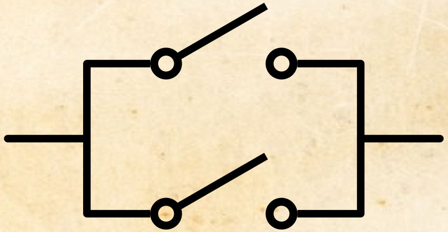
- Symbols:
  - Logic: ∧ Programming: **&** Set notation: ∩



Disjunction: **OR**

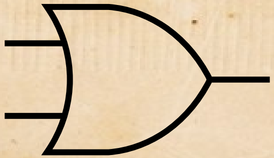


- Example English statement
  - You may be injured if you are hit or you fall
- Switch equivalent

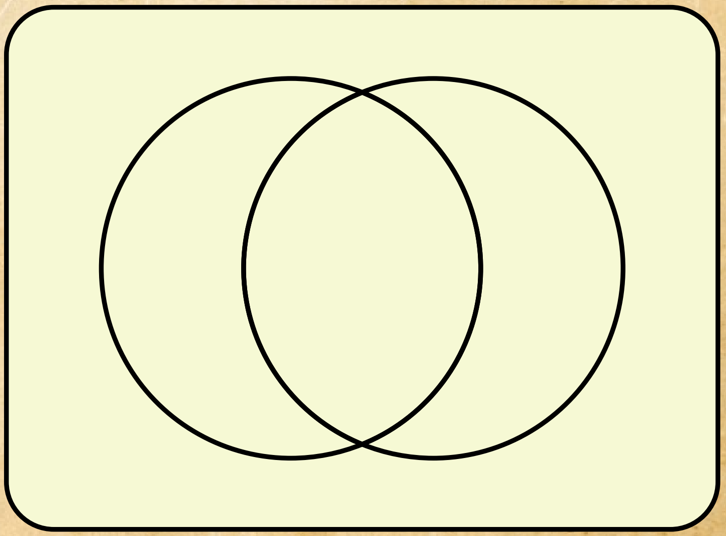


- Symbols:
  - Logic: $\vee$ Programming: | Set notation: $\cup$

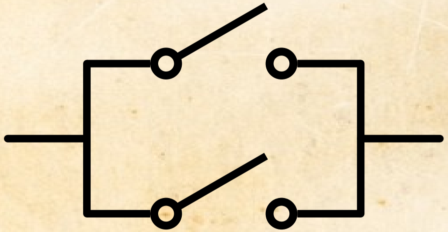
Disjunction: **OR**



- Example English statement
  - You may be injured if you are hit or you fall

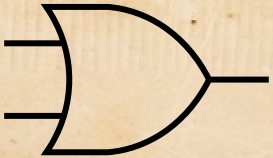


- Switch equivalent

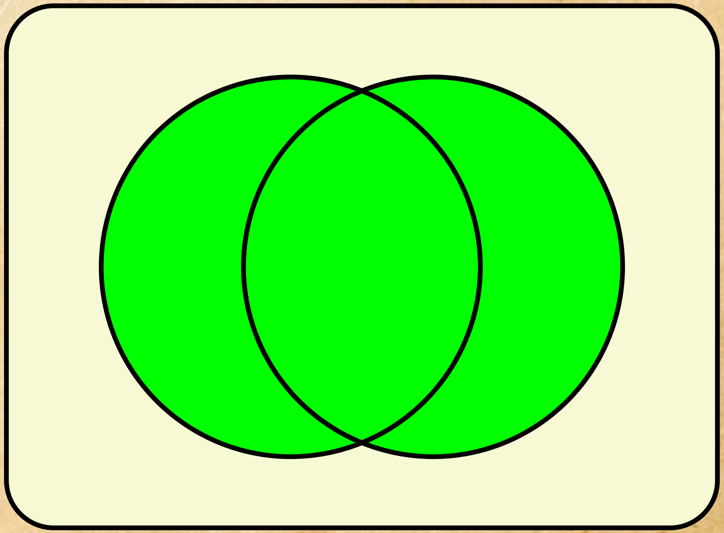


- Symbols:
  - Logic: $\lor$ Programming: | Set notation: $\cup$

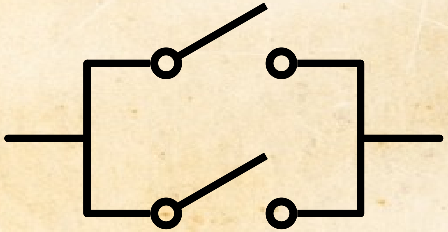
Disjunction: **OR**



- Example English statement
  - You may be injured if you are hit or you fall

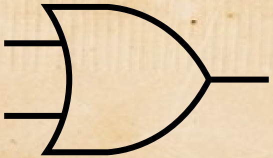


- Switch equivalent



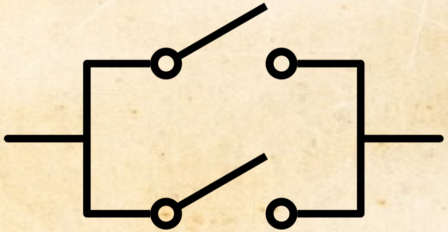
- Symbols:
  - Logic: $\vee$ Programming: | Set notation: $\cup$

Disjunction: **OR**



- Example English statement
  - You may be injured if you are hit or you fall

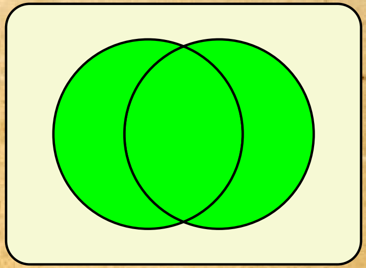
- Switch equivalent



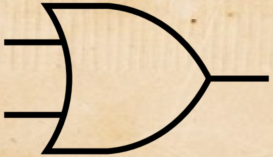
- Symbols:
  - Logic: ∨ Programming: | Set notation: ∪

Truth Table for OR

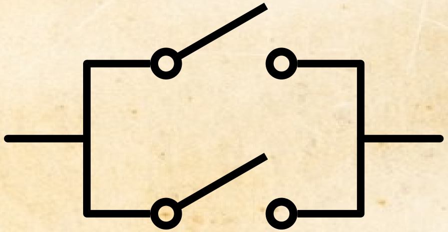
a	b	a ∨ b
0	0	
0	1	
1	0	
1	1	



Disjunction: **OR**



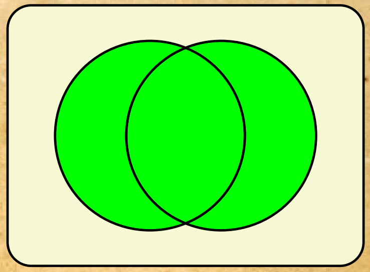
- Example English statement
  - You may be injured if you are hit or you fall
- Switch equivalent



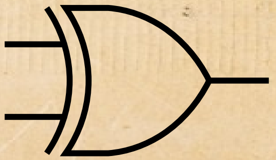
- Symbols:
  - Logic: $\vee$ Programming: | Set notation: $\cup$

Truth Table for OR

a	b	a ∨ b
0	0	0
0	1	1
1	0	1
1	1	1



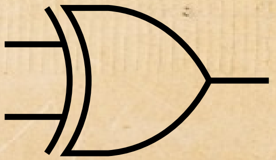
Exclusive Disjunction: **XOR**



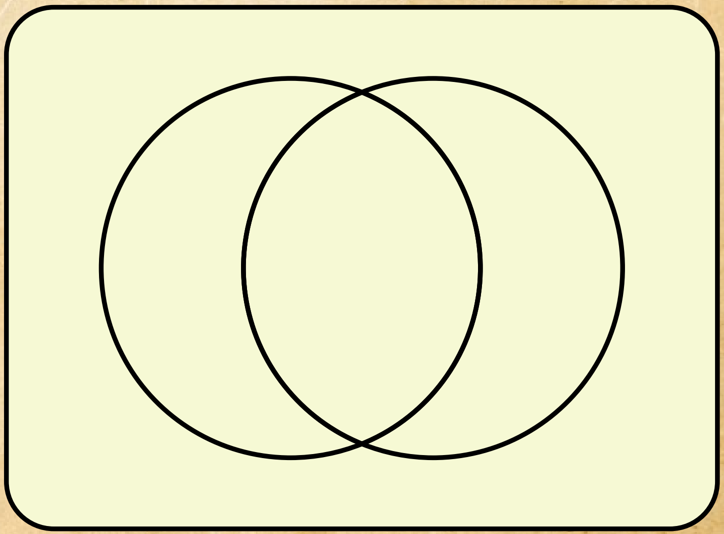
- Example English statement
  - Pick either heads or tails
  - I will watch the movie on Monday or Tuesday

- Symbols:
  - Logic: $\veebar$ Programming: ^ Set notation: *none*

Exclusive Disjunction: **XOR**

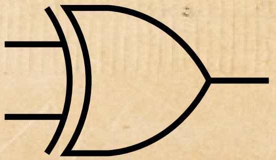


- Example English statement
  - Pick either heads or tails
  - I will watch the movie on Monday or Tuesday

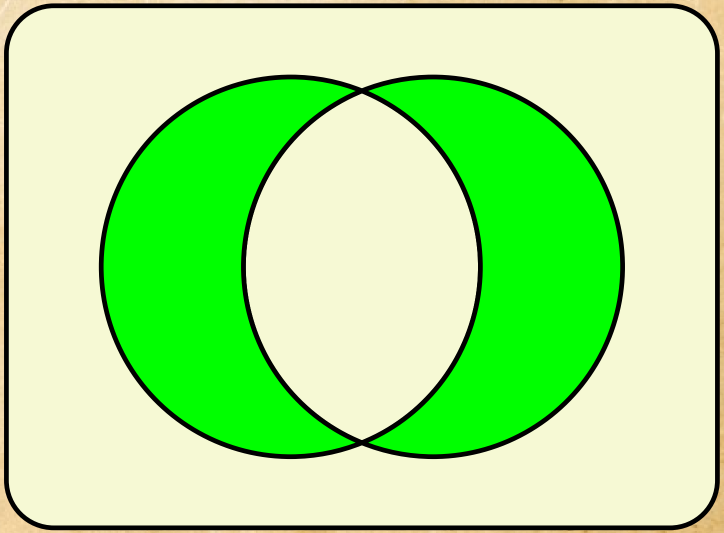


- Symbols:
  - Logic: $\veebar$ Programming: ^ Set notation: *none*

Exclusive Disjunction: **XOR**

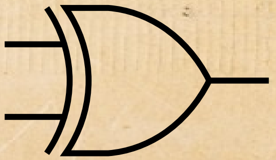


- Example English statement
  - Pick either heads or tails
  - I will watch the movie on Monday or Tuesday



- Symbols:
  - Logic: $\veebar$ Programming: ^ Set notation: *none*

Exclusive Disjunction: **XOR**

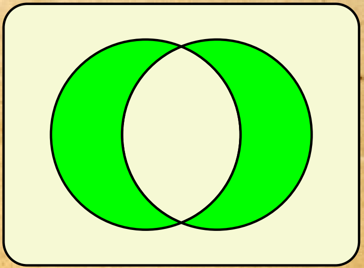


- Example English statement
  - Pick either heads or tails
  - I will watch the movie on Monday or Tuesday

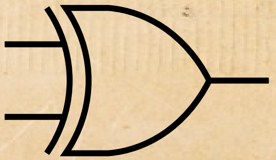
Truth Table for XOR

a	b	a ∨ b
0	0	
0	1	
1	0	
1	1	

- Symbols:
  - Logic: $\veebar$ Programming: ^ Set notation: *none*



Exclusive Disjunction: **XOR**

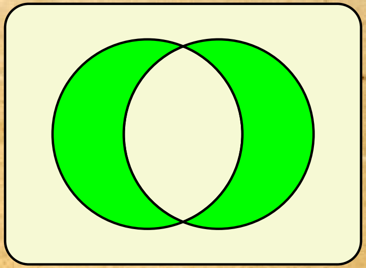


- Example English statement
  - Pick either heads or tails
  - I will watch the movie on Monday or Tuesday

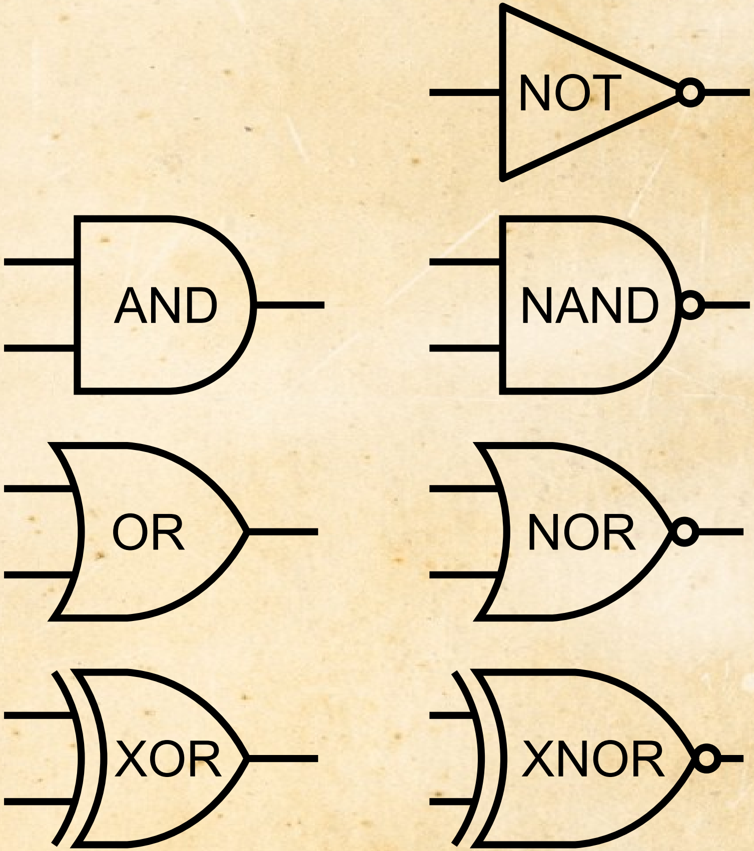
Truth Table for XOR

a	b	$a \vee b$
0	0	0
0	1	1
1	0	1
1	1	0

- Symbols:
  - Logic: $\veebar$ Programming: ^ Set notation: *none*



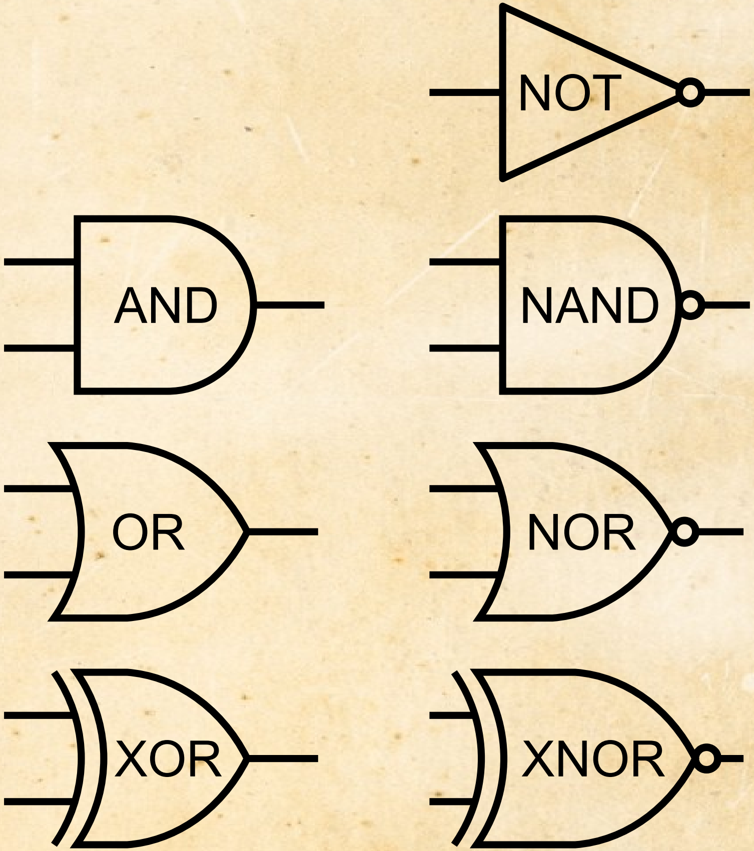
Inverted Logic Gates



Truth Table for AND and NAND

a	b	a ∧ b	¬(a ∧ b)
0	0		
0	1		
1	0		
1	1		

Inverted Logic Gates



Truth Table for AND and NAND

a	b	a ∧ b	¬(a ∧ b)
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Implication and Equivalence

Truth Table for IMPLY

a	b	A ⇒ b
0	0	1
0	1	1
1	0	0
1	1	1

Truth Table for EQUIV

a	b	A ⇔ b
0	0	1
0	1	0
1	0	0
1	1	1

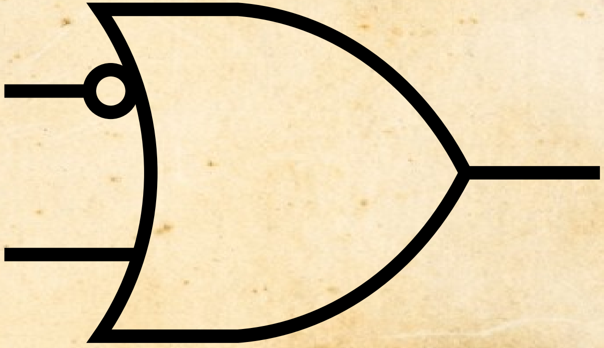
Implication and Equivalence

Truth Table for IMPLY

a	b	$A \Rightarrow b$
0	0	1
0	1	1
1	0	0
1	1	1

Truth Table for EQUIV

a	b	$A \Leftrightarrow b$
0	0	1
0	1	0
1	0	0
1	1	1



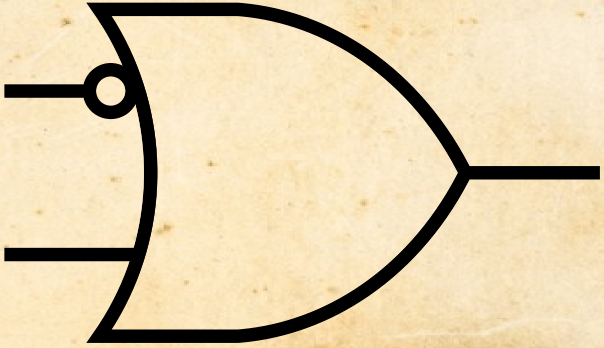
Implication and Equivalence

Truth Table for IMPLY

a	b	A ⇒ b
0	0	1
0	1	1
1	0	0
1	1	1

Truth Table for EQUIV

a	b	A ⇔ b
0	0	1
0	1	0
1	0	0
1	1	1



Properties of Elementary Algebra

- Associative Property

$$(a+b)+c = a+(b+c)$$

$$(a \cdot b) \cdot c = a \cdot (b \cdot c)$$

- Commutative Property

$$a+b = b+a \qquad a \cdot b = b \cdot a$$

- Distributive Property

$$a(b+c) = (a \cdot b)+(a \cdot c)$$

Associative Property

$$\left(a \cdot b\right) \cdot c = a \cdot \left(b \cdot c\right)$$

A	B	C	A∧B	(A∧B)∧C	B∧C	A∧(B∧C)
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

Associative Property

$$(a \cdot b) \cdot c = a \cdot (b \cdot c)$$

A	B	C	A∧B	(A∧B)∧C	B∧C	A∧(B∧C)
0	0	0	0	**0**	0	**0**
0	0	1	0	**0**	0	**0**
0	1	0	0	**0**	0	**0**
0	1	1	0	**0**	1	**0**
1	0	0	0	**0**	0	**0**
1	0	1	0	**0**	0	**0**
1	1	0	1	**0**	0	**0**
1	1	1	1	**1**	1	**1**

$$(A \wedge B) \wedge C = A \wedge (B \wedge C)$$

Associative Property

$$(a+b)+c = a+(b+c)$$

A	B	C	A∨B	(A∨B)∨C	B∨C	A∨(B∨C)
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

Associative Property

$$(a+b)+c = a+(b+c)$$

A	B	C	A∨B	(A∨B)∨C	B∨C	A∨(B∨C)
0	0	0	0	**0**	0	**0**
0	0	1	0	**1**	1	**1**
0	1	0	1	**1**	1	**1**
0	1	1	1	**1**	1	**1**
1	0	0	1	**1**	0	**1**
1	0	1	1	**1**	1	**1**
1	1	0	1	**1**	1	**1**
1	1	1	1	**1**	1	**1**

$$(A \vee B) \vee C = A \vee (B \vee C)$$

Distributive Rule $a(b+c) = (a \cdot b) + (a \cdot c)$

A	B	C	B∨C	A∧(B∨C)	A∧B	A∧C	(A∧B)∨(A∧C)
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

$$A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$$

Distributive Rule $a(b+c)=(a\cdot b)+(a\cdot c)$

A	B	C	B∨C	A∧(B∨C)	A∧B	A∧C	(A∧B)∨(A∧C)
0	0	0	0	**0**	0	0	**0**
0	0	1	1	**0**	0	0	**0**
0	1	0	1	**0**	0	0	**0**
0	1	1	1	**0**	0	0	**0**
1	0	0	0	**0**	0	0	**0**
1	0	1	1	**1**	0	1	**1**
1	1	0	1	**1**	1	0	**1**
1	1	1	1	**1**	1	1	**1**

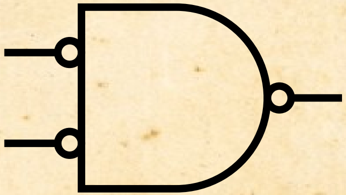
$$A\wedge(B\vee C) = (A\wedge B)\vee(A\wedge C)$$

De Morgan’s Laws

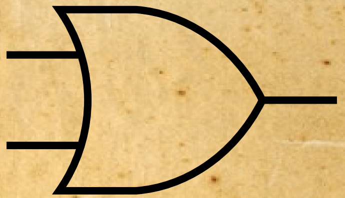
A	B	¬A	¬B	¬A∧¬B	¬(¬A∧¬B)	A∨B

De Morgan’s Laws

A	B	¬A	¬B	¬A∧¬B	¬(¬A∧¬B)	A∨B
0	0	1	1	1	**0**	**0**
0	1	1	0	0	**1**	**1**
1	0	0	1	0	**1**	**1**
1	1	0	0	0	**1**	**1**

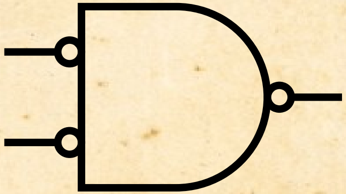


$$\neg(\neg \boldsymbol{A} \wedge \neg \boldsymbol{B}) = \boldsymbol{A} \vee \boldsymbol{B}$$



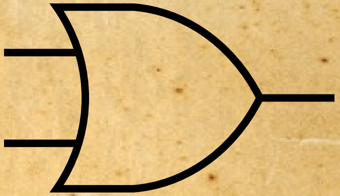
De Morgan’s Laws

A	B	¬A	¬B	¬A∧¬B	¬(¬A∧¬B)	A∨B
0	0	1	1	1	**0**	**0**
0	1	1	0	0	**1**	**1**
1	0	0	1	0	**1**	**1**
1	1	0	0	0	**1**	**1**



$$\neg(\neg \boldsymbol{A} \wedge \neg \boldsymbol{B}) = \boldsymbol{A} \vee \boldsymbol{B}$$

$$\overline{\overline{A} \wedge \overline{B}} = A \vee B$$

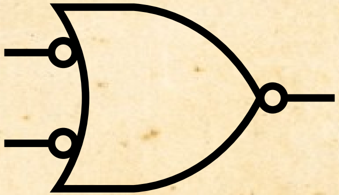


De Morgan’s Laws

A	B	¬A	¬B	¬A∨¬B	¬(¬A∨¬B)	A∧B

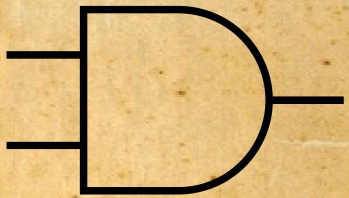
De Morgan's Laws

A	B	¬A	¬B	¬A∨¬B	¬(¬A∨¬B)	A∧B
0	0	1	1	1	**0**	**0**
0	1	1	0	1	**0**	**0**
1	0	0	1	1	**0**	**0**
1	1	0	0	0	**1**	**1**



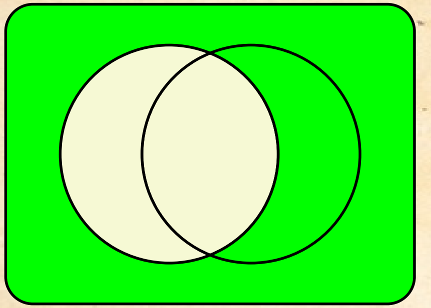
$$\neg(\neg \boldsymbol{A} \vee \neg \boldsymbol{B}) = \boldsymbol{A} \wedge \boldsymbol{B}$$

$$\overline{\overline{A} \vee \overline{B}} = A \wedge B$$

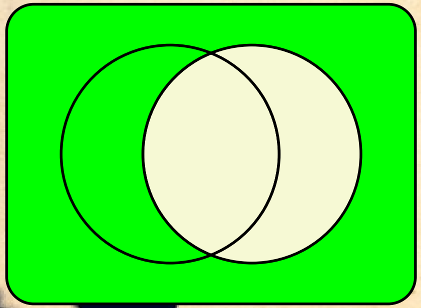


De Morgan’s Laws

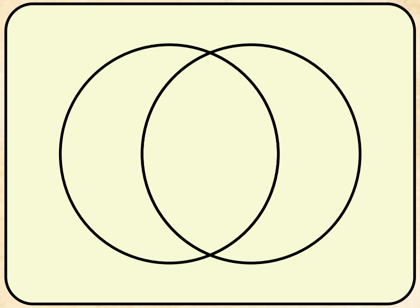
- It also works with sets



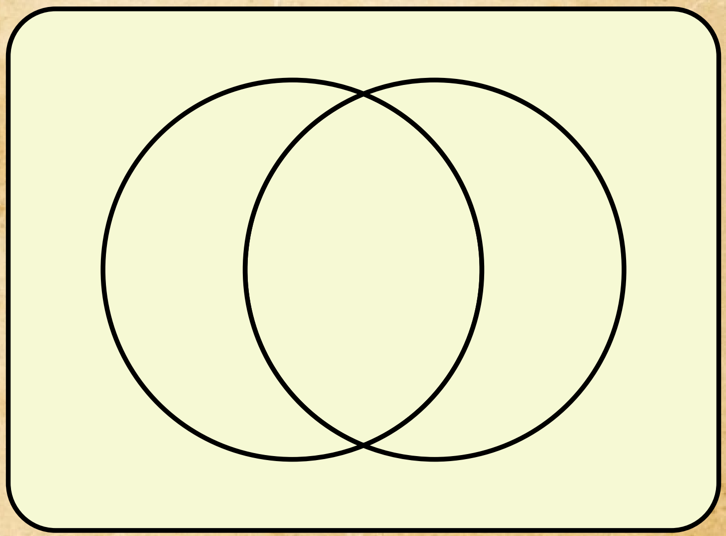
$\boldsymbol{A'}$



$\boldsymbol{B'}$



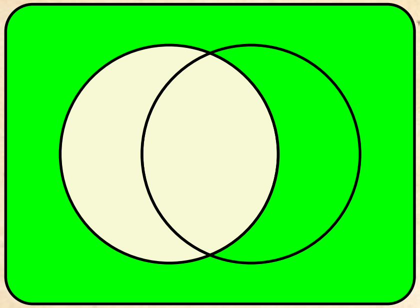
$\boldsymbol{A' \cap B'}$



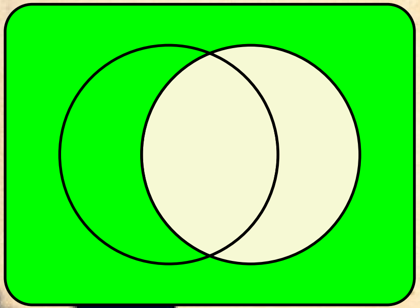
$\boldsymbol{(A' \cap B')'}$

De Morgan’s Laws

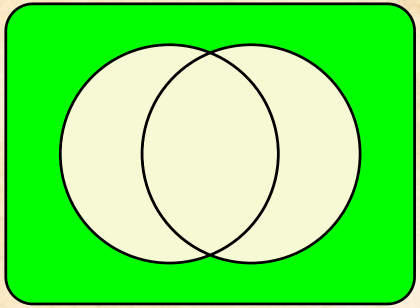
- It also works with sets



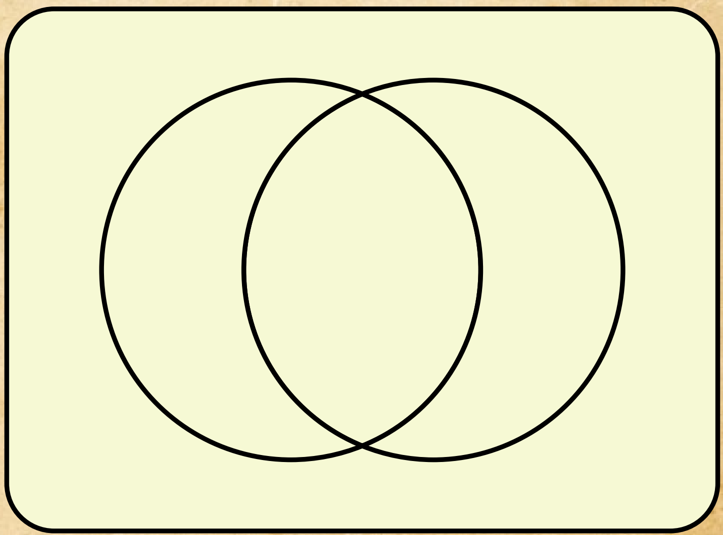
$\boldsymbol{A'}$



$\boldsymbol{B'}$



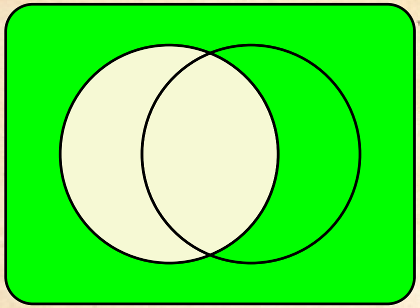
$\boldsymbol{A' \cap B'}$



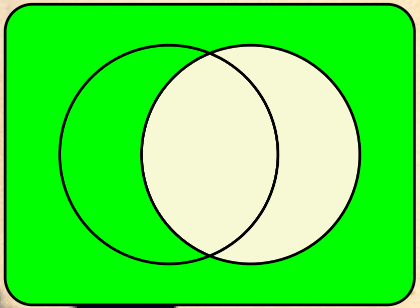
$\boldsymbol{(A' \cap B')'}$

De Morgan’s Laws

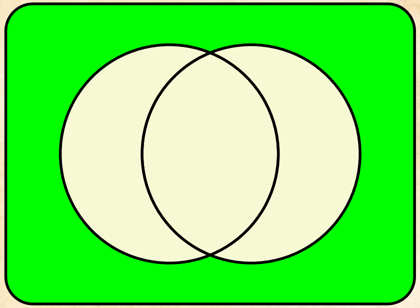
- It also works with sets



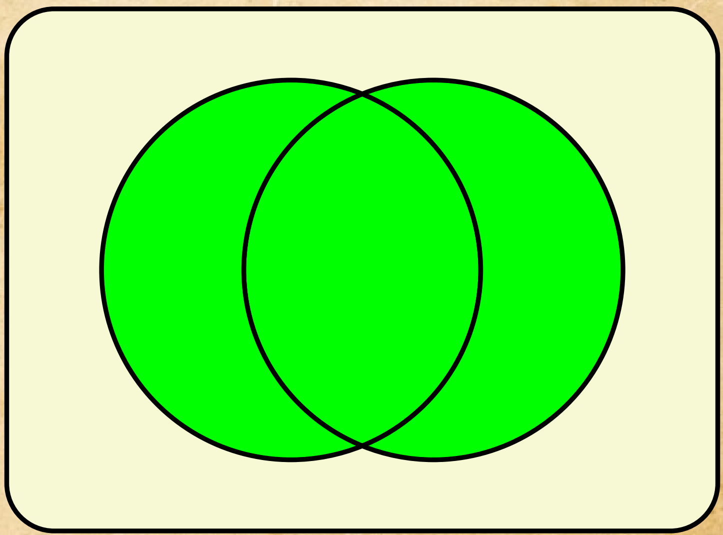
A'



B'



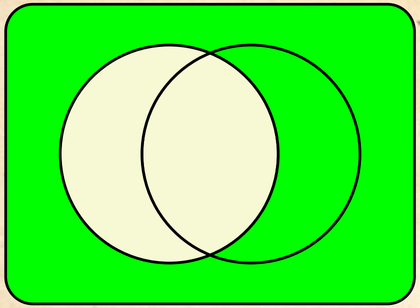
$A' \cap B'$



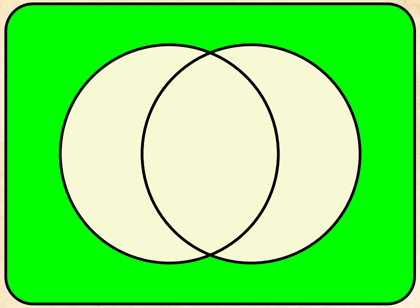
$(A' \cap B')'$

De Morgan's Laws

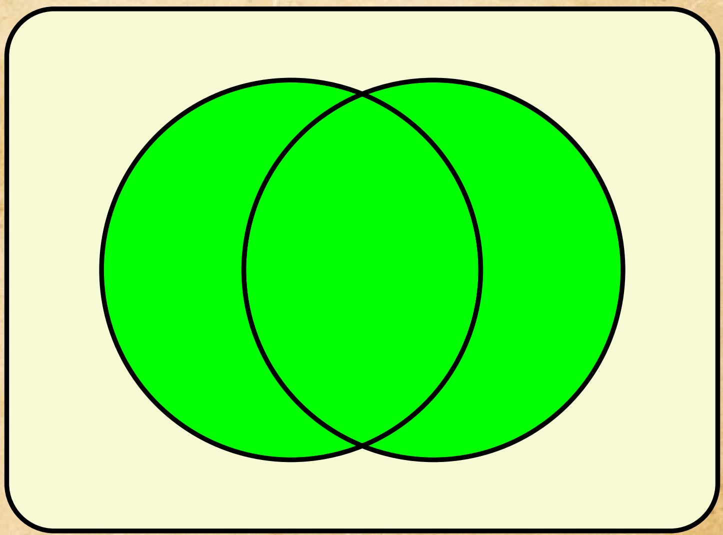
- It also works with sets



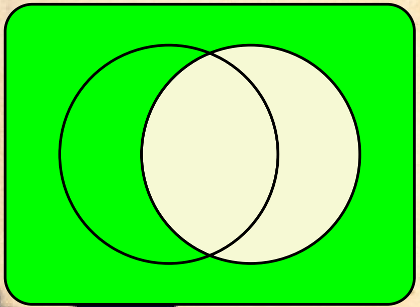
A'



$A' \cap B'$



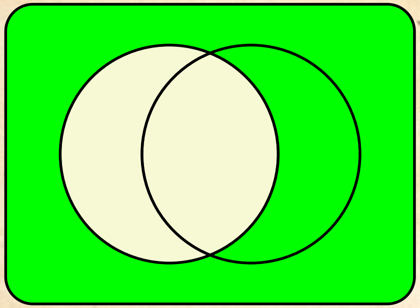
$(A' \cap B')' = A \cup B$



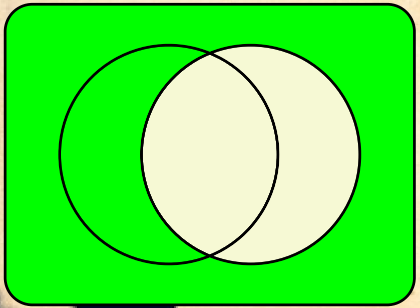
B'

De Morgan’s Laws

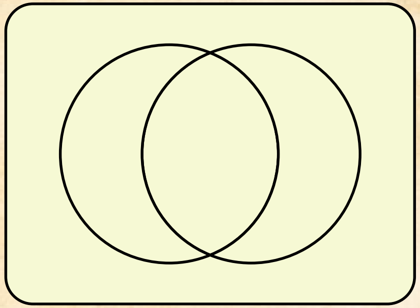
- It also works with sets



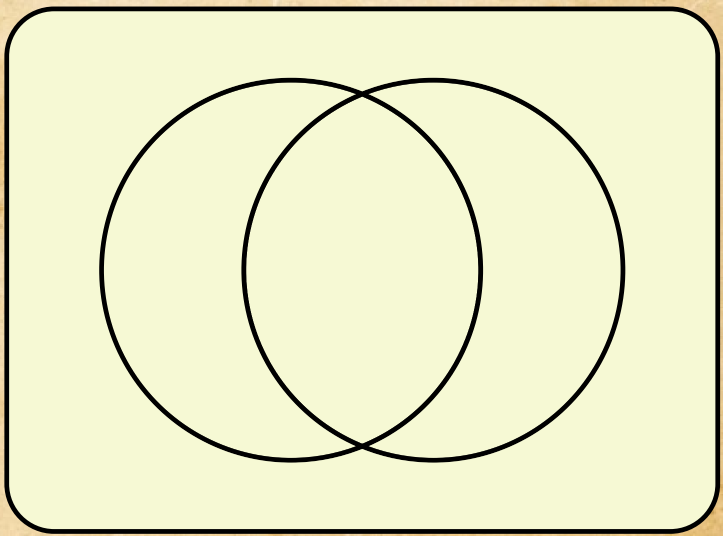
$\boldsymbol{A'}$



$\boldsymbol{B'}$



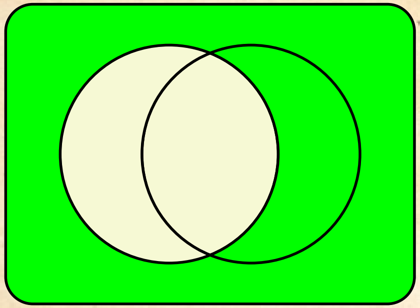
$\boldsymbol{A' \cup B'}$



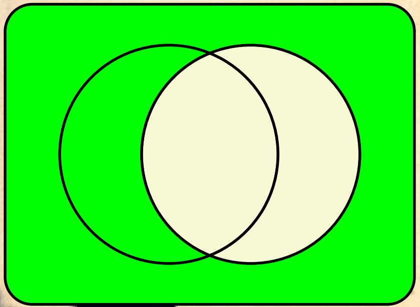
$\boldsymbol{(A' \cup B')'}$

De Morgan’s Laws

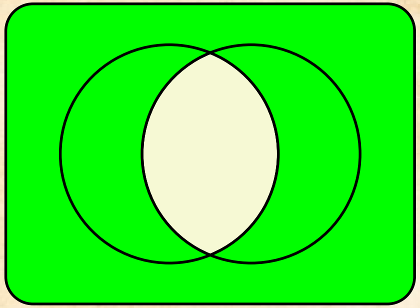
- It also works with sets



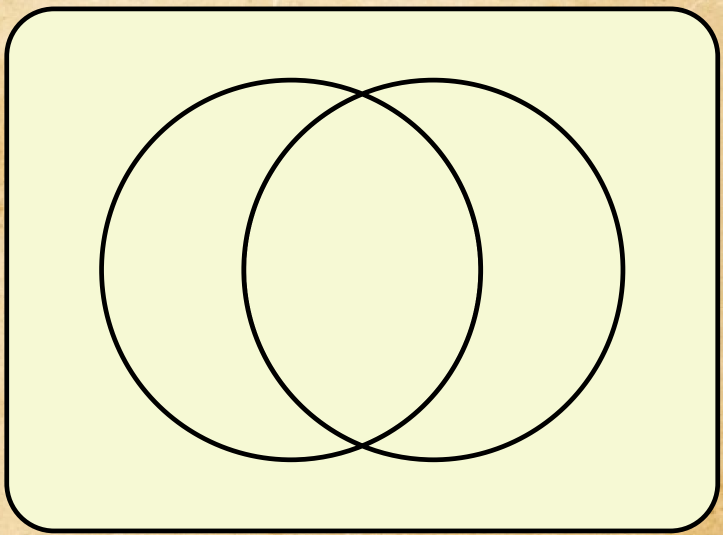
A'



B'



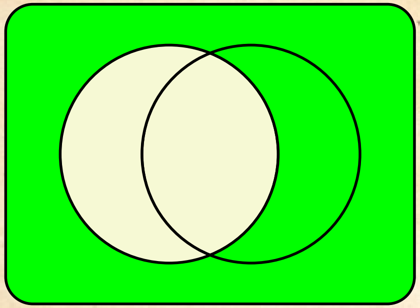
$A' \cup B'$



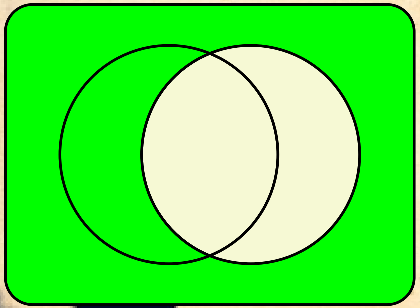
$(A' \cup B')'$

De Morgan’s Laws

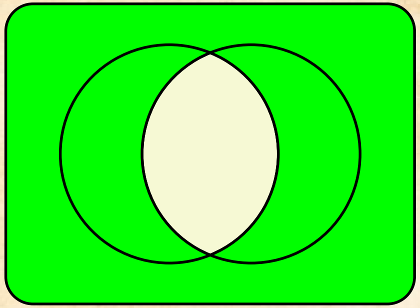
- It also works with sets



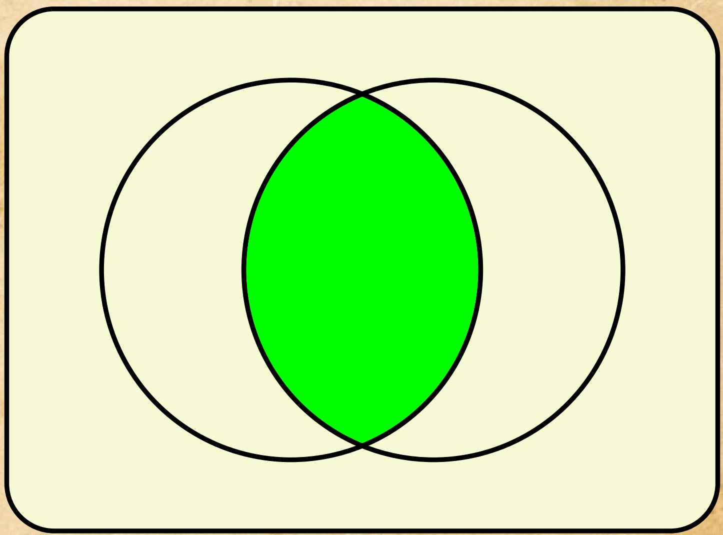
A'



B'



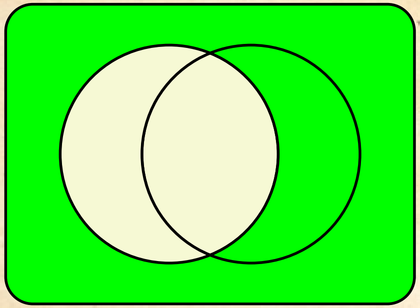
$A' \cup B'$



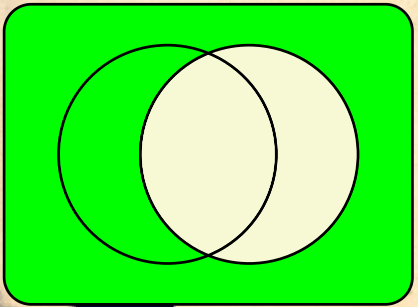
$(A' \cup B')'$

De Morgan’s Laws

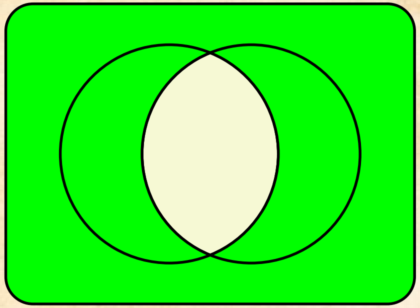
- It also works with sets



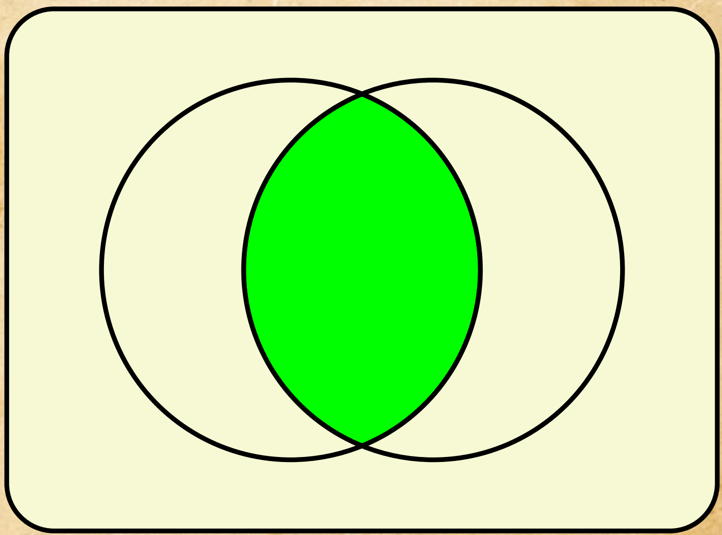
A'



B'



$A' \cup B'$



$(A' \cup B')' = A \cap B$

Symbols

<table>
<tr><th></th><th></th><th></th><th></th><th></th><th colspan="2">Java, C</th></tr>
<tr><th></th><th>Operation</th><th>Sets</th><th>Logic</th><th>Alternate</th><th>Bitwise</th><th>Logical</th></tr>
<tr><td>negation</td><td>NOT</td><td>'</td><td>¬</td><td></td><td>~</td><td>!</td></tr>
<tr><td>conjunction</td><td>AND</td><td>∩</td><td>∧</td><td>·</td><td>&</td><td>&&</td></tr>
<tr><td>inclusive disjunction</td><td>OR</td><td>∪</td><td>∨</td><td>+</td><td>|</td><td>||</td></tr>
<tr><td>exclusive disjunction</td><td>XOR</td><td></td><td>⊻</td><td>⊕</td><td>^</td><td></td></tr>
</table>

Boolean Logic

Introduction