CS1112
Introduction to Logic

Lecturer:

Professor Barry O'Sullivan

Office: 2-65, Western Gateway Building

email: *b.osullivan@cs.ucc.ie*

http://osullivan.ucc.ie/teaching/cs1112/

Introduction to Logic

What is logic?

Why do we need it?

Logic's role in Computer Science

Logic

Logic is the study of rational argument.

- Given a set of facts, and an argument claiming that a conclusion follows from those facts, how do we decide whether or not the argument is justified?

- In order to do this, we need a precise language to state the facts, the conclusions and the steps of the argument.

Which of the following are valid arguments?

If Ireland are playing in Dublin, then they wear green jerseys
Ireland are playing in Dublin
Therefore Ireland are wearing green jerseys

If Ireland are playing in Dublin, then they wear red jerseys with yellow spots
Ireland are playing in Dublin
Therefore Ireland are wearing red jerseys with yellow spots

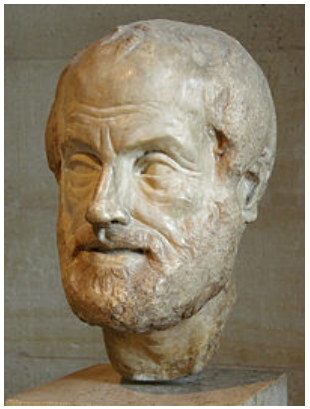
If Ireland are playing in Dublin, then they wear green jerseys
Ireland are wearing green jerseys
Therefore Ireland are playing in Dublin

Everyone born on the island of Ireland is entitled to Irish citizenship
William was born on the island of Ireland
Therefore William is entitled to Irish citizenship

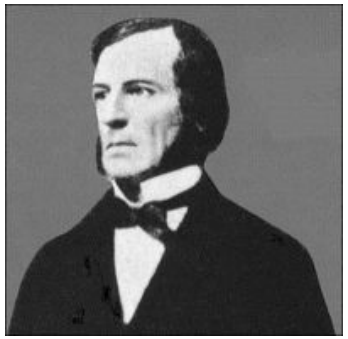
Everyone born on the island of Ireland is entitled to Irish citizenship
Stephen was not born on the island of Ireland
Therefore Stephen is not entitled to Irish citizenship

A very brief history of logic

- studied since (at least) the Ancient Greeks



- Aristotle (c350BC) analysed arguments, and his methods were used for more or less 2000 years



- George Boole, here at University College Cork, in 1854, published "An Investigation of the Laws of Thought ..."
- Boole's work made modern digital computers possible

- Also led to the development of symbolic logic, started by Gottlob Frege in 1879

The Study of *Rational* Argument

- This is not about emotions, slamming doors, hidden subtexts, fairness, charm, persuasion, or shouting the loudest in the pub.

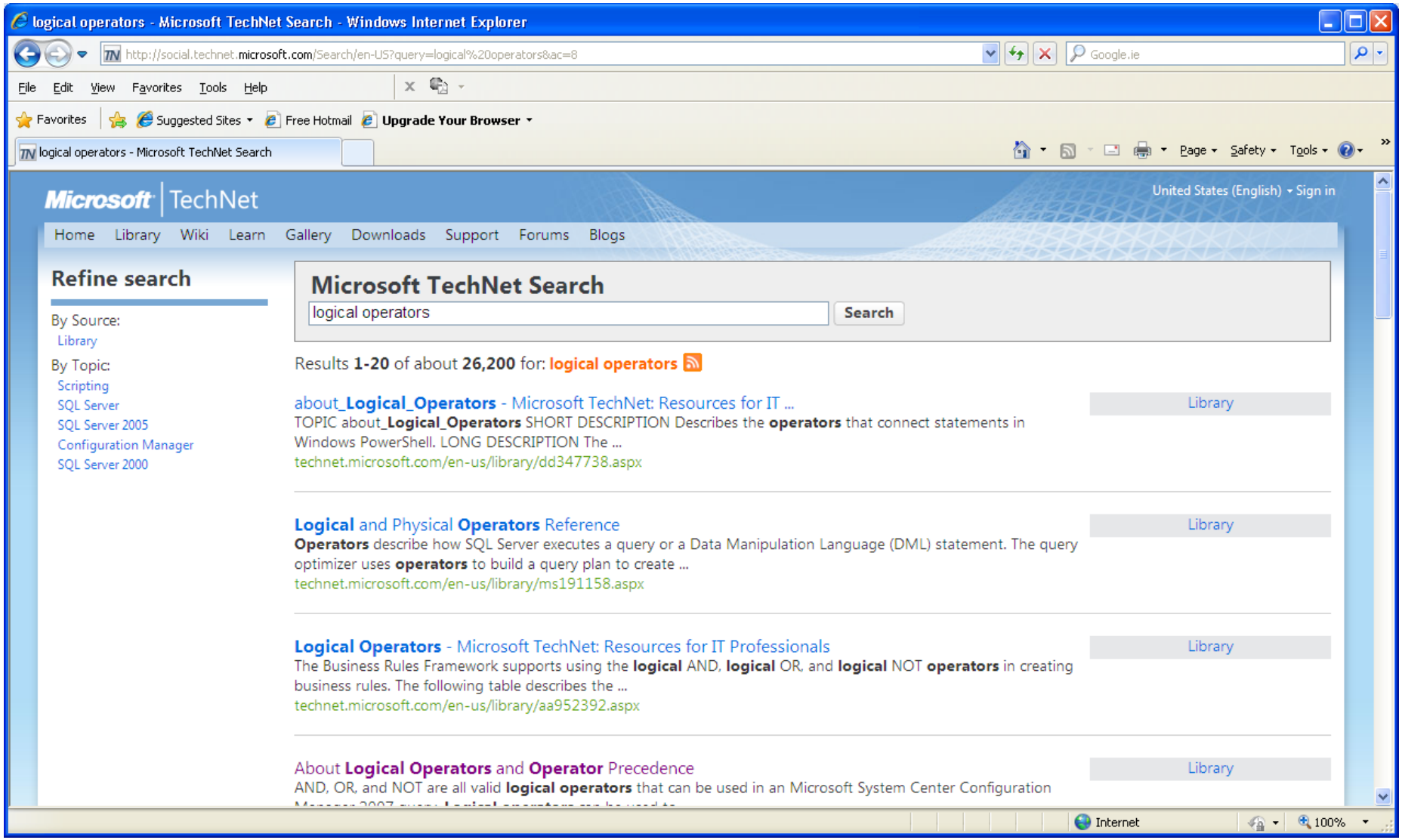
- It is about cold, calm, rational argument.
  - It may help you in normal arguments if you can spot logical flaws, or statements that don't make sense, or ambiguities or mistakes in what is being asked for, but it is not going to tell you when to raise your voice, or when to call in a favour ...

- The aim:
  - strip away the emotions and background knowledge, and understand what is being asked, and what the consequences are
  - learn to state what is required clearly and precisely
  - learn to spot and create valid arguments to justify why a conclusion follows from the initial facts
  - learn to create arguments that justify whether a solution meets its requirements

The role of logic in Computer Science

Logic is fundamental to computer science. We use logic to:

- design circuits, as studied in CS1110
- state inside programs which branch of an "if" statement is to be followed, or whether the body of a "while" loop is to be entered
- specify the requirements for programs
- design routines which test other programs
- prove that a program meets its specification
- prove that a security system stops unauthorised access
- verify that circuit designs do the job they are supposed to do
- check that databases contain consistent information
- process user requests for detailed search in web searches or library catalogues or product catalogues
- check whether orders for computers specify a configuration that will work
- build systems that understand natural language
- build "expert systems" that replicate expert human behaviour
- build artificially intelligent systems


logical operators - Microsoft TechNet Search - Windows Internet Explorer
http://social.technet.microsoft.com/Search/en-US?query=logical%20operators&ac=8
File Edit View Favorites Tools Help
Favorites Suggested Sites Free Hotmail Upgrade Your Browser
logical operators - Microsoft TechNet Search
Page Safety Tools
Microsoft TechNet
United States (English) Sign in
Home Library Wiki Learn Gallery Downloads Support Forums Blogs
Refine search
By Source:
Library
By Topic:
Scripting
SQL Server
SQL Server 2005
Configuration Manager
SQL Server 2000
Microsoft TechNet Search
logical operators
Search
Results 1-20 of about 26,200 for: logical operators
about_Logical_Operators - Microsoft TechNet: Resources for IT ...
TOPIC about_Logical_Operators SHORT DESCRIPTION Describes the operators that connect statements in Windows PowerShell. LONG DESCRIPTION The ...
technet.microsoft.com/en-us/library/dd347738.aspx
Library
Logical and Physical Operators Reference
Operators describe how SQL Server executes a query or a Data Manipulation Language (DML) statement. The query optimizer uses operators to build a query plan to create ...
technet.microsoft.com/en-us/library/ms191158.aspx
Library
Logical Operators - Microsoft TechNet: Resources for IT Professionals
The Business Rules Framework supports using the logical AND, logical OR, and logical NOT operators in creating business rules. The following table describes the ...
technet.microsoft.com/en-us/library/aa952392.aspx
Library
About Logical Operators and Operator Precedence
AND, OR, and NOT are all valid logical operators that can be used in an Microsoft System Center Configuration
Library
Internet
100%

Logic and Human Reasoning

But there is a problem:

- humans, in general, are not good at logic

- we are not good at stripping away our emotions
- we jump to conclusions, based on what we want to happen
- we find it difficult to follow formal arguments

- In computer science, making logical mistakes can be serious, and even fatal
  - analysing security code for bank intranet
  - designing processes for making pharmaceuticals
  - implementing emergency sequences in power stations

The need for logical testing

- think of the specification for a small part of a larger program
  - you must write a method which takes numbers as input, and returns true or false
  - the method must return false if the input is even

- Someone has written the method, and you are now trying to decide whether it is correct, and whether it is safe to incorporate in your larger program. You have to test it. So what do you need to test?

  a) what happens on input of randomly generated even numbers?
  b) what happens on the input of randomly generated odd numbers?
  c) generate some random input, which produces the output true - what was the input?
  d) generate some random input, which produces the output false - what was the input?

Understanding Specifications

You software must ensure at least one of these conditions:
(i)whenever the user is logged out, their balance is hidden
(ii)the balance cannot be visible at the same time as a transaction is being processed

You are offered a solution that guarantees that whenever a transaction is being processed, the system forbids the user being logged out at the same time as their balance is being displayed

Does the solution meet the specification?

So what does CS1112 look at?

First, we consider the logic of simple statements, which are either true or false

propositional logic

- how to write them
- how to build complex statements
- how to determine whether they are true or false
- how to construct proofs that they are true

... using similar methods to those of CS1110

In CS1113 we will move on to more complicated statements which involve variables and collections of objects, and we will do similar things with them.

predicate logic

All of this builds on functions, relations and sets.

Simple statements

An atomic statement, or proposition, is phrase or sentence that declares a fact, and is either true or false.
Questions, commands, and so on are not propositions.

Examples:
1. The lecturer for CS1112 in 2015/16 is O'Sullivan
2. Dublin is the capital of Ireland
3. Enda Kenny played football for Bohemians
4. Homer Simpson lives in Cork
5. 7 + 2 = 9
6. 5 / 2 = 17
7. Mitt Romney is president of the USA

are all propositions. 1, 2, and 5 are true. 3, 4,6 and 7 are false.

Not propositions

8. Are we nearly there yet?
9. Look before you leap.
10. Yield at the junction.
11. Ouch!
12. x = y+2
13. Select Name from Project_Members where ID=109

are not propositions. 8 is a question. 9, 10 and 13 are commands. 11 is an exclamation. 12 might be true or might not, depending on what values we give to x and y, so since it has unbound variables, it is not a proposition.

Utterances of these types are not part of propositional logic.

Truth Values

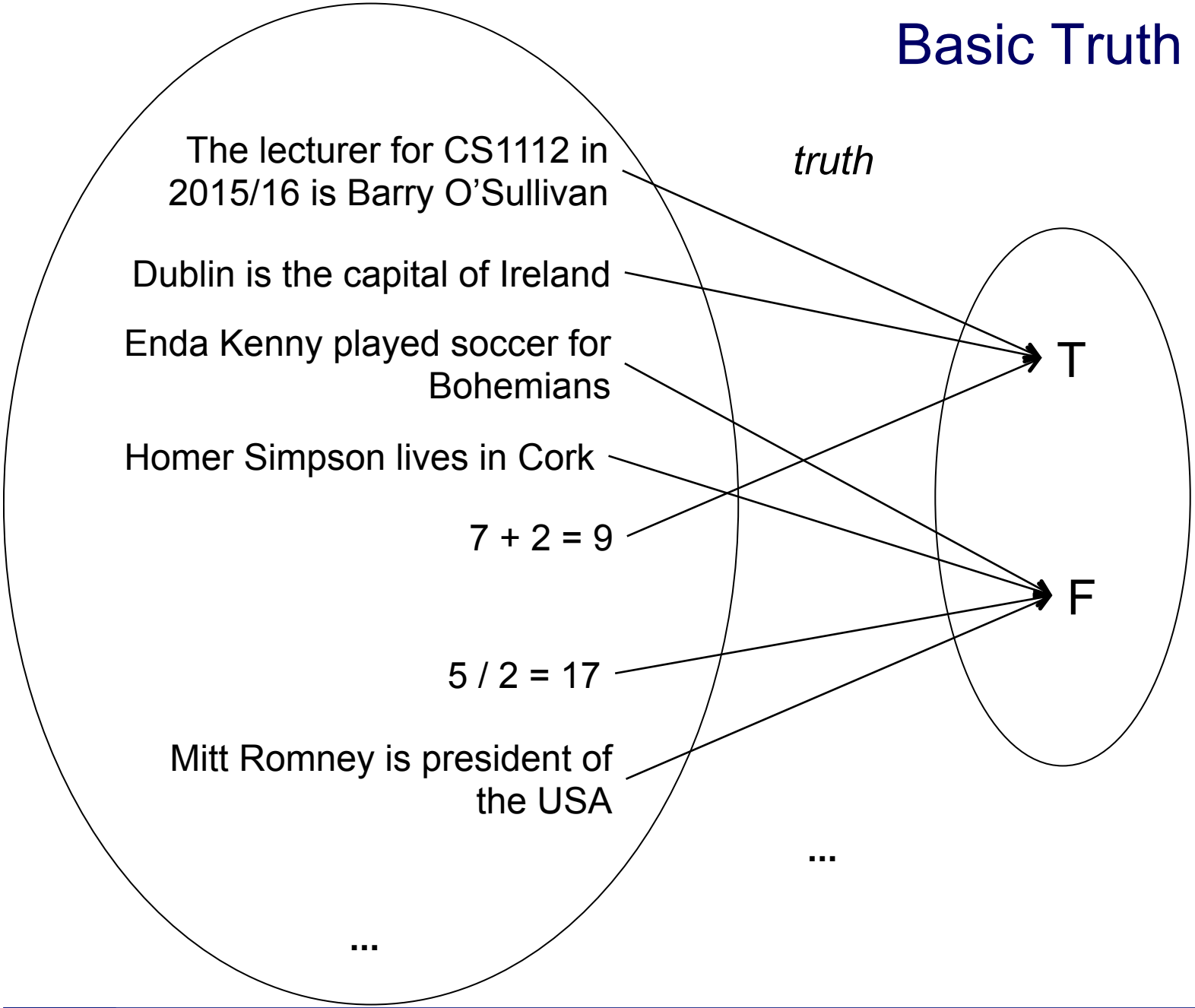
"Dublin is the capital of Ireland" is a true proposition
"Homer Simpson lives in Cork" is a false proposition

True (T) and false (F) are known as the truth values of propositions.

To each proposition, we can assign a unique truth value (and so in some sense there is a function from the set of all propositions to the set {T,F}, called the truth function).

Imagine a big table or arrow diagram which tells us which statements map to true, and which map to false.

Basic Truth Function



Compound Statements

We can build more complicated statements by combining simple statements:

14. The CS1112 lecturer in 2015/16 is O'Sullivan **and** 7+2=9
15. Dublin is the capital of Ireland **or** Enda Kenny played soccer for Bohemians.
16. Dublin is the capital of Ireland **and** Homer Simpson lives in Cork.

The truth value depends on the truth values of the atomic propositions and the connective used to combine them

- 14 is true, 15 is true, and 16 is false

The need for a formal language

In order to work out the truth value for more complicated sentences, we need to define a function which acts on the truth values of the atomic propositions, and returns a truth value for the compound statement.

We need a language to state the function precisely. We will use:

propositional symbols

- p, q, r, ..., p', p'', ... to represent propositions
- T, F to represent the truth values true and false
- $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$ to represent connectives which combine different propositions into a compound statement.

logical connectives

Negation (¬)

The first connective doesn't actually connect two propositions -- instead, we apply it to a single proposition. We call it "not", and we write it as "¬".

The truth function for ¬ simply switches the truth value of the proposition it is applied to.

We can represent this using a table, in which we consider all possible truth values for the starting proposition, and the result of applying the function.

p	$\neg p$
T	F
F	T

Note: compare this to the truth tables you cover in CS1110

Examples of negation

If p represents the proposition "Dublin is the capital of Ireland", we know p takes the value T (i.e. p is a true statement). Therefore $\neg p$ takes the value F (i.e. $\neg p$ is a false statement).

We can interpret $\neg p$ as being "It is not the case that Dublin is the capital of Ireland" or more naturally "Dublin is not the capital of Ireland".

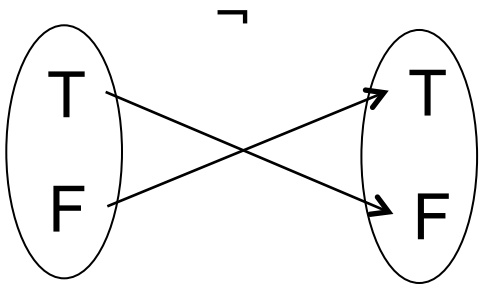
q = "Homer Simpson lives in Cork" has value F
$\neg q$ = "Homer Simpson does not live in Cork" has value T

r = "7 + 2 = 9" has the value T
$\neg r$ = "7 + 2 ≠ 9" has the value F

p	$\neg p$
T	F
F	T

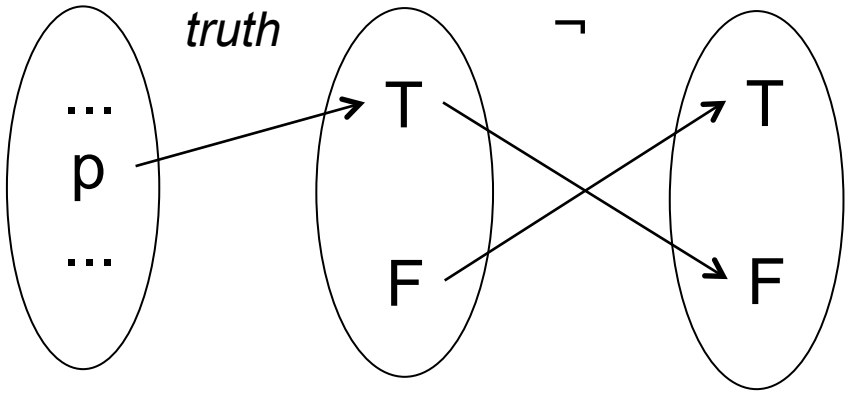
Truth Function for ¬

The function for ¬ is a mapping from {T,F} to {T,F}.



To apply it to propositions, we compose it with the basic truth function. So if the proposition symbol *p* stands for the proposition "Dublin is the capital of Ireland", then *truth(p)* = T, so we have:

¬ ° *truth* (p) = ¬ (T) = F



but this is too cumbersome, so we normally write ¬p

Exercise

Let *p* be the proposition "Paris is in Ireland"
Let *q* be the proposition “Ireland beat Germany last month"
Let *r* be the proposition "Bono is U2's singer"

Express the following using propositional symbols and logical connectives:

It is not the case that Paris is in Ireland
Ireland did not beat Germany last month

Translate the following into English:

$\neg r$

Conjunction (∧)

Conjunction connects two propositions. The conjunction of two propositions is true if both of the smaller propositions are true; otherwise it is false.

For conjunction we use the symbol ^ (or sometimes "&&" or AND) placed between the two propositions we are joining

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

Note: these are all the possible combinations of truth values for p and q

Note: compare this to the truth tables you saw in CS1110

Examples of Conjunction

Let *p* be "Dublin is the capital of Ireland"
Let *q* be "The lecturer for CS1112 in 2015/16 is O'Sullivan"
p has value *T*, *q* has value *T*, and so $p \wedge q$ must have value *T*.

In other words, it is true that "Dublin is the capital of Ireland and the lecturer for CS1112 in 2015/16 is O'Sullivan"
This is the normal interpretation of "and" in English.

p = "Dublin is the capital of Ireland" (*T*)
r = "Homer Simpson lives in Cork" (*F*)
so $p \wedge r$ must have the value *F*

p = "5 / 2 = 17" (*F*)
q = "Homer Simpson lives in Cork" (*F*)
so $p \wedge q$ must have the value *F*

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

Exercise

Let *p* be "Cork is in Munster"
Let *q* be "Roses are red"
Let *r* be "Mice like to eat chili peppers"
Let *s* be "Bono is in Snow Patrol"

Express the following in symbols and connectives:
Cork is in Munster and roses are red

Express the following in English: $q \wedge r$

If we now know that *p* is true, *q* is true, *r* is false, and *s* is false, what is the truth value of the following?

$p \wedge q$
$p \wedge r$
$r \wedge s$

Next lecture ...

Disjunction

conditional statements

bi-conditional