1982

Computer-Based Logic and Argument Analysis: Analytics

Preston K. Covey
Carnegie Mellon University, dtrollcovey@gmail.com

Follow this and additional works at: http://repository.cmu.edu/philosophy

Part of the Philosophy Commons

This Article is brought to you for free and open access by the Dietrich College of Humanities and Social Sciences at Research Showcase @ CMU. It has been accepted for inclusion in Department of Philosophy by an authorized administrator of Research Showcase @ CMU. For more information, please contact research-showcase@andrew.cmu.edu.
COMPUTER-BASED LOGIC AND ARGUMENT ANALYSIS:

ANALYTICS

Submitted to *Teaching Philosophy*, with versions to be submitted to *Computers and the Humanities* and *Educational Technology*.

Preston K. Covey, Jr.

Carnegie-Mellon University

Copyright (C) 1982
I. INTRODUCTION

I have developed a package of computer-assisted instruction programs at Carnegie-Mellon University to teach first-order logic and to explore its applications to the analysis of relatively sophisticated (in our case, philosophic) arguments. The package is called ANALYTICS and presently consists of the following seven programs of three types:

QUESTION-AND-ANSWER INSTRUCTIONAL PROGRAMS IN SENTENTIAL LOGIC


VALID: Seven lessons on the truth-functional proof of validity and intuitive explanations of the basic rules of inference for sentential logic.

DRILL-AND-PRACTICE PROGRAMS IN SYMBOLIZATION AND TRUTH-VALUE ANALYSIS

SYMBOL: Generates symbolization exercises and (when required) answers on selected levels of difficulty with various combinations of sentential connectives. Students may also set the parameters to have the program generate exercises of any complexity of their own choosing. Generates sentence schemas with only the connectives themselves in English. Employs a variety of connectives (like 'however,' 'unless,' 'just in case').

SYMBO2: Otherwise like SYMBOL but generates actual English sentences from a library of atomic sentences created by the instructor.

TRUTH: Generates sentence schemas with English connectives like SYMBOL with truth-value assignments for the sentential variables. Asks the student to analyze the truth-value of a molecular sentence of that form given the assigned truth-values. Leads the student through the truth-functional analysis and provides hints (or answers) if requested.

ARGUMENT-ANALYSIS PROGRAMS WITH PROOF-CHECKER

RECON: Provides English argument texts from libraries of stored arguments for the student to reconstruct in valid symbolic form. Asks the student to assign constants to the atomic sentences of the argument, to identify and symbolize the stated premises and conclusion of the argument, to supply any 'missing' premise(s) needed to make the argument valid, and then to prove the argument valid by deriving the conclusion. The program provides hints in all phases of argument
reconstruction as well as an English translation, on request, of any lines of the argument or derivation so that the student can think about the content as well as the logical (symbolized) form of the argument simultaneously (and, as presented on the screen, side-by-side). The hints also guide the student in the construction of plausible 'missing' premises. So, the program facilitates—and reinforces—the need to play form (the constraint of formal validity) off against content (and non-formal constraints like plausibility) in the analytic-intuitive search for 'missing' or operative but unstated assumptions of an argument. The program's proof-checker monitor's each step of the derivation of an arguments conclusion for well-formedness and validity.

**ARGUE**: Provides English argument texts and word problems for analysis from libraries of stored problems, but unlike **RECON** also handles arguments formalizable in quantificational logic (with identity) and operates in 'free-form' mode in which the student can enter and manipulate his own symbolized argument and English constant assignments. **ARGUE** can then provide an English translation at any point of a derivation, as with stored problems. The program can be used for purely formal derivational work as well as exercise in reconstructing English arguments with their underlying principles or 'hidden' assumptions. Besides providing hints and immediate error messages, the program greatly facilitates the mere manipulation of symbolic formulae in derivational work and provides clear formating in the display of conditional proofs.

We employ these programs in a required freshman core curriculum course to teach first-order formal logic, with applications to argument analysis, in just over half a semester. The class average on standard examinations has improved one grade level since the introduction of the programs to the course; the programs consistently poll as the most valuable element of the course; and the **ARGUE** program in particular receives a very respectable 4.2 average on the student rating scale where 5 is the highest rating for effectiveness. This, in spite of the fact that most of our students are shy of both computers and formalism upon entry to the course.

The programs utilize the notational and natural deduction system found in the most popular elementary logic texts (such as Copi and Kahane) and have their greatest utility in introductory logic courses that stress the formalization of English arguments (as opposed to more mathematically inclined courses or ones favoring a Fitch-style natural deduction system). Thus they would be well applied in the majority of introductory logic courses, especially ones that wish to teach a fair complement of formal logic in minimal time in order to include course segments on informal or inductive logic.

These programs are written in BASIC Plus-2, currently run on our DEC 2060's, and will address any hard-copy terminal as well as Decscope or Perkin-Elmer CRT's. These programs are accessible at Carnegie-Mellon University through EDUNET (and we can make provision for them to address other CRT terminal types on request). A user manual and further information may be obtained by writing Professor Preston K. Covey, Jr., Department of History and Philosophy, Carnegie-Mellon University, Pittsburgh, PA 15213 or by calling (412) 683-3154.

In the table of contents or menu of the **ANALYTICS** package illustrated below, three programs in addition to the seven described above are listed. **EQUIV** and **QUANT** are not presently complete. **BERTIE** is a modified version of the program developed at Dartmouth under that name. Students have access to this program for doing symbolic derivation problems that have no
associated English argument texts. While ARGUE also allows work with purely symbolic derivations, I reserve its use for English argument analysis.
II. THE ANALYTICS PACKAGE

1. ANALYTICS: PROGRAM CONTENTS AND OPERATION

The ANALYTICS program itself is just a table-of-contents or menu program that allows you easy selection among and access to the programs in formal logic and argument analysis designed for the Philosophic Methods and Social Values course, 80-100. 'ANALYTICS' is the name of a package of computer programs, each of which can be run separately and directly, without going through ANALYTICS. The instructions that follow will allow you to run and sample the programs if you have access through EDUNET or TELENET (write or call Preston Covey to make the necessary arrangements).

Each of the programs contained in ANALYTICS can be run by using the following command schema:

    RUN<PC0K>Programname <CR>

The easiest way to remember about the placement of spaces in typing this command sequence is simply to remember to insert no spaces. A run command like the above includes my directory name, PC0K, because the system must be told where/in what directory the program is to be found. In place of 'programname' you would of course type the actual name of the program you wanted to run. (Remember: '<CR>' is the reminder to hit the 'Return' bar upon entering a command or information that you want the system or a program to act upon.)

Until you know your way around the programs and the computer better, you're best advised to always run the ANALYTICS program itself and then choose among the programs contained within it according to the assignments in your 80-100 syllabus. To run the ANALYTICS program and gain access to its several contents, you simply type the following (when prompted by the '@' sign -- what you type is in boldface):

    @Run<PC0K>Analytics <CR>

The results of a successful execution of this run command is depicted on the next page.
The following programs are available:

1 - SENT : Sentential Logic Lessons
2 - SYMBOL : Symbolization Exercises -- Sentential Connectives
3 - SYMB2 : Symbolization Exercises -- English Sentences
4 - TRUTH : Truth-value Analysis Exercises
5 - VALIDITY : Rules of Inference and Valid Argument Forms
6 - EQUIV : Logical Equivalence and Rules of Replacement
7 - QUANT : Quantificational Logic Lessons [OFF LINE]
8 - BERTIE : Proof Checker for Symbolic Logic
9 - RECON : Argument Reconstruction Program
10 - ARGUE : Free-form Argument Construction Program

Which would you care to do?

<1-10 or STOP> ?

At this point, where ANALYTICS is prompting me for a response/choice with its question mark ?, I could either elect to stop and exit ANALYTICS by typing the command 'STOP', or I could select one of the listed programs to run by typing the appropriate number 1, 2, 3, etc...or 10. Let's pick it up from ANALYTICS' question and see what would happen if I elected to stop (What I type is represented in boldface):

Which would you care to do?

<1-10 or STOP> ? STOP

RIGHTO! GOODBYE!

@

Here ANALYTICS says good-bye and returns me to the system's '@' sign. (It also erases any record of what terminal type I might have said I was using. This is why, when first using these programs, you should exit any program via ANALYTICS using the STOP command.)

Often, your options or choices within a program will be presented in abbreviated form between angle brackets < >, like the options to choose programs 1, 2, 3, etc...or 10 or STOP in the example above: <1-10 or STOP>. Your options of this sort will either be explained or clear to commonsense in the context of the programs.
2. INSTRUCTIONAL PROGRAMS: SENT, VALIDITY, EQUIV, QUANT

These programs each consist of instructional material, like that in a textbook, organized into 'frames' of discrete information upon which you are questioned at regular intervals. Unlike a textbook, the programs require you to respond to the material and they respond to you in turn: they require you to be actively responsive and they are interactive. Thus they are presumably more effective media for learning than textbook reading.

At the end of each lesson (which take 10 to 15 minutes to complete) you are given choices regarding what to do next. Follow your syllabus. You are given the option to exit these programs only at the end of each lesson. The instructions needed to negotiate the lessons are contained therein.

In the sample 'tour' provided into the programs below (Section 5), the table of contents or lessons for the SENT and VALIDITY programs are provided. (We will not be using the QUANT or EQUIV programs.)

This brief 'tour' also illustrates how the programs are all looped together in ANALYTICS.

3. THE DRILL-AND-PRACTICE PROGRAMS: SYMBOL, SYMBO2, TRUTH

These programs are subdivided into lesson types of graduated orders of difficulty. You choose the lesson/level assigned in your syllabus and do as many problems as you need to feel mastery of the exercise (five or more in each lesson/level).

These programs generate their own problems (the problems are not stored or 'hard-coded' in the programs, but created by the programs); and the programs know the correct answer in each case and will correct you accordingly. Read the instructions and note the commands available for these programs (Sections 5.2 through 5.4 below -- the instructions are also available within the programs themselves by use of the INS command).

NOTE: Because these programs generate exercises and their answers ad infinitum, they will keep giving you exercises until you tell them to stop by using the STOP command (in response to their prompt 'What next?'), or until you tell them to start another lesson by using the START command (in response to 'What next?).

NOTE: You can exit or stop these programs or go back to the 'start' to choose a new lesson/level only at the points where the programs ask you 'What next?'. Once you have begun a single exercise, you must and should finish it. Be aware that help or hints are available, and that the program will guide you through the exercise or give you the answer if you fail after a couple tries.

4. ARGUMENT ANALYSIS / PROOF-CHECKING PROGRAMS: BERTIE, RECON, ARGUE

These problem-solving guidance programs have their own manuals devoted to them. The ARGUE program is featured in section III below, with basic instructions on its operation and
sample interactions and exercises in argument analysis.

5. A BRIEF TOUR WITH SAMPLE INTERACTIONS:
SENT, SYMBOL, SYMB02, TRUTH, VALIDITY

The following sections contain a guided tour of the programs the students use to learn the truth-functional apparatus in their initial assignments in their core curriculum course. The tour will follow the order in which students are introduced to the programs in their syllabus. It will include a look at the lesson content of each program and, for the drill-and-practice programs, a listing of the instructions and commands.

We will take this tour by running and looping through ANALYTICS, so you can see how all the programs are connected through this menu program. The tour will be continuous, from entry into to exiting from ANALYTICS, but I will highlight the segments within each program with numbered subsections as follows:

5.1 The SENT Program
5.2 The SYMBOL Program
5.3 The SYMB02 Program
5.4 The TRUTH Program
5.5 The VALIDITY Program

We will begin by running ANALYTICS. What you would type in the following interactions will be printed in boldface. What the system or programs print will be in normal typeface. In running ANALYTICS I will first select the SENT program, selection 1 in the listing below.
5.1 THE SENT PROGRAM -- VIA ANALYTICS

@Run<PC0K> ANALYTICS

The following programs are available:

1 - SENT : Sentential Logic Lessons
2 - SYMBOL : Symbolization Exercises -- Sentential Connectives
3 - SYMBO2 : Symbolization Exercises -- English Sentences
4 - TRUTH : Truth-value Analysis Exercises
5 - VALIDITY : Rules of Inference and Valid Argument Forms
6 - EQUIV : Logical Equivalence and Rules of Replacement
7 - QUANT : Quantificational Logic Lessons
8 - BERTIE : Logic Proof Checker
9 - RECON : Argument Reconstruction Program
10 - ARGUE : Free-form Argument Construction Program

Which would you care to do?

<1-10 OR STOP> ? 1

What type of terminal are you on?

1 - Hard copy
2 - Perkin-Elmer 1100
3 - Perkin-Elmer 550
4 - VT52 Decscope

? 1

Do you want an introduction <Y or N> ? N

The following [SENT] tutorials and drills are available:

0 - Introduction
1 - Negation
2 - Review of negation
3 - Conjunction
4 - Shorthand truth-value analysis
5 - Review of disjunction
6 - Disjunction 1: The ambiguity of disjunction
7 - Disjunction 2: The rule of disjunction
8 - Conditional 1: What conditionals mean
9 - Conditional 2: The rule for 'IF'
10 - Contradiction


Which would you enjoy doing?

Enter the number <0 - 10 or STOP> ? STOP

The following [ANALYTICS] programs are available:

1 - SENT : Sentential Logic Lessons
2 - SYMBOL : Symbolization Exercises — Sentential Connectives
3 - SYMBOL2 : Symbolization Exercises — English Sentences
4 - TRUTH : Truth-value Analysis Exercises
5 - VALIDITY : Rules of Inference and Valid Argument Forms
6 - EQUIV : Logical Equivalence and Rules of Replacement
7 - QUANT : Quantificational Logic Lessons
8 - BERTIE : Logic Proof Checker
9 - RECON : Argument Reconstruction Program
10 - ARGUE : Free-form Argument Construction Program

Which would you care to do?

<1-10 OR STOP> ? 2

5.2 THE SYMBOL PROGRAM

[Note: To do the sample exercise shown below you must have done SENT lessons 1, 2 and 4!]

SYMBOL: SYMBOLIZING SENTENTIAL CONNECTIVES

Do you want instructions <Y or N>? Y

These drills are designed to give you practice translating English connectives into symbolic form. You will be given sentence forms with sentential connectives in ordinary English and asked to

1) Identify the logical forms

   NEGATION - N
   CONJUNCTION - CJ
   DISJUNCTION - D
   CONDITIONAL - CD
   BICONDITIONAL - B

2) Type the symbolic forms.

The symbols for the connectives are:
The following commands are available:

'RETURN' - go to next problem when in response to 'What next?'
END or STOP - exit program in response to 'What next?'
INS - get program instructions in response to 'What next?'
START - choose another lesson in response to 'What next?'
HELP - when in response to 'What next?' will give this
list of commands.

Hit RETURN to continue.

The following [SYMBOL] lessons are available:

1 - Conjunction
2 - Disjunction
3 - Conjunction plus Disjunction
4 - Conditional
5 - Conditional plus Conjunction/Disjunction
6 - Biconditional
7 - Mixed Bag: Random Problems
8 - Free Form: Construct Your Own Problems

Enter lesson number or stop <1-8 or STOP> ? 1

OK, let's give it a go.

It is not the case that (not (it is not the case that P, however I) and G).
What form is this sentence? CONJUNCTION

No, try again.

? NEGATION

Right! Good!

SYMBOLIZATION:

? ---P & I & G
Not quite. Try again.

? -(--P & I) & G

Wrong again. It's

-(-(P & I) & G)

What next? START

The following [SYMBOL] lessons are available:

1 - Conjunction
2 - Disjunction
3 - Conjunction plus Disjunction
4 - Conditional
5 - Conditional plus Conjunction/Disjunction
6 - Biconditional
7 - Mixed Bag: Random Problems
8 - Free Form: Construct Your Own Problems

Enter lesson number or stop <1-8 or STOP>? STOP

The following [ANALYTICS] programs are available:

1 - SENT: Sentential Logic Lessons
2 - SYMBOL: Symbolization Exercises -- Sentential Connectives
3 - SYMBO2: Symbolization Exercises -- English Sentences
4 - TRUTH: Truth-value Analysis Exercises
5 - VALIDITY: Rules of Inference and Valid Argument Forms
6 - EQUIV: Logical Equivalence and Rules of Replacement
7 - QUANT: Quantificational Logic Lessons
8 - BERTIE: Logic Proof Checker
9 - RECON: Argument Reconstruction Program
10 - ARGUE: Free-form Argument Construction Program

Which would you care to do?

<1-10 OR STOP>? 3

5.3 THE SYMBO2 PROGRAM

SYMBO2: SYMBOLIZING ENGLISH SENTENCES
These drills are designed to give you practice translating English sentences into symbolic form.

You will be given molecular sentences with sentential variables assigned to each component atomic sentence. You will then be asked to symbolize the sentence using the assigned sentential variables and our conventional symbols for the sentential connectives.

The symbols for the connectives are:

- V - DISJUNCTION
- & - CONJUNCTION
- => - CONDITIONAL
- <=> - BICONDITIONAL
- - NEGATION

Hit RETURN to continue.

The following commands are available:

'RETURN' - go to next problem when in response to 'What next?'
END or STOP - exit program in response to 'What next?'
INS - get program instructions in response to 'What next?'
START - choose another lesson in response to 'What next?'
HELP - when in response to 'What next?' will give this list of commands.

Hit RETURN to continue.

The following [SYMBO2] lessons are available:

1 - Conjunction
2 - Disjunction
3 - Conjunction plus Disjunction
4 - Conditional
5 - Conditional plus Conjunction/Disjunction
6 - Biconditional
7 - Mixed Bag: Random Problems
8 - Free Form: Construct Your Own Problems

Enter lesson number or stop <1-8 or STOP> ? 1

OK, let's give it a go.
G - the life of a cancerous cell is sacred
H - living things automatically possess a right to life
I - persons have a special right to life
J - people have rights

It's not the case that although the life of a cancerous cell is not sacred, living things do not automatically possess a right to life; and people don't have rights.

SYMBOLIZATION:
? -G & H & -J

Not quite. Try again.
? -(G & H) & J

Wrong again. It's

-(G & -H) & -J

What next? <CR>

G - human fetuses are persons
H - people have rights
I - all forms of life are deserving of rights
J - life per se is sacred

Life per se is sacred and all forms of life are deserving of rights.

SYMBOLIZATION:
? START

Not quite. Try again.
? STOP

Wrong again. It's

J & I

What next? START

The following [SYMO2] lessons are available:

1 - Conjunction
2 - Disjunction
3 - Conjunction plus Disjunction
4 - Conditional
5 - Conditional plus Conjunction/Disjunction
6 - Biconditional
7 - Mixed Bag: Random Problems
8 - Free Form: Construct Your Own Problems

Enter lesson number or stop <1-8 or STOP> ? STOP

The following [ANALYTICS] programs are available:

1 - SENT : Sentential Logic Lessons
2 - SYMBOL : Symbolization Exercises -- Sentential Connectives
3 - SYMB02 : Symbolization Exercises -- English Sentences
4 - TRUTH : Truth-value Analysis Exercises
5 - VALIDITY : Rules of Inference and Valid Argument Forms
6 - EQUIV : Logical Equivalence and Rules of Replacement
7 - QUANT : Quantificational Logic Lessons
8 - BERTIE : Logic Proof Checker
9 - RECON : Argument Reconstruction Program
10 - ARGUE : Free-form Argument Construction Program

Which would you care to do?

<1-10 OR STOP> ? 4

5.4 THE TRUTH PROGRAM

[NOTE: The sample exercise below requires having done SENT lessons 1, 2 and 4!]

TRUTH: TRUTH-FUNCTIONAL ANALYSIS OF CONNECTIVES

Do you want instructions <Y or N> ? Y

These drills are designed to give you practice translating English connectives into symbolic form and in determining the truth-value of sentences according to the truth-functional rules governing the sentential connectives. You will be given a list of sentential variables and their truth-values. You will then be given sentence forms with sentential connectives in ordinary English. Based on the assigned truth-values and the logical form of the sentence, you will be asked to determine the truth-value of the sentence.

If you fail to give the correct answer, you will be asked to
work through the 'short-hand' truth table for the sentence.
If you give the correct answer, the truth table exercise is optional.

If you need a hint or aid when the program asks you a question, like '<T or F>?' or 'What next?' type HELP.

Hit RETURN to continue.

The following commands are available:

'RETURN' - go to next problem when in response to 'What next?'
END or STOP - exit program in response to 'What next?'
INS - get program instructions in response to 'What next?'
START - choose another lesson in response to 'What next?'
TT - get truth table of the connectives in response to 'What next?'
HELP - when in response to 'What next?' will give this list of commands.
- when in response to the question '<T or F>?' will get either the SYMBOLIC TRANSLATION or TRUTH-TABLE RULES.

Hit RETURN to continue.

The following [TRUTH] lessons are available:

1 - Conjunction
2 - Disjunction
3 - Conjunction plus Disjunction
4 - Conditional
5 - Conditional plus Conjunction/Disjunction
6 - Biconditional
7 - Mixed Bag: Random Problems
8 - Free Form: Construct Your Own Problems

Enter lesson number or stop <1-8 or STOP> ? 1

OK, let's give it a go.

Variables and values:

\begin{align*}
\text{G} & \quad \text{F} \\
\text{H} & \quad \text{F} \\
\text{I} & \quad \text{T} \\
\text{J} & \quad \text{T}
\end{align*}
J but (G, nevertheless J).

T or F ? HELP

Symbolic form or truth tables <S or T> ? T

<table>
<thead>
<tr>
<th>P</th>
<th>Q</th>
<th>¬P</th>
<th>P&amp;Q</th>
<th>PVQ</th>
<th>P→Q</th>
<th>P←Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>

Hit RETURN to continue.

T or F ? HELP

Symbolic form or truth tables <S or T> ? S

J & (G & J)

Hit RETURN to continue.

T or F ? F

Correct! Or lucky!

Do you want to work through truth table <Y or N> ? Y

J & (G & J)

T  F  T

? F

Do you want to see the truth-table <Y or N> ? Y

J & (G & J)

T  F  T

F

Table is complete and correct.

What next ? START

The following [TRUTH] lessons are available:
1 - Conjunction
2 - Disjunction
3 - Conjunction plus Disjunction
4 - Conditional
5 - Conditional plus Conjunction/Disjunction
6 - Biconditional
7 - Mixed Bag: Random Problems
8 - Free Form: Construct Your Own Problems

Enter lesson number or stop 〈1-8 or STOP〉 ? STOP

The following [ANALYTICS] programs are available:

1 - SENT : Sentential Logic Lessons
2 - SYMBOL : Symbolization Exercises — Sentential Connectives
3 - SYMBO2 : Symbolization Exercises — English Sentences
4 - TRUTH : Truth-value Analysis Exercises
5 - VALIDITY : Rules of Inference and Valid Argument Forms
6 - EQUIV : Logical Equivalence and Rules of Replacement
7 - QUANT : Quantificational Logic Lessons
8 - BERTIE : Logic Proof Checker
9 - RECON : Argument Reconstruction Program
10 - ARGUE : Free-form Argument Construction Program

Which would you care to do?

〈1-10 OR STOP〉 ? 5

5.5 THE VALIDITY PROGRAM

VALIDITY: Rules of Inference and Valid Argument Forms

Do you want an introduction to this lesson set?
〈Y or N〉 ? N

The following lessons explaining the validity of inference rules are available:

0 - Introduction
1 - The CONJUNCTION rule
2 - The SIMPLIFICATION rule
3 - The ADDITION rule
4 - The DISJUNCTIVE SYLLOGISM rule — Validity Test Explained
5 - The MODUS PONENS and MODUS TOLLENS rules — CONDITIONALS Explained
6 - The CONSTRUCTIVE DILEMMA rule
7 - The HYPOTHETICAL SYLLOGISM rule

Type the number of the rule you would like to prove valid and have explained.

<0 - 7 or STOP> ? STOP

The following [ANALYTICS] programs are available:

1 - SENT : Sentential Logic Lessons
2 - SYMBOL : Symbolization Exercises -- Sentential Connectives
3 - SYMBO2 : Symbolization Exercises -- English Sentences
4 - TRUTH : Truth-value Analysis Exercises
5 - VALIDITY : Rules of Inference and Valid Argument Forms
6 - EQUIV : Logical Equivalence and Rules of Replacement
7 - QUANT : Quantificational Logic Lessons
8 - BERTIE : Logic Proof Checker
9 - RECON : Argument Reconstruction Program
10 - ARGUE : Free-form Argument Construction Program

Which would you care to do?.
<1-10 OR STOP> ? STOP

RIGHTO! GOOD-BYE!
III. A PROOF-CHECKING PROGRAM: ARGUE

1. ARGUE'S FUNCTIONS

ARGUE is designed to give you practice in constructing derivations in symbolic notation. Practice in doing derivations with ARGUE will exercise your sense of logical form and reinforce your judgments about validity within the very precise constraints of symbolic logic. ARGUE will check each line of any derivation you construct for the following crucial logical properties:

1. Well-Formedness: ARGUE will check any symbolic formula you enter in a derivation to see that it is a well-formed formula (WFF). It will give you an error message when any formula is not well formed.

2. Validity: ARGUE will check any line you enter in a derivation to see that it is a valid move according to our given rules of inference and replacement. (See Appendix II for summary of given rules.) When you enter any line in a derivation, you must justify it by citing the previous line(s) of the derivation and/or the rule that allows you to derive or enter the new line. ARGUE will check your justification to see:

   Whether you have cited correct or sufficient line numbers to justify the new line.

   Whether you have cited the correct rule to justify the new line.

If you fail to justify any line in a derivation correctly, or if the line you enter is not allowed or valid by the cited rule, ARGUE will give you an error message. ARGUE thus prompts you to detect and correct your errors immediately. ARGUE allows you three different options for practicing derivations:

1. 'Request a Problem': You can work on stored problems from ARGUE's problem sets. These problems are supplied with stored hints and solutions that you can see by using the HINT, HELP, and UNCLE commands. (For a description of ARGUE's problem sets, see Appendix I. For a description of ARGUE's various commands, see Section 3 below. See Section 2.1 below for illustration of how to work with ARGUE's stored problems.) When you have successfully derived the conclusion assigned in a given problem, ARGUE will congratulate you (as illustrated in Section 2.1 below).

2. 'Enter a Conclusion to Derive': You can enter your own conclusion (in symbolic notation) to derive, and thus create your own derivation problems to work on. When you have successfully derived your stated conclusion in valid fashion, ARGUE will congratulate you (as illustrated in Section 2.2 below).

3. 'Or Type 'Begin': You can simply begin a derivation without specifying any target conclusion that you want to derive. ARGUE will allow you to proceed as you will, checking each step for validity. In this 'free-form' mode ARGUE doesn't know what in particular you want to derive, because you have not specified any particular
conclusion; so it will not congratulate you at any point in your derivation, but will simply prompt you for successive steps in your derivation until you request a stored problem, ask to start another line of derivation or stop. Section 2.3 below illustrates this 'free-form' option in ARGUE.

These three options are best illustrated with some sample interactions. Let's take a look at what it's like to run and work with ARGUE.
2. RUNNING THE ARGUE PROGRAM: OPTIONS & KEY COMMANDS

You can run the ARGUE program either by running the ANALYTICS menu program and choosing option 10 [ARGUE], or by running ARGUE directly, as follows. When you have the system's attention (the @ sign), what you type is represented in boldface below. The program's response is in normal typeface ( <CR> indicates the required carriage return):

@Run<PC0K>ARGUE <CR>

IF YOU NEED INSTRUCTIONS, TYPE 'INS'; OTHERWISE

ENTER A CONCLUSION TO DERIVE, REQUEST A PROBLEM, OR TYPE 'BEGIN'

? INS <CR>

ARGUE IS A COMPUTER PROGRAM THAT CHECKS DERIVATIONS IN SENTENTIAL AND QUANTIFICATIONAL LOGIC.

ARGUE ACCEPTS THE FOLLOWING COMMANDS:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET N</td>
<td>OPENS PROBLEM SET N</td>
</tr>
<tr>
<td>PROBLEM N</td>
<td>ASKS ARGUE TO GIVE YOU PROBLEM NUMBER N</td>
</tr>
<tr>
<td>HINT</td>
<td>REQUESTS A HINT ABOUT SOLVING THE PROBLEM</td>
</tr>
<tr>
<td>HELP N</td>
<td>PRINTS N (MORE) LINES OF ARGUE'S STORED SOLUTION</td>
</tr>
<tr>
<td>UNCLE</td>
<td>GIVE UP? ARGUE PRINTS A SOLUTION</td>
</tr>
<tr>
<td>NEW</td>
<td>ALLOWS YOU TO BEGIN ANOTHER PROBLEM OR DERIVATION</td>
</tr>
<tr>
<td>LIST</td>
<td>LISTS YOUR DERIVATION (TO GET A CLEAN COPY)</td>
</tr>
<tr>
<td>LIST M-N</td>
<td>LISTS LINES M THROUGH N OF YOUR DERIVATION</td>
</tr>
<tr>
<td>DELETE N</td>
<td>ERASES YOUR DERIVATION FROM LINE N TO THE LAST LINE</td>
</tr>
<tr>
<td>REMOVE N</td>
<td>ERASES LINE N ONLY (DERIVATION IS RENUMBERED; USE LIST)</td>
</tr>
<tr>
<td>RULES</td>
<td>PRINTS RULES OF INFERENCE AND REPLACEMENT</td>
</tr>
<tr>
<td>STOP / BYE</td>
<td>STOPS THE PROGRAM (EXITS THROUGH ANALYTICS)</td>
</tr>
<tr>
<td>INS</td>
<td>PRINTS THIS LIST OF COMMANDS</td>
</tr>
<tr>
<td>/</td>
<td>TYPE THE SLASH '/' TO JUSTIFY EACH LINE IN YOUR DERIVATION FOLLOWED BY THE RULE AND LINE NUMBERS APPEALED TO</td>
</tr>
</tbody>
</table>

ENTER A CONCLUSION TO DERIVE, REQUEST A PROBLEM, OR TYPE 'BEGIN'

?

The interaction above illustrates (1) the ARGUE program's opening response, (2) the program's prompt, the question mark '?', in response to which you would type a command (or a line in a derivation), (3) the commands available in ARGUE (printed in response to the INS command), and (4) the three options allowed you for working in ARGUE: 'Enter a conclusion to derive, request a problem, or type 'Begin'. I will illustrate each of these options in the three sections that follow, and highlight the commands you should be familiar with at each stage. A description of the commands will be found in Section 3 below. I will begin with the middle
option, requesting a problem, since it is the one you are most apt to elect in your initial work with ARGUE.

2.1 '... REQUEST A PROBLEM . . .' 

When the ARGUE program prompts you to select one of its three options for practicing derivations and you wish to work on some stored problem(s) (which are assigned in the course syllabus and described in Appendix I), you need first to tell ARGUE which of its problem sets to open by using the SET command.

2.1.1 Open a Problem Set: The SET and PROblem Commands

In order to work any problem you must first ask ARGUE to open a problem set. The available problem sets are identified by number (1, 2, 3 . . . 5). (See Appendix I for a description of available problem sets. These are assigned in your syllabus, but may be reviewed and reworked any time you want.)

You request a problem set \( N \) in response to ARGUE's prompt, \(?\), by typing the command \( \text{SET} \ N \), where in place of the \( N \) you would type the number of the set you want. To get ARGUE to open problem set 1, you would type, in response to the prompt \(?\) : \( \text{SET} \ 1 \). (You may type any commands or other input to ARGUE in either lower or upper case, in any combination.)

ARGUE will tell you how many problems are contained in the requested set and then again give you its three basic options: 'Enter a conclusion to derive, request a problem,' or type 'Begin'. After opening a given set \( N \), you may pursue any of these options and whenever you request a problem by number, say, problem \( M \), ARGUE will give you problem \( M \) from set \( N \). Whenever you request a problem, ARGUE will assume that you want a problem from the problem set last opened. (You can open a different problem set by using the set command at any time that ARGUE has prompted you with the question mark \(?\), even in the middle of doing a given problem.)

To request a given problem, once you have opened a problem set, you use the Problem command. Notice (in the illustration that follows) that the Problem command can be abbreviated to Pro. For example, to request the first problem in a set you could type, in response to ARGUE's prompt \(?\) : either Problem 1, or Prob 1, or Pro 1. In general, ARGUE's commands (and rule names) can be abbreviated to their first three or four letters.

In the illustrative interaction that follows, I will run ARGUE, request a problem (without first opening a problem set, to show you how ARGUE responds), then open a problem set, request a problem, and (in the middle of the problem) request that ARGUE open a different problem set and give me a problem from that set -- all to show you how easily you can flip in and out of different problem sets with the SET command. (Again, what I type -- or what you would type -- is represented in boldface. ARGUE's responses are in normal typeface. Commentary which is not part of the input/output will be between square brackets [ ... ]. I will often dispense with the \(<\text{CR}>\) to remind you to hit the Return bar after typing your input in the illustrations provided below.)
IF YOU NEED INSTRUCTIONS, TYPE 'INS'; OTHERWISE

ENTER A CONCLUSION TO DERIVE, REQUEST A PROBLEM, OR TYPE 'BEGIN'

? Pro 4 <CR>

YOU MUST FIRST OPEN A PROBLEM SET WITH THE 'SET' COMMAND

ENTER A CONCLUSION TO DERIVE, REQUEST A PROBLEM, OR TYPE 'BEGIN'

? Set 2

THERE ARE 13 PROBLEMS IN THIS SET

ENTER A CONCLUSION TO DERIVE, REQUEST A PROBLEM, OR TYPE 'BEGIN'

? Pro 4

DERIVE: -G

1  F => -G
2  -L => -F
3  G => F
4  -L
5  ? Set 3 <CR>

[Notice that after setting up the problem by giving me a conclusion to
derive and four premises from which to derive it, ARGUE then prompts
me for input with its question mark '?' at line 5.

At this point I have elected not to do the problem but rather to open
Problem Set 3. ARGUE then responds as follows:]

THERE ARE 15 PROBLEMS IN THIS SET

ENTER A CONCLUSION TO DERIVE, REQUEST A PROBLEM, OR TYPE 'BEGIN'

? Pro 2 <CR>

DERIVE: -(L & K)

1  -L V -K
2  ? Set 1 <CR>

THERE ARE 26 PROBLEMS IN THIS SET

ENTER A CONCLUSION TO DERIVE, REQUEST A PROBLEM, OR TYPE 'BEGIN'
2.1.2 Working a Problem: Key Commands

The illustration above shows you that you can request a new problem or problem set at any point while working a given problem where ARGUE has prompted you for input with the '?'.

The illustrations below show you the effects of certain key commands in the course of actually working through a problem.

The Slash Command '/': Justifying the Lines of a Derivation

Whenever you enter a symbolic formula on a line of a derivation, you must give a justification for that line by citing the rule that warrants the line and the numbers of any previous line(s) to which you appeal.

You indicate to ARGUE that you are going to give a justification for a line of derivation that you have just entered by typing the slash '/' after entering your formula followed by the appropriate rule and line number(s). For example, given the problem below, you could proceed as follows (where what you type is in boldface):

DERIVE: P & R

1 P & Q / PREMISE
2 S & R / PREMISE
3 ? P / SIMPL 1
4 ? R / 2 SIMPR
5 ? P&R/3 4CONJ

CONGRATULATIONS! YOUR DERIVATION IS COMPLETE.

LIST YOUR DERIVATION, ASK FOR A PROBLEM, OR TYPE 'NEW'.

When you have succeeded in deriving the assigned conclusion, ARGUE will congratulate you (as above) and give you the options of getting a clean listing of your derivation, requesting another problem, or starting anew. Typing the NEW command at this juncture, or at any point
where ARGUE prompts you with its query '?' will simply get you ARGUE's original three basic options: Enter a conclusion to derive, request a problem, or type 'Begin'. More on LISTing your derivation later. For now you should take note of the following:

1. Whenever you enter a formula, like P, on a line of derivation, like line 3 above, you must also enter a justification preceded by the slash / on that line.

2. The justification of a line of derivation, following the slash /, consists of a rule citation and, where previous lines are appealed to, line numbers.

3. Some rules, like the PREMISE introduction rule, do not require the citation of previous lines because they allow the introduction of lines that do not depend logically on previous lines. Notice that in introducing the two premises in the problem above, ARGUE cited only the PREMISE rule preceded by the slash /.

4. You may cite the rule either before the line numbers or vice versa (compare lines 3 and 4 above) -- the order does not matter.

5. You need not put any spaces between any of the entries in a line of derivation except between line numbers (see line 5 above) . . .

6. You must put either a space or a comma between line numbers when more than one line number is cited in a justification: For example, on line 5 above, had we typed 34 instead of 3 4, ARGUE would have thought we were referring to line 34 rather than lines 3 and 4 (and would have given us an error message to the effect that the line number was too large and too few lines were appealed to for the CONJunction rule).

7. It does not matter how many spaces you put between entries in a line of derivation, so long as you put no spaces in the name or abbreviation of the cited rule.

8. You may use abbreviations for the rules of inference and replacement. (The first three or four letters of the rule name or the initials of the rule will do: legal abbreviations are given on your rule summaries and in ARGUE's rule file, which you can have ARGUE print out by typing RULES in response to the query '?'.)

9. You may type your input to ARGUE in any combination of upper and lower case.

10. You must put the slash / between the formula you enter as a line of derivation and its justification -- unless you use the slash command to have ARGUE type the line for you (see the next section below).

The Slash Command ' / ': Commanding ARGUE to Type a Line of Derivation

It is often possible to get ARGUE to type a desired line of a derivation for you (and thereby save yourself both effort and the risk of making a typographical error) by using the slash / as
a command. When prompted by ARGUE's query ? to enter a line in a derivation, you might type the slash / plus the rule you want to apply plus the line numbers you want the rule applied to. Then hit the carriage return. This will have the effect of telling ARGUE: 'Give me what I can get by the cited rule from the cited lines.' If, and only if, there is a valid consequence that can be directly derived (in one step) from the cited lines by the rule you cite, then ARGUE will print out that result for you with its justification. If there is no valid consequence that can be obtained by the cited rule from the cited lines in one step, ARGUE will of course give you an appropriate error message to that effect. Were we to use the slash command in this way on the problem in the previous illustration, the results would look like this (where what we type is printed in boldface):

DERIVE: P & R

1 P & Q / PREMISE
2 S & R / PREMISE

3 ? / 1 SIMPL
3 P / 1 SIMPL

4 ? / 2 SIMPR
4 R / 2 SIMPR

5 ? / 3 4 CONJ
5 P & R / 3 4 CONJ

CONGRATULATIONS! YOUR DERIVATION IS COMPLETE

ENTER A CONCLUSION TO DERIVE, REQUEST A PROBLEM, OR TYPE 'BEGIN'

When the formulae that you want to derive are longer and more complex than the above, this feature of ARGUE's slash command is especially helpful. In using this convenience, however, you should note carefully the following:

1. The slash command cannot be used with all rules to get ARGUE to print the line you want. It can only be used when the result of applying the cited rule to the cited lines would be unique or unambiguous. For example, were you to type / PREMISE in response to ARGUE's query ?, ARGUE could not print out a premise for you because it could not possibly know which of the infinite possible premises you might want.

2. In certain cases where the application of a rule to certain lines might allow more than one valid result, the ambiguity can be resolved by specifying an appropriate rule
variation. Certain rules (like SIMPlification) that warrant some basic logical move (like deriving one conjunct from a conjunction) but that could have more than one result (like the derivation of either the left or the right conjunct), have different variations to get different results. Look at lines 3 and 4 in the illustration above: the rule specification SIMPL gave us the left conjunct from line 1; the rule specification SIMPR gave us the right conjunct from line 2. The SIMPlification rule thus has two specifications to allow unambiguous application of the slash command. The case is similar with DSL and DSR, CDI and CDII (see your rule summary, in your logic text or Appendix II to the ARGUE manual).

3. In cases where a replacement rule could be applied to more than one connective in a formula, ARGUE will always apply the rule to the major connective. Thus, for example, if you use the slash command to apply the COMmutation rule to the formula P V (Q & R), ARGUE will commute the disjunctive part of the formula (since the 'V' is the major connective) as follows: (Q & R) V P.

4. When using the slash command to get ARGUE to type formulae for you and when there is a choice about the order in which the components (say, the conjuncts in a conjunction) occur, you should type the line numbers of the components (conjuncts) in the order in which you want them combined (conjoined). Look at line 5 in the illustration above. I typed / 3 4 CONJ and ARGUE formed the resultant conjunction with the conjunct from line 3 first and the conjunct from line 4 second. Had I typed / 4 3 CONJ ARGUE would have formed the conjunction in the reverse order: R & P rather than P & R.

ARGUE's rule set is given in the Appendix. Special ARGUE commands that allow work with English argument texts are given in the following section below. Sample I/O interactions with ARGUE are contained in Section 4, and sample exercises in the formal construction and analysis of principles and English arguments are illustrated in Section 5.
3. SPECIAL ARGUE COMMANDS

The ARGUE program operates with the same commands as Dartmouth's BERTIE, but it also has some special commands of its own that are useful for using ARGUE to construct and test your own arguments in English.

In the sample interactions that follow, notice that in the first example I have entered my own conclusion to derive along with my own premise. ARGUE then congratulates me when I have derived the designated conclusion, just as with a stored problem. Notice that after I have completed my derivation, ARGUE still allows me to LIST it or get ENGLISH translations.

Notice in the second example below that when I type 'BEGIN' in response to ARGUE's prompt, the program allows me to pursue any line of derivation: because ARGUE doesn't know in this case what my objective might be, it will let me continue until I type 'NEW' to get a new set of options (or 'STOP' in order to exit the program). The two examples below illustrate the two 'free-form' uses of ARGUE for purposes of entering one's own arguments or derivations. Also illustrated are the use of the slash '/ ' command to get ARGUE to type a line, and use of the LIST and NEW commands -- all of which are available in BERTIE as well. In addition, the interactions below illustrate three commands that are special to ARGUE (and RECON -- both of which handle ENGLISH argument texts):

ASSIGN: This command allows you to assign English translations to variables and constants so that ARGUE can print out a rough English translation of any line(s) of your argument.

Here it is important to remember the following conventions:
Let F through S stand for either sentences or predicates
Indicate a one or two-place predicate by one or two subscripts
For example: Fx is a one-place, Fxy a two-place predicate
Let x, y, z, or w be individual variables
Let a, b, c, or d be individual constants
If you wish to use the same letter more than once (as in the second example below), you may use numbered letters:
For example: Flx, F2x, F3x...etc
    a1, a2, a3....; x1, x2, x3...etc.

ALIST: This command will print out a list of the variable/constant assignments

ENG: This command will print your derivation with English, if you have entered English translations using ASSIGN.
    ENG N gives the English for line N
    ENG N-M gives the English for lines N through M
EXAMPLE 1: 'Enter your own conclusion to derive...'

@Run<PCOK>ARGUE

Free-form Argument Construction Program

Type 'INS' for instructions, otherwise,

Enter a conclusion to derive, request a problem, or type 'BEGIN'
? G & -P

BEGIN YOUR DERIVATION:

1 ? -(G => P) / PREMISE

2 ? ASSIGN

Variable or constant: ? G

Now enter an English assignment: ? God has committed a grievous crime

2 ? ASSIGN

Variable or constant: ? P

Now enter an English assignment: ? God will be punished

2 ? /1 IMPL

2 -(-G V P) / 1 IMPL

3 ? /2 DEM

3 G & -P / 2 DEM

CONGRATULATIONS! YOUR DERIVATION IS COMPLETE.

List your derivation, ask for a problem, or type 'NEW'.
? LIST

DERIVE: G & -P

1 -(G => P) / PREMISE
2 -(-G V P) / 1 IMPL
3 G & -P / 2 DEM
List your derivation, ask for a problem, or type 'NEW'.

? ENG 1

1 \(-(G \Rightarrow P)\) \hspace{1cm} \text{NOT (IF GOD HAS COMMITTED A GREIVOUS CRIME THEN GOD WILL BE PUNISHED)}

List your derivation, ask for a problem, or type 'NEW'.

? ENG 3

3 \(G \& \neg P\) \hspace{1cm} \text{GOD HAS COMMITTED A GREIVOUS CRIME AND NOT (GOD WILL BE PUNISHED)}

List your derivation, ask for a problem, or type 'NEW'.

? NEW

EXAMPLE 2: \ldots\text{or type 'BEGIN'}

[After listing my last derivation I typed 'NEW' to begin anew on something else. In response, ARGUE prompts me as usual:]

Enter a conclusion to derive, request a problem, or type 'BEGIN'

? BEGIN

Begin your derivation:

1 \(? -(x)(P_{lx} \Rightarrow P_{2x})\) \hspace{1cm} \text{/ PREMISE}

2 \(? \text{ASSIGN} \)

Variable or constant: \(? P_{lx} \)

Now enter an English assignment: \(? x \text{ is prosecuted} \)

2 \(? \text{ASSIGN} \)

Variable or constant: \(? P_{2x} \)

Now enter an English assignment: \(? x \text{ is punished} \)

2 \(? \text{ALIST} \)

\(P_{lx}: x \text{ IS PROSECUTED} \)

\(P_{2x}: x \text{ IS PUNISHED} \)
2 ? LIST

1 \(-(x)(P1x \Rightarrow P2x)\) / PREMISE
2 ? /1 IMPL
2 \-(x)(\neg P1x \vee P2x) / 1 IMPL
3 ? /2 DEM
3 \-(x)(\neg P1x \& \neg P2x) / 2 DEM
4 ? /3 QN
4 (Ex)(P1x \& \neg P2x) / 3 QN
5 ? LIST

1 \-(x)(P1x \Rightarrow P2x) / PREMISE
2 \-(x)(\neg P1x \vee P2x) / 1 IMPL
3 \-(x)(\neg P1x \& \neg P2x) / 2 DEM
4 (Ex)(P1x \& \neg P2x) / 3 QN

5 ? ENG 1

1 \-(x)(P1x \Rightarrow P2x)

\text{NOT (FOR ALL } x, \text{ IF } x \text{ IS PROSECUTED THEN } x \text{ IS PUNISHED)}

5 ? ENG 4

4 (Ex)(P1x \& \neg P2x)

\text{FOR SOME } x, x \text{ IS PROSECUTED AND NOT (x IS PUNISHED)}

5 ? NEW

Enter a conclusion to derive, request a problem, or type 'BEGIN'

?

At this point you could STOP and exit the program, enter your own conclusion to derive, request a problem from a problem set, or begin again as above.

Remember: Whatever English assignments you make within a derivation are forgotten by ARGUE as soon as you go on to a NEW derivation.
When you are working on a stored problem in one of ARGUE's problem sets, you may wish to review the original argument text given at the beginning of the problem. The following is an illustration of the use of the TEXT command for this purpose, along with the ENG and ALIST commands in an ARGUE problem:

Persons have a right to life, allright! And it's morally wrong to kill what's got a right to life. So, it's certainly morally wrong to kill a human fetus!

Supply the UNSTATED GENERAL PREMISE required to make this argument valid and derive the conclusion.

**DERIVE:** \( (x)(Fx \rightarrow Mx) \)

1 \( (x)(Px \rightarrow Rx) \) \hspace{1cm} / PREMISE
2 \( (x)(Rx \rightarrow Mx) \) \hspace{1cm} / PREMISE
3 ? TEXT

Persons have a right to life, allright! And it's morally wrong to kill what's got a right to life. So, it's certainly morally wrong to kill a human fetus!

3 ? ALIST

Px: x IS A PERSON
Rx: x HAS A RIGHT TO LIFE
Mx: IT'S MORALLY WRONG TO KILL x
Fx: x IS A HUMAN FETUS

3 ? ENG

**DERIVE:** \( (x)(Fx \rightarrow Mx) \)

FOR ALL x, IF x IS A HUMAN FETUS THEN IT'S MORALLY WRONG TO KILL x

1 \( (x)(Px \rightarrow Rx) \)

FOR ALL x, IF x IS A PERSON THEN x HAS A RIGHT TO LIFE

2 \( (x)(Rx \rightarrow Mx) \)

FOR ALL x, IF x HAS A RIGHT TO LIFE THEN IT'S MORALLY WRONG TO KILL x

3 ?
4. SAMPLE I/O INTERACTIONS WITH ARGUE AND ENGLISH ARGUMENTS

The following is a print-out of some sample interactions with the ARGUE program. It includes illustration of: (1) running the program; (2) program commands; (3) use of the ENGLISH command to obtain English translations of symbolized material; (4) use of the HINT command to obtain hints for solving the problem; (5) use of the LIST command to obtain a clean listing of the derivation so far; (6) use of the TEXT command to retrieve the statement of the problem and argument text; (7) use of the slash '/ ' command to have the program type out lines or make substitutions according to the legal rules of inference or replacement; (8) use of the SET and PROBLEM commands for gaining access to other problem sets and problems (which one may do at any point while in the program).

Sample student input is what is typed in after the program prompt, which is the question mark '?'. Our sample student in this illustration will first read through the stored hints for the problem and then work through and list the derivation. This is a simple exercise in supplying a 'missing' premise and proving the validity of the resulting argument by deriving the conclusion.

Somewhat more challenging exercises in formalization, the construction of unstated assumptions, and derivation are then given.

@RUN ARGUE

Free-form Argument Construction Program

Type 'INS' for instructions, otherwise,

Enter a conclusion to derive, request a problem, or type 'BEGIN'

? INS

ARGUE is designed to aid you in learning to construct your own arguments to given conclusions in valid deductive form while allowing you to consider the content and plausibility of your argument.

The following commands are available:

- SET N Opens problem set number N
- PRO N Gives you problem number N
- STOP Ends the program
- NEW Starts a new problem or argument; ends current one
- HINT Gives a hint about solving the problem
- DEL N Erases your proof from line N to the end
- REM N Removes line N only
- LIST Lists your proof with line justifications
- LIST M-N Lists lines M through N of your proof
- ENG Lists your proof with English translations
- ENG M-N Translates lines of M through N of your proof
- RULES Gives a list of the valid rules
- ASSIGN Allows you to assign English translations to new constants
Enter a conclusion to derive, request a problem, or type `BEGIN`

? SET 2
There are 5 problems in this set.

Enter a conclusion to derive, request a problem, or type `BEGIN`

? PROBLEM 2

Persons have a right to life, all right! And it's morally wrong to kill what's got a right to life. So, it's certainly morally wrong to kill a human fetus!

Supply the UNSTATED GENERAL PREMISE required to make this argument valid and derive the conclusion.

Let: $P_x = \text{`x is a person'}$
$R_x = \text{`x has a right to life'}$
$M_x = \text{`It's morally wrong to kill x'}$
$F_x = \text{`x is a human fetus'}$
$t / u = \text{`Arbitrary thing t / u'}$

DERIVE: $(\forall x)(F_x \implies M_x)$

1. $(\forall x)(P_x \implies R_x)$ / PREMISE
2. $(\forall x)(R_x \implies M_x)$ / PREMISE
3. ? HINT

REMEMBER TO USE THE ENG COMMAND IN ORDER TO DISPLAY THE ENGLISH VERSION OF THE ARGUMENT ONCE THE ORIGINAL STATEMENT OF THE ARGUMENT HAS SCROLLED OFF THE SCREEN.

5 Hints remaining.

3 ? ENGLISH

DERIVE: $(\forall x)(F_x \implies M_x)$

FOR ALL $x$, IF $x$ IS A HUMAN FETUS THEN IT'S MORALLY WRONG TO KILL $x$

1. $(\forall x)(P_x \implies R_x)$
   
   FOR ALL $x$: IF $x$ IS A PERSON THEN $x$ HAS A RIGHT TO LIFE

2. $(\forall x)(R_x \implies M_x)$
   
   FOR ALL $x$: IF $x$ HAS A RIGHT TO LIFE THEN IT'S MORALLY WRONG TO KILL $x$

3. ? HINT
You must supply the tacit assumption needed to make the argument valid as an EXPLICIT PREMISE. Isn't it obvious? Either of these will do:

(a) Human fetuses are persons
(b) Human fetuses have a right to life

But while either will make the argument VALID, (b) makes premise (1) superfluous whereas (a) posits a connection between persons and rights.

4 Hints remaining.

3 ? HINT

Since the original argument posits a connection between being a person and having a right to life, your reconstruction of the argument should preserve this connection: the argument seems tacitly to assume that it is ON THE BASIS OF BEING PERSONS that human fetuses have a right to life

The argument tacitly assumes that HUMAN FETUSES ARE PERSONS.
So, state this!
3 Hints remaining.

3 ? HINT

Use UNIVERSAL INSTANTIATION (UI) on the premises in order to display their underlying sentential structure. You must instantiate to pseudo-names (t) in order to later derive the given conclusion by

UNIVERSAL GENERALIZATION: What goes for EVERYTHING x goes for ARBITRARY THING t [by UI]; and what goes for ANY ARBITRARY THING t goes for EVERYTHING x [by UG].

2 Hints remaining.

3 ? HINT

Take 'Human fetuses are persons' (x) (Fx => Px) as an explicit premise

Get:  Pt => Rt  by UI
       Rt => Mt  by UI
       Pt => Mt  by HS
       Ft => Pt  by UI  and  Ft => Mt  by HS

1 Hints remaining.

3 ? HINT

Given: IF ANY ARBITRARY THING t is a human fetus, it's wrong to kill t

Then: For EVERYTHING x: If x is a human fetus, it's wrong to kill x

Thus, you can get (x) (Fx => Mx) from Ft => Mt by UG
where  Ft => Mt  was previously derived using UI

0 Hints remaining.

3 ? LIST
Persons have a right to life, allright! And it's morally wrong to kill what's got a right to life. So, it's certainly morally wrong to kill a human fetus!

Supply the UNSTATED GENERAL PREMISE required to make this argument valid and derive the conclusion.

Let:  
Px = 'x is a person'
Rx = 'x has a right to life'
Mx = 'It's morally wrong to kill x'
Fx = 'x is a human fetus'
t / u = 'Arbitrary thing t / u'

CONGRATULATIONS! YOUR DERIVATION IS COMPLETE.
List your derivation, ask for a problem, or type 'NEW'.
? LIST

DERIVE: \( (x)(Fx \Rightarrow Mx) \)

1 \( (x)(Rx \Rightarrow Rx) \)  / PREMISE
2 \( (x)(Fx \Rightarrow Mx) \)  / PREMISE
3 \( (x)(Fx \Rightarrow Px) \)  / PREMISE
4 \( Pt \Rightarrow Rt \)  / 1 UI
5 \( Rt \Rightarrow Mt \)  / 2 UI
6 \( Ft \Rightarrow Pt \)  / 3 UI
7 \( Pt \Rightarrow Mt \)  / 4, 5 HS
8 \( Ft \Rightarrow Mt \)  / 6, 7 HS
9 \( (x)(Fx \Rightarrow Mx) \)  / 8 UG

List your derivation, ask for a problem, or type 'NEW'.
? SET 51
There are 9 problems in this set.

Enter a conclusion to derive, request a problem, or type 'BEGIN'
? PROBLEM 1

Is preventing serious hurt to other people the only legitimate ground for justifying coercion or prohibition by law? Is HURT the only form of HARM?

Well, advertising the pleasures and techniques of sodomy on a large billboard in public is surely offensive even if it's not seriously hurtful to people. But if preventing serious hurt to others is the only legitimate ground for justifying prohibition by law, then (i) ONLY what is seriously hurtful may legitimately be prohibited and (ii) what is offensive but not seriously hurtful may NOT be prohibited. Therefore, serious hurt to others is not the only ground for justifying prohibition by law.

Be sure all the stated and additional required premises are symbolized. Then DERIVE the conclusion. [Use Indirect Proof: CP + REDUCTIO.]

Let: \( a \) = the act of advertising sodomy in public
\( d \) = the act of preventing actual hurt to other people
\( Gx = x \) is the only legitimate ground for justifying prohibition
\( Lx = x \) may legitimately be prohibited
\( Hx = x \) is seriously hurtful to other people; \( Ox = x \) is offensive

DERIVE: \(-Gd\)

1 \( Oa \& -Ha \)  / PREMISE
2 ? PROBLEM 4
About my skinning my own dog alive -- when it's done secretly, say, in the privacy of my basement, it is neither hurtful to people nor offensive to people. So, although it is morally objectionable, my skinning my dog alive may NOT legitimately be prohibited so long as it's done secretly.

SYMBOLIZE the stated premise of this argument below, and SUPPLY the unstated premises required to obtain the conclusion. One of the missing premises is a tacitly assumed principle of legitimate prohibition: you must construct the conditions it alleges to be NECESSARY and/or SUFFICIENT for justifying coercive prohibition.

Let:  
Lx = x may legitimately be prohibited  
Hx = x is hurtful to persons  
Ox = x is offensive to persons  
Sx = x is done secretly  
Mx = x is morally objectionable  
a = my skinning my dog alive

DERIVE: Ma & (Sa => -La)
5. SAMPLE ARGUE EXERCISES

Of the following ten sample exercises Problems 1 - 4 are taken from a problem set for the computer-assist program called ARGUE that guides argument reconstruction and deductive derivation (by means of hints and a proof-checker).

Problems 5 - 10 are sample examination questions, which could also be entered and worked through using the ARGUE program's proof-checker and ability to represent the symbolized form of an argument and its English translation side-by-side.

While formal reconstruction, symbolization, and derivation are central to these problems, it should be evident that they also involve tasks in philosophic and dialectical analysis. Difficulties of interpretation encountered in symbolization are often philosophically germaine.

Problems 1 - 4 involve supplying missing premises, constructing general principles (specifying necessary or sufficient conditions) of legitimate social constraint, and assessing the plausibility of principles by deriving their logical implications for given cases. These problems are based on chapters two and three of Joel Feinberg's Social Philosophy (Englewood Cliffs, NJ: Prentice-Hall, 1973).

Problems 5 - 10 involve similar tasks and relate specifically to John Stuart Mill's On Liberty and an actual case of controversial paternalism (which I have called 'the Williams case'). We use Mill and an assortment of such problem cases as the basis for seeking a system of plausible and consistent principles of legitimate social constraint -- again, within a framework of strict formal-logical constraints. (The justifiability of the latter qua social-intellectual constraints also happens to be topical to the course.) These problems are dialectically related in ways that should be evident.

Problem 1

Is preventing serious hurt to others the only legitimate ground for justifying coercion or prohibition by law? Is hurt the only form of harm?

Well, advertising the pleasures and techniques of sodomy on a large billboard in Times Square is surely offensive even if it's not hurtful to anyone. But if preventing serious hurt to others is the only legitimate ground for justifying prohibition by law, then what is offensive but not seriously hurtful may not be prohibited.

Symbolize the stated and any additional premises required to derive, by reductio ad absurdum, the conclusion that serious hurt to others is not the only ground for legitimate prohibition by law.
Let:  
\[ a = \text{the act of advertising sodomy in public} \]
\[ L_x = x \text{ may legitimately be prohibited by law} \]
\[ H_x = x \text{ is seriously hurtful to others} \]
\[ O_x = x \text{ is offensive to others} \]

**Problem 2**

Only what is seriously hurtful to others may legitimately be prohibited *unless* being otherwise harmful to others besides the agent (say, being a hazard or an invasion of an important interest) is a sufficient condition for legitimizing prohibition.

Now, in the case of advertising sodomy on a large billboard in Times Square, let's even grant that it does not seriously hurt people. But do we not have an important interest in going about our business in public free from embarrassing nuisances and distractions?

Symbolize the premises (stated and implied) required to derive the conclusion that:

There are cases where something that's not actually hurtful to people may legitimately be prohibited, and, in general, whatever is an invasion of our important and common interests may legitimately be prohibited by law.

Then derive this conclusion.

Let:  
\[ a = \text{the act of advertising sodomy in public (say, Times Square)} \]
\[ L_x = x \text{ may legitimately be prohibited} \]
\[ H_x = x \text{ is actually hurtful to people} \]
\[ I_x = x \text{ is the invasion of an important interest of others} \]

**Problem 3**

About my skinning my dog alive -- when it's done secretly, say, in the privacy of my basement, it is neither hurtful to other people nor offensive to people. So, although it is morally outrageous, my skinning my own dog alive may *not* be legitimately prohibited by law so long as it's done secretly.

Symbolize the stated and additional premises required to derive the conclusion of this argument. Then do the derivation. One of the missing premises is a tacitly assumed principle of legitimate prohibition: you must specify the conditions that it assumes are necessary or sufficient for justifying legal prohibition such that my skinning my dog alive is legal.
Let: \( a = \) my skinning my own dog alive

\[ Lx = x \text{ may legitimately be prohibited by law} \]

\[ Hx = x \text{ is hurtful to other people} \]

\[ Ox = x \text{ is offensive to other people} \]

\[ Mx = x \text{ is morally objectionable} \]

\[ Sx = x \text{ is done secretly} \]

**Problem 4**

(Note: A more creative problem of this sort would normally come after several more dialectically related problems like the three above.)

We would like a principle of legitimate social coercion that spelled out both the limiting and justifying -- necessary and sufficient -- conditions of this 'necessary evil.'

Construct and symbolize a principle such that:

1. Your principle states what you now take to be the more plausible candidates for necessary and/or sufficient conditions of legitimate social coercion, chosen from among the candidate conditions (Ix through Px) below. Your principle will state, in effect: **Only and/or all actions that are _____ may legitimately be prohibited**, where you are to fill in the blank. Your principle may certainly posit conjunctive or disjunctive conditions.

2. Your principle has, together with plausible additional assumptions that you state as explicit premises, the logical consequence that certain specific bad actions or practices \( a \) and \( b \) may legitimately be prohibited (you choose the examples) and that certain other permissible actions \( c \) and \( d \) may not be prohibited. Use the ASSIGN command to assign your chosen cases a constant (for example: \( a \) = addictive drug abuse) for purposes of using the ENGLISH command to obtain translations of your symbolized principle and the derivation of its consequences in your chosen cases.

Derive the consequences of your principle for your chosen cases (and be prepared to argue the plausibility of your principle against possible counter-examples in class).

Let: \( Lx = x \text{ may legitimately be prohibited} \)

\[ Ix = x \text{ is 'prejudicial' (Mill's term) to important interests of others} \]

\[ Ox = x \text{ is offensive to others} \]

\[ Mx = x \text{ is morally objectionable by current public standards} \]

\[ Px = x \text{ is harmful or hazardous to the agent him/herself} \]
A Problem Case in Point: Williams

Williams would like to be released from the hospital to which he was taken a year ago after an automobile accident (explosion) left him blind, unable to use his hands or legs, and with deforming burns over 80% of his body. His burns have not healed; if he is released from the hospital and allowed to go home, infection will set in and he will die. But Williams, who has been declared sane and mentally competent by the hospital psychiatrist, wants to die. Before his accident he enjoyed skiing, motorcycle racing, hang-gliding, and scuba diving. Blind and crippled, he will no longer be able to engage in these activities. Nor will he be able to resume his career as a real-estate broker. At age 27, Williams does not look forward to the kind of life he will live; on top of that, he does not like the life he does live. Each day he suffers excruciating pain when his entire body below the neck must be submerged in a chemical bath as treatment for his burns. The process of removing old and applying new dressings to his wounds is also exceedingly painful. Pain medications are of little help. Williams believes that even if he could adjust to life as a blind man and a cripple, going through the burn treatments for the sake of that life prospect is not worth it. His doctor, with the support of the hospital, is refusing to allow Williams to be removed from the hospital by his family or by hired transporters.

The reason is this: The doctor believes that Williams will one day be glad that he was not allowed to die. (Cf. Mill on 'general presumption.') It is true, as a matter of fact, that the vast majority of burn patients want to die while they are in the hospital, but when the treatments are over they are thankful that they were not permitted to have their own way, and that they are still alive. Williams has sought a court decision on his case; he believes the hospital has no legal right to keep him under its protection, especially since on the scene of the accident he requested that he not be taken to any hospital.

Problem 5

One exception to Mill's anti-paternalistic presumption that a mature individual should be considered the best judge of what's in his best interests is, according to Mill, "when he attempts to decide irrevocably now what will be best for his interest at some future and distant time. The presumption . . . is only legitimate where the judgment is grounded on actual . . . present experience; not where it is formed antecedently to [relevant] experience."

We might recall here Mill's view on the justifiability of coercive paternalistic constraint in the case of selling oneself into slavery (something that could, like death for Williams, appear ever so much more attractive than life in one's present and foreseeable state).

We might paraphrase this principle as follows: When a person's decision will prove irrevocable, he should not be presumed the best judge of what's in his best interests unless he both knows the far-reaching consequences of his decision in plain fact and has the experience to appreciate these consequences and weigh them against those of alternative courses of action.

Now, while Williams has been informed of the likely long-term benefits of continuing in treatment and while he knows full well the fatal consequences of discontinuing treatment, we
might nonetheless argue on the basis of the above principle that Williams should not be presumed to be the best judge of what's in his best interest in this case.

Symbolize the general paternalistic principle paraphrased above and additional premises (plausible in the Williams case) that will allow you to derive the conclusion that Williams should not be presumed the best judge of what's in his best interests regarding treatment. Then derive this conclusion.

Let:  

\( a = \) Williams;  
\( d = \) the act of discontinuing Williams' treatment

\( J_{xy} = x \text{ is presumed to be the best judge of whether } y \text{ is in his interest} \)

\( I_{xy} = x\text{'s decision to } y \text{ would prove irrevocable and far-reaching} \)

\( K_{xy} = x \text{ knows the far-reaching consequences (costs versus benefits)} \)
\( \text{of } y \text{ and alternative courses of action} \)

\( H_{xy} = x \text{ has the experience to appreciate, weigh and balance the} \)
\( \text{far-reaching consequences of } y \text{ against those of alternative courses of action} \)

Problem 6

Symbolize the stated premises of the following argument and any additional premise(s) plausible to assume about Williams and his doctor that will allow you to derive the given conclusion. (You will need to reflect imaginatively on the question of what important interests the doctor, any doctor, may have in refusing to release Williams from treatment.)

Is premise (2) an accurate or plausible version of Mill's so-called Harm Principle? What does he have in mind by an action's being 'prejudicial' to the interests of others? Explain.

(1) Even if \( x \) can accomplish \( y \) only if \( z \) does \( w \), and \( x \) has an important interest in doing so,

but \( z \) has an important interest in not doing \( w \),

then it's prejudicial to an important interest of \( z \)'s to force \( z \) to do \( w \)

(2) If it is prejudicial to an important interest of \( z \)'s to force \( z \) to do \( w \),

then it's not the case that \( z \) should be forced to do \( w \)

Therefore, Williams' doctor should not be forced (say, by the court) to allow Williams' release from treatment.
Let: \( a = \) Williams; \( d = \) Williams' doctor

\[ b = \text{Williams' getting himself brought home} \]

\[ c = \text{The doctor's allowing Williams to be carried out of the hospital} \]

\[ Mxy = x \text{ can manage to accomplish } y \]

\[ Qzw = z \text{ does } w \]

\[ Ixy = x \text{ has an important interest in accomplishing } y \]

\[ Nzw = z \text{ has an important interest in not doing } w \]

\[ Pzw = \text{It is prejudicial to important interests of } z \text{ to force him to do } w \]

\[ Fzw = z \text{ should be forced to do } w \]

**Problem 7**

Assume as your first premise the same general principle that was the first premise in the argument of the last problem.

But let's take a different case to test the plausibility of the general principle that was the second premise in the argument of the last problem. The assignments below will tell the story:

Let: \( a = \) A poor struggling father of 23 children, call him Abe

\[ d = \text{A derelict drunken dipsomaniac, call her Dipsy} \]

\[ b = \text{Abe's buying bread for his starving family with his cash that Dipsy stole} \]

\[ c = \text{Dipsy's coughing up the cash that she stole from Abe} \]

Use the same factors and other assignments as in the last problem.

Symbolize the general principle that was the first premise in the last problem as the first premise for your derivation.

Symbolize the second principle/premise from the last problem as a **hypothesis**.

Assuming that Dipsy **should** be forced to cough up the cash that she stole from Abe, symbolize this and the plausible assumptions concerning Abe and Dipsy that will then allow you to perform a **reductio ad absurdum** on the hypothesis.

**Problem 8**

Recall Mill’s position that "no one is warranted in saying to another human being of ripe years, that he shall not do with his life . . . what he chooses . . . . Interference . . . to overrule his
We might abstract from this statement the following general principle, supposing that we are speaking within the domain of 'purely' self-regarding actions by mature individuals ("of ripe years"):

For any person x, action y, and person z: it's never the case that x's reasonable or general presumption that y is not in the best interests of z is a sufficient condition to make it permissible for x to interfere to constrain z from doing y.

Or: It's never the case that it's permissible for x to interfere to constrain z from doing y if x has only a reasonable or general presumption that y is not in the best interests of z.

Given Feinberg's discussion of the grounds for justified paternalistic interference with the attempt at suicide, the following case of b, c, and d provides a putative counter-example to the above principle. Consider:

\[ d = \text{A known manic-depressive alcoholic who has been drinking and again is in the depths of depression. He has just come from a bar where he was crying in his cups about how he just couldn't go on any more.}\]
\[ \text{The barkeep, while sympathetic, replied that d has had these bouts before over the years, and, having attempted suicide on several occasions, has always been glad later, upon sobering up, to have failed in the attempt.} \]

\[ b = \text{The benevolent barkeep who has now followed d to the middle of a high bridge where d is climbing over the railing in an apparent attempt to jump into the freezing river below.} \]

\[ c = d's \text{ committing suicide by climbing over the railing and jumping into the freezing river to drown.} \]

Symbolize two premises about b, c, and d that (1) are each plausible in terms of Feinberg's analysis of justified paternalistic interference in cases of attempted suicide and (2) together allow you to derive the negation of the general principle above.

Then symbolize an alternative general principle to the effect that it's not the case that x's having reasonable/general presumption is sufficient to make it permissible for x to interfere to constrain z from doing y.

Be sure that your revised principle is consistent with the two assumptions about b, c, and d that you used in your previous derivation.

Consider the ambiguity in the general proposition that 'it's not the case that x's having reasonable presumption is sufficient....' Does this mean (1) that it's never sufficient or (2) that it's not always sufficient? To which of these two interpretations does Mill seem committed? Explain, briefly.
Let:  
\( b \) = The benevolent barkeep in the case in question
\( c \) = d's climbing over the railing to jump into the freezing river
\( d \) = The manic-depressive alcoholic in the case in question
\( P \text{xyz} \) = x has reasonable/general presumption to the effect that
\( y \) is not in the best interests of \( z \)
\( I \text{xyz} \) = It is permissible for x to interfere to constrain \( z \) from doing \( y \)

Problem 9

Consider the following paternalism principle and the story accompanying it below:

**Paternalism Principle (PP)**

It is morally permissible for a person \( x \) to force a person \( y \) to perform an action \( z \) if, and only if, \( y \) will violate someone's rights by not doing \( z \) or it is the case both that \( x \) with good reason believes that doing \( z \) is in \( y \)'s best interest and \( x \) either has important information that \( y \) lacks about the consequences of not doing \( z \) or has good reason to believe that, were \( y \) calmly to deliberate about being forced to do \( z \), \( y \) would agree to have \( x \) force him to do it.

**The Case of Nick and Tina**

Nick, heavily dependent on cigarettes, is extremely considerate of others. Outside of his own home he smokes only when those around him have consented to his doing so. But, when he smokes, he smokes very heavily. He knows that smoking is very bad for him and he would like to quit. However, though he has tried more than once to stop, he has not succeeded. Tina, a close friend of Nick's, can't understand why he continues to befoul his system and stain his fingers, teeth, and lungs with tobacco residue. She and Nick have talked about the matter many times. And, in a recent discussion she has told Nick that, she intends to lock him in his own apartment for a week and simply make it impossible for him to get any cigarettes. In response, Nick firmly told Tina that he did not want her to force him to kick his habit even if she could guarantee a 'cure' thereby. He appreciated her concern, but the problem was one that he felt he had to handle on his own, even if his own efforts should end in failure.

Symbolize Principle PP.

Symbolize three additional premises that (1) are plausible to assume in the given case and (2) will allow you to derive the conclusion that it is not morally permissible for Tina to force Nick to stop smoking.

Be sure none of the premises you posit turn out to be superfluous by deriving this conclusion.
Let:  
\[ b = \text{Benevolent Tina} \]
\[ c = \text{Nick's cutting out cigarettes} \]
\[ d = \text{Nicotine-dependent Nick} \]
\[ F_{xyz} = \text{It is morally permissible for } x \text{ to force } y \text{ to do } z \]
\[ R_{yz} = y \text{ will violate someone's rights by not doing } z \]
\[ G_{xzy} = x \text{ with good reason believes that doing } z \text{ is in } y's \text{ best interest} \]
\[ I_{xyz} = x \text{ has important information that } y \text{ lacks about the consequences of not doing } z \]
\[ H_{xyz} = x \text{ has good reason to believe that, were } y \text{ clamy to deliberate about being forced to do } z, \text{ he would agree to have } x \text{ force him to do it.} \]

**Problem 10**

(Note: This problem raises issues respecting the proper treatment of subjunctives and counterfactuals, which we must explain but ultimately finesse in the framework of first-order logic.)

Now, look here: it's simply not enough that a person be mentally competent in order to be a competent judge of what's in his best interest. Ignorance of likely harmful consequences can provide grounds to interfere, as Mill seems to allow in the case of a person who is about to cross a dangerous bridge: we may "seize him and turn him back," alter his intended course, in order to inform him of the danger, on the general presumption that, once he fully appreciates his alternatives and their consequences, he will alter his intended course and be glad for our interference. Presumably, once we inform him of the danger, if he is still in no state or position to fully appreciate the information—say, because he is drunk—we may detain him even further, provided that we have good reason to presume that if he \textit{were} in a position to fully appraise all the relevant consequences he \textit{would} choose not to cross the bridge. Being informed about alternatives and consequences may not be sufficient for being in a position to appraise them fully or competently.

One could argue on similar grounds in the Williams case that if Williams does undergo the full course of therapy, then he would come to consent to being detained in treatment (granted, in retrospect) and constraining him to undergo the full course of therapy would be permissible paternalism.

Using all but only the assignments below, symbolize this conclusion, general principles abstracted from the passage above, and other premises concerning the Williams case that will allow you to derive the conclusion.

Then derive the conclusion, being sure that you have no superfluous premises. You may use
Conditional Proof.

Let:  \( a = \) Williams;  \( c = \) the full course of therapy

\( M_x = x \) is mentally competent in general

\( J_{xy} = x \) is a competent judge of whether \( y \) is in his best interest

\( G_{xy} = x \) does undergo \( y \)

\( P_{xy} = \) Constraining \( x \) to undergo \( y \) is permissible paternalism

\( O_{xy} = x \) would come to appreciate the opportunities afforded by \( y \)

\( I_{xy} = x \) is informed about the likely consequences of \( y \)

\( L_{xy} = x \) lacks the experience to fully appreciate all the consequences of \( y \)

\( S_{xy} = x \) is in a state to fully appreciate the likely consequences of \( y \)

\( R_{xy} = x \) would presumably come to consent to \( y \) in retrospect
APPENDIX
THE PROOF-CHECKER'S RULES OF INference AND REPLACEMENT

CONTENTS

SENTENTIAL RULES OF INFERENCE

SENTENTIAL RULES OF REPLACEMENT

CONDITIONAL PROOF RULE & INDIRECT PROOF STRATEGY

QUANTIFICATIONAL RULE SCHEMAS

RESTRICTIONS ON THE USE OF UNIVERSAL GENERALIZATION

RESTRICTIONS ON THE USE OF EXISTENTIAL INSTANTIATION
### Sentential Logic: Rules of Inference

**Premise (PREM or P)**
A premise may be introduced on any line of a derivation, except within Conditional Proof.

**Excluded-Middle Introduction (E-MI or EMI):** \( P \lor \neg P \)
At any point in a derivation one may introduce a sentence of the above form.

<table>
<thead>
<tr>
<th>Conjunction (CONJ)</th>
<th>Addition (ADD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P ) ( Q ) ( P \land Q )</td>
<td>( P ) ( P \land Q ) ( P \lor Q )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Simplification (SIMPL)</th>
<th>Simplification (SIMPR)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P \land Q ) ( P )</td>
<td>( P \land Q ) ( Q )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Disjunctive Syllogism (DSL)</th>
<th>Disjunctive Syllogism (DSR)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P \lor Q ) ( \neg Q ) ( P )</td>
<td>( P \lor Q ) ( \neg P ) ( Q )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Modus Ponens (MP)</th>
<th>Modus Tollens (MT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P \to Q ) ( P ) ( Q )</td>
<td>( P \to Q ) ( \neg Q ) ( \neg P )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Constructive Dilemma (CDI)</th>
<th>Constructive Dilemma (CDII)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P \lor Q ) ( P \to R ) ( Q \to R ) ( R )</td>
<td>( P \lor Q ) ( P \to R ) ( Q \to S ) ( R \lor S )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hypothetical Syllogism (HS)</th>
<th>Reductio Ad Absurdum (RED)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P \to Q ) ( Q \to R ) ( P \to R )</td>
<td>( P \to (Q \land \neg Q) ) ( \neg P )</td>
</tr>
</tbody>
</table>
SENTENTIAL LOGIC: RULES OF REPLACEMENT

Note: The dual colon "::" indicates that the respective formulae are logically equivalent — i.e., that either formula may be replaced by the other in any line of a derivation, OR that either may be derived from the other in a derivation.

Double Negation (DN):  
\[ P :: \neg\neg P \]

Commutation (COM):  
\[ P \& Q :: Q \& P \]
\[ P \lor Q :: Q \lor P \]
\[ P \leftrightarrow Q :: Q \leftrightarrow P \]

Transposition (TRANS):  
\[ P \rightarrow Q :: \neg Q \rightarrow \neg P \]

Equivalence (EQUIV):  
\[ P \leftrightarrow Q :: (P \rightarrow Q) \& (Q \rightarrow P) \]
\[ P \leftrightarrow Q :: (P \& Q) \lor (\neg P \& \neg Q) \]

Implication (IMPL):  
\[ P \rightarrow Q :: \neg P \lor Q \]

De Morgan (DEM):  
\[ \neg P \& \neg Q :: \neg(P \lor Q) \]
\[ \neg P \lor \neg Q :: \neg(P \& Q) \]
\[ P \& Q :: \neg(\neg P \lor \neg Q) \]
\[ P \lor Q :: \neg(\neg P \& \neg Q) \]

Exportation (EXP):  
\[ (P \& Q) \rightarrow R :: P \rightarrow (Q \rightarrow R) \]

Tautology (TAUT):  
\[ P :: P \lor P \]
\[ P :: P \& P \]
CONDITIONAL PROOF RULE & INDIRECT PROOF STRATEGY

CONDITIONAL PROOF STRATEGY: HYPOTHESIS + CP

With conditional proof your aim is to derive a conditional, say: \( P \rightarrow Q \)

1. Assume the **ANTECEDENT** of the conditional as a **HYPOTHESIS**.

   [Be sure to enter any premises BEFORE you enter your hypothesis!]

2. Derive the **CONSEQUENT** of the conditional you wish to derive.

3. **DISCHARGE** your hypothesis by conditionalization:

   Form the desired conditional \( (P \rightarrow Q) \), citing the CP rule and the line of your hypothesis \( (P) \) through the line of the consequent \( (Q) \):

   **THUS:**
   
   1 \( P \) / HYPOTHESIS
   : 
   2 \( Q \) / [Cite rules/lines by which derived]
   3 \( P \rightarrow Q \) / CP 1-2 [Note use of dash '-']

INDIRECT PROOF STRATEGY: CP + REDUCTIO

Where you wish to derive some sentence, say: \( P \)

1. Assume as a **HYPOTHESIS** the **NEGATION** of the sentence to be derived:

   1 \( -P \) / HYPOTHESIS

2. Derive a **CONTRACTION**: 2 \( Q \& -Q \) / [Rule/line cit.]

3. Apply **CP**:

   3 \( -P \rightarrow (Q \& -Q) \) / CP 1-2

4. Apply the **REDUCTIO** rule: 4 \( P \) / REDUCTIO, 3

   **THUS:**
   
   1 \( -P \) / HYPOTHESIS
   : 
   2 \( Q \& -Q \) / [Cite rules/lines by which derived]
   3 \( -P \rightarrow (Q \& -Q) \) / CP 1-2
   4 \( P \) / REDUCTIO, 3
### QUANTIFICATIONAL RULE SCHEMA SUMMARY

**UI** To constants: 

<table>
<thead>
<tr>
<th>Px</th>
<th>Pa</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x)(Px =&gt; Qx)</td>
<td>Pa =⇒ Qa</td>
</tr>
</tbody>
</table>

or pseudo-names: 

<table>
<thead>
<tr>
<th>Px</th>
<th>Pt</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x)(Px =&gt; Qx)</td>
<td>Pt =⇒ Qt</td>
</tr>
</tbody>
</table>

**UG** ONLY from pseudo-names NOT occurring in lines obtained by EI: 

<table>
<thead>
<tr>
<th>Pt</th>
<th>Pt =⇒ Qt</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x)Px</td>
<td>(x)(Px =&gt; Qx)</td>
</tr>
</tbody>
</table>

**ON** 

<table>
<thead>
<tr>
<th>(x)Px</th>
<th>-(Ex)-Px</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x)-Px : -(Ex)Px</td>
<td></td>
</tr>
<tr>
<td>-(x)Px : (Ex)-Px</td>
<td></td>
</tr>
<tr>
<td>-(x)-Px : (Ex)Px</td>
<td></td>
</tr>
</tbody>
</table>

(x)Px & Qx : -(Ex)-(Px & Qx) 
(x)-(Px & Qx) : -(Ex)(Px & Qx) 
-(x)(Px & Qx) : (Ex)-(Px & Qx) 
-(x)-(Px & Qx) : (Ex)(Px & Qx) 

Steps: 
1. Change quantifier: (x)-Px -->> (Ex)-Px 
2. Negate quantifier: -(Ex)-Px 
3. Negate after quantifier: -(Ex)--Px 
4. Drop any double negation: -(Ex)Px 

**EG** From constants: 

<table>
<thead>
<tr>
<th>Pa</th>
<th>Pa &amp; Qa</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Ex)Px</td>
<td>(Ex)(Px &amp; Qx)</td>
</tr>
</tbody>
</table>

or pseudo-names: 

<table>
<thead>
<tr>
<th>Pt</th>
<th>Pt &amp; Qt</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Ex)Px</td>
<td>(Ex)(Px &amp; Qx)</td>
</tr>
</tbody>
</table>

**EI** ONLY to pseudo-names not used in a previous line: 

<table>
<thead>
<tr>
<th>(Ex)Px</th>
<th>-(Ex)-(Px &amp; Qx)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pu</td>
<td>-(Pu &amp; Qu)</td>
</tr>
</tbody>
</table>
RESTRICTIONS ON UNIVERSAL GENERALIZATION (UG)

1. You may NOT universally generalize from constants

You may universally generalize only from pseudo-names, never from constants. Generalizing from an individual case is obviously INVALID by commonsense, and proven to be so by the following argument schema and interpretation, whereby the premise (1) is obviously true but the conclusion (2) is obviously false:

Let: \( P_x = x \text{ was president of the U.S.}; \quad a = \text{Abe Lincoln} \)

\[ \begin{align*}
1 & \quad P_a \\
2 & \quad (x)P_x \\
\text{INVALID:} & \quad \text{Everything was president of the U.S.}
\end{align*} \]

2. You may NOT UG from pseudo-names that occur in a line obtained by El

When a given pseudo-name (say, \( t \)) ever occurs in a line that is obtained by El (as \( t \) does in line 3 below), you may not universally generalize from it -- even if the pseudo-name was itself originally obtained by UI (as \( t \) was at line 2 below). The following derivation and interpretation shows that this move can lead from truth (Everyone has a parent) to falsity (Someone is everyone's parent). Therefore, it is INVALID:

Let: \( \text{The Domain} = \text{people}; \quad P_{xy} = x \text{ is a parent of } y \)

\[ \begin{align*}
1 & \quad (x)(E_y)P_{yx} \\
2 & \quad (E_y)P_{yt} \\
3 & \quad P_{ut} \\
\text{INVALID >>>} & \quad (x)P_{ux} \\
4 & \quad (E_y)(x)P_{yx} \\
\end{align*} \]

\[ \begin{align*}
& \quad \text{Everyone has a parent} \\
& \quad / \ UI, 1 \\
& \quad / \ El, 2 [\text{Cannot UG from } t] \\
& \quad \text{Some person } u \text{ is everyone's parent} \\
& \quad / \ EG, 4 \text{ Someone's everyone's p}
\end{align*} \]
RESTRICTIONS ON EXISTENTIAL INSTANTIATION (EI)

1. You may NOT existentially instantiate to constants

You may existentially instantiate only to pseudo-names, never to constants: the following argument schema, existentially instantiating to a constant, is shown to be INVALID by the following interpretation, whereby the premise (1) is obviously true but the conclusion (2) is obviously false.

Let: \( P_x = x \) is president of the U.S.; \( d = \) Princess Diana

\[
\begin{align*}
1 & \ (Ex)P_x \quad \text{Someone is president of the U.S.} \\
\text{INVALID:} & \ 2 \ P_d \quad \text{Princess Diana is president of the U.S.}
\end{align*}
\]

2. You may NOT EI to a pseudo-name already introduced in the derivation

When a given pseudo-name (say, \( t \)) has been previously introduced in a derivation (say, by UI at line 2 in the example below), you must existentially instantiate to a different pseudo-name (say, \( u \)). The following interpretation shows that existentially instantiating to a pseudo-name already introduced can lead from truth (Everyone has a parent) to falsity (Someone is his own parent). The INVALID move is at line 3; line 4 is legal by EG.

Let: The Domain = people; \( P_{xy} = x \) is a parent of \( y \)

\[
\begin{align*}
1 & \ (x)(Ey)P_{yx} \quad \text{Everyone has a parent} \\
2 & \ (Ey)Pyt \quad / \ UI, 1 \\
\text{INVALID:} & \ 3 \ P_{tt} \quad \text{Someone } t \text{ is his own parent} \\
4 & \ (Ex)P_{xx} \quad \text{There is someone who's his own parent}
\end{align*}
\]
I am indebted to my former colleague Anita Allen for finding and writing up this case for use in the course. This case contains more factors than are accounted for in the selected exercises that follow. An actual case and not a mere 'thought experiment,' it provides a rich source of factors not explicitly accounted for in, say, Mill's principles or discussion of paternalism. Another advantage of this particular case is that there exists a gruesome and poignant documentary film to bring the issues that elude neat formalization home to the students. In this way we attempt to catch the students between the 'rock' of a hard case and the 'hard place' of having to formalize their working hypotheses or principles regarding what should be decided in such cases. The 'Williams' case is also an interesting one to use because the institution that originally controlled the lease on the film will not release the film, because of its truly painful impact, for use in freshman classes. Thus, control of the film of the case is itself a pertinent problem case under the topic of paternalism -- and one that bears directly on the students themselves.

I am indebted to my colleague Jonathan Pressler for this problem from an examination for one of his sections of the course.