CHAPTER 7
PROOFS: MAIN

The system of derivations, or "proofs" as it is sometimes called, differs in several important ways from both tables and trees. First, unlike tables and trees, derivations use permissive rules rather than ordering rules. These rules permit us to write certain formulas down, provided that certain other formulas have already been listed. We are not forced to use any particular rule at any particular point. Second, the purpose of derivations is different from the purpose of tables and trees. While tables and trees are primarily designed to show us whether or not an argument is valid, derivations show us why an argument is valid (assuming that it is), and they help us explain to other people how the conclusion of that argument follows from its premises.

A derivation is best viewed as simply a list of formulas. Each formula listed is either a premise of the argument, or the result of applying a rule on one or two earlier lines.

The derivation begins by listing the argument's premises. (The conclusion is often listed to the right of the final premise.) The objective is to get from the premises to the conclusion. Once this is accomplished, the derivation is finished. We have shown a way to arrive at the conclusion of the argument from its premises. (The list of formulas is said to be a derivation of the formula written on the last line -- the conclusion -- from the initial premises.)

Once we have constructed a derivation we can use it to help explain to others how to get from the argument's premises to its conclusion by using only simple and obviously correct reasoning processes.

There are numerous different systems of derivations. The one we will be developing here contains two different types of rules. First, it contains Rules of Inference. (There are, as you will see, nine different Rules of Inference.) Though all the rules are required, the Rules of Inference are the primary rules in the system and they need to be mastered first. In addition to these rules, however, the system also contains rules we will refer to as "Rules of Replacement." After we have explored all of the Rules of Inference we will examine the Rules of Replacement. Once we have included both sets of rules in the system, theoretically we will be able to derive the conclusion of any valid argument from its premises.

THE RULES OF INFERENCE

MODUS PONENS

n. **(♥>♦)**
p. **♥**

———

♦ **n, p, MP**

The rule Modus Ponens, abbreviated MP, tells us that if a horseshoe claim has already been listed on any earlier line of a derivation, and the left side of that horseshoe claim has also been listed on another line, we may write its right side down on any later line we wish. When we write the new formula down we justify it by citing both of the required earlier lines, followed by "MP."

AN EXAMPLE

1. ((P . Q) > - R)
2. (P . Q)
3. - R 1, 2, MP

A PROBLEM

1. (M > ((- P v R) > Q))
2. M
3. (- P v R) / Q

MODUS TOLLENS

n.	**(♥ > ♦)**		
p.	**- ♦**		
	- ♥		**n, p, MT**

The rule Modus Tollens, abbreviated MT, tells us that if a horseshoe claim has already been listed on any earlier line of a derivation, and a tilde its right side has also been listed on another line, we may write a tilde its left side down on any later line we wish. Once we have listed the new formula, we justify it by citing both of the required earlier lines, followed by "MT."

AN EXAMPLE

1.	((T v - Q) > - R)	
2.	(T v - Q)	
3.	(S > R)	
4.	- R	1, 2, MP
5.	-S	3, 4, MT

A PROBLEM

1.	- S	
2.	(T > S)	
3.	(- T > (P > S))	/ -P

HYPOTHETICAL SYLLOGISM

n.	**(♥ > ♦)**		
p.	**(♦ > ♣)**		
	(♥ > ♣)		**n, p, HS**

The rule Hypothetical Syllogism, abbreviated HS, tells us that if a horseshoe claim has already been listed on any earlier line of a derivation, and a second horseshoe claim, whose right side exactly matches the left side of the other horseshoe claim, has also been listed, we may write down a new horseshoe claim whose left side is the non-matching left side of the one horseshoe claim, and whose right side is the non-matching right side of the other horseshoe. The new claim is justified HS.

AN EXAMPLE

1.	(P > (Q . R))	
2.	((Q . R) > S)	
3.	(P > S)	1, 2, HS

A PROBLEM

1.	(P > Q)	
2.	((P > R) > - T)	
3.	(S > T)	
4.	(Q > R)	/ -S

ABSORPTION

n. (♥>♦)

(♥>(♥.♦)) n, Abs

Unlike MP, MT, and HS, which require that two earlier lines be listed for the rule to be used, Absorption (Abs) needs only one earlier line. That line must be a horseshoe claim. The rule Abs then tells us that we may write down a new horseshoe claim. The formula on the left of this horseshoe claim must match the left side of the earlier horseshoe claim, while its right side will be a dot claim. On the left of the dot we put the left side of the original horseshoe, while on the right side of the dot we put the right side of that horseshoe.

AN EXAMPLE

1. (P > Q)	
2. ((P . Q) > R)	
3. (P > (P . Q))	1, Abs
4. (P > R)	2, 3, HS

A PROBLEM

1. - (R . S)	
2. (R > S)	
3. (- R > T)	/ T

If you bothered comparing the four rules we have just introduced you might have noticed a couple of things about them. First, they all deal with claims that have as their main connectives, a horseshoe. Second, while two of these rules, viz., MP and MT, tell us how to break up a formula whose main connective is a horseshoe, the other two rules tell us how to build a formula that has a horseshoe as its main connective. This last point is, we think, especially helpful. Whenever you want to dig a part of a horseshoe formula out of that formula you should consider the possibility of using either MP or MT. And whenever you want to build a formula that is not yet listed and is a claim that has a horseshoe as its main connective, you should consider the possibility of using either HS or Abs.

SIMPLIFICATION

n. (♥.♦)

♥ n, Simp

Like both Modus Ponens and Modus Tollens, Simplification tells us how to use a claim that occurs earlier in a derivation. However, it deals with dots rather than horseshoes. It tells us that we can always write the left side of a dot claim down. (Oddly enough, it does not allow us to write the right side down.) When we use this rule, all we need to do is to cite the line whose left side we are pulling down, and "Simp."

AN EXAMPLE

1. (P > Q)	
2. (- Q . R)	
3. (- P > S)	
4. - Q	2, Simp
5. - P	1, 4, MT
6. S	3, 5, MP

A PROBLEM

1. ((P > T) . R)
2. (Q > P)
3. - T / -Q

CONJUNCTION

n. ♥
p. ♦

(♥.♦) n, p, Conj

Like Simplification, Conjunction also deals with a dot claim. However, it tells you how to build one, rather than destroy it. All you need are both sides of the dot claim you want, and you can list it. You justify the new line you are writing down by citing the two earlier lines required and writing "Conj."

AN EXAMPLE

1.	(P > Q)	
2.	(- Q . R)	
3.	((- P . - Q) > S)	
4.	- Q	2, Simp
5.	- P	1, 4, MT
6.	(- P . - Q)	4, 5, Conj
7.	S	3, 6, MP

A PROBLEM

1. (P . R)
2. (P > S)
3. ((P . S) > (S > T)) / (P > T)

Conjunction is a very powerful rule. It permits you to build infinitely many different formulas, beginning with just two. Thus, P and Q, yield (P.Q), and this, together with P gives (P.(P.Q)), etc. As with other building rules, however, the idea is to build a formula that you can then use to do something with.

Let's turn now to two rules that involve wedges, rather than dots.

ADDITION

n. ♥

p. (♥v♦) n, Add

Like Conjunction, Addition is a building rule; and also like Conjunction, it can be used to build infinitely many formulas. The rule tells us that we can add on a wedge anything to any formula we want. We sometimes like to call this rule "The Magic Rule" because it permits us to add on absolutely any formula. Like the rabbit, it comes out of the proverbial hat.

AN EXAMPLE

1. ((P v - Q) > R)	
2. (P . - S)	
3. P	2, Simp
4. (P v - Q)	3, Add
5. R	1, 4, MP
6. (R v (S = - T))	5, Add

A PROBLEM

1. ((P > - Q) > - R)	
2. (P > - S)	
3. ((- R v T) > - J)	
4. (- S > - Q)	/ (- J . - R)

DISJUNCTIVE SYLLOGISM

n. **(♥v♦)**
p. **-♥**

♦ **n, p, DS**

Our next rule, Disjunctive Syllogism (DS), tells us how to destroy a wedge claim. In effect, it says that if we already have a wedge claim listed on an earlier line and a tilde its left side listed on another earlier line, we may write down the right side of the wedge claim at any later point. The move is justified by citing the two required lines and DS.

AN EXAMPLE

1. (P > Q)	
2. (P v R)	
3. - Q	
4. - P	1, 3, MT
5. R	2, 4, DS
6. (R v - S)	5, Add

A PROBLEM

1. (- P . Q)	
2. (P v (R > S))	
3. (T > R)	/ (T > S)

CONSTRUCTIVE DILEMMA

n. **((♥>♦).(♣>♠))**
p. **(♥v♣)**

(♦v♠) **n, p, CD**

Constructive Dilemma, abbreviated CD, is a complicated rule. It says that if we already have a dot claim, both sides of which are horseshoes, and also a wedge claim, whose left side matches the left side of the horseshoe claim on the left of the dot, and whose right side matches the left side of the horseshoe claim on the right of the dot, we can build a wedge claim whose left side matches the right side of the former horseshoe and whose right side matches the right side of the latter horseshoe. (Examine the rule above carefully.)

AN EXAMPLE

1. ((P > Q) . (R > S))	
2. (P v R)	
3. (Q v S)	1, 2, CD

A PROBLEM

1. (P>Q)	
2. (Pv(R>S))	
3. -Q	
4. (T>W)	
5. (Qv(TvR))	/ (WvS)

THE RULES OF REPLACEMENT

The Rules of Replacement differ from the Rules of Inference in several ways. First, they do not require that you use them on the main connective. Instead, they can be applied to a part of a formula. Second, they require only one earlier line of the appropriate sort, and not, as most of the Rules of Inference do, two. Finally, you can work backwards with these rules. (All of this will, we hope, become clearer soon.)

Each of the rules, in effect, asserts that a formula of one sort can either replace, or be replaced by, a formula of another sort. To use the rule, you simply make the exchange and cite the earlier line you are making the exchange on. If you are exchanging part of a formula on an earlier line with another unit, the portion of the earlier line not replaced is simply copied.

DOUBLE NEGATION

--♥ = ♥ **n, DN**

This rule tells us that we may either chop off two tildes or add two tildes to any formula, or any part of any formula, which has already been listed. The justification consists in citing the line number of the formula on which the exchange has been made, followed by the abbreviation for the rule. Study the example below carefully.

AN EXAMPLE

1. (P > - - Q)	
2. ((P > R) > - - S)	
3. (Q > R)	
4. (P > Q)	1, DN
5. (P > R)	3, 4, HS
6. - - S	2, 5, MP
7. S	6, DN
8. (S v - T)	7, Add
9. - - (S v - T)	8, DN

A PROBLEM

1. - - ((- -P v Q) . S)	
2. - P	/ Q

COMMUTATION

(♥.♦) = (♦.♥)	**n, Com**
(♥v♦) = (♦v♥)	**n, Com**

The rule Commutation, abbreviated Com, is a flipping principle. It applies to both dots and wedges, and it allows us to reverse the two sides of the wedge, or dot claim.

AN EXAMPLE

1. ((P v Q) > R)	
2. ((S > (Q v P)) . - R)	
3. ((Q v P) > R)	1, Com
4. (S > (Q v P))	2, Simp
5. (S > R)	3, 4, HS
6. (- R . (S > (Q v P)))	2, Com
7. - R	6, Simp
8. - S	5, 7, MT

A PROBLEM

1. - - (P v Q)	
2. ((Q v P) > (R . T))	/ T

TRANSPOSITION

(♥>♦) = (-♦>-♥)	**n, Trans**

Like Commutation, Transposition is a flipping principle. However, it flips horseshoe claims, rather than dots or wedges. One other important difference between Transposition and Commutation should also be noted. When the two sides of the horseshoe claim are flipped, they either both get a tilde, or they both lose a tilde. Transposition is abbreviated Trans.

AN EXAMPLE

1. (P > - (Q > R))	
2. (S > (- R > - Q))	
3. (S > (Q > R))	2, Trans
4. (- - (Q > R) > - P)	1, Trans
5. ((Q > R) > - P)	4, DN
6. (S > - P)	3, 5, HS

A PROBLEM

1. ((- Q > - R) . S)	
2. - - ((R > Q) > T)	/ (S.T)

ASSOCIATION

(♥ . (♦ . ♣)) = ((♥ . ♦) . ♣)	**n, Assoc**
(♥ v (♦ v ♣)) = ((♥ v ♦) v ♣)	**n, Assoc**

The rule Association, abbreviated Assoc, is a parenthesis-moving rule. It applies only to dots and wedges. Roughly, it says that when you have a complex claim, which contains two wedges, or two dots, you can shift the innermost pair of parentheses to the left or right one. (Look at the rule and the example below to see how this is done.)

AN EXAMPLE

1. (((P v Q) v R) > T)	
2. ((S . (U . W)) > ((Q v R) v P)))	
3. ((P v (Q v R)) > T)	1, Assoc
4. (((Q v R) v P) > T)	3, Com
5. ((S . (U . W)) > T)	2, 4, HS
6. (((S . U) . W) > T)	5, Assoc

A PROBLEM

1. - (P v Q)	
2. (P v (Q v R))	
3. (- S > -R)	/ S

EXPORTATION

(♥ > (♦ > ♣)) = ((♥ . ♦) > ♣)   n, Exp

The rule Exportation, abbreviated Exp, like Association is a parenthesis-moving rule, but it applies to horseshoes. It says that if you have a claim which contains two horseshoes, and the right horseshoe is surrounded by a pair of parentheses, you can move the parentheses to the left one unit, but the first horseshoe converts into a dot. Alternately, if you have a pair of formulas surrounded by a dot, and then a horseshoe, followed by another formula, you can move the parentheses right one unit, but the dot converts into a horseshoe. (See the example below.)

AN EXAMPLE

1. (P > (Q > R))	
2. (S > (P . Q))	
3. ((P . Q) > R)	1, Exp
4. (S > R)	2, 3, HS

A PROBLEM

1. (P > (Q > R))	
2. (- R . S)	
3. (T > (P . Q))	/ (S . - T)

DISTRIBUTION

(♥ v (♦ . ♣)) = ((♥ v ♦) . (♥ v ♣))   n, Dist

(♥ . (♦ v ♣)) = ((♥ . ♦) v (♥ . ♣))   n, Dist

Distribution, abbreviated Dist, is probably the most complex rule of all. It tells us how to manipulate claims that contain combinations of wedges and dots. One version of it tells us that if we have a wedge claim, the right side of which is a dot, we can replace this with a dot claim. This dot claim will consist in two wedge claims. On the left side of each of these wedge claims we list the formula that was on the left of the original wedge. On the right side of the leftmost wedge claim we put the formula that was on the left side of the original dot, and on the right side of the rightmost wedge claim we put the right side of the original dot. The rule, of course, also works in reverse. It tells us that if we have a dot claim, both sides of which are wedge formulas, and if the left sides of both of these wedged claims matches, we can build a wedge claim. On the left of this wedge claim we put the common formula that appeared on the left side of both of the original wedge claims. On its right side we build a dot claim. On the left side of this dot we put the formula that appeared on the right side of the leftmost wedge, and on the right side of this dot we put the right side of the rightmost wedge. (See the top version of the rule above, and the example provided below.)

AN EXAMPLE

1. ((P v (Q . R)) > (S v (U . W)))	
2. (- T > ((P v Q) . (P v R)))	
3. (- T > (P v (Q . R)))	2, Dist
4. (- T > (S v (U . W)))	1, 3, HS
5. (- T > ((S v U) . (S v W)))	4, Dist

The other version of Distribution is exactly like the one just described, with dots and wedges exchanged throughout.

AN EXAMPLE

1. ((- P . Q) v (- P . R))	
2. (P v ((Q v R) > - S))	
3. (-P . (Q v R))	1, Dist
4. - P	3, Simp
5. ((Q v R) > - S)	2, 4, DS
6. ((Q v R) . - P)	3, Com
7. (Q v R)	6, Simp
8. - S	5, 7, MP

A PROBLEM

1. (P v (Q . R))	
2. - Q	/ P

DEMORGAN

-(♥.♦) = (-♥v-♦)	**n, DeM**
-(♥v♦) = (-♥.-♦)	**n, DeM**

One version of DeMorgan, abbreviated DeM, tells us that if we have a negated formula whose right side is a dot claim, we may eliminate this tilde, put a tilde in front of the formulas on both sides of the dot, and change the dot to a wedge. Alternately, if we have a wedge claim, both sides of which are negated, we may eliminate these two tildes, replace the wedge with a dot, and add a tilde in front of the resulting formula. The other version of DeM proceeds in exactly the same way, with wedges and dots replaced throughout.

AN EXAMPLE

1. ((- (P . Q) v R) > T)	
2. (((- P v - Q) v R) > T)	1, DeM
3. ((- P v (- Q v R)) > T)	2, Assoc

A PROBLEM

1. - (P . Q)	
2. - - P	
3. (- Q > R)	/ (P . R)

TAUTOLOGY

♥ = (♥v♥)	**n, Taut**
♥ = (♥.♥)	**n, Taut**

Tautology, abbreviated Taut, is a very simple rule. It tells us that we may add to any formula wedge itself, or dot itself. Alternately, if we have a claim that reads either a formula wedge itself, or a formula dot itself, we may reduce this to that formula alone.

AN EXAMPLE

1. (P > (- Q > - P))	
2. (P > (P > Q))	1, Trans
3. ((P . P)>Q)	2, Exp
4. (P > Q)	3, Taut
5. ((P v P) > Q)	4, Taut

A PROBLEM

1. ((P > - Q) . (R > - Q))	
2. (Q v T)	
3. (P v R)	/ (T . (P > - Q))

IMPLICATION

(♥>♦) = (-♥v♦)	**n, Impl**

Implication, abbreviated Impl, is a relatively easy rule, but an important one nonetheless. It tells us how to convert a horseshoe claim into a wedge claim, and vice versa. In effect, it says that if we have a horseshoe claim we can convert the horseshoe into a wedge, though the formula on the left side of this wedge must be negated as we do so. Alternately, it tells us we can convert a tilde wedge claim into a horseshoe claim by simply dropping the tilde from the formula on the left side of the wedge and changing the wedge into a horseshoe.

AN EXAMPLE

1. ((P > - Q) v R)	
2. ((- P v - Q) v R)	1, Impl
3. (- P v (- Q v R))	2, Assoc
4. (P > (- Q v R))	3, Impl

A PROBLEM

1. (Q v - P)
2. ((R > S) v T)
3. - T
4. (P v R)   / ((Q v S) . (- R v S))

EQUIVALENCE

(♥ = ♦) = ((♥ > ♦) . (♦ > ♥))   n, Equiv

(♥ = ♦) = ((♥ . ♦) v (- ♥ . - ♦))   n, Equiv

Our last rule, Equivalence, abbreviated Equiv, tells us how to work with a triple bar claim. Virtually whenever such a claim occurs in either a premise or the conclusion, we need to use one of the two versions of this rule. One version changes the triple bar into a double horseshoe claim, which is glued together with a dot; while the other version builds a wedge claim, both sides of which are dot claims. (Please examine the rule above, and example below, carefully.)

AN EXAMPLE

1. ((P = Q) > (R = S))	
2. - (R . S)	
3. ((P > Q) . (Q > P))	
4. (P = Q)	3, Equiv
5. (R = S)	1, 4, MP
6. ((R . S) v (- R . - S))	5, Equiv
7. (- R . - S)	2, 6, DS

A PROBLEM

1. (P = - Q)
2. - - Q
3. (P v R)   / (- P . R)

Although we will normally use only one of the two versions of Equivalence in a given problem, it isn't a bad idea to begin by listing both, and then selecting the one that turns out to be most useful. Unless you are expected to solve the problem in as few steps as possible (which will be the case in both the Examination and the Practice Exercises in the computer program that accompanies this text), or you see how to solve the problem on the fly, you should write down both versions of Equivalence.

GENERAL STRATEGIES AND HELPFUL SUGGESTIONS

If you are going to succeed at solving difficult derivations you need to learn to think about them in the right sort of way. Though there is no substitute for practice, the following suggestions may prove useful.

1. Always look carefully at the conclusion before doing anything else.
    a. If it contains a letter not found in any of the premises, at some point you will have to use Addition. Either you will need to Add the letter without a tilde, or you will need to Add a tilde then that letter. If one way isn't working out, try the other.
    b. If the conclusion is a dot claim, try to figure out how to get one of its sides. Then try to get the other side, and use Conj.
    c. If the conclusion is a triple bar claim, you will almost certainly have to use Equiv. Try translating the conclusion into both of its forms, and then try obtaining one of them.

d. If the conclusion is a horseshoe claim, there are several ways in which it might be obtained.

(1) HS is the most likely. Look and see if any of the premises have parts that look like the left side of the conclusion. Look and see if any of the premises have parts that look like the right side of the conclusion. If you find premises of both sorts, and HS is not immediately applicable, think in terms of trying to use the Rules of Replacement in order to get the right side of one to match the left side of the other.

(2) Another likely candidate here is Impl. Convert the conclusion into a tilde wedge. See if either it, or a part of it, resembles any of the premises.

(3) If the right side of the conclusion is a dot claim, and one of the elements of that dot matches the formula located on the left side of the horseshoe, try to obtain the conclusion through Abs.

e. If the conclusion is a wedge claim, look at each of the premises and see if any of them contain wedges, or horseshoes. If so, CD may be worth thinking about. Alternately, Addition is a possibility, though it doesn't work too often.

f. If the conclusion is complex and its main connective is a tilde, think in terms of using DeM.

(1) If the conclusion has the form, -(♥.♦), you might be able to get it by first getting -♥, or -♦, and then using Addition (and Commutation, if necessary).

(2) If it has the form: -(♥v♦) you might try getting -♥ and -♦, and then conjoining these before using DeM.

(3) If the conclusion has the form: -(♥>♦), try using Impl to convert the horseshoe into a tilde wedge, and then use DeM.

2. Now look carefully at the premises. Ask yourself if there are any Rules of Inference you can use on the lines available.

a. If any of the premises are dot claims, use Simp to pull out their left sides. Then use Com to flip the two sides around and pull out the other side.

b. If any of the premises are triple bar claims, use both versions of Equiv on them.

c. If two or more premises are horseshoe claims, see if you can't get the right side of one to match the left side of the other, and then use HS.

d. If the same letters or formulas occur on the same sides of two horseshoe claims, try using Transposition on one of them.

e. Whenever a tilde occurs on the outside of a formula, use DeM to push it in.

3. Once you have made some moves with the premises, go back to the bottom of the problem. The idea is to work from bottom up, and then turn around and work from top down, gradually working your way into the middle of the problem.

4. If you can't solve a problem, put it away. Go on to another problem. Or try doing something else. Sometimes our minds seem to work best on the problem when we aren't consciously thinking about it.

5. On those problems you cannot solve, if someone shows you a solution to them, try to see which rule you missed. (Most of us have a rule or two that we always seem to overlook.) Find your own problem rule, and then, whenever you are having difficulties with a derivation, think about using that rule.

6. Keep the list of rules below readily at hand. The Building Rules tell you what formulas you will need to have if you are trying to build a formula whose main connective is of the sort listed. These are the rules you should think in terms of when you are working up from the bottom of the problem. The Destroying Rules, on the other hand, tell you how to use formulas whose main connectives are of the sort listed. (They are the rules you want to think in terms of when you are working from the top of the problem down.)

BUILDING RULES		DESTROYING RULES	
HS	>	MP	>
Abs	>	MT	>
Conj	.	Simp	.
Add	v	DS	v
CD	v		

Good luck and happy hunting.

PROBLEMS

A. Construct proofs of each of the following arguments. You only need to use the Rules of Inference. If you cannot construct such a proof, the computer program can construct it for you.

1. ((P>Q)>(- R.S))
   (P>(M.N))
   (((M.N)>Q).S)
   (RvT) / T

2. P
   ((PvQ)> - R)
   (S>R)
   ((P. - S)>M) / (MvN)

3. (P>Q)
   (R>S)
   (PvR) / ((P.Q)vS)

4. ((P>(P.Q))> - R)
   (S>R)
   (P>Q)
   (- S>(R>P)) / (S>Q)

B. Construct proofs of each of the following arguments. You only need to use the Rules of Inference. Note: The computer program cannot solve these proofs.

1. (P>Q)
   (R>S)
   (PvR)
   (((P.Q)v(R.S))>(Q>R)) / (P>S)

2. (P>Q)
   (- Q.(R=S))
   ((- P. - Q)>(R>Q))
   (Rv(Q>S)) / ((P>(P.S))v(L>M))

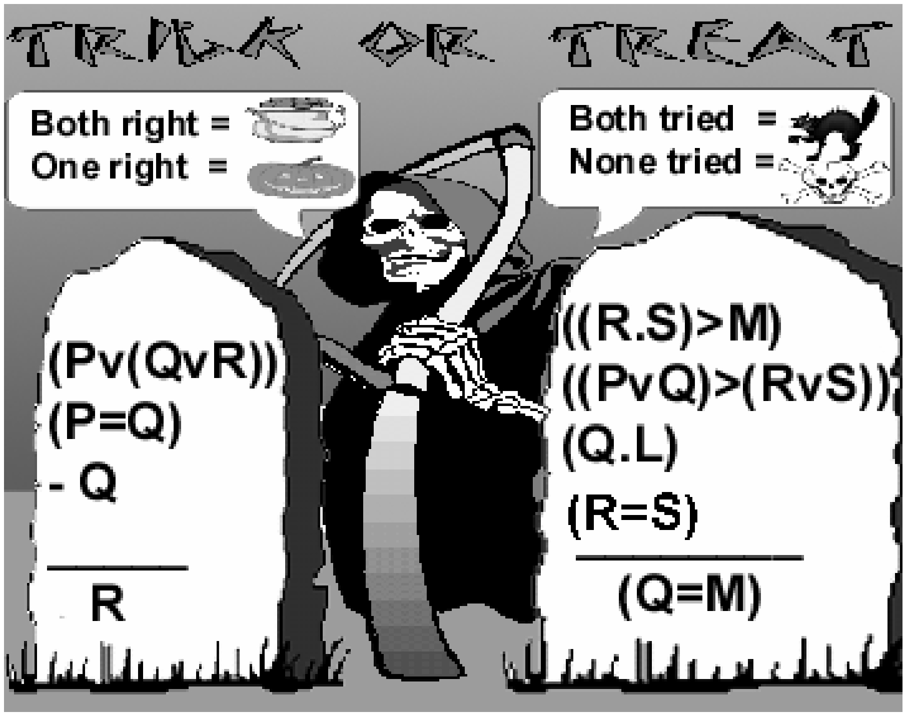
C. Construct proofs of each of the following arguments. You will need to use both the Rules of Inference and the Rules of Replacement. If you cannot construct such a proof, the computer program can construct it for you.

1. - T
   ((T>L)>(M>N)) / (- N> - M)

2. (R>P)
   ((- P. - Q) vR) / (P=R)

3. (P=Q)
   (- P. (Q=M)) / - M

4. ((P.Q)>R)
   ((Q>R)>S) / (Sv - P)


TRICK OR TREAT
Both right =
One right =
Both tried =
None tried =
(Pv(QvR))
(P=Q)
- Q
R
((R.S)>M)
((PvQ)>(RvS))
(Q.L)
(R=S)
(Q=M)