

# Tarski's World: Revised and Expanded Edition

Dave Barker-Plummer

Jon Barwise

John Etchemendy

in collaboration with

Albert Liu

CENTER FOR THE STUDY  
OF LANGUAGE  
AND INFORMATION





Copyright © 2007  
CSLI Publications  
Center for the Study of Language and Information  
Leland Stanford Junior University  
Printed in the United States  
11 10 09 08 07 1 2 3 4 5

*Library of Congress Cataloging-in-Publication Data*

Barker-Plummer, Dave.  
Tarski's world / by Dave Barker-Plummer, Jon Barwise,  
John Etchemendy.— Rev. and expanded.

p. cm. — (CSLI lecture notes ; no. 169)

Includes indexes.

ISBN 1-57586-484-3 (pbk. : alk. paper)

1. First-order logic. 2. Tarski's world. I. Barwise, Jon.  
II. Etchemendy, John, 1952– III. Title. IV. Series.

BC128.B35 2004

511.3–dc22 2004017514

CIP

∞ The acid-free paper used in this book meets the minimum requirements of the American National Standard for Information Sciences—Permanence of Paper for Printed Library Materials, ANSI Z39.48-1984.

CSLI was founded in 1983 by researchers from Stanford University, SRI International, and Xerox PARC to further the research and development of integrated theories of language, information, and computation. CSLI headquarters and CSLI Publications are located on the campus of Stanford University.

CSLI Publications reports new developments in the study of language, information, and computation. Please visit our web site at

<http://cslipublications.stanford.edu/>

for comments on this and other titles, as well as for changes and corrections by the authors and publisher.

*To the people who made Tarski's World a reality:  
Rick Wong, Rolf van Widenfelt, Steve Loving,  
Alan Bush, Mike Lenz, Eric Ly, Pete Murray,  
Dan Fish, Kalpana Bharadwaj,  
Christopher Fuselier and Albert Liu.*



---

# Contents

<b>How to Use This Book</b>	<b>xi</b>
To the student	xii
To the instructor	xiii
Acknowledgements	xv
What's new in this edition?	xv
More acknowledgements	xvi

## I Exercises

<b>1</b>	<b>Instructions About the Exercises</b>	<b>3</b>
<b>2</b>	<b>Exercises on Propositional Logic</b>	<b>9</b>
<b>3</b>	<b>Exercises on First-order Logic</b>	<b>25</b>
<b>4</b>	<b>More Theoretical Exercises</b>	<b>53</b>

## II Using the Software

<b>5</b>	<b>Using Tarski's World</b>	<b>67</b>
5.1	Getting started	67
5.1.1	Opening saved files	68
5.1.2	Starting new files	68

5.1.3	Saving a file	68
5.1.4	Closing Tabs	69
5.1.5	Reverting a File	69
5.1.6	Printing	70
5.1.7	Quitting (Exiting) Tarski's World	70
5.2	The World Panel	70
5.2.1	Adding blocks	70
5.2.2	Selecting blocks	70
5.2.3	Moving blocks	70
5.2.4	Sizing and shaping blocks	70
5.2.5	Naming blocks	71
5.2.6	Deleting blocks	71
5.2.7	Cutting, copying, and pasting blocks	71
5.2.8	Hiding labels	72
5.2.9	2-D view	72
5.2.10	Rotating Worlds	72
5.3	The Sentence Panel	72
5.3.1	Writing formulas	72
5.3.2	Commenting your sentences	73
5.3.3	Creating a list of sentences	73
5.3.4	Moving from sentence to sentence	73
5.3.5	Deleting sentences	74
5.3.6	Typing symbols from the keyboard	74
5.3.7	Cutting, copying, and pasting	75
5.4	Verifying syntax and truth	75
5.5	Playing the game	76
5.5.1	Picking blocks and sentences	76
5.5.2	Backing up and giving up	76
5.5.3	When to play the game	77
5.6	Preferences	78
<b>6</b>	<b>Using Submit</b>	<b>81</b>
6.1	Getting started	81
6.2	Choosing files to submit	83
6.3	How you know your files were received	85
6.4	Saving your user data	85



## Appendixes

- A First-order Logic 89**
- A.1 First-order languages 89
  - A.2 Individual constants 90
  - A.3 Predicate symbols 90
  - A.4 Atomic sentences 91
  - A.5 Connectives 92
  - A.6 Variables 95
  - A.7 Atomic wffs 96
  - A.8 Quantifiers 96
  - A.9 Wffs and sentences 97
  - A.10 Satisfaction and truth 100
  - A.11 Game rules 101
  - A.12 Logical equivalences 103
  - A.13 Validity and logical consequence 104
- B Using Tarski's World 5.x 107**
- B.1 Getting started 107
    - B.1.1 Launching Tarski's World 107
    - B.1.2 The main windows 107
    - B.1.3 Opening saved files 109
    - B.1.4 Starting new files 110
    - B.1.5 Saving a file 110
    - B.1.6 Quitting (Exiting) Tarski's World 110
  - B.2 The world window 111
    - B.2.1 Adding blocks 111
    - B.2.2 Naming blocks 111
    - B.2.3 Moving blocks 111
    - B.2.4 Sizing and shaping blocks 112
    - B.2.5 Deleting blocks 112
    - B.2.6 Hiding labels 112
    - B.2.7 2-D view 112
    - B.2.8 Rotating Worlds 113

B.3	The keyboard and sentence windows	113
B.3.1	Writing formulas	113
B.3.2	Commenting your sentences	114
B.3.3	Creating a list of sentences	114
B.3.4	Moving from sentence to sentence	114
B.3.5	Deleting sentences	115
B.3.6	Typing symbols from the keyboard	115
B.3.7	Cutting, copying, and pasting	116
B.3.8	Printing	116
B.4	The evaluation box / sentence inspector	116
B.4.1	Verifying syntax and truth	116
B.5	Playing the game	117
B.5.1	Picking blocks and sentences	118
B.5.2	Backing up and giving up	118
B.5.3	When to play the game	119

**General Index**      **121**

**File Index**      **125**

---

## How to Use This Book

Tarski's World is a computer-based introduction to one of the most significant and widely applied intellectual developments of the twentieth-century: first-order logic. While it grew out of work in the philosophy of mathematics, first-order logic has become a requisite tool for the study of a multitude of disciplines, from philosophy and mathematics, its original inspirations, to linguistics, psychology, computer science, and artificial intelligence.

First-order logic is the most basic system of logic. While the language it is based on uses very few primitive concepts, and so is easily learned, it has proven to be a powerful language, one capable of expressing many important notions. Indeed, it is frequently claimed that the language of first-order logic is the appropriate language for the conduct of all rigorous discourse. We do not in fact agree with this opinion, since there are many demonstrably richer languages which are every bit as precise and rigorous. Nevertheless, first-order logic stands in a privileged position; all more powerful logics are merely enrichments in one direction or other of the first-order case. It is clearly the right starting point for any student who needs to understand logic.

First-order logic has two main parts, *syntax* and *semantics*. On the syntactic side we have notions like:

1. predicate and individual symbol
2. connective and quantifier symbol
3. sentence and well-formed formula (“wff”)
4. free and bound variable
5. inference rule
6. provable wff

On the semantic side we have notions like:

1. relation and individual
2. world, model, or relational structure
3. truth and satisfaction
4. entailment
5. valid wff

Most computer-based treatments of logic concentrate almost entirely on concepts from the list of syntactic notions, in particular the last two. But the main lesson of the last fifty years' research in logic has been that the items on the second, semantic list are by far the most fundamental. In designing Tarski's World, our goal was to provide an introduction to the semantic side of logic. The program is named after one of the pioneers of the semantic approach to logic, the famous Polish-American logician, Alfred Tarski.

### **To the student**

Tarski's World makes learning the basic ideas of first-order logic much more interesting, fun, and efficient than any other method we have found. Part one of the book consists of over one hundred exercises which you can complete, most using the the Tarski's World application.

If you are using Tarski's World in conjunction with a logic class, then we suggest that you proceed as follows. Begin by skimming Chapter 5 for an overview of the Tarski's World program, and appendix A for a brief introduction to FOL. Then you should work carefully through Section 5.5, **Playing the game**, to make sure you understand how and when to play the game, and why it works the way it does. Finally, when you are comfortable with the program, plunge into the exercises in Chapters 2–4, referring to other parts of the book as needed. The table of contents and index will help you find your way if you get stuck on something.

If you are working through Tarski's World on your own rather than in a logic class, then you should start by reading appendix A, to get an overview of FOL. Then read through Chapter 5 for an introduction to the Tarski's World application. Refer back to appendix A as needed. You should be able to read through the chapter in a couple of hours. After that, you will feel comfortable with Tarski's World. Once you have finished that, you should be ready to start work on Chapters 2–4. As you work through the exercises, you will need to consult various sections

of appendix A from time to time. Be sure to submit your solutions to the Grade Grinder to receive feedback on how you are progressing.

### **To the instructor**

Our motivation for developing Tarski's World was to make teaching first-order logic easier and more fun for us. There were two particularly vexing problems.

One had to do with getting introductory students to understand the central semantical ideas of first-order logic. The way these ideas are usually presented in textbooks makes them so abstract that students have a hard time understanding the point. Tarski's World makes them very concrete and easy to understand, and so makes our job, both in the classroom and during office hours, much more pleasant.

The other problem had to do with teaching students how to express themselves in the first-order language. Some students catch on quite quickly, while others need a lot of help and practice. Unfortunately, it is help that is hard to give. For example, when you give translation exercises, there is no single right answer: anything logically equivalent to a right answer is also a right answer. So someone has to read the answers carefully and try to see if they are logically equivalent to the right answer—an undecidable question, of course.

Tarski's World allows us to solve this problem in two ways. First, it opens up many ways other than translation to teach what first-order sentences mean. If you scan through our exercises and think about them, you will see that they employ all the various ways that we learn any new language. With Tarski's World, we are not limited to the abstract task of translating back and forth from English: we can directly describe worlds, use the language to identify objects, construct worlds satisfying a description, and so forth. What's more, when it does come to translation, Tarski's World allows a better way to check if the answer is correct. It allows the student to test the truth-value of the translation against the truth-value of the original English sentence in a variety of worlds, to see if they are always the same. These tests won't mistakenly reject logically equivalent translations, as syntactically based tests invariably do.

In designing Tarski's World, we wanted a tool that could be used in two ways. First of all, we wanted to use it as an integral part of our basic logic course. In such a course, we tailor many of our classroom

examples to Tarski's World, and assign a large number of exercises from Chapters 2–4 of the book. Most of these exercises may be submitted to the Grade Grinder, our Internet-based grading service.

The second use we put Tarski's World to is with more advanced logic courses. In teaching such courses, we usually find a few students who really do not know how to express themselves in first-order logic. Rather than let these students flounder, we wanted a tool that we could simply hand them, and let them work through on their own. Tarski's World also serves this function remarkably well.

As a result, we think we have come up with a tool that is flexible enough to be of use to almost anyone teaching first-order logic. We hope that you find it to be as helpful as we have. We welcome suggestions for improvements in later versions, both from you and from your students.

This book is intended for instructors who want to use Tarski's World as a supplement to some other logic text, or in a course not devoted primarily to logic. Our stand-alone courseware package *Language, Proof and Logic*, would be more appropriate for instructors teaching a course devoted to FOL. *Language, Proof and Logic* contains Tarski's World as well as two other applications: Fitch, a program for constructing natural deduction proofs, and Boole, a program for constructing truth tables. Purchasers of *Language, Proof and Logic* also have access to the Grade Grinder. The book contains (optional) chapters on set theory, on inductive definitions, and on such topics as resolution and unification. *Language, Proof and Logic* is published by CSLI Publications, and is distributed by the University of Chicago Press. More information is available from the Language, Proof and Logic web site, <http://lpl.stanford.edu>.

If you find Tarski's World useful, you might also be interested in *Hyperproof* and *Turing's World*, two other instructional packages that we have developed for use in logic courses. *Hyperproof* is an introduction to analytical reasoning, built on the semantic perspective presented in Tarski's World. *Turing's World* is a self-contained introduction to Turing machines, one of the fundamental notions of logic and computer science. Like Tarski's World, both of these programs are published by CSLI Publications and distributed by the University of Chicago Press. At present, they are available only for the Macintosh.

## Acknowledgements

Our primary debt of gratitude goes to the programmers who made Tarski's World such a wonderful program. The original version was written by Rick Wong and Rolf van Widenfelt, under the guidance of Steve Loving. Developing the original program took over three years from our initial conception to the first released version. Rick, Rolf, and Steve's work on the program was supported by the Faculty Author Development Program at Stanford University. The current Macintosh version of Tarski's World incorporates numerous improvements to the original program, and itself took many years to complete. The programming was carried out by Mike Lenz, except for the desktop grading facility, which was written by Alan Bush.

The growing success of the package over the past few years convinced us to develop versions for other machines. This development has been overseen by the (alphabetically) third author. The NeXT and Windows versions share a computational engine, but required completely distinct user interface code. The common engine was written by Eric Ly, Pete Murray, and Dan Fish. Eric wrote the NeXT interface single-handedly in a matter of months, a great tribute both to Eric and to the NeXTstep development platform. The Windows interface was much more difficult, due to the complexities of the Windows environment. The interface was written by Christopher Fuselier and Kalpana Bharadwaj, building on earlier code written by Pete and Dan. Except for the original version, all work was funded by the Center for the Study of Language and Information.

People who have been involved in software development will realize what a tremendous debt we owe to all ten of our programmers. The creativity, originality, and pure hard work that goes into developing an ambitious piece of software is too often overlooked or undervalued. Finally, but perhaps most important, we have gotten a lot of personal pleasure out of working with all of our Tarski's World programmers over the past decade, and take our hats off to them for a job well done.

## What's new in this edition?

This edition incorporates some major changes. First and most obvious is the addition of the Grade Grinder service to the package. This grading service allows students to submit answers to exercises over the Internet to an automated grading server, and to receive commentary

on their work almost immediately by email. The Grade Grinder thus serves as an always-available teaching assistant that can point out many of the errors typically made by students in working through exercises. Students can correct many of these errors by themselves, once they have been pointed out to them. This has the effect of freeing instructors from grading these misunderstandings and leaving more time for aiding students with deeper conceptual misunderstandings. In addition, the Grade Grinder frees the instructor from the burdens of managing electronic submission of homework, for example the storage, opening and management of the many files that are submitted in response to homework exercises.

In order for an instructor to receive grade reports from their students, it is necessary for them to register at

`http://tarskis-world.stanford.edu`

An important change to the software is the simplification of the semantics of the predicate `Between`. In this edition, an object can only be between two others if they are all on the same row, column or diagonal of the checkerboard. We believe that the advantages of a rule that is simple to state outweigh the possible unnaturalness of the definition.

The original introduction to the text was written at a time when students could not be assumed to be familiar with basic operations like opening, saving and printing files on a computer. We have revised the introduction to reflect the fact that these operations are now likely to be familiar to all students using our book.

Finally we have added to the exercises in the package. This edition contains more than thirty exercises not found in the previous edition.

The Tarski's World web site is

`http://tarskis-world.stanford.edu`

Users should consult this site for updated software, errata, and other information concerning the package.

Bug reports and other correspondence concerning the software can be addressed to `TWbugs@csl.stanford.edu`

### **More acknowledgements**

Since the previous edition of this book, Albert Liu has taken over the development of Tarski's World. The current version of Tarski's World incorporates numerous of his improvements. Albert Liu, Richard Sanders and Tom Robertson contributed code to `Submit` and the `Grade Grinder`.



Ben Davidson was largely responsible for the modifications necessary to the Grade Grinder, and for identifying necessary changes to the text, which enable us to offer the grading service to readers of this book.



**Part I**

---

**Exercises**



---

## Instructions About the Exercises

This book came packaged with software that you must have to use the book. In the software package, you will find a CD-ROM containing the Tarski's World application. The CD-ROM also contains an electronic copy of the book, in case you prefer reading it on your computer. When you buy the package, you also get access to the Grade Grinder, an Internet grading service that can check whether your work is correct. Most of the exercises in this book require that you create a file or files using Tarski's World and then submit these using the program Submit. When you do this, your solutions are submitted to our grading server which assesses your files and sends a report to you and (optionally) your instructor.

Exercises in the book are numbered  $n.m$ , where  $n$  is the number of the chapter and  $m$  is the number of the exercise in that chapter. Exercises whose solutions consist of one or more files that you are to submit to the Grade Grinder are indicated with an arrow ( $\nearrow$ ), so that you know the solutions are to be sent off into the Internet ether. Exercises whose solutions are to be turned in (on paper) to your instructor are indicated with a pencil ( $\pencil$ ). For example, exercises might look like this:

$\nearrow$  **Exercise 1.1** Use Tarski's World to build a world in which the following sentences are all true. . . .

$\pencil$  **Exercise 1.2** Turn in an informal proof that the following argument is logically valid. . . .

The arrow on Exercise 1.1 tells you that the world you create using Tarski's World is to be submitted electronically, and that there is

nothing else to turn in. The pencil on Exercise 1.2 tells you that your solution should be turned in directly to your instructor, on paper.

Some exercises ask you to turn in something to your instructor in addition to submitting a file electronically. These are indicated with both an arrow and a pencil (↗|✎). This is also used when the exercise *may* require a file to be submitted, but may not, depending on the solution. For example, the next exercise might ask:

↗|✎ **Exercise\* 1.3** Is the following argument valid? If so, turn in an informal proof of its validity. If not, use Tarski’s World to build a counterexample and submit your world as World 1.3.

Here, we can’t tell you definitely whether you’ll be submitting a file or turning something in without giving away an important part of the exercise, so we mark the exercise with both symbols.

By the way, in giving instructions in the exercises, we will reserve the word “submit” for electronic submission, using the Submit program. We use “turn in” when you are to turn in the solution to your instructor.

Exercises may also have from one to three stars (\*, \*\*, \*\*\*), as a rough indication of the difficulty of the problem. We think that the exercise above would be a little more difficult than average.

When you create files to be submitted to the Grade Grinder, it is important that you name them correctly. Sometimes we will tell you what to name the files, but more often we expect you to follow a few standard conventions. Our naming convention is simple. Your file should be called **World n.m** or **Sentences n.m**, where *n.m* is the number of the exercise. The key thing is to get the right exercise number in the name, since otherwise your solution will be graded incorrectly. We’ll remind you of these naming conventions a few times, but after that you’re on your own.

**Your First Exercise** Here’s your first Submit exercise. Make sure you actually do it, right now if possible. It will teach you how to use Submit to send files to the Grade Grinder, a skill you definitely want to learn. You will need to know your email address, your instructor’s name and email address, and your Book ID number before you can do the exercise. If you don’t know any of these, talk to your instructor first. Your computer must be connected to the Internet to submit files. If it’s not, use a public computer at your school or at a public library.

✦ **Exercise 1.1** (Submit) We're going to step you through the process of submitting a file to the Grade Grinder. The file is called **World Submit Me 1**. It is a world file, but you won't have to open it using Tarski's World in order to submit it. We'll pretend that it is an exercise file that you've created while doing your homework, and now you're ready to submit it. More complete instructions on running Submit are contained in Chapter 6.

1. Find the program Submit on the CD-ROM that came with your book. Submit has a blue and yellow icon and appears inside a folder called **Submit Folder**. Once you've found it, double-click on the icon to launch the program. If you have installed the software onto your hard disk, the folder will be within the **TW Software** folder created by the installation.
2. After a moment, you will see the main Submit window, which has a rotating cube in the upper-left corner. The first thing you should do is fill in the requested information in the five fields. Enter your Book ID first, then your name and email address. You have to use your complete email address—for example, *claire@cs.nevada-state.edu*, not just *claire* or *claire@cs*—since the Grade Grinder will need the full address to send its response back to you. Also, if you have more than one email address, you have to use the same one every time you submit files, since your email address and Book ID together are how Grade Grinder will know that it is really you submitting files. Finally, fill in your instructor's name and complete email address. Be very careful to enter the correct and complete email addresses!
3. If you are working on your own computer, you might want to save the information you've just entered on your hard disk so that you won't have to enter it by hand each time. You can do this by choosing **Save As...** from the **File** menu. This will save all the information except the Book ID in a file called **Submit User Data**. Later, you can launch Submit by double-clicking on this file, and the information will already be entered when the program starts up.
4. We're now ready to specify the file to submit. Click on the button **Choose Files To Submit** in the lower-left corner. This opens a window showing two file lists. The list on the left shows files on your computer—currently, the ones inside the **Submit Folder**—while the one on the right (which is currently empty) will list files you want to

submit. We need to locate the file **World Submit Me 1** on the left and copy it over to the right. The file is located in the Tarski's World exercise files folder. To find this folder you will have to navigate among folders until it appears in the file list on the left. Start by clicking once on the **Submit Folder** button above the left-hand list. A menu will appear and you can then move up to higher folders by choosing their names (the higher folders appear lower on this menu). Move to the next folder up from the **Submit Folder**, which should be called **TW Software**. When you choose this folder, the list of files will change. On the new list, find the folder **Tarski's World Folder** and double-click on its name to see the contents of the folder. The list will again change and you should now be able to see the folder **TW Exercise Files**. Double-click on this folder and the file list will show the contents of this folder. Toward the bottom of the list (you will have to scroll down the list by clicking on the scroll buttons), you will find **World Submit Me 1**. Double-click on this file and its name will move to the list on the right.

5. When you have successfully gotten the file **World Submit Me 1** on the righthand list, click the **Done** button underneath the list. This should bring you back to the original Submit window, only now the file you want to submit appears in the list of files. (Macintosh users can get to this point quickly by dragging the files they want to submit onto the Submit icon in the Finder. This will launch Submit and put those files in the submission list. If you drag a folder of files, it will put all the files in the folder onto the list.)
6. When you have the correct file on the submission list, click on the **Submit Files** button under this list. Submit will ask you to confirm that you want to submit **World Submit Me 1**, and whether you want to send the results just to you or also to your instructor. In this case, select **Just Me**. When you are submitting finished homework exercises, you should select **Instructor Too**. Once you've chosen who the results should go to, click the **Proceed** button and your submission will be sent. (With real homework, you can always do a trial submission to see if you got the answers right, asking that the results be sent just to you. When you are satisfied with your solutions, submit the files again, asking that the results be sent to the instructor too. But don't forget the second submission!)



7. In a moment, you will get a dialog box that will tell you if your submission has been successful. If so, it will give you a “receipt” message that you can save, if you like. If you do not get this receipt, then your submission has not gone through and you will have to try again.
8. A few minutes after the Grade Grinder receives your file, you should get an email message saying that it has been received. If this were a real homework exercise, it would also tell you if the Grade Grinder found any errors in your homework solutions. You won’t get an email report if you put in the wrong, or a misspelled, email address. If you don’t get a report, try submitting again with the right address.
9. When you are done, choose **Quit** from the **File** menu. Congratulations on submitting your first file.

Here’s an important thing for you to know: when you submit files to the Grade Grinder, Submit sends a copy of the files. The original files are still on the disk where you originally saved them. If you saved them on a public computer, it is best not to leave them lying around. Put them on a floppy disk that you can take with you, and delete any copies from the public computer’s hard disk.

More detailed instructions on using Submit can be found in Chapter 6 on page 81.



---

## Exercises on Propositional Logic

The following three chapters contain many valuable exercises intended for both beginning and more advanced logic students. If you have already studied some logic, you may find some of the exercises quite easy. Still, it is a good idea to run through all the exercises, since they build on each other. Before trying any of these exercises, however, please read Chapter 1 and complete exercise 1.1 on page 4.

The exercises are ordered according to the complexity of the first-order sentences involved. Within this ordering, we use a series of stars ( $\star$ ,  $\star\star$ ,  $\star\star\star$ ) to indicate the difficulty of the problem. Exercises with no stars are the most basic. Those with three stars are quite challenging and provide good term projects for the interested student.

**Exercise 2.1** (Basic sentences) First-order logic assumes that every predicate is interpreted by a determinate property or relation. By a *determinate* property, we mean a property such that, given any object, there is always a fact of the matter about whether the object has the property or not. This exercise will help you see exactly how Tarski's World interprets the various predicates.

Open the files called Wittgenstein's World and Wittgenstein's Sentences. You will find these in the folder TW Exercise Files. In these files, you will see a blocks world and a list of atomic sentences. (We have added comments to some of the sentences. Comments are prefaced by a semicolon (“;”), which tells Tarski's World to ignore the rest of the line.)

1. Move through the sentences using the arrow keys on your keyboard, mentally assessing the truth value of each sentence in the given

world. Use the **Verify** button to check your assessments. (Since the sentences are all atomic sentences the **Game** button will not be helpful.) If you are surprised by any of the evaluations, try to figure out how your interpretation of the predicate differs from the correct interpretation. The correct interpretation is given in Table 2 on page 92, but try to work it out for yourself if you can.

2. Next change *Wittgenstein's World* in many different ways, seeing what happens to the truth of the various sentences. The main point of this is to help you figure out how Tarski's World interprets the various predicates. For example, what does *BackOf*(d, c) mean? Do two things have to be in the same column for one to be in back of the other?
3. Play around as much as you need until you are sure you understand the meanings of the atomic sentences in this file. For example, in the original world none of the sentences using *Adjoins* comes out true. You should try to modify the world to make some of them true. As you do this, you will notice that large blocks cannot adjoin other blocks.
4. In doing this exercise, you will no doubt notice that *Between* does not mean exactly what the English *between* means. This is due to the necessity of interpreting *Between* as a determinate predicate. For simplicity, we insist that in order for *b* to be between *c* and *d*, all three must be in the same row, column, or diagonal.
5. When you are finished, close the files, but do not save the changes you have made to them.

There is nothing to submit or turn in for this exercise.

✦ **Exercise 2.2** (Copying some sentences) The following are all well-formed sentences of our language. Start a new sentence file and copy them into it. Check each after you write it to see that it is a sentence. If you make a mistake, edit it before going on. Save your sentence list as *Sentences 2.2*. (If you have already played around with writing sentences and don't feel the need for this exercise, you can skip it. We will not use the sentence list you create.)

1.  $\text{Tet}(a)$
2.  $\text{FrontOf}(a, b)$
3.  $\neg \text{Between}(a, b, c)$
4.  $\text{Between}(a, b, c) \wedge \text{Between}(a, c, b)$

5.  $\text{FrontOf}(a, c) \rightarrow \text{Between}(d, e, c)$
6.  $(\text{Tet}(a) \wedge \text{FrontOf}(a, b)) \rightarrow \text{Between}(a, d, e)$

Submit the file that you have created.

✦ **Exercise 2.3** (Fixing some expressions) Most of the following are not quite well-formed sentences of our language. Start a new sentence file and copy them into it, adding whatever punctuation (parentheses and commas) is necessary to make them sentences. With some of them, there is more than one way to make them a sentence. Use **Verify** to make sure your entries are well-formed sentences. If you have any trouble with these, try referring to Section A.9, page 97.

1.  $\text{Cube}(a) \wedge \text{Cube}(b) \vee \text{Dodec}(b)$
2.  $\text{Tet}(a) \wedge \text{Small}(a) \rightarrow \text{BackOf}(a, b)$
3.  $\text{Cube}(c) \wedge \text{Small}(c) \wedge \text{LeftOf}(c, b)$
4.  $\text{Tet}(a) \rightarrow \text{Small}(a) \vee \text{Medium}(a)$
5.  $\text{Tet}(a) \leftrightarrow \text{Cube}(b) \leftrightarrow \text{Dodec}(c)$
6.  $\text{Between}(cba)$

Submit your sentence list as **Sentences 2.3**.

The next few exercises deal with sentences that can be built up using just the connectives  $\wedge$ ,  $\vee$ , and  $\neg$ . If you do not know what these connectives mean, read Section A.5, page 92.

✦ **Exercise 2.4** (Basic propositional logic) In this exercise you are asked to evaluate some sentences built up from atomic sentences using the propositional connectives  $\wedge$ ,  $\vee$ ,  $\neg$ . Run through **Boole's Sentences**, evaluating them in **Wittgenstein's World**. (If you made changes to **Wittgenstein's World** while doing Exercise 2.1, close the file and open it again to get back the original version. When it asks you if you want to save the changes you made, click **No**.) If you make a mistake, play the game to see where you have gone wrong. Don't go from one sentence to the next until you understand why it has the truth value it does. Do you see the importance of parentheses?

After you understand all of the sentences, go back and see which of the false sentences you can make true by adding, deleting or moving parentheses but without making any other changes. Submit your file as **Sentences 2.4**.

↗ **Exercise 2.5** (Building a world) Open *Quinn's Sentences*. Build a single world where all the sentences in this file are true. As you work through the sentences, you will find yourself successively modifying the world. Whenever you make a change in the world, be careful that you don't make one of your earlier sentences false. When you are finished, verify that all the sentences are really true. Submit your world as *World 2.5*.

↗ **Exercise 2.6** (Describing a simple world) Open *Boole's World*. Start a new sentence file, named *Sentences 2.6*, where you will describe some features of this world. Check each of your sentences to see that it is indeed a sentence and that it is true in this world.

1. Notice that *f* (the large dodecahedron in the back) is not in front of *a*. Use your first sentence to say this.
2. Notice that *f* is to the right of *a* and to the left of *b*. Use your second sentence to say this.
3. Use your third sentence to say that *f* is either in back of or smaller than *a*.
4. Express the fact that both *e* and *d* are between *c* and *a*.
5. Note that neither *e* nor *d* is larger than *c*. Use your fifth sentence to say this.
6. Notice that *e* is neither larger than nor smaller than *d*. Use your sixth sentence to say this.
7. Notice that *c* is smaller than *a* but larger than *e*. State this fact.
8. Note that *c* is in front of *f*; moreover, it is smaller than *f*. Use your eighth sentence to state these things.
9. Notice that *b* is in the same row as *a* but is not in the same column as *f*. Use your ninth sentence to express this fact.
10. Notice that *e* is not in the same column as either *c* or *d*. Use your tenth sentence to state this.

Now let's change the world so that none of the above mentioned facts hold. We can do this as follows. First move *f* to the front right corner of the grid. (Be careful not to drop it off the edge. You might find it easier to make the move from the 2-D view. If you accidentally drop it, just open *Boole's World* again.) Then move *e* to the back left corner of the grid and make it large. Now none of the facts hold; if your answers to 1–10 are correct, all of the sentences should now be false. Verify

that they are. If any are still true, can you figure out where you went wrong? Submit your sentences when you think they are correct. There is no need to submit the modified world file.

✦ **Exercise 2.7** (Some translations) Tarski's World provides you with a very useful way to check whether your translation of a given English sentence is correct. If it is correct, then it will always have the same truth value as the English sentence, no matter what world the two are evaluated in. So when you are in doubt about one of your translations, simply build some worlds where the English sentence is true, others where it is false, and check to see that your translation has the right truth values in these worlds. You should use this technique frequently in all of the translation exercises.

Start a new sentence file, and use it to enter translations of the following English sentences into first-order logic. You will only need to use the connectives  $\wedge$ ,  $\vee$ , and  $\neg$ .

1. *Either **a** is small or both **c** and **d** are large.*
2. ***d** and **e** are both in back of **b**.*
3. ***d** and **e** are both in back of **b** and larger than it.*
4. *Both **d** and **c** are cubes, however neither of them is small.*
5. *Neither **e** nor **a** is to the right of **c** and to the left of **b**.*
6. *Either **e** is not large or it is in back of **a**.*
7. ***c** is neither between **a** and **b**, nor in front of either of them.*
8. *Either both **a** and **e** are tetrahedra or both **a** and **f** are.*
9. *Neither **d** nor **c** is in front of either **c** or **b**.*
10. ***c** is either between **d** and **f** or smaller than both of them.*
11. *It is not the case that **b** is in the same row as **c**.*
12. ***b** is in the same column as **e**, which is in the same row as **d**, which in turn is in the same column as **a**.*

Before you submit your sentence file, do the next exercise.

**Exercise 2.8** (Checking your translations) Open Wittgenstein's World. Notice that all of the English sentences from Exercise 2.7 are true in this world. Thus, if your translations are accurate, they will also be true in this world. Check to see that they are. If you made any mistakes, go back and fix them. But as we have stressed, even if one of your sentences comes out true in Wittgenstein's World, it does not mean that

it is a proper translation of the corresponding English sentence. All you know for sure is that your translation and the original sentence have the same truth value in this particular world. If the translation is correct, it will have the same truth value as the English sentence in *every* world. Thus, to have a better test of your translations, we will examine them in a number of worlds, to see if they have the same truth values as their English counterparts in all of these worlds.

Let's start by making modifications to *Wittgenstein's World*. Make all the large or medium objects small, and the small objects large. With these changes in the world, the English sentences 1, 3, 4, and 10 become false, while the rest remain true. Verify that the same holds for your translations. If not, correct your translations. Next, rotate your modified *Wittgenstein's World* 90° clockwise. Now sentences 5, 6, 8, 9, and 11 should be the only true ones that remain.

Let's check your translations in another world. Open *Boole's World*. The only English sentences that are true in this world are sentences 6 and 11. Verify that all of your translations except 6 and 11 are false. If not, correct your translations.

Now modify *Boole's World* by exchanging the positions of *b* and *c*. With this change, the English sentences 2, 5, 6, 7, and 11 come out true, while the rest are false. Check that the same is true of your translations.

There is nothing to submit except **Sentences 2.7**.

The remaining exercises of this chapter use the full set of propositional connectives, including  $\rightarrow$  and  $\leftrightarrow$ . If you do not know what these symbols mean, read Section A.5, page 92.

✦ **Exercise 2.9** (Evaluating sentences in a world) Run through *Abelard's Sentences*, evaluating them in *Wittgenstein's World*. If you make a mistake, play the game to see where you have gone wrong. Once you have gone through all the sentences, go back and make all the false ones true by changing one or more names used in the sentence. Submit your edited sentences as **Sentences 2.9**.

✦ **Exercise\* 2.10** (Name that object) Open *Sherlock's World* and *Sherlock's Sentences*. You will notice that none of the objects in this world has a name. Your task is to assign the names *a*, *b*, and *c* in such a way that all the sentences in the list come out true. Submit the modified world as **World 2.10**.



✦ **Exercise 2.11** (Describing a world) Launch Tarski's World and choose **Hide Labels** from the **Display** menu. Then, with the labels hidden, open *Montague's World*. In this world, each object has a name, and no object has more than one name. Start a new sentence file where you will describe some features of this world. Check each of your sentences to see that it is indeed a sentence and that it is true in this world.

1. Notice that if **c** is a tetrahedron, then **a** is not a tetrahedron. (Remember, in this world each object has exactly one name.) Use your first sentence to express this fact.
2. However, note that the same is true of **b** and **d**. That is, if **b** is a tetrahedron, then **d** isn't. Use your second sentence to express this.
3. Observe that if **b** is a tetrahedron, then **c** isn't. Express this.
4. Notice that if **a** is a cube and **b** is a dodecahedron, then **a** is to the left of **b**. Use your next sentence to express this fact.
5. Use your next sentence to express the fact that if **b** and **c** are both cubes, then they are in the same row but not in the same column.
6. Use your next sentence to express the fact that **b** is a tetrahedron only if it is small. [Check this sentence carefully. If your sentence evaluates as false, then you've got the arrow pointing in the wrong direction.]
7. Next, express the fact that if **a** and **d** are both cubes, then one is to the left of the other. [Note: You will need to use a disjunction to express the fact that one is to the left of the other.]
8. Notice that **d** is a cube if and only if it is either medium or large. Express this.
9. Observe that if **b** is neither to the right nor left of **d**, then one of them is a tetrahedron. Express this observation.
10. Finally, express the fact that **b** and **c** are the same size if and only if one is a tetrahedron and the other is a dodecahedron.

Save your sentences as **Sentences 2.11**. Now choose **Show Labels** from the **Display** menu. Verify that all of your sentences are indeed true. When verifying the first three, pay particular attention to the truth values of the various constituents. Notice that sometimes the conditional has a false antecedent and sometimes a true consequent. What it never has is a true antecedent and a false consequent. In each of these three cases, play the game committed to true. Make sure you understand why the game proceeds as it does.

✦ **Exercise 2.12** (Translation) Translate the following English sentences into FOL. Your translations will use all of the propositional connectives.

1. *If  $\mathbf{a}$  is a tetrahedron then it is in front of  $\mathbf{d}$ .*
2.  *$\mathbf{a}$  is to the left of or right of  $\mathbf{d}$  only if it's a cube.*
3.  *$\mathbf{c}$  is between either  $\mathbf{a}$  and  $\mathbf{e}$  or  $\mathbf{a}$  and  $\mathbf{d}$ .*
4.  *$\mathbf{c}$  is to the right of  $\mathbf{a}$ , provided it (i.e.,  $\mathbf{c}$ ) is small.*
5.  *$\mathbf{c}$  is to the right of  $\mathbf{d}$  only if  $\mathbf{b}$  is to the right of  $\mathbf{c}$  and left of  $\mathbf{e}$ .*
6. *If  $\mathbf{e}$  is a tetrahedron, then it's to the right of  $\mathbf{b}$  if and only if it is also in front of  $\mathbf{b}$ .*
7. *If  $\mathbf{b}$  is a dodecahedron, then if it isn't in front of  $\mathbf{d}$  then it isn't in back of  $\mathbf{d}$  either.*
8.  *$\mathbf{c}$  is in back of  $\mathbf{a}$  but in front of  $\mathbf{e}$ .*
9.  *$\mathbf{e}$  is in front of  $\mathbf{d}$  unless it (i.e.,  $\mathbf{e}$ ) is a large tetrahedron.*
10. *At least one of  $\mathbf{a}$ ,  $\mathbf{c}$ , and  $\mathbf{e}$  is a cube.*
11.  *$\mathbf{a}$  is a tetrahedron only if it is in front of  $\mathbf{b}$ .*
12.  *$\mathbf{b}$  is larger than both  $\mathbf{a}$  and  $\mathbf{e}$ .*
13.  *$\mathbf{a}$  and  $\mathbf{e}$  are both larger than  $\mathbf{c}$ , but neither is large.*
14.  *$\mathbf{d}$  is the same shape as  $\mathbf{b}$  only if they are the same size.*
15.  *$\mathbf{a}$  is large if and only if it's a cube.*
16.  *$\mathbf{b}$  is a cube unless  $\mathbf{c}$  is a tetrahedron.*
17. *If  $\mathbf{e}$  isn't a cube, either  $\mathbf{b}$  or  $\mathbf{d}$  is large.*
18.  *$\mathbf{b}$  or  $\mathbf{d}$  is a cube if either  $\mathbf{a}$  or  $\mathbf{c}$  is a tetrahedron.*
19.  *$\mathbf{a}$  is large just in case  $\mathbf{d}$  is small.*
20.  *$\mathbf{a}$  is large just in case  $\mathbf{e}$  is.*

Save your list of sentences as `Sentences 2.12`. Before submitting the file, you should complete Exercise 2.14.

✦ **Exercise\* 2.13** (Building a world) Build a world in which all of the English sentences listed in Exercise 2.12 are true. Now make sure that all your translations are also true. If one of your translations is false, see whether the original English sentence is true. If it is, then there is something wrong with your translation. Play the game to try to figure out what the problem is. Submit your world as `World 2.13`. In order for us to grade your files, you must submit both `World 2.13` and `Sentences 2.12` at the same time.

**Exercise 2.14** (Checking your translations) Open **Bolzano's World**. Notice that all the English sentences from Exercise 2.12 are true in this world. Thus, if your translations are accurate, they will also be true in this world. Check to see that they are. If you made any mistakes, go back and fix them.

Remember that even if one of your sentences comes out true in **Bolzano's World**, it does not mean that it is a proper translation of the corresponding English sentence. If the translation is correct, it will have the same truth value as the English sentence in *every* world. So let's check your translations in some other worlds.

Open **Wittgenstein's World**. Here we see that the English sentences 3, 5, 9, 11, 12, 13, 14, and 20 are false, while the rest are true. Check to see that the same holds of your translations. If not, correct your translations (and make sure they are still true in **Bolzano's World**).

Next open **Leibniz's World**. Here half the English sentences are true (1, 2, 4, 6, 7, 10, 11, 14, 18, and 20) and half false (3, 5, 8, 9, 12, 13, 15, 16, 17, and 19). Check to see that the same holds of your translations. If not, correct your translations.

Finally, open **Venn's World**. In this world, all of the English sentences are false. Check to see that the same holds of your translations and correct them if necessary.

There is no need to submit any files for this exercise, but don't forget to submit **Sentences 2.12**.

✦ **Exercise\* 2.15** (Figuring out sizes and shapes) Start a new sentence file and use it to translate the following English sentences.

1. If  $\mathbf{a}$  is a tetrahedron, then  $\mathbf{b}$  is also a tetrahedron.
2.  $\mathbf{c}$  is a tetrahedron if  $\mathbf{b}$  is.
3.  $\mathbf{a}$  and  $\mathbf{c}$  are both tetrahedra only if at least one of them is large.
4.  $\mathbf{a}$  is a tetrahedron but  $\mathbf{c}$  isn't large.
5. If  $\mathbf{c}$  is small and  $\mathbf{d}$  is a dodecahedron, then  $\mathbf{d}$  is neither large nor small.
6.  $\mathbf{c}$  is medium only if none of  $\mathbf{d}$ ,  $\mathbf{e}$ , and  $\mathbf{f}$  are cubes.
7.  $\mathbf{d}$  is a small dodecahedron unless  $\mathbf{a}$  is small.
8.  $\mathbf{e}$  is large just in case it is a fact that  $\mathbf{d}$  is large if and only if  $\mathbf{f}$  is.
9.  $\mathbf{d}$  and  $\mathbf{e}$  are the same size.
10.  $\mathbf{d}$  and  $\mathbf{e}$  are the same shape.

11.  $f$  is either a cube or a dodecahedron, if it is large.

12.  $c$  is larger than  $e$  only if  $b$  is larger than  $c$ .

Save these sentences as Sentences 2.15. Then see if you can figure out the sizes and shapes of  $a$ ,  $b$ ,  $c$ ,  $d$ ,  $e$ , and  $f$ . You will find it helpful to approach this problem systematically, filling in the following table as you reason about the sentences:

	$a$	$b$	$c$	$d$	$e$	$f$
<i>Shape:</i>						
<i>Size:</i>						

When you have filled in the table, use it to guide you in building a world in which the twelve English sentences are true. Verify that your translations are true in this world as well. Submit both your sentence file and your world file.

✦ **Exercise 2.16** (Parentheses) Show that the sentence

$$\neg(\text{Small}(a) \vee \text{Small}(b))$$

is not a consequence of the sentence

$$\neg\text{Small}(a) \vee \text{Small}(b)$$

You will do this by submitting a counterexample world in which the second sentence is true but the first sentence is false.

✦ **Exercise 2.17** (More parentheses) Show that

$$\text{Cube}(a) \wedge (\text{Cube}(b) \vee \text{Cube}(c))$$

is not a consequence of the sentence

$$(\text{Cube}(a) \wedge \text{Cube}(b)) \vee \text{Cube}(c)$$

You will do this by submitting a counterexample world in which the second sentence is true but the first sentence is false.

The next few exercises exploit the propositional equivalences described in appendix A, section A.12 on page 103. They demonstrate that the language of propositional logic can be reduced by discarding some connectives, without affecting the expressive power of the language (but while sacrificing convenience.)

✦ **Exercise 2.18** (Redundancy of conditionals) The file *Gentzen's Sentences* contains sentences involving  $\leftrightarrow$  and  $\rightarrow$  in the odd-numbered slots. In the even numbered slots, write equivalent sentences using only the connectives  $\wedge$ ,  $\vee$  and  $\neg$ . Evaluate your sentences in *Bolzano's World*, *Boole's World* and *Wittgenstein's World* before submitting your sentences. The sentences in the even numbered slots should always have the same truth value as the sentence preceding them.

✦ **Exercise 2.19** (DeMorgan Equivalences) Open the file *DeMorgan's Sentences*. Construct a world where all the *odd* numbered sentences are true. Notice that no matter how you do this, the even numbered sentences also come out true. Submit the world as *World 2.19.1*. Next build a world where all the odd numbered sentences are *false*. Notice that no matter how you do it, the even numbered sentences also come out false. Submit this as *World 2.19.2*.

📎 **Exercise 2.20** (Explaining de Morgan) In Exercise 2.19, you noticed an important fact about the relation between the even and odd numbered sentences in *DeMorgan's Sentences*. Explain in terms of the meaning of the connectives why each even numbered sentence always has the same truth value as the odd numbered sentence that precedes it. Turn in your explanation.

✦ **Exercise 2.21** (Negation normal form) A sentence is in *negation normal form* (NNF) if all occurrences of  $\neg$  apply directly to atomic sentences. Any formula involving only the connectives  $\wedge$ ,  $\vee$  and  $\neg$  can be put into negation normal form using the equivalences (our use of the symbol  $\Leftrightarrow$  is explained in section A.12 on page 103):

$$\begin{aligned}\neg\neg A &\Leftrightarrow A \\ \neg(A \wedge B) &\Leftrightarrow \neg A \vee \neg B \\ \neg(A \vee B) &\Leftrightarrow \neg A \wedge \neg B\end{aligned}$$

Open *Turing's Sentences*. You will find the following five sentences, each followed by an empty sentence position.

1.  $\neg(\text{Cube}(a) \wedge \text{Larger}(a, b))$
3.  $\neg(\text{Cube}(a) \vee \neg\text{Larger}(b, a))$
5.  $\neg(\neg\text{Cube}(a) \vee \neg\text{Larger}(a, b) \vee a \neq b)$
7.  $\neg(\text{Tet}(b) \vee (\text{Large}(c) \wedge \neg\text{Smaller}(d, e)))$
9.  $\text{Dodec}(f) \vee \neg(\text{Tet}(b) \vee \neg\text{Tet}(f) \vee \neg\text{Dodec}(f))$

In the empty positions, write the negation normal form of the sentence above it. Then build any world where all of the names are in use. If you have gotten the negation normal forms correct, each even numbered sentence will have the same truth value in your world as the odd numbered sentence above it. Verify that this is so in your world. Submit the modified sentence file as **Sentences 2.21**.

✦ **Exercise 2.22** (Converting CNF to DNF) A sentence is in *disjunctive normal form* (DNF) if it is the disjunction of one or more conjunctions of one or more literals. Distribution of  $\wedge$  over  $\vee$  allows you to translate any sentence in negation normal form into a sentence in disjunctive normal form.

$$P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$$

Similarly, a sentence is in *conjunctive normal form* (CNF) if it is the conjunction of one or more disjunctions of one or more literals. Distribution of  $\vee$  over  $\wedge$  allows you to translate any sentence in negation normal form into a sentence in conjunctive normal form.

$$P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$$

**Open CNF Sentences.** In this file you will find the following conjunctive normal form sentences in the odd numbered positions, but you will see that the even numbered positions are blank.

1.  $(\text{LeftOf}(a, b) \vee \text{BackOf}(a, b)) \wedge \text{Cube}(a)$
3.  $\text{Larger}(a, b) \wedge (\text{Cube}(a) \vee \text{Tet}(a) \vee a = b)$
5.  $(\text{Between}(a, b, c) \vee \text{Tet}(a) \vee \neg \text{Tet}(b)) \wedge \text{Dodec}(c)$
7.  $\text{Cube}(a) \wedge \text{Cube}(b) \wedge (\neg \text{Small}(a) \vee \neg \text{Small}(b))$
9.  $(\text{Small}(a) \vee \text{Medium}(a)) \wedge (\text{Cube}(a) \vee \neg \text{Dodec}(a))$

In the even numbered positions you should fill in a DNF sentence logically equivalent to the sentence above it. Check your work by opening several worlds and checking to see that each of your sentences has the same truth value as the one above it. Submit the modified file as **Sentences 2.22**.

✦ **Exercise 2.23** (Converting to CNF via NNF) **Open More CNF Sentences.** In this file you will find the following sentences in every third position.


1.  $\neg[(\text{Cube}(a) \wedge \neg\text{Small}(a)) \vee (\neg\text{Cube}(a) \wedge \text{Small}(a))]$
4.  $\neg[(\text{Cube}(a) \vee \neg\text{Small}(a)) \wedge (\neg\text{Cube}(a) \vee \text{Small}(a))]$
7.  $\neg(\text{Cube}(a) \wedge \text{Larger}(a, b)) \wedge \text{Dodec}(b)$
10.  $\neg(\neg\text{Cube}(a) \wedge \text{Tet}(b))$
13.  $\neg\neg\text{Cube}(a) \vee \text{Tet}(b)$

The two blanks that follow each sentence are for you to first transform the sentence into negation normal form, and then put that sentence into CNF. Again, check your work by opening several worlds to see that each of your sentences has the same truth value as the original. When you are finished, submit the modified file as **Sentences 2.23**.

✦ **Exercise 2.24** (Elimination of  $\vee$ ) Each of the sentences in **Gentzen's Other Sentences** involve the use of both  $\wedge$  and  $\vee$ . In each even numbered position write a sentence equivalent to the one above it, but that uses only  $\wedge$  and  $\neg$ . Before submitting your work, check that the truth values of the pairs of sentences agree in a number of worlds.

✦ **Exercise 2.25** (Elimination of  $\wedge$ ) This exercise is the same as the previous one, except that here we ask you to write a sentence that uses only  $\vee$  and  $\neg$  and is equivalent to the one above it.

✦ **Exercise 2.26** (Equivalences in the blocks language) In the blocks language used in Tarski's World there are a number of equivalent ways of expressing some of the predicates. Open **Bernays' Sentences**. You will find a list of atomic sentences, where every other sentence is left blank. In each blank, write a sentence that is equivalent to the sentence above it, but does not use the predicate used in that sentence. (In doing this, you may presuppose any general facts about Tarski's World, for example that blocks come in only three shapes.) If your answers are correct, the odd numbered sentences will have the same truth values as the even numbered sentences in every world. Check that they do in **Ackermann's World**, **Bolzano's World**, **Boole's World**, and **Leibniz's World**. Submit the modified sentence file as **Sentences 2.26**.

✦  **Exercise 2.27** (Context sensitivity of predicates) We have stressed the fact that FOL assumes that every predicate is interpreted by a determinate relation, whereas this is not the case in natural languages like English. Indeed, even when things seem quite determinate,

there is often some form of context sensitivity. In fact, we have built some of this into Tarski's World. Consider, for example, the difference between the predicates **Larger** and **BackOf**. Whether or not cube **a** is larger than cube **b** is a determinate matter, and also one that does not vary depending on your perspective on the world. Whether or not **a** is back of **b** is also determinate, but in this case it does depend on your perspective. If you rotate the world by  $90^\circ$ , the answer might change.

Open Austin's Sentences and Wittgenstein's World. Evaluate the sentences in this file and tabulate the resulting truth values in a table like the one below. We've already filled in the first column, showing the values in the original world. Rotate the world  $90^\circ$  clockwise and evaluate the sentences again, adding the results to the table. Repeat until the world has come full circle.

	Original	Rotated $90^\circ$	Rotated $180^\circ$	Rotated $270^\circ$
1.	FALSE			
2.	FALSE			
3.	TRUE			
4.	FALSE			
5.	TRUE			
6.	FALSE			

You should be able to think of an atomic sentence in the blocks language that would produce a row across the table with the following pattern:

TRUE   FALSE   TRUE   FALSE

Add a seventh sentence to **Austin's Sentences** that would display the above pattern.

Are there any atomic sentences in the language that would produce a row with this pattern?

FALSE   TRUE   FALSE   FALSE

If so, add such a sentence as sentence eight in **Austin's Sentences**. If not, leave sentence eight blank.

Are there any atomic sentences that would produce a row in the table containing exactly three **TRUE**'s? If so, add such a sentence as number nine. If not, leave sentence nine blank.

Submit your modified sentence file as **Sentences 2.27**. Turn in your completed table to your instructor.



↗ **Exercise 2.28** (Name that object) Open Rebus' Sentences and Rebus' World. Modify the world by assigning names to the blocks in such a way that the sentences are all true. Submit your world.



### 3

---

## Exercises on First-order Logic

The first exercises in this chapter involve sentences that contain a single instance of one of the quantifier symbols,  $\exists$  and  $\forall$ , and propositional combinations of such sentences. If you are not familiar with these symbols, see Section A.8, page 96.

✦ **Exercise 3.1** (Evaluating sentences in a world) Open *Peirce's World* and *Peirce's Sentences*. There are 30 sentences in this file. Work through them, assessing their truth and playing the game when necessary. Make sure you understand why they have the truth values they do. (You may need to switch to the 2-D view for some of the sentences.) After you understand each of the sentences, go back and make the false ones true by adding or deleting a negation sign. Submit the file when the sentences all come out true in *Peirce's World*.

✦ **Exercise 3.2** (Building a world) Open *Aristotle's Sentences*. Each of these sentences is of one of the four forms treated in Aristotle's logic:

All A's are B's  
No A's are B's  
Some A's are B's  
Some A's are not B's

Build a single world where all the sentences in the file are true. As you work through the sentences, you will find yourself successively modifying the world. Whenever you make a change in the world, you had better go back and check that you haven't made any of the earlier sentences false. Then, when you are finished, verify that all the sentences are really true. Save your world as *World 3.2*.

✦ **Exercise 3.3** (Fixing some expressions) Open the sentence file *Bernstein's Sentences*. The expressions in this list are not quite well-formed sentences of our language, but they can all be made sentences by slight modification. Turn them into sentences *without adding or deleting any quantifier symbols*. With some of them, there is more than one way to make them a sentence. Use **Verify** to make sure your results are sentences and then submit the corrected file.

✦ **Exercise 3.4** (Fixing some more expressions) Open the sentence file *Schönfinkel's Sentences*. Again, the expressions in this list are not well-formed sentences. Turn them into sentences, but this time, do it *only* by adding quantifier symbols or variables, or both. Do not add any parentheses. Use **Verify** to make sure your results are sentences and submit the corrected file.

✦ **Exercise\* 3.5** (Name that object) Open *Maigret's World* and *Maigret's Sentences*. The goal is to try to figure out which objects have names, and what they are. You should be able to figure this out from the sentences, all of which are true. Once you have come to your conclusion, assign the six names to objects in the world in such a way that all the sentences do indeed evaluate as true. Submit your modified world.

☞ **Exercise 3.6** (A common translation mistake) When we get around to translating English sentences containing quantifiers, we will see that sentences of the following forms are translated in quite different ways:

All A's are B's  
Some A's are B's

The former are translated as:

$$\forall x (A(x) \rightarrow B(x))$$

whereas the latter are translated as:

$$\exists x (A(x) \wedge B(x))$$

Beginning students are often tempted to translate the latter more like the former, say as:

$$\exists x (A(x) \rightarrow B(x))$$

This is in fact an extremely unnatural sentence of first-order logic. It is meaningful, but it doesn't mean what you might think. This exercise is designed to show you exactly what a sentence of this form means.

Open *Edgar's Sentences* and evaluate them in *Edgar's World*. Make sure you understand why each of them has the truth value it does. Play the game if any of the evaluations surprise you. Which of these sentences would be a good translation of *There is a tetrahedron that is large*? (Clearly this English sentence is false in *Edgar's World*, since there are no tetrahedra at all.) Which sentence would be a good translation of *There is a cube between **a** and **b***? Which would be a good translation of *There is a large dodecahedron*? Express in clear English the claim made by each sentence in the file and turn in your answers to your instructor.

✦ **Exercise 3.7** (Fixing ungrammatical expressions) Open *Bozo's Sentences 1* and *Leibniz's World*. Some of the expressions in this file are not wffs, some are wffs but not sentences, and one is a sentence but false. Read and assess each one. See if you can adjust each one to make it a true sentence with as little change as possible. Try to capture the intent of the original expression, if you can tell what that was (if not, don't worry). Use **Verify** to make sure your results are true sentences and then submit your file.

✦ **Exercise 3.8** (Fixing ungrammatical expressions) Open *Bozo's Sentences 2* and *Leibniz's World*. Most of the expressions in this file are not sentences. Some are not wffs, while others are wffs but not sentences. Read and assess each one. If it is not a wff, fix it. If it is not a sentence, adjust it so as to make it a true sentence with as little change as possible. If it is a false sentence, try to make it true, again with as little change as possible. See if you can capture the intent of the original expression. Save your list of sentences as *Sentences 3.8*.

✦ **Exercise 3.9** (Describing a world) Open *Reichenbach's World 1*. Start a new sentence file where you will describe some features of this world using sentences of the simple Aristotelian forms. Check each of your sentences to see that it is indeed a sentence and is true in this world.

1. Use your first sentence to describe the size of all the tetrahedra.
2. Use your second sentence to describe the size of all the cubes.
3. Use your third sentence to express the truism that every dodecahedron is either small, medium, or large.

4. Notice that some dodecahedron is large. Express this fact.
5. Observe that some dodecahedron is not large. Express this.
6. Notice that some dodecahedron is small. Express this fact.
7. Observe that some dodecahedron is not small. Express this.
8. Notice that some dodecahedron is neither large nor small. Express this.
9. Express the observation that no tetrahedron is large.
10. Express the fact that no cube is large.

Save your list of sentences as **Sentences 3.9**. Now change the sizes of the objects in the following way: make one of the cubes large, one of the tetrahedra medium, and all the dodecahedra small. With these changes, the following should come out false: 1, 2, 4, 7, 8 and 10. If not, then you have made an error in describing the original world. Can you figure out what it is? Try making other changes and see if your sentences have the expected truth values.

✦ **Exercise 3.10** (Translating existential noun phrases) The first thing you have to learn in order to translate quantified expressions is how to treat complex noun phrases, expressions like “some boy living in Omaha” or “every girl living in Duluth.” In this exercise we will concentrate on the former sort of noun phrase, those whose most natural translation involves an existential quantifier. Typically, these will be noun phrases starting with one of the determiners “some,” “a,” and “an,” including noun phrases like “something.”

- Start a new sentence file and enter translations of the following English sentences. Each will use the symbol  $\exists$  exactly once. None will use the symbol  $\forall$ . As you go, check that your entries are well-formed sentences. By the way, you will find that many of these English sentences are translated using the same first-order sentence.
  1. *Something is large.*
  2. *Something is a cube.*
  3. *Something is a large cube.*
  4. *Some cube is large.* [Hint: This sentence means the same thing as *Something is both a cube and large.*]
  5. *Some large cube is to the left of **b**.*
  6. *A large cube is to the left of **b**.*
  7. ***b** has a large cube to its left.*

8. ***b** is to the right of a large cube.* [Hint: This translation should be almost the same as the last, but it should contain the predicate symbol `RightOf`.]
9. *Something to the left of **b** is in back of **c**.*
10. *A large cube to the left of **b** is in back of **c**.*
11. *Some large cube is to the left of **b** and in back of **c**.*
12. *Some dodecahedron is not large.*
13. *Something is not a large dodecahedron.*
14. *It is not the case that something is a large dodecahedron.*
15. ***b** is not to the left of a cube.* [Warning: This sentence is ambiguous. Can you think of two importantly different translations? One starts with  $\exists$ , the other starts with  $\neg$ . Use the second of these for your translation, since this is the most natural reading of the English sentence.]

Save your list of sentences as **Sentences 3.10**.

- **Open Montague's World.** Notice that all the English sentences above are true in this world. Check that all your translations are also true. If not, you have made a mistake. Can you figure out what is wrong with your translation?
- **Move the large cube to the back right corner of the grid.** Observe that English sentences 5, 6, 7, 8, 10, 11 and 15 are now false, while the rest are still true. Check that the same holds of your translations. If not, you have made a mistake. Can you figure out what is wrong with your translation?
- **Now make the large cube small.** The English sentences 1, 3, 4, 5, 6, 7, 8, 10, 11, and 15 are false in the modified world, the rest are true. Again, check that your translations have the same truth values. If not, figure out what is wrong.
- **Finally, move **c** straight back to the back row, and make **b** large.** All the English sentences other than 1, 2, and 13 are false. Check that the same holds for your translations. If not, figure out where you have gone wrong.

✦📖 **Exercise 3.11** (Common mistakes, part 2) In this exercise we return to the point made in Exercise 3.6, page 26. Glance back at that exercise to recall the basic point. Now open *Allan's Sentences*. In this file, sentences 1 and 4 are the correct translations of *Some dodecahedron*

*is large* and *All tetrahedra are small*, respectively. Let's investigate the logical relations between these and sentences 2 and 3.

1. Construct a world in which sentences 2 and 4 are true, but sentences 1 and 3 are false. Save it as **World 3.11.1**. This shows that sentence 1 is not a consequence of 2, and sentence 3 is not a consequence of 4.
2. Can you construct a world in which sentence 3 is true and sentence 4 is false? If so, do so and save it as **World 3.11.2**. If not, explain why you can't and what this shows.
3. Can you construct a world in which sentence 1 is true and sentence 2 is false? If so, do so and save it as **World 3.11.3**. If not, explain why not.

Submit any world files you constructed and turn in any explanations to your instructor.

✦ **Exercise 3.12** (Translating universal noun phrases) Universal noun phrases are those that begin with determiners like “every,” “each,” and “all.” These are usually translated with the universal quantifier. Sometimes noun phrases beginning with “no” and with “any” are also translated with the universal quantifier.

Start a new sentence file, and enter translations of the following sentences. This time each translation will contain exactly one  $\forall$  and no  $\exists$ .

1. *All cubes are small.*
2. *Each small cube is to the right of **a**.*
3. ***a** is to the left of every dodecahedron.*
4. *Every medium tetrahedron is in front of **b**.*
5. *Each cube is either in front of **b** or in back of **a**.*
6. *Every cube is to the right of **a** and to the left of **b**.*
7. *Everything between **a** and **b** is a cube.*
8. *Everything smaller than **a** is a cube.*
9. *All dodecahedra are not small.* [Note: Most people find this sentence ambiguous. Can you find both readings? One starts with  $\forall$ , the other with  $\neg$ . Use the former, the one that means all the dodecahedra are either medium or large.]
10. *No dodecahedron is small.*



11. *a does not adjoin everything.* [Note: This sentence is ambiguous. We want you to interpret it as a denial of the claim that *a* adjoins everything.]
12. *a does not adjoin anything.* [Note: These last two sentences mean different things, though they can both be translated using  $\forall$ ,  $\neg$ , and Adjoins.]
13. *a is not to the right of any cube.*
14. ( $\star$ ) *If something is a cube, then it is not in the same column as either a or b.* [Warning: While this sentence contains the noun phrase “something,” it is actually making a universal claim, and so should be translated with  $\forall$ . You might first try to paraphrase it using the English phrase “every cube.”]
15. ( $\star$ ) *Something is a cube if and only if it is not in the same column as either a or b.*

Now let’s check the translations in some worlds.

- Open *Claire’s World*. Check to see that all the English sentences are true in this world, then make sure the same holds of your translations. If you have made any mistakes, fix them.
- Adjust *Claire’s World* by moving *a* directly in front of *c*. With this change, the English sentences 2, 6, and 12–15 are false, while the rest are true. Make sure that the same holds of your translations. If not, try to figure out what is wrong and fix it.
- Next, open *Wittgenstein’s World*. Observe that the English sentences 2, 3, 7, 8, 11, 12, and 13 are true, but the rest are false. Check that the same holds for your translations. If not, try to fix them.
- Finally, open *Venn’s World*. English sentences 2, 4, 7, and 11–14 are true; does the same hold for your translations?

When you are satisfied that your translations are correct, submit your sentence file.

↗ **Exercise 3.13** (Translation) Open *Leibniz’s World*. This time, we will translate some sentences while looking at the world they are meant to describe.

- Start a new sentence file, and enter translations of the following sentences. Each of the English sentences is true in this world. As you go, check to make sure that your translation is indeed a true sentence.

1. *There are no medium-sized cubes.*
  2. *Nothing is in front of **b**.*
  3. *Every cube is either in front of or in back of **e**.*
  4. *No cube is between **a** and **c**.*
  5. *Everything is in the same column as **a**, **b**, or **c**.*
- Now let's change the world so that none of the English sentences is true. We can do this as follows. First change **b** into a medium cube. Next, delete the leftmost tetrahedron and move **b** to exactly the position just vacated by the late tetrahedron. Finally, add a small cube to the world, locating it exactly where **b** used to sit. If your answers to 1–5 are correct, all of the translations should now be false. Verify that they are.
  - Make various changes to the world, so that some of the English sentences come out true and some come out false. Then check to see that the truth values of your translations track the truth values of the English sentences.

So far, most of the sentences we have looked at have had at most one quantifier. In the next few exercises, we delve into sentences that contain more than one instance of  $\forall$ , or more than one instance of  $\exists$ .

✦ **Exercise 3.14** (Vacuously true generalizations) Open **Dodgson's Sentences**. Note that the first sentence says that every tetrahedron is large.

1. **Open Peano's World**. Sentence 1 is clearly false in this world, since the small tetrahedron is a “counterexample” to the universal claim. What this means is that if you play the game committed to the falsity of this claim, then when Tarski's World asks you to pick an object you will be able to pick the small tetrahedron and win the game. Try this.
2. Delete this counterexample and verify that sentence 1 is now true.
3. Now open **Peirce's World**. Verify that sentence 1 is again false, this time because there are three counterexamples. (Now if you play the game committed to the falsity of the sentence, you will have three different winning moves when asked to pick an object: you can pick any of the small tetrahedra and win.)
4. Delete all three counterexamples, and evaluate the claim. Is the result what you expected? The generalization is true, because there

are no counterexamples to it. But it is what we call a *vacuously* true generalization, since there are no objects that satisfy the antecedent. That is, there are no tetrahedra at all, small, medium, or large.

5. Confirm that all of sentences 1–3 are vacuously true in the current world.
6. Two more vacuously true sentences are given in sentences 4 and 5. However, these sentences are different in another respect. Each of the first three sentences could have been non-vacuously true in a world, but these latter two can only be true in worlds containing no tetrahedra. That is, the only way they can be true is to be vacuously true. Let's call generalizations with this property “inherently vacuous.” Thus a sentence of the form  $\forall x (A(x) \rightarrow B(x))$  is inherently vacuous if any world in which it is true is also a world in which  $\forall x \neg A(x)$  is true.
7. Add a sixth generalization to the file that is vacuously true in Peirce's World but non-vacuously true in Peano's World. (In both cases, make sure you use the unmodified worlds.) Save your new sentence file as Sentences 3.14.

In everyday conversation, it is rare to encounter a vacuously true generalization. When we do, we feel that the speaker has misled us. For example, suppose a professor claims “Every freshman who took the class got an A,” when in fact no freshman took her class. Here we wouldn't say that she lied, but we would certainly say that she misled us. Her claim suggests that there were freshman in the class, and if there were no freshman, then that's what she would have said if she were being forthright. This is why *inherently* vacuous claims like sentence 5 strike us as counterintuitive: we can see that they cannot be true without being misleading.

✦ **Exercise 3.15** (Evaluating multiple quantifier sentences) Open up Peano's World and Peano's Sentences. The sentence file contains 30 assertions that Alex made about this world. Evaluate Alex's claims. If you have trouble with any, play the game (several times if necessary) until you see where you are going wrong. Then change each of Alex's false claims into a true claim. If you can make the sentence true by adding a clause of the form  $x \neq y$ , do so. Otherwise, see if you can turn the false claim into an interesting truth: don't just add a negation sign to the front of the sentence. Submit your corrected list of sentences.

✦ **Exercise\* 3.16** (Building a world) *Open Ramsey's Sentences*. Build a world in which sentences 1–10 are all true at once. These sentences all make either “particular” claims (that is, they contain no quantifiers) or “existential” claims (that is, they assert that things of a certain sort exist). Consequently, you could make them true by successively adding objects to the world. But part of the exercise is to make them all true *with as few objects as possible*. You should be able to do it with six objects, total. So rather than adding objects for each new sentence, try adjusting the world, and only add new objects when necessary. Again, be sure to go back and check that all the sentences are true when you are finished. [Hint: To make all the sentences true with this small a world, one of the objects will have to have two names.] Save your world as *World 3.16*.

✦ **Exercise 3.17** (Modifying the world) Sentences 11–20 of *Ramsey's Sentences* all make “universal” claims. That is, they all say that every object in the world has some property or other. Check to see whether *World 3.16* satisfies the universal claims expressed by these sentences. If not, modify it so it makes all 20 sentences true at once. Save the modified world as *World 3.17*.

✦ **Exercise 3.18** (Expanding a world) In the real world, things change in various ways. They come, move around, and go. And as things change, so do the truth values of sentences.

- In this exercise, the goal is to change *World 3.17* to make as many of Ramsey's sentences false as you can. But here's the catch: you can only add objects of various sizes and shapes; don't change the existing objects in any way. Save your world as *World 3.18*.
- (★) Do you notice anything about which sentences you can make false in this way and which you cannot? Try to give a fairly clear and intuitive account of which sentences you cannot make false in this way. We will return to this topic in Exercise 4.14, page 60.

In order for us to grade your files, you must submit both *World 3.17* and *World 3.18* at the same time.

✦ **Exercise 3.19** (Simple multiple quantifier sentences) The file *Frege's Sentences* contains 14 sentences; the first seven begin with a pair of existential quantifiers, the second seven with a pair of universal quantifiers. Go through the sentences one by one, evaluating them in *Peirce's World*. Though you probably won't have any trouble

understanding these sentences, don't forget to use the game if you do. When you understand all the sentences, modify the size and location of a single block so that the first seven sentences are true and the second seven false. Submit the resulting world.

Now that you have plenty of experience with quantifiers, we present exercises in which both universal and existential quantifiers get mixed together.

✦ **Exercise 3.20** (Mixed quantifier sentences with identity) Open *Leibniz's World* and use it to evaluate the sentences in *Leibniz's Sentences*. Make sure you understand all the sentences and follow any instructions in the file. Submit your modified sentence list.

✦ **Exercise 3.21** (Building a world) Open *Buridan's Sentences*. Build a world in which all ten sentences are true. Submit your world.

✦ **Exercise 3.22** (Consequence) These two English sentences are consequences of the ten sentences in *Buridan's Sentences*.

1. *There are no cubes.*
2. *There is exactly one large tetrahedron.*

Because of this, they must be true in any world in which *Buridan's Sentences* are all true. So of course they must be true in *World 3.21*, no matter how you built it.

- Translate the two sentences, adding them to the list in *Buridan's Sentences*. Name the expanded list *Sentences 3.22*. Verify that they are all true in *World 3.21*.
- Modify the world by adding a cube. Try placing it at various locations and giving it various sizes to see what happens to the truth values of the sentences in your file. One or more of the original ten sentences will always be false, though different ones at different times. Find a world in which only one of the original ten sentences is false and name it *World 3.22.1*.
- Next, get rid of the cube and add a second large tetrahedron. Again, move it around and see what happens to the truth values of the sentences. Find a world in which only one of the original ten sentences is false and name it *World 3.22.2*.

Submit your sentence file and two world files.

↗ **Exercise 3.23** (Independence) Show that the following sentence is independent of those in *Buridan's Sentences*, that is, neither it nor its negation is a consequence of those sentences.

$$\exists x \exists y (x \neq y \wedge \text{Tet}(x) \wedge \text{Tet}(y) \wedge \text{Medium}(x) \wedge \text{Medium}(y))$$

You will do this by building two worlds, one in which this sentence is false (call this *World 3.23.1*) and one in which it is true (*World 3.23.2*)—but both of which make all of *Buridan's sentences* true.

↗ **Exercise 3.24** (Simple mixed quantifier sentences) Open *Hilbert's Sentences* and *Peano's World*. Evaluate the sentences one by one, playing the game if an evaluation surprises you. Once you understand the sentences, modify the false ones by adding a single negation sign so that they come out true. The catch is that you aren't allowed to add the negation sign to the front of the sentence! Add it to an atomic formula, if possible, and try to make the claim nonvacuously true. (This won't always be possible.) Make sure you understand both why the original sentence is false and why your modified sentence is true. When you're done, submit your sentence list with the changes.

↗ **Exercise 3.25** (It's a small world after all) Create a world *containing at most three objects* in which the nine sentences in *Ockham's Sentences* are all true. Save this world as *World 3.25*. We will be using it later.

↗ **Exercise 3.26** (Building a world) Create a world in which all ten sentences in *Arnault's Sentences* are true. Save your world as *World 3.26*.

↗ **Exercise\* 3.27** (Numerical sentences) By a “numerical claim” we mean one that says that there are a certain number of objects, or a certain number with some property or other. In earlier exercises, we have already come across some simple numerical claims. This exercise will help you recognize numerical claims when you come across them in our first-order language. Open *Whitehead's Sentences*.

1. The first sentence says that there are at least two objects, and the second sentence says that there are at most two objects. (Do you see how they manage to say these things?) Build a world where the first two sentences are both true.

2. Sentence 3 is the conjunction of the first two. Hence it asserts, in one sentence, that there are exactly two objects. Check to see that it is true in the world you have just built.
3. The fourth sentence is in fact equivalent to the third sentence. It is a shorter way of saying that there are exactly two objects. Use the game to see why it is true in a world where there are two objects, but false in worlds with more or less than two objects.
4. Sentence 5 appears, at first sight, to assert that there are at least three objects, so it should be false in a world with two objects. Check to see if it is indeed false in such a world. Why isn't it? Play the game to confirm your suspicions.
5. The sixth sentence actually manages to express the claim that there are at least three objects. Do you see how it's different from the fifth sentence? Check to see that it is false in the current world, but is true if you add another object to the world.
6. The seventh sentence says that there are exactly three objects in the world. Check to see that it is true in the world with three objects, but false if you either delete an object or add another object.
7. Sentence 8 asserts that *a* is a large object, and in fact the *only* large object. To see just how the sentence manages to say this, start with a world with three small objects and name one of them *a*. Play the game committed to true to see why the sentence is false. Now make object *a* large. Play the game committed to false to see why it is true. Finally, make one of the other objects large as well, and play the game committed to true to see why it is false.
8. Sentence 8 asserted that *a* was the only large object. How might we say that there is exactly one large object, without using a name for the object? Compare sentence 8 with sentence 9. The latter asserts that there is something which is the only large object. Check to see that it is true only in worlds in which there is exactly one large object.
9. Construct a world in which sentence 10 is true. Save your world as World 3.27.1.
10. Make sentences 11 and 12 true in a single world. Save your world as World 3.27.2.
11. Sentence 13 is another way to assert that there is a unique dodecahedron. That is, sentence 13 is equivalent to sentence 10. Can you

see why? Check three worlds to see that the two sentences are true in the same worlds—those in which there is a single dodecahedron.

12. Sentence 14 says that there are exactly two tetrahedra. Check that it is true in such worlds, but false if there are fewer or more than two.

✦ **Exercise\*\* 3.28** (The Russellian analysis of definite descriptions) First-order logic has only two quantifiers, whereas English has many determiners, words like “some” and “every,” that combine with nouns to produce noun phrases (like “some cube,” “every cube”). Other determiners include numbers (as in “two cubes”) and the definite article “the” (as in “the cube”). Bertrand Russell proposed that a sentence like *The cube is small* should be analyzed as asserting that there is exactly one cube, and it is small. According to this analysis, the sentence will be false if there is no cube, or if there is more than one, or if there is exactly one, but it’s not small. If this analysis is correct (and many do not think it is), then such sentences can easily be expressed in first-order logic.

1. In exercise 3.27 on page 36 we saw two ways for saying that there is a single dodecahedron (sentences 10 and 13). Open *Russell’s Sentences*, the first sentence here uses the second method for asserting that there is a single cube. Compare sentence 1 with sentence 2. Sentence 2 is the Russellian analysis of our sentence *The cube is small*. Construct a world in which sentence 2 is true.
2. Construct a world in which sentences 2-7 are all true. (Sentence 7 contains the Russellian analysis of *The small dodecahedron is to the left of the medium dodecahedron*.)

Submit your world.

✦ **Exercise 3.29** (Describing a world) Open *Peano’s World*. Start a new sentence file where you will describe some features of this world. Again, be sure to check each of your sentences to see that it is indeed a sentence and is true.

1. Notice that every dodecahedron is small. Use your first sentence to say this.
2. State the fact that there is a medium sized cube.
3. Next, assert that there are at least two cubes.



4. Express the fact that there is a tetrahedron between two dodecahedra.
5. Notice that it is not the case that every cube is in front of a dodecahedron. Say this.

Save your list of sentences as **Sentences 3.29**. Now let's change the world so that none of the above facts hold. We can do this by first changing the medium cube into a dodecahedron, and then moving the leftmost dodecahedron to the front row. If your answers to 1-5 are correct, all of the sentences should now be false.

✦ **Exercise 3.30** (Translating mixed quantifier sentences) When an English sentence contains more than one quantified noun phrase, translating it can seem quite confusing unless it is approached in a very systematic manner. It often helps to have a number of intermediate steps, where quantified noun phrases are treated one at a time. For example, suppose we wanted to translate the sentence *Each cube is to the left of a tetrahedron*. Here, there are two quantified noun phrases: *each cube* and *a tetrahedron*. We can start by dealing with the first noun phrase, temporarily treating the complex phrase *is-to-the-left-of-a-tetrahedron* as a single unit. In other words, we can think of the sentence as a single quantifier sentence, on the order of *Each cube is small*. The translation would look like this:

$$\forall x (\text{Cube}(x) \rightarrow x\text{-is-to-the-left-of-a-tetrahedron})$$

Of course, this is not a sentence in our language, so we need to translate the expression *x-is-to-the-left-of-a-tetrahedron*. But we can think of this expression as a single quantifier sentence, at least if we pretend that *x* is a name. It has the same general form as the sentence *b is to the left of a tetrahedron*, and would be translated as:

$$\exists y (\text{Tet}(y) \wedge \text{LeftOf}(x, y))$$

Substituting this in the above, we get the desired translation of the original English sentence:

$$\forall x (\text{Cube}(x) \rightarrow \exists y (\text{Tet}(y) \wedge \text{LeftOf}(x, y)))$$

This multi-step process usually makes translation of multiple quantifier sentences much easier than if we tried it in a single blow. Eventually, though, you will be able to go through the intermediate steps in your head. This exercise is designed to give you a feel for the intermediate stages in this translation process.

- **Open Montague's Sentences.** This file contains expressions that are halfway between English and first-order logic. Our goal is to edit this file until it contains translations of the following English sentences. You should read the English sentence, make sure you understand how we got to the halfway point, and then complete the translation by replacing the hyphenated expression with a wff of first-order logic.

1. *Every cube is to the left of every tetrahedron.* [In the sentence window, you see the halfway completed translation, together with some blanks that need to be replaced by wffs. Commented out below this, you will find an intermediate "sentence." Make sure you understand how we got to this intermediate stage of the translation. Then complete the translation by replacing the blank with

$$\forall y (\text{Tet}(y) \rightarrow \text{LeftOf}(x, y)).$$

Once this is done, check to see if you have a well-formed sentence. Does it look like a proper translation of the original English? It should.]

2. *Every small cube is in back of a large cube.*
3. *Some cube is in front of every tetrahedron.*
4. *A large cube is in front of a small cube.*
5. *Nothing is larger than everything.*
6. *Every cube in front of every tetrahedron is large.*
7. *Everything to the right of a large cube is small.*
8. *Nothing in back of a cube and in front of a cube is large.*
9. *Anything with nothing in back of it is a cube.*
10. *Every dodecahedron is smaller than some tetrahedron.*

Save your list of sentences as **Sentences 3.30**.

- **Open Peirce's World.** Notice that all the English sentences are true in this world. Check to see that all of your translations are true as well. If they are not, see if you can figure out where you went wrong.
- **Open Leibniz's World.** Note that the English sentences 5, 6, 8, and 10 are true in this world, while the rest are false. Verify that your translations have the same truth values. If not, fix them.
- **Open Ron's World.** Here, the true sentences are 2, 3, 4, 5, and 8. Check that your translations have the right values, and correct them if they don't.

✦ **Exercise 3.31** (More multiple quantifier sentences) Now, we will try translating some multiple quantifier sentences completely from scratch. You should try to use the step-by-step procedure.

- Start a new sentence file and translate the following English sentences.
  1. *Every tetrahedron is in front of every dodecahedron.*
  2. *No dodecahedron has anything in back of it.*
  3. *No tetrahedron is the same size as any cube.*
  4. *Every dodecahedron is the same size as some cube.*
  5. *Anything between two dodecahedra is a cube.* [Note: This use of *two* really can be paraphrased using *between a dodecahedron and a dodecahedron.*]
  6. *Every cube falls between two objects.*
  7. *Every cube with something in back of it is small.*
  8. *Every dodecahedron with nothing to its right is small.*
  9. (★) *Every dodecahedron with nothing to its right has something to its left.*
  10. *Any dodecahedron to the left of a cube is large.*
- Open **Bolzano's World**. All of the above English sentences are true in this world. Verify that all your translations are true as well.
- Now open **Ron's World**. The English sentences 4, 5, 8, 9, and 10 are true, but the rest are false. Verify that the same holds of your translations.
- Open **Claire's World**. Here you will find that the English sentences 1, 3, 5, 7, 9, and 10 are true, the rest false. Again, check to see that your translations have the appropriate truth value.
- Finally, open **Peano's World**. Notice that only sentences 8 and 9 are true. Check to see that your translations have the same truth values.

✦ **Exercise\* 3.32** (Sentences that need paraphrasing before translation) Some English sentences do not easily lend themselves to direct translation using the step-by-step procedure discussed above. With such sentences, however, it is often quite easy to come up with an English paraphrase that is amenable to the procedure. Consider, for example, *If a freshman takes a logic class, then he or she must be smart*. The step-by-step procedure does not work here. But we can paraphrase the sentences as *Every freshman who takes a logic class must be smart*, and this is easily treated by the procedure.

- Translate the following sentences by first giving a suitable English paraphrase.
  1. *Only large objects have nothing in front of them.*
  2. *If a cube has something in front of it, then it's small.*
  3. *Every cube in back of a dodecahedron is also smaller than it.*  
 [Warning: This is an example of what is known as a “donkey” sentence, following a notorious example *Every farmer who owns a donkey beats it*. What makes such a sentence a bit tricky is the existential noun phrase in the relative clause which serves as the antecedent of the pronoun “it” in the verb phrase. This combination in effect forces us to translate the existential noun phrase with a universal quantifier. First, the donkey sentence would be paraphrased as *For every farmer and every donkey, if the farmer owns the donkey, then he beats it*. This sentence clearly needs two universal quantifiers in its translation. Several of the sentences that follow in this and the next exercise are donkey sentences.]
  4. *If  $e$  is between two objects, then they are both small.*
  5. *If a tetrahedron is between two objects, then they are both small.*

Save your list of sentences as Sentences 3.32.

- Open Ron's World. Recall that there are lots of hidden things in this world. Each of the above English sentences is true in this world, so the same should hold of your translations. Check to see that it does.
- Now open Bolzano's World. In this world, only sentence 3 is true. Check that the same holds of your translations.
- Next open Wittgenstein's World. In this world, only the English sentence 5 is true. Verify that your translations have the same truth values.

✦ **Exercise 3.33** (More sentences that need paraphrasing before translation) Translate the following sentences by first giving a suitable English paraphrase.

1. *Every dodecahedron is as large as every cube.* [Hint: Since we do not have anything corresponding to *as large as* (by which we mean at least as large as) in our language, you will first need to paraphrase this predicate using *larger than or same size as*.]
2. *If a cube is to the right of a dodecahedron but not in back of it, then it is as large as the dodecahedron.*
3. *No cube with nothing to its left is between two cubes.*

4. *The only large cubes are **b** and **c**.*
5. *At most **b** and **c** are large cubes.* [Note: There is a significant difference between this sentence and the previous one. This one does not imply that **b** and **c** are large cubes, while the previous sentence does.]

Open Ron's World. Each of the above English sentences is true in this world, so the same should hold of your translations. Check to see that it does. Now open Bolzano's World. In this world, only sentences 3 and 5 are true. Check that the same holds of your translations. Next open Wittgenstein's World. In this world, only the English sentences 2 and 3 are true. Verify that your translations have the same truth values. Submit your sentence file.

✦ **Exercise\* 3.34** (Name that object) Open Carroll's World and Hercules's Sentences. Try to figure out which objects have names, and what they are. You should be able to figure this out from the sentences, all of which are true. Once you have come to your conclusion, add the names to the objects and check to see if all the sentences are true. Submit your modified world.

✦ **Exercise\* 3.35** (Definite descriptions and numerical quantifiers) In this exercise we will try our hand translating English sentences involving numerical claims and definite descriptions. For purposes of this exercise, we will assume that the Russellian analysis of definite descriptions, described in Exercise 3.28, page 38, is correct.

- Translate the following English sentences.
  1. *There are at least two dodecahedra.*
  2. *There are at most two tetrahedra.*
  3. *There are exactly two cubes.*
  4. *There are only three things that are not small.*
  5. *The small tetrahedron has nothing in front of it.*
  6. *The tetrahedron with something in front of it is large.*
  7. *No dodecahedron is in back of the large cube.*
  8. (★★) *The medium cube is to the right of the large cube.*
  9. (★★) *The only thing with nothing to its right is the medium cube.*
  10. (★★) *The smallest cube is medium.*

Save your list of sentences as Sentences 3.35.

- Open Peano's World. Note that all of the English sentences are true in this world. Check to see that your translations are as well.

- Open **Bolzano's World**. Here sentences 1, 3, and 7 are the only true ones. Verify that your translations have the right truth values in this world.
- Open **Skolem's World**. Only sentences 5 and 7 are true in this world. Check your translations.
- Finally, open **Montague's World**. In this world, sentences 2, 3, 5, 7, and 10 are the only true ones. Check your translations.

✦ **Exercise\* 3.36** (Saying more complicated things) Open **Skolem's World**. Create a new sentence file and describe the following features of **Skolem's World**.

1. Use your first sentence to say that there are only cubes and tetrahedra.
2. Next say that there are exactly three cubes.
3. Express the fact that every cube has a tetrahedron that is to its right but is neither in front of or in back of it.
4. Express the fact that at least one of the tetrahedra is between two other tetrahedra.
5. Notice that the further back something is, the larger it is. Say this.
6. Note that none of the cubes is to the right of any of the other cubes. Try to say this.
7. Observe that the small tetrahedron is in front of but to neither side of all the other tetrahedra. State this.

Save your list of sentences as **Sentences 3.36**. If you have expressed yourself correctly, there is very little you can do to **Skolem's World** without making at least one of your sentences false. Basically, all you can do is “stretch” things out, that is, move things apart while keeping them aligned. To see this, try making the following changes.

1. Add a new tetrahedron to the world. Find one of your sentences that comes out false. Move the new tetrahedron so that a different sentence comes out false.
2. Change the size of one of the objects. What sentence now comes out false?
3. Change the shape of one of the objects. What sentence comes out false?
4. Slide one of the cubes to the left. What sentence comes out false?
5. Rearrange the three cubes. What goes wrong now?

✦ **Exercise 3.37** (Translation) Open Peirce's World. Look at it in 2-D to remind yourself of the hidden objects. Start a new sentence file where you will translate the following English sentences. Again, be sure to check each of your translations to see that it is indeed a true sentence.

1. *Everything is either a cube or a tetrahedron.*
2. *Every cube is to the left of every tetrahedron.*
3. *There are at least three tetrahedra.*
4. *Every small cube is in back of a particular large cube.*
5. *Every tetrahedron is small.*
6. *Every dodecahedron is smaller than some tetrahedron.* [Note: This is vacuously true in this world.]

Now let's change the world so that none of the English sentences are true. (We can do this by changing the large cube in front to a dodecahedron, the large cube in back to a tetrahedron, and deleting the two small tetrahedra in the far right column.) If your answers to 1–5 are correct, all of your translations should be false as well. If not, you have made a mistake in translation. Make further changes, and check to see that the truth values of your translations track those of the English sentences. Submit your sentence file.

✦ **Exercise\*\* 3.38** (More translations for practice) This exercise is just to give you more practice translating sentences of various sorts. They are all true in Skolem's World, in case you want to look while translating.

- Translate the following sentences.
  1. *Not every cube is smaller than every tetrahedra.*
  2. *No cube is to the right of anything.*
  3. *There is a dodecahedron unless there are at least two large objects.*
  4. *No cube with nothing in back of it is smaller than another cube.*
  5. *If any dodecahedra are small, then they are between two cubes.*
  6. *If a cube is medium or is in back of something medium, then it has nothing to its right except for tetrahedra.*
  7. *The further back a thing is, the larger it is.*
  8. *Everything is the same size as something else.*
  9. *Every cube has a tetrahedron of the same size to its right.*

10. *Nothing is the same size as two (or more) other things.*
  11. *Nothing is between objects of shapes other than its own.*
- Open Skolem's World. Notice that all of the above English sentences are true. Verify that the same holds of your translations.
  - This time, rather than open other worlds, make changes to Skolem's World and see that the truth value of your translations track that of the English sentence. For example, consider sentence 5. Add a small dodecahedron between the front two cubes. The English sentence is still true. Is your translation? Now move the dodecahedron over between two tetrahedra. The English sentence is false. Is your translation? Now make the dodecahedron medium. The English sentence is again true. How about your translation?

Submit your sentence file.

✦ **Exercise\* 3.39** (More translations) The following English sentences are true in Gödel's World. Translate them, and make sure your translations are also true. Then modify the world in various ways, and check that your translations track the truth value of the English sentence.

1. *Nothing to the left of **a** is larger than everything to the left of **b**.*
2. *Nothing to the left of **a** is smaller than anything to the left of **b**.*
3. *The same things are left of **a** as are left of **b**.*
4. *Anything to the left of **a** is smaller than something that is in back of every cube to the right of **b**.*
5. *Every cube is smaller than some dodecahedron but no cube is smaller than every dodecahedron.*
6. *If **a** is larger than some cube then it is smaller than every tetrahedron.*
7. *Only dodecahedra are larger than everything else.*
8. *All objects with nothing in front of them are tetrahedra.*
9. *Nothing is between two objects which are the same shape.*
10. *Nothing but a cube is between two other objects.*
11. ***b** has something behind it which has at least two objects behind it.*
12. *More than one thing is smaller than something larger than **b**.*

Submit your sentence file.



✦ **Exercise\* 3.40** (Translating extended discourse) The problems of translation are much more difficult when we look at extended discourse, where more than one sentence comes in. This exercise will help you get a feeling for the difficulty.

- Open Reichenbach's World 1 and examine it. Check to see that all of the sentences in the following discourse are true in this world.

*There are (at least) two cubes. There is something between them.*

*It is a medium dodecahedron. It is in front of a large dodecahedron.*

*These two are left of a small dodecahedron. There are two tetrahedra.*

Translate this discourse into a single first-order sentence. Check to see that your translation is true. Now check to see that your translation is false in Reichenbach's World 2.

- Open Reichenbach's World 2. Check to see that all of the sentences in the following discourse are true in this world.

*There are two tetrahedra. There is something between them. It is a medium dodecahedron. It is in front of a large dodecahedron. There are two cubes. These two are left of a small dodecahedron.*

Translate this into a single first-order sentence. Check to see that your translation is true. Now check to see that your translation is false in Reichenbach's World 1. However, note that the English sentences in the two discourses are in fact exactly the same; they have just been rearranged! The moral of this exercise is that the correct translation of a sentence into first-order logic (or any other language) can be very dependent on context. Submit your sentence file.

✦ **Exercise\* 3.41** (Ambiguity) Use Tarski's World to create a new sentence file and use it to translate the following sentences into FOL. Each of these sentences is ambiguous, so you should have two different translations of each. Put the two translations of sentence 1 in slots 1 and 2, the two translations of sentence 2 in slots 3 and 4, and so forth.

1. *Every cube is between a pair of dodecahedra.*
2. *Every cube to the right of a dodecahedron is smaller than it is.*
3. *Cube **a** is not larger than every dodecahedron.*
4. *No cube is to the left of some dodecahedron.*
5. *(At least) two cubes are between (at least) two dodecahedra.*

Now open Carroll's World. Which of your sentences are true in this world? You should find that exactly one translation of each sentence is true. If not, you should correct one or both of your translations.

Notice that if you had had the world in front of you when you did the translations, it would have been harder to see the ambiguity in the English sentences. The world would have provided a context that made one interpretation the natural one. Submit your sentence file.

✦|📖 **Exercise 3.42** (Block parties) The interaction of quantifiers and negation gives rise to subtleties that can be pretty confusing. Open *Löwenheim's Sentences*, which contains eight sentences divided into two sets. Suppose we imagine a column containing blocks to be a *party* and think of the blocks in the column as the attendees. We'll say a party is *lonely* if there's only one block attending it, and say a party is *exclusive* if there's any block who's not there (i.e., who's in another column).

1. Using this terminology, give simple and clear English renditions of each of the sentences. For example, sentence 2 says *some of the parties are not lonely*, and sentence 7 says *there's only one party*. You'll find sentences 4 and 9 the hardest to understand. Construct a lot of worlds to see what they mean.
2. With the exception of 4 and 9, all of the sentences are equivalent to other sentences on the list, or to negations of other sentences (or both). Which sentences are 3 and 5 equivalent to? Which sentences do 3 and 5 negate?
3. Sentences 4 and 9 are logically independent: it's possible for the two to have any pattern of truth values. Construct four worlds: one in which both are true (World 3.42.1), one in which 4 is true and 9 false (World 3.42.2), one in which 4 is false and 9 true (World 3.42.3), and one in which both are false (World 3.42.4).

Submit the worlds you've constructed and turn the remaining answers in to your instructor.

📖 **Exercise\* 3.43** (Quotations) Translate the following into FOL. Explain the meanings of the names, predicates, and function symbols you use, and comment on any shortcomings in your translations.

1. *There's a sucker born every minute.*
2. *Whither thou goest, I will go.*
3. *Soothsayers make a better living in the world than truthsayers.*
4. *To whom nothing is given, nothing can be required.*
5. *If you always do right, you will gratify some people and astonish the rest.*

✦ **Exercise 3.44** (Evaluating sentences in a world) Open Leibniz's World and Zorn's Sentences. The sentences in this file contain both quantifiers and the identity symbol. Work through them, assessing their truth and playing the game when necessary. After you're sure you understand why the sentences get the values they do, modify the false ones to make them true. You can make any change you want *except* adding or deleting a negation sign.

☞ **Exercise\*\* 3.45** (Challenging quotations) Translate the following into FOL, introducing names, predicates, and function symbols as needed. As usual, explain your predicates and function symbols, and any shortcomings in your translations. If you assume a particular domain of discourse, mention that as well.

1. *Only the brave know how to forgive.*
2. *No man is an island.*
3. *I care for nobody, not I,  
If no one cares for me.*
4. *Every nation has the government it deserves.*
5. *There are no certainties, save logic.*
6. *Misery (that is, a miserable person) loves company.*
7. *All that glitters is not gold.*
8. *There was a jolly miller once  
Lived on the River Dee.*
9. *If you praise everybody, you praise nobody.*
10. *Something is rotten in the state of Denmark.*

✦ **Exercise 3.46** (Describing a world) Let's try our hand describing a world using multiple quantifiers. Open Finsler's World and start a new sentence file.

1. Notice that all the small blocks are in front of all the large blocks. Use your first sentence to say this.
2. With your second sentence, point out that there's a cube that is larger than a tetrahedron.
3. Next, say that all the cubes are in the same column.
4. Notice, however, that this is not true of the tetrahedra. So write the same sentence about the tetrahedra, but put a negation sign out front.

5. Every cube is also in a different row from every other cube. Say this.
6. Again, this isn't true of the tetrahedra, so say that it's not.
7. Notice there are different tetrahedra that are the same size. Express this fact.
8. But there aren't different cubes of the same size, so say that, too.

Are all your translations true in Finsler's World? If not, try to figure out why. In fact, play around with the world and see if your first-order sentences always have the same truth values as the claims you meant to express. Check them out in König's World, where all of the original claims are false. Are your sentences all false? When you think you've got them right, submit your sentence file.

✦ **Exercise 3.47** (Name that object) Open *Marple's Sentences* and *Marple's World*. Modify the world by assigning names to the blocks in such a way that the sentences are all true. Submit your world.

✦ **Exercise 3.48** (Name that object) Open *Deckard's Sentences* and *Rebus' World*. Modify the world by assigning names to the blocks in such a way that the sentences are all true. Submit your world.

✦ **Exercise 3.49** (Building another world) It is not possible to create a world in which all of the sentences in *Arnault's Sentences* are false. Create a world in which as many of the sentences as possible are false. Submit your world as *World 3.49*.

Just as there is a redundancy in the collection of propositional connectives that we have introduced (see exercises 2.18, 2.24 and 2.25) we only need one of the quantifiers, as demonstrated by the following equivalences.

$$\begin{aligned}\exists xP(x) &\Leftrightarrow \neg\forall x\neg P(x) \\ \forall xP(x) &\Leftrightarrow \neg\exists x\neg P(x)\end{aligned}$$

✦ **Exercise 3.50** (Redundancy of Quantifiers 1) The file *Barwise's Sentences* contains sentences involving both quantifiers in the odd numbered positions. In each even numbered position write a sentence that is equivalent to the one above, but which does not use the existential quantifier. Simplify your answer by eliminating double negations where possible.

↗ **Exercise 3.51** (Redundancy of Quantifiers 2) This exercise is just like the preceding one, except that you are asked to write sentences that do not involve the universal quantifier in the even numbered positions. Don't forget to simplify your answers where possible.



## More Theoretical Exercises

The following exercises introduce and explore some more theoretical topics. They assume that you have already become a fairly proficient user of the first-order language. You might find them fun.

✦ **Exercise 4.1** (Games of incomplete information) Sometimes you can know that a sentence is true in a world without knowing how to play the game and win. For example, if you know that a given sentence is valid (see Section A.13, page 104), then you know that it will be true in any world Tarski's World can produce. However, you may not know how to play the game and win.

Open Mostowski's World. Translate the following into first-order logic. Then, without using the 2-D view, make as good a guess as you can about whether the sentences are true or not in the world. Once you have assessed a given sentence, use **Verify** to see if you are right. Then, with the correct truth value checked, see how far you can go in playing the game. Quit whenever you get stuck, and play again. Can you predict in advance when you will be able to win? Do not look at the 2-D view until you have finished the whole exercise.

1. *There are at least two tetrahedra.*
2. *There are at least three tetrahedra.*
3. *There are at least two dodecahedra.*
4. *There are at least three dodecahedra.*
5. *Either there is a small tetrahedron behind a small cube or there isn't.*
6. *Every large cube is in front of something.*

7. *Every tetrahedron is in back of something.*
8. *Every small cube is in back of something.*
9. *Every cube has something behind it.*
10. *Every dodecahedron is small, medium, or large.*
11. *If  $e$  is to the left of every dodecahedron, then it is not a dodecahedron.*

Now modify the world so that the true sentences are still true, but so that it will be clear how to play the game and win. Submit your sentence file.

✦📖 **Exercise\* 4.2** (Validity<sub>*I*</sub> versus Validity<sub>*U*</sub>) Before doing this exercise, read Section A.13, page 104. As we point out in that discussion, there is a difference between validity for interpreted languages, like the language used by Tarski's World, and validity for partially uninterpreted languages, of the kind studied in most logic texts. In the following exercises, we explore this topic a bit.

In order to talk coherently about the two notions and compare them, let's use a subscript *I* for the notion of validity applied to an interpreted language, and *U* for the notion applied when the predicate symbols (other than =) are treated as uninterpreted. Anything that is valid<sub>*U*</sub> is valid<sub>*I*</sub>, but in general the converse does not hold. Thus, for example, the sentence

$$\forall x \forall y (\text{LeftOf}(x, y) \rightarrow \text{RightOf}(y, x))$$

is valid<sub>*I*</sub> but not valid<sub>*U*</sub>. If a sentence is valid<sub>*I*</sub> but not valid<sub>*U*</sub>, then there must be a way to reinterpret the predicate symbols so that the result can be falsified in some world.

1. Open Carnap's Sentences and Bolzano's World. Paraphrase each sentence in English and verify that it is true in the given world.
2. For each sentence, decide whether you think it is true in *all* worlds or not, that is, whether it is valid<sub>*I*</sub> or not. If it is not valid<sub>*I*</sub>, find a world in which the sentence comes out false and name the world World 4.2.x, where x is the number of the sentence. [Hint: Exactly three of them are not valid<sub>*I*</sub>.]
3. Which of these sentences are valid<sub>*U*</sub>? [Hint: Three are.]
4. For each sentence which is valid<sub>*I*</sub> but not valid<sub>*U*</sub>, think of a way to reinterpret the predicates in the sentence so that the result can be falsified in some world.



Turn in your answers to parts one, three and four to your instructor; submit any worlds you built in part two.

✦|📖 **Exercise\*\* 4.3** (Validity<sub>I</sub> versus non-logical truth in all worlds) Another distinction Tarski's World helps us to understand is the difference between sentences that are valid<sub>I</sub> and sentences that are, for reasons that have nothing to do with logic, true in all worlds. The notion of valid<sub>I</sub> has to do with a sentence being true simply in virtue of the *meaning* of the sentence, and so no matter how the world is. However, some sentences are true no matter how the world is, but not because of the meaning of the sentence or its parts, but because of, say, laws governing the world. We can think of the constraints imposed by the innards of Tarski's World as physical laws governing how the world can be. For example, the sentence which asserts that there are at most 12 objects happens to hold in all the worlds that we can construct with Tarski's World, but it is not valid<sub>I</sub>, let alone valid<sub>U</sub>.

**Open Post's Sentences.** Classify each sentence in one of the following ways: (A) valid<sub>I</sub>, (B) true in all worlds that can be depicted using Tarski's World, but not valid<sub>I</sub>, or (C) falsifiable in some world that can be depicted by Tarski's World. For each sentence of type (C), build a world in which it is false, and save it as **World 4.3.x**, where x is the number of the sentence. For each sentence of type (B), use a pencil and paper to depict a world in which it is false. (In doing this exercise, assume that there are only three sizes of objects, so that **Medium** simply means *neither small nor large*. However, it is not plausible to assume that **Cube** means *neither a dodecahedron nor tetrahedron*, so you should not assume anything like this.)

✦|📖 **Exercise\* 4.4** (Some argument patterns) We will say that an argument from premises P to a conclusion C is valid (*I* or *U*) if it is impossible for P to be true without C also being true. In this exercise we consider some common patterns of inference, some of which are valid, and some invalid. Assess the validity<sub>U</sub> of each pattern, writing an informal argument justifying the validity of those patterns that you think are valid. Turn in your assessments.

For each invalid pattern, give a counterexample using Tarski's World. To give a counterexample in these cases, you will have to come up with sentences of the blocks language that fit the pattern, and a world that makes those specific premises true and the conclusion false. In the

sentence file, list the premises first and the conclusion last. Save your files as **World 4.4.n** and **Sentences 4.4.n** where  $n$  is the number of the argument form. Submit all the world and the sentence files.

1. *Affirming the Consequent*: From  $A \rightarrow B$  and  $B$ , infer  $A$ .
2. *Modus Tollens*: From  $A \rightarrow B$  and  $\neg B$ , infer  $\neg A$ .
3. *Strengthening the Antecedent*: From  $B \rightarrow C$ , infer  $(A \wedge B) \rightarrow C$ .
4. *Weakening the Antecedent*: From  $B \rightarrow C$ , infer  $(A \vee B) \rightarrow C$ .
5. *Strengthening the Consequent*: From  $A \rightarrow B$ , infer  $A \rightarrow (B \wedge C)$ .
6. *Weakening the Consequent*: From  $A \rightarrow B$ , infer  $A \rightarrow (B \vee C)$ .
7. *Constructive Dilemma*: From  $A \vee B$ ,  $A \rightarrow C$ , and  $B \rightarrow D$ , infer  $C \vee D$ .
8. *Transitivity of the Biconditional*: From  $A \leftrightarrow B$  and  $B \leftrightarrow C$ , infer  $A \leftrightarrow C$ .

✦|📖 **Exercise 4.5** (Valid or Falsifiable?) For each of the following sentences decide whether you think that it is valid<sub>I</sub>. If it is not, create a world which makes it false and submit the file as **World 4.5.n** where  $n$  is the number of the sentence. For the remainder of the sentences, turn in an explanation of why the sentence cannot be falsified in Tarski's World.

1.  $\neg \forall x \text{ Small}(x) \leftrightarrow \forall x \neg \text{Small}(x)$
2.  $(\forall x \text{ Cube}(x) \vee \forall x \text{ Dodec}(x)) \leftrightarrow \forall x (\text{Cube}(x) \vee \text{Dodec}(x))$
3.  $(\forall x \text{ Medium}(x) \wedge \forall x \text{ Tet}(x)) \leftrightarrow \forall x (\text{Medium}(x) \wedge \text{Tet}(x))$
4.  $\neg \exists x \text{ Dodec}(x) \leftrightarrow \exists x \neg \text{Dodec}(x)$
5.  $(\exists x \text{ Medium}(x) \vee \exists x \text{ Smaller}(x, b)) \leftrightarrow \exists x (\text{Medium}(x) \vee \text{Smaller}(x, b))$
6.  $(\exists x \text{ SameSize}(x, a) \wedge \exists x \text{ Small}(x)) \leftrightarrow \exists x (\text{SameSize}(x, a) \wedge \text{Small}(x))$
7.  $\forall x \text{ Cube}(x) \rightarrow \exists x \text{ Cube}(x)$
8.  $\forall x \text{ Cube}(x) \leftrightarrow \exists x \text{ Cube}(x)$

✦|📖 **Exercise\* 4.6** (Consistency) A sentence is said to be *inconsistent* if, due simply to its meaning, there is no way it could be true. Conversely, it is *consistent* if, so far as its meaning goes, it could have been true. More generally, a set  $T$  of sentences is said to be consistent if and only if all of the sentences in  $T$  could be true simultaneously, again, so far as their meanings go. Like validity, the notion of consistency can be divided into *consistent<sub>I</sub>* and *consistent<sub>U</sub>*, depending on whether the meanings of the predicates are assumed to be fixed.

1. Show that a sentence  $A$  is consistent if and only if  $\neg A$  is not valid.
2. Show that a set  $T$  of sentences is consistent just in case  $\exists x (x \neq x)$  is not a consequence of  $T$ .
3. Show that if a set is consistent $_I$ , it is consistent $_U$ .
4. Determine whether the following set of sentences is consistent. If it is, build a world. If it is not, use informal methods of proof to derive a contradiction from the set. [Hint: Translate these into first-order logic. Then use Tarski's World to build a world in which all the sentences are true.] Submit your world as World 4.6 or turn in your informal proof to your instructor.
  - (a) *Every cube is to the left of every tetrahedron.*
  - (b) *There are no dodecahedra.*
  - (c) *There are exactly four cubes.*
  - (d) *There are exactly four tetrahedra.*
  - (e) *No tetrahedron is large.*
  - (f) *Nothing is larger than anything to its right.*
  - (g) *One thing is to the left of another just in case the latter is behind the former.*

Save your world as World 4.6.

✦ **Exercise\* 4.7** (More about consistency) Open Padoa's Sentences. Any three of the sentences in Padoa's Sentences form a consistent set. There are four sets of three sentences, so to show this, build four worlds, World 4.7.123, World 4.7.124, World 4.7.134, and World 4.7.234, where the four sets are true. (Thus, for example, sentences 1, 2 and 4 should be true in World 4.7.124.)

📎 **Exercise\* 4.8** (Consistency $_I$ ) Give an informal proof that the four sentences in Padoa's Sentences taken together are inconsistent $_I$ .

📎 **Exercise\* 4.9** (Consistency $_U$ )

1. Reinterpret the predicates Tet and Dodec so that sentence 3 from Padoa's Sentences comes out true in World 4.9.124. Since this is the only sentence that uses these predicates, it follows that all four sentences would, with this reinterpretation, be true in this world. (This shows that the set is consistent $_U$ .)
2. Reinterpret the predicate Between in such a way that World 4.9.123 makes all the sentences in Padoa's Sentences true.

✦ **Exercise\* 4.10** (Null quantification) There is nothing in the rules for forming wffs that requires that the variable  $x$  occur free in  $A$  when forming either  $\forall xA$  or  $\exists xA$ . If it isn't, it is said to be a case of *null quantification*. This and the following exercise will help you to recognize instances of null quantification, and to see what sentences with null quantifiers mean.

Open **Null Quantification Sentences**. In this file you will find sentences in the odd numbered slots. Notice that each sentence is obtained by putting a quantifier in front of a sentence in which the quantified variable is not free.

1. Open **Gödel's World** and evaluate the truth of the first sentence. Do you understand why it is false? Repeatedly play the game committed to the truth of this sentence, each time choosing a different block when your turn comes around. Not only do you always lose, but your choice has no impact on the remainder of the game. Frustrating, eh?
2. Check the truth of the remaining sentences and make sure you understand why they have the truth values they do. Play the game a few times on the second sentence, committed to both true and false. Notice that neither your choice of a block (when committed to false) nor Tarski's World's choice (when committed to true) has any effect on the game.
3. In the even numbered slots, write the sentence from which the one above it was obtained. Check that the even and odd numbered sentences have the same truth value, no matter how you modify the world. This is because they are logically equivalent. Save and submit your sentence file.

📎 **Exercise\* 4.11** (More null quantification)

1. How can you describe the semantic function of null quantification?
2. Consider the following sentence

$$\exists y(\text{Tet}(y) \wedge \forall y(\text{Cube}(y) \rightarrow \text{Small}(y)))$$

Neither of the quantifiers in this sentences are null, however the second quantifier binds the same variable as the first. Figure out what this sentence says by assessing its truth and playing the game. Write a sentence equivalent to it, but which uses two different variables.

3. Do you think there could ever be a need to append a quantifier  $\forall y$  or  $\exists y$  to a formula that already contains the variable  $y$  bound by another quantifier (as in the sentence above?)


✦📖 **Exercise\*\*\* 4.12** (Numbers of variables) Tarski's World only allows you to use six variables. Let's explore what kind of limitation this imposes on our language.

1. Translate the sentence *There are at least two objects*, using only the predicate  $=$ . How many variables do you need?
2. Translate *There are at least three objects*. How many variables do you need?
3. It is impossible express the sentence *There are at least seven objects* using only  $=$  and the six variables available in Tarski's World, no matter how many quantifiers you use. Try to prove this. [Warning: This is true, but it is very challenging to prove. Contrast this problem with the one below.] Submit your two sentences and turn in your proof.

✦ **Exercise\*\* 4.13** (Reusing variables) In spite of the above exercise, there are in fact sentences we can express using just the six available variables that can only be true in worlds with at least seven objects. For example, in Robinson's Sentences, we give such a sentence, one that only uses the variables  $x$  and  $y$ .

1. Open this file. Build a world where there are six small cubes arranged on the front row and test the sentence's truth. Now add one more small cube to the front row, and test the sentence's truth again. Then play the game committed (incorrectly) to false. Can you see the pattern in Tarski's World's choice of objects? When it needs to pick an object for the variable  $x$ , it picks the leftmost object to the right of all the previous choices. Then, when it needs to pick an object for the variable  $y$ , it picks the last object chosen. Can you now see how the reused variables are working?
2. In the previous exercise, we asked you to prove that you could not express the existence of seven objects using only six variables. Yet Robinson's sentence guarantees the existence of seven objects using only two variables (and could obviously be continued to guarantee the existence of more). Can you explain the apparent conflict? [Hint: Rotate your world  $90^\circ$  and evaluate Robinson's sentence again. Is this sentence equivalent to the claim that there are at least seven objects?]
3. Now delete one of the cubes, and play the game committed (incorrectly) to true. Do you see why you can't win?

4. Now write a sentence that says there are at least four objects, one in front of the next. Use only variables  $x$  and  $y$ . Build some worlds to check whether your sentence is true under the right conditions. Submit your sentence file.

 **Exercise\*\*\* 4.14** (Persistence through expansion) As we saw in Exercise 3.18, page 34, some sentences simply can't be made false by adding objects of various sorts to the world. Once they are true, they stay true. For example, the sentence *There is at least one cube and one tetrahedron*, if true, cannot be made false by adding objects to the world. This exercise delves into the analysis of this phenomenon in a bit more depth.

Let's say that a sentence  $A$  is *persistent through expansion* if, whenever it is true, it remains true no matter how many objects are added to the world. (In logic books, this is usually called just persistence, or persistence under extensions.) Notice that this is a semantic notion. That is, it's defined in terms of truth in worlds. But there is a corresponding syntactic notion. Call a sentence *existential* if the only logical symbols it contains are  $\exists, \wedge, \vee, \neg$ , and  $=$ , and if no occurrence of the existential quantifier falls in the scope of a negation sign.

- Show that  $\text{Cube}(a) \rightarrow \exists x \text{ FrontOf}(x, a)$  is equivalent to an existential sentence.
- Is  $\exists x \text{ FrontOf}(x, a) \rightarrow \text{Cube}(a)$  equivalent to an existential sentence? Show that every existential sentence is persistent through expansion. [Hint: You will have to prove something slightly stronger, by induction on wffs. If you are not familiar with induction on wffs, just try to understand why this is the case. If you are familiar with induction, try to give a rigorous proof.] Conclude that every sentence equivalent to an existential sentence is persistent through expansion.

It is a theorem, due to Tarski and Łoś (a Polish logician whose name is pronounced more like “wash” than “loss”), that any sentence which is persistent through expansion is logically equivalent to an existential sentence. Since this is the converse of what you were asked to prove, we can conclude that a sentence is persistent through expansion if and only if it is equivalent to an existential sentence. This is a classic example of a theorem that gives a syntactic characterization of some semantic notion. For a proof of the theorem, see any textbook in model theory.

✦ **Exercise 4.15** (Contracting a world) Translate the following into first-order logic. Then open World 3.25, the world you built for Exercise 3.25, page 36. Remove objects from this world to make the sentences true.

1. *There are fewer than three objects.*
2. *Nothing but dodecahedra have things in front of them.*
3. *The large things are exactly the tetrahedra.*
4. *Something is neither a cube nor a tetrahedron.*

Now open Ockham's Sentences. Recall that all of these sentences were true in the original version of World 3.25. Check which of them are true in your contracted version of this world. Save your world as World 4.15 and your sentences as Sentences 4.15. In order for us to grade your files, you must submit both World 3.25 and World 4.15 at the same time.

☞ **Exercise\*\*\* 4.16** (Persistence through contractions)

1. Give the natural semantic characterization of sentences that are persistent through contractions of a world.
2. Show that a sentence  $A$  is persistent through contractions if and only if the sentence  $\neg A$  is persistent through expansions.

✦ **Exercise 4.17** (Invariance under motion, part 1) The real world does not hold still the way the world of mathematical objects does. Things move around. The truth values of some sentences change with such motion, while the truth values of other sentences don't. Start a new sentence file and translate the following English sentences into first-order logic. Then move objects around in World 3.25 to make the sentences true.

1. *Nothing is between any other things.*
2. *If one object is to the right of another, then the first is either a dodecahedron or a cube.*
3. *The tetrahedron is in front of everything else.* [Note: translate this sentence using the Russellian analysis of definite descriptions (see exercise 3.28, page 38.)]

Save your sentences as Sentences 4.17. Then check to see how many of the sentences in Ockham's Sentences are false in the altered world.

✦ **Exercise\* 4.18** (Invariance under motion, part 2) Open Ockham's World and Ockham's Sentences. Verify that all the sentences are true in the given world. Make as many of Ockham's Sentences false as you can by just moving objects around. Don't add or remove any objects from the world, or change their size or shape. You should be able to make false (in a single world) all of the sentences containing any spatial predicates, that is, containing *LeftOf*, *RightOf*, *FrontOf*, *BackOf*, or *Between*. (However, this is a quirk of this list of sentences, as we will see in the next exercise.) Save the world as **World 4.18**.

👉 **Exercise\*\*\* 4.19** (Invariance under motion, part 3) Call a sentence *invariant under motion* if, for every world, the truth value of the sentence (whether true or false) does not vary as objects move around in that world.

1. Prove that if a sentence does not contain any spatial predicates, then it is invariant under motion.
2. Give an example of a sentence containing a spatial predicate that is nonetheless invariant under motion.
3. Give another such example. But this time, make sure your sentence is not logically equivalent to any sentence that doesn't contain spatial predicates.

✦ **Exercise 4.20** (Persistence under growth, part 1) In the real world, things not only move around, they also grow larger. (Some things also shrink, but ignore that for now.) Starting with Ockham's World, make the following sentences true by allowing some of the objects to grow:

1.  $\forall x \neg \text{Small}(x)$
2.  $\exists x \exists y (\text{Cube}(x) \wedge \text{Dodec}(y) \wedge \text{Larger}(y, x))$
3.  $\forall y (\text{Cube}(y) \rightarrow \forall v (v \neq y \rightarrow \text{Larger}(v, y)))$
4.  $\neg \exists x \exists y (\neg \text{Large}(x) \wedge \neg \text{Large}(y) \wedge x \neq y)$

How many of Ockham's Sentences are false in this world? Save your world as **World 4.20**.

👉 **Exercise\*\*\* 4.21** (Persistence under growth, part 2) Say that a sentence *A* is *persistent under growth* if, for every world in which *A* is true, *A* remains true if some or all of the objects in that world get larger. Thus, *Large(a)* and  $\neg \text{Small}(a)$  are persistent under growth, but *Smaller(a, b)* isn't. Give a syntactic definition of as large a set as you



can for which every sentence in the class is persistent under growth. Can you prove that all of these sentences are persistent under growth?

✦|📝 **Exercise\* 4.22** (Persistence through change in perspective) **Open Skolem's World.** So far, we have been viewing our worlds from a fixed perspective, that of the user of the computer. But imagine that there are other people around the grid, some at the rear, some at other sides. You will agree with them about the truth of some sentences, but disagree about the truth of others.

- Translate the following sentences into first-order logic. Check to make sure your translations are true in Skolem's World.
  1. *There are only cubes and tetrahedra.*
  2. *There is a cube between two cubes.*
  3. *There is no cube to the left of another cube.*
  4. *There is no cube in back of a larger cube.*
  5. *Every cube has a tetrahedron to its right.*
  6. *Every cube has a tetrahedron either to its left or right.*
  7. *The rearmost cube is large.*
  8. *No two cubes are the same size.*
  9. *Nothing is the same size as anything in back of it.*

Save your list of sentences as **Sentences 4.22.1**.

- Which of the sentences are true in the world as viewed from the rear (i.e., rotated  $180^\circ$ )? How about the world rotated  $90^\circ$  clockwise? For each of the sentences, give first-order versions which express the same thing, but from the two other perspectives. Thus, for example, if the original sentence was **FrontOf(a,b)** then the two sentences would be **BackOf(a,b)** and **LeftOf(a,b)**, respectively. Save the new sentence lists as **Sentences 4.22.2** (for the “rear perspective” sentences) and **Sentences 4.22.3** (for the “clockwise  $90^\circ$ ” sentences). Check to see that your sentences have the correct truth values when Skolem's World is rotated appropriately.
- Which of the following English sentences also exhibit this sort of dependence on perspective?
  1. *The enemy is retreating.*
  2. *The local high school is the best in the state.*
  3. *The tallest man is at the front of the line.*
  4. *The tallest tree is in front of the others.*
  5. *The largest prime less than 100 is greater than 50.*

Explain your answers.

☞ **Exercise\* 4.23** (Inexpressibility) Let's say that two worlds  $W_1$  and  $W_2$  are *elementarily equivalent* for our language just in case exactly the same sentences of the language are true in both worlds. This notion can be applied using any first-order language, and with any two worlds. Sometimes worlds will be elementarily equivalent for one language, but not for another.

Worlds can be quite different but still be elementarily equivalent with respect to some language. This happens when there are differences in the worlds that simply cannot be captured by any sentence of the given language. This is a very important notion in first-order logic (one that was introduced, by the way, by Tarski). We can illustrate it quite simply with Tarski's World.

- Build a world in which the following sentences are true:
  1. *There is an object in the front row.*
  2. *There is an object in the back row.*
 Save your world as **World 4.23.1**.
- Using our first-order language, give as faithful a rendering as possible of sentence 2. Check that it is true in your world.
- Now alter the world by moving the object in the back row forward one row. Show that the translation you came up with is still true. Thus, your translation of the last sentence must not have captured what was said by the English sentence. Save your world as **World 4.23.2**.
- (★★) Prove that, in fact, the two worlds are elementarily equivalent.

This shows that the property of being in the back row cannot be expressed in the language we've used in Tarski's World, no matter how clever or complicated a sentence we might come up with. Of course one could consider a richer language, where the predicate `InBackRow` was included. But the phenomenon of essentially distinct, but elementarily equivalent worlds is almost always present in first-order languages.

## Part II

---

# Using the Software

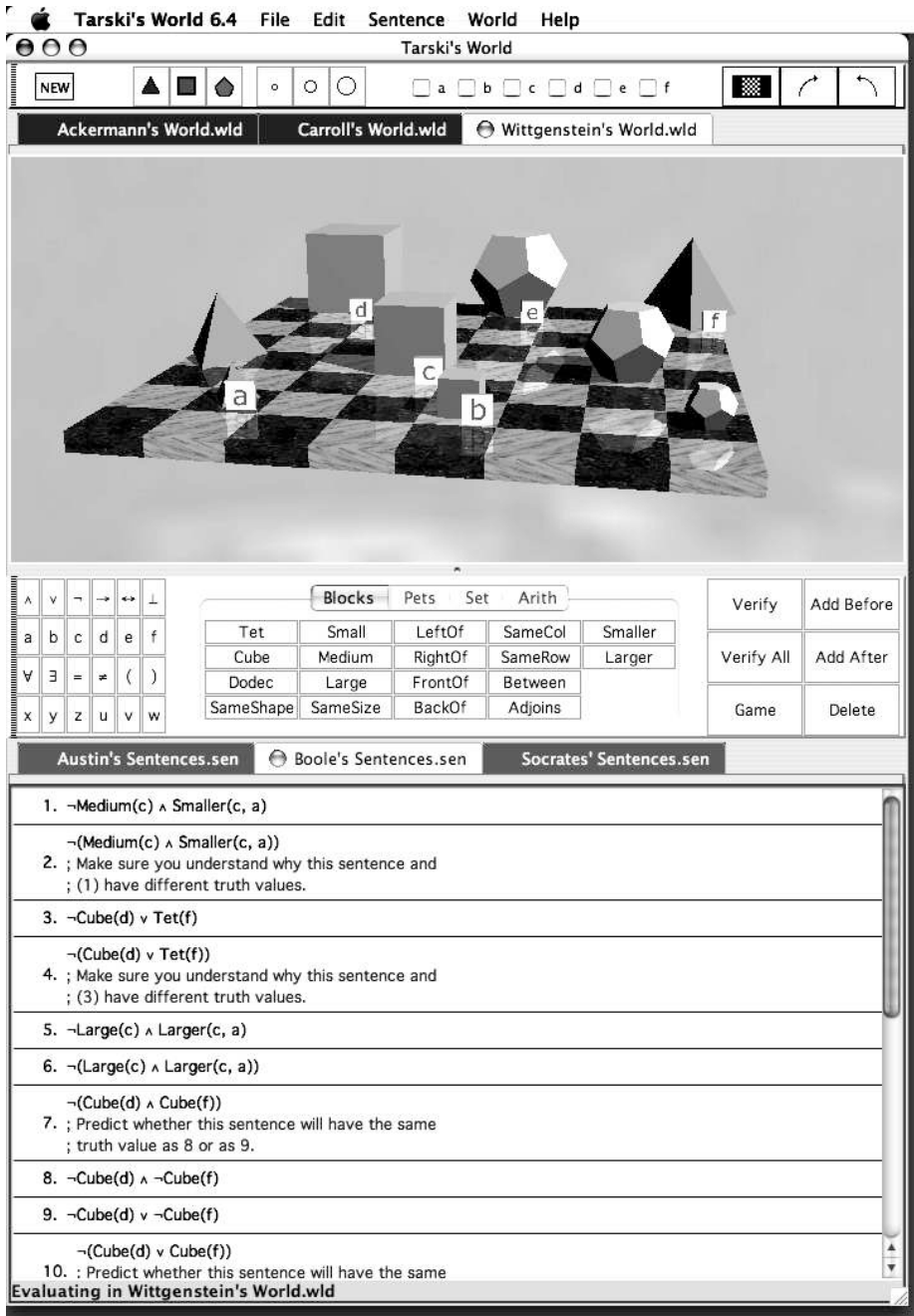


FIGURE 1 Main window of Tarski's World.

## Using Tarski's World

Tarski's World lets you represent simple, three-dimensional worlds inhabited by geometric blocks of various kinds and sizes, and test first-order sentences to see whether they are true or false in those worlds. We begin with instructions on how to start and stop Tarski's World, and explain the basic layout of the screen.

### 5.1 Getting started

The Tarski's World application is contained inside the folder called **Tarski's World Folder**. Also in this folder is a folder called **TW Exercise Files**, in which you will find the Tarski's World exercise files referred to in the book.

When Tarski's World is running you will see a large window divided into two sections. The upper *world panel* contains a checkerboard on which blocks are placed, called a world, and a tool bar for manipulating the content of this world. Immediately above the world is a tab which contains the name of the world. Initially this is **Untitled World**.

The *sentence panel* is the white panel at the bottom of the window. At first it contains only the numeral "1" inside. This is where sentences are entered and evaluated to see whether they are true or false in the world represented in the world window. Feel free to type something in the sentence window, say, "I'd rather be in Philadelphia." Immediately above the sentences is a tab which contains the name of the collection of sentences. Initially this is **Untitled Sentences**.

The *sentence toolbar* appears above the sentence panel. We generally use these tools to enter sentences of first-order logic. Feel free to play around by clicking on the buttons in the sentence toolbar.

### 5.1.1 Opening saved files

Both worlds and sentence lists can be saved as files on your disk. Indeed, many prepackaged world and sentence files come with Tarski's World. To open a saved file, you use the **Open. . .** command on the **File** menu.

To open a file, pull down the **File** menu and choose **Open. . .**. A file dialog will appear which allows you to navigate to the file that you wish to open. You will have to navigate to the right folder to find the prepackaged files, which are in **TW Exercise Files**. Find this folder, select it, and then click **Open**, or simply double-click on the name. Feel free to open one of the files you see, say, **Ackermann's World**, but if you make any changes to the world, don't save them.

When you open a file, a new tab will be created above the new sentence or world panel.<sup>1</sup> This tab will contain the name of the file that you opened. To return to viewing any other world or sentence file, just click on its tab, and it will reappear.

### 5.1.2 Starting new files

If you want to start a new world or sentence file, choose **New** from the **File** menu. You may then specify whether you want a new world or new sentence file from the menu which appears. The **New World** and **New Sentences** commands create a new empty world or sentence panel as appropriate. These are created as new tabs within the collection of worlds or sentences.

The command **New Random World** on the **New** menu creates a new world, and populates it with randomly chosen blocks.

The **New Window** item on the **New** menu creates a new window identical to the initial main window.

You may have noticed that there is another **New** command on the **File** menu. depending on which panel is active, this reads **New Sentences** or **New World**, and is equivalent to the corresponding item on the **New** submenu. This item also has a shortcut.

### 5.1.3 Saving a file

If you want to save a file, use the **Save** submenu from the **File** menu. There are items here which allow you to save the current world, **Save World** or **Save World As...**, the current sentences, or all worlds and

---

<sup>1</sup>There is one exception to this rule, and that is when the current tab is one of the "Untitled" tabs that has not been changed. In this case the old tab will be replaced by the new one.

sentences in all tabs.

If the file has never been saved before, a dialog box will appear giving you the option of naming the file you are about to create. If you were to hit the return key, or click the **Save** button, the file would be saved with the default name. You should type in some other name before hitting the return key or clicking **Save**. You should also make sure you are saving the file where you want it. Check the directory name at the top of the save dialog box. If you're not in the folder where you want to save the file, navigate to the right one by clicking on this name.

You may have noticed that there is another save command on the **File** menu, depending on which panel is active, this reads **Save Sentences** or **Save World**, and is equivalent to the corresponding item on the **Save** menu. This item also has a shortcut.

Once a file has been saved, the name of the file appears in the corresponding tab. If you are working on a named file, the **Save** and **Save As...** commands behave differently. The first will save a new version of the file under the same name, and the old version will be gone. The second gives you a chance to create a new file, with a new name, and keeps the old file, with its name. For this reason, **Save As...** is the safer of the two options.

You can also access the save commands by control-clicking (Macintosh) or right-clicking (Windows) on the corresponding tab.

All files created by Tarski's World can be read by either the Macintosh or Windows version of the application.

#### 5.1.4 Closing Tabs

When you are done with a world or sentence file, you can close it using the **Close** commands on the **File** menu. As usual, there is a command which closes the active tab whether it is a world or sentence, and a submenu which allows you to close the tab of your choice. The close commands can also be accessed from the tab's menu.

#### 5.1.5 Reverting a File

If you want to reload a tab from its corresponding file, you can do so using the **Revert** submenu on the **File** menu. You will be asked first whether you want to save the changes that you have made to the file (to a different file), and then the content of the current tab will be replaced from the file. This command can also be accessed from the tab's menu.

### 5.1.6 Printing

To print your sentences or world, choose the appropriate **Print** command from the **File** menu, or from the tab's popup menu. If your computer is not connected to a printer, this probably won't work.

### 5.1.7 Quitting (Exiting) Tarski's World

Eventually you will want to leave Tarski's World. To do this, choose **Quit** from the application menu (**Exit** from the **File** menu on Windows). If you've made any unsaved changes to the files, Tarski's World will give you a chance to save them.

## 5.2 The World Panel

### 5.2.1 Adding blocks

To put a block on the grid, simply click the **New** button on the tool bar. Try this out. The size and shape of block that is created can be controlled by setting a preference (see section 5.6). A small cube is created by default.

### 5.2.2 Selecting blocks

A block can be selected by clicking on it. The block will change color to indicate its selection. To unselect a block, click elsewhere in the world window.

To select more than one block, hold down the shift key while clicking on the blocks. If many blocks are selected, and you want to deselect one of them, click on it while holding down the shift key.

### 5.2.3 Moving blocks

To move a block, position the cursor over the block and drag it to the desired position. (That is, move the mouse's arrow over the block and then, with the button depressed, move the mouse until the block is where you want it.) If multiple blocks are selected, they will all move.

If you move a block (or blocks) too close to the edge it will fall off.

### 5.2.4 Sizing and shaping blocks

To change a block's shape, select it and click on one of the shape buttons on the toolbar. These display a triangle, square and pentagon and change the shape to tetrahedron, cube and dodecahedron, respectively. If multiple blocks are selected all will changed to the new shape.

Similarly, to change a block's size, select it and click on one of the



size buttons on the toolbar. These display circles of small, medium and large sizes. If multiple blocks are selected all will changed to the new size.

### 5.2.5 Naming blocks

When a block is selected, the name checkboxes on the toolbar are activated. To add a name to the selected block, click on the appropriate checkbox. If the box is already checked, the name will be removed from the block.

In first-order logic, one object can have several names, but two objects cannot share the same name. Hence Tarski's World lets you give a block more than one name, but once a name is used, that name cannot be assigned to another block.

### 5.2.6 Deleting blocks

To delete a block, drag the block off the edge of the grid and drop it. Alternatively, select the appropriate block or blocks and hit the **Delete** key.

### 5.2.7 Cutting, copying, and pasting blocks

If you want to copy some blocks from one file to another, use the cut, copy, and paste functions.

If you select blocks and then choose **Cut** or **Copy** from the **Edit** menu, the blocks are stored on the computer's clipboard. The difference between the two commands is that **Cut** deletes the blocks from their present position, while **Copy** leaves them in place. You can't see the contents of the clipboard, but the blocks will be there until you cut or copy something else to the clipboard.

Once some blocks are on the clipboard, they can be pasted into a different (or the same) world. Just select the relevant tab and choose **Paste** from the **Edit** menu. A copy of the blocks on the clipboard will be inserted.

You can paste several copies if you want to, even into the same world. Tarski's World will attempt to paste the blocks in the same configuration as they were cut, but will need to move them if there are already blocks in any of those positions. Because two blocks cannot have the same name, pasted blocks will have their names removed.

### 5.2.8 Hiding labels

Whenever you name a block, Tarski's World labels the block with its name. Of course, in the real world we only wear name tags at unpleasant social occasions. Like us, blocks in Tarski's World can have names without wearing labels. To hide the labels, simply choose **Hide Labels** from the **World** menu. To redisplay the labels, choose **Show Labels** from the **World** menu.

This command toggles the display of labels in all open worlds.

### 5.2.9 2-D view

Labels aren't the only things that can hide. Sometimes a small block can be obscured from view by another block in front of it. To get a bird's eye view of the world, choose **2-D View** from the **World** menu. To get back to the usual perspective, choose **3-D View** from the **World** menu. These commands can also be accessed from the tool bar using the button which looks like a small version of the checkerboard.

Blocks can be moved, selected, and changed from the 2-D view in exactly the same way as the 3-D view. (You can even change to the 2-D view in the middle of playing the game; sometimes you will have to in order to pick an appropriate block, or to see what Tarski's World is referring to.)

### 5.2.10 Rotating Worlds

To rotate a world by 90 degrees in either direction, choose **Rotate World Clockwise** or **Rotate World Counterclockwise** from the **World** menu. Such a rotation counts as a change to the world and will be saved when you save the world.

You can also rotate the world from the tool bar using the arrow buttons.

## 5.3 The Sentence Panel

There are two ways to enter formulas into the sentence window, from the sentence toolbar or from the keyboard. Most people find it easier to use the toolbar than the keyboard.

### 5.3.1 Writing formulas

Tarski's World makes writing first-order formulas quite painless. As you may have noticed while playing with the sentence toolbar, when you enter a predicate, like **Tet** or **BackOf**, the insertion point locates itself

in the appropriate position for entering “arguments”—variables ( $u, v, w, x, y, z$ ) or individual constants ( $a, b, c, d, e, f$ ).

What this means is that a sentence like `BackOf(a,b)` can be entered into the sentence list with three mouse clicks in the toolbar: first on the `BackOf` button, then on the `a` button, then on the `b` button. To enter the same thing from the keyboard would require 11 keystrokes.

In order to allow you to write more readable formulas, Tarski’s World treats brackets (“[ ]”) and braces (“{ }”) as completely equivalent to parentheses. Thus, for example, you could write `[LeftOf(a, b) ^ Large(a)]` and Tarski’s World will read this sentence as `(LeftOf(a, b) ^ Large(a))`. But you have to type brackets and braces from the keyboard.

### 5.3.2 Commenting your sentences

You can add comments to your sentences in a way that will be ignored by the program when it is checking to see if they are well formed or true. You do this by prefacing each line of text you want ignored by a semicolon (;). This will cause Tarski’s World to ignore anything that follows on the same line. Tarski’s World displays all of the characters in the comment in red to remind of their (in)significance.

### 5.3.3 Creating a list of sentences

To create a whole list of sentences, you first enter one sentence, and then choose **Add Sentence After** from the **Sentence** menu. You are given a new, numbered line, and can then enter a new sentence. If you hit the Return key, this will *not* start a new sentence, but will simply break your existing sentence into two lines. Use **Add Sentence After!**

Instead of choosing **Add Sentence After** from the **Sentence** menu, you can do this from the toolbar by clicking the Add After button or you can do it directly from the keyboard in two ways. You can type Shift-Return (that is, type Return while holding the shift key down) or use the keyboard equivalent shown in the menu.

To insert a new sentence in your list *before* the current sentence, choose **Add Sentence Before** from the **Sentence** menu, or using the **Add Before** button on the toolbar.

### 5.3.4 Moving from sentence to sentence

You will often need to move from sentence to sentence within a list of sentences. You can move the insertion point with the up and down arrow keys ( $\uparrow, \downarrow$ ) on the keyboard or by clicking on the sentence of

TABLE 1 Keyboard equivalents for typing symbols.

Symbol	Key	Symbol	Key
$\neg$	~	$\neq$	#
$\wedge$	&	$\vee$	
$\rightarrow$	\$	$\leftrightarrow$	%
$\forall$	@	$\exists$	/
$\subseteq$	_	$\in$	\

interest with the mouse. The left and right arrow keys ( $\leftarrow$ ,  $\rightarrow$ ) on the keyboard also move the insertion point, but only within a single sentence.

If you hold down the Option key, the up arrow takes you to the first sentence of the list, the down arrow takes you the last sentence of the list, and the left and right arrows take you to the beginning and the end of the current sentence.

### 5.3.5 Deleting sentences

To delete a whole sentence and renumber the sentences that remain, choose **Delete Sentence** from the **Sentence** menu. First make sure the insertion point is somewhere in the sentence you want to delete.

Note that you cannot highlight parts of two different sentences and then delete them. If you want to delete a sentence boundary, you must use the command **Delete Sentence** from the **Sentence** menu.

### 5.3.6 Typing symbols from the keyboard

Sentences can be entered into the sentence window by typing them on the physical keyboard. When typing predicates in the blocks language, you must be sure to spell them correctly and to capitalize the first letter (since otherwise they will be interpreted as names, not predicates). You also have to insert your own punctuation: parentheses after the predicate, and commas to separate multiple “arguments” (as in  $\text{Between}(a, x, z)$ ). To get the logical symbols use the keyboard equivalents shown in Table 1.

Either the sentence window or the Keyboard window must be “active” before typing on the physical keyboard will have any effect. If you type and nothing shows up, that’s because the world panel is currently the active panel. To activate the other panel, just click in it somewhere.

You can change the size of the font used to display sentences using the **Text size** submenu on the **Sentence** menu.

### 5.3.7 Cutting, copying, and pasting

If you want to change the order of the sentences in a list, or copy a sentence from one file to another, use the cut, copy, and paste functions.

If you highlight a string of symbols and then choose **Cut** or **Copy** from the **Edit** menu, the string of symbols is stored on the computer's clipboard. The difference between the two commands is that **Cut** deletes the highlighted symbols from their present position, while **Copy** leaves them in place. You can't see the contents of the clipboard, but the symbols will be there until you cut or copy something else to the clipboard.

Once something is on the clipboard, it can be pasted anywhere you want it. Just put the insertion point at the desired place and choose **Paste** from the **Edit** menu. A copy of the string of symbols on the clipboard will be inserted. You can paste several copies at several different points, if you want to.

You can copy sentences out of Tarski's World and paste them into Fitch or Boole, and vice versa.

## 5.4 Verifying syntax and truth

As you will learn, only some strings of symbols are grammatically correct, or well formed, as we say in logic. These expressions are usually called *well-formed formulas*, or *wffs*. And only some of these are appropriate for making genuine claims about the world. These are called *sentences*. Sentences are wffs with no free variables. You will learn about these concepts in the text.

To see if what you have written in the sentence window is a sentence, and if so, whether it is true in the world currently displayed, click on the Verify button in the toolbar, or type Command-Return (Control-Return on Windows). If you want to check a whole list of sentences, choose **Verify All Sentences** from the **Sentence** menu. Alternatively, use the **Verify All** button on the tool bar.

When you verify a sentence, the results are displayed in the margin to the left of the sentence number: "T" or "F" indicates that the sentence is true or false in the world, "\*" indicates that the formula is not well-formed or not a sentence, while "+" indicates that the formula is a sentence of first-order logic, but not evaluable in the current world. If

you are unsure why a sentence is not evaluable, verifying the sentence again will result in a dialog explaining the reason.

The evaluations are removed when the sentence or world is changed.

## 5.5 Playing the game

When you stake out a claim about a world with a complex sentence, you are committed not only to the truth of that sentence, but also to claims about its component sentences. For example, if you are committed to the truth of a conjunction  $A \wedge B$  (read “A and B”) then you are also committed both to the truth of A and to the truth of B. Similarly, if you are committed to the truth of the negation  $\neg A$  (read “not A”), then you are committed to the falsity of A.

This simple observation allows us to play a game that reduces complex commitments to more basic commitments. The latter claims are generally easier to evaluate. The rules of the game are part of what you will learn in the body of this book. Here, we will explain the kinds of moves you will make in playing the game.

To play the game, you need a guess about the truth value of the current sentence in the current world. This guess is your initial commitment. The game is of most value when this commitment is wrong, even though you won't be able to win in this case.

To start the game, click the **Game** button on the sentence tool bar. Tarski's World will begin by asking you to indicate your initial commitment. At this point, how the game proceeds depends on both the form of the sentence and your current commitment. A summary of the rules can be found in Table 9.1 in Chapter 9 of the textbook.

### 5.5.1 Picking blocks and sentences

As you see from the game rules, at certain points you will be asked to pick one sentence from a list of sentences. You do this by clicking on the desired sentence and then clicking **OK**.

At other points in the game, you will be asked to pick a block satisfying some formula. You do this by moving the cursor over the desired block and selecting it. Then click **OK**. If necessary, Tarski's World assigns a name to the chosen block, for example n1, and labels it.

### 5.5.2 Backing up and giving up

Tarski's World never makes a mistake in playing the game. It will win if it is possible for it to win, that is, if your initial commitment was wrong.

However, *you* may make a mistake, and so lose a game you could have won. All it takes is some bad choices along the way. Tarski's World will take advantage of you. It will not tell you that you made a bad move until it has won, when it will inform you that you could have won. What this means is that there are two ways for you to lose: if you were wrong in your initial assessment, or if you make a faulty choice in the play of the game. To put this more positively, if you win a game against the computer, then you can be quite sure that your initial assessment of the sentence, as well as all subsequent choices, were correct.

To make up for the edge the computer has, Tarski's World allows you to retract any choices you have made, no matter how far into the game you've gone. So if you think your initial assessment was correct but that you've made a bad choice along the way, you can always retract some moves by clicking on the **Back** button. If your initial assessment really *was* correct, you should, by using this feature, eventually be able to win. If you can't, your initial commitment was wrong.

If, halfway through the play of the game, you realize that your assessment was wrong and understand why, you can stop the game by clicking the **End** button. This ends the game, but does not shut down Tarski's World.

### 5.5.3 When to play the game

In general, you won't want to play the game with every sentence. The game is most illuminating when you have incorrectly assessed a sentence's truth value, but are not sure why your assessment is wrong. When this happens, you should always play the game without changing your commitment. Tarski's World will win, but in the course of winning, it will usually make clear to you exactly why your assessment was wrong. That's the real value of the game.

You might wonder what happens when you play the game with a correct assessment. In this case, if you play your cards right, you are guaranteed to win. But Tarski's World does not simply give up. At those points in the game when it needs to make choices, it will make them more or less randomly, hoping that you will blunder somewhere along the line. If you do, it will seize the opportunity and win the game. But, as we have noted, you can always renege by backing up.

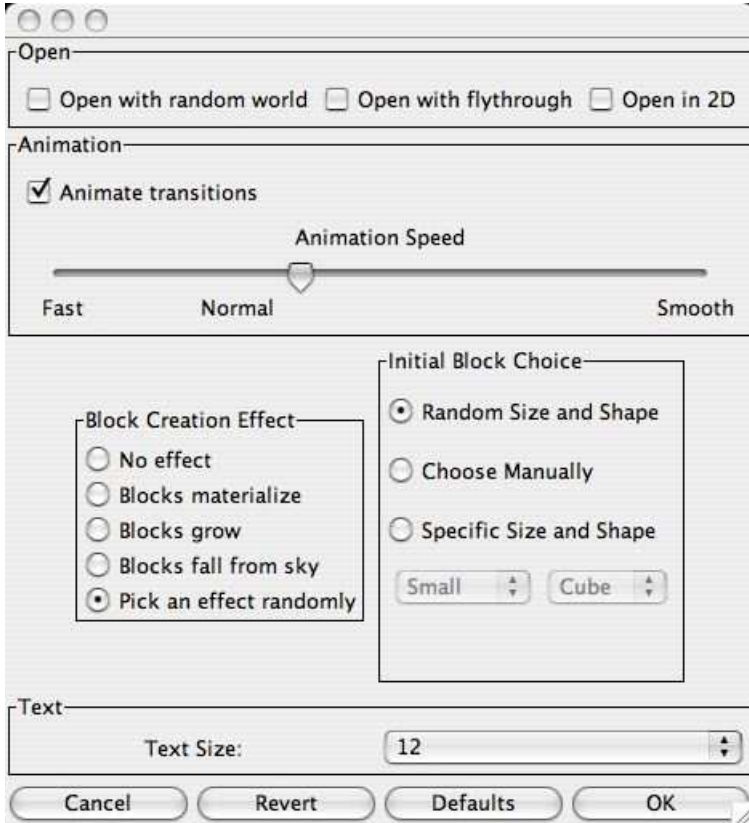


FIGURE 2 Tarski's World Preferences Dialog

## 5.6 Preferences

Some aspects of the behavior of Tarski's World can be controlled using the preferences dialog. This can be accessed by choosing the **Preferences...** command from the application menu (**Edit** Menu on Windows). The preferences dialog is shown in figure 2.

The first row of preferences are checkboxes which allow you to switch on some options for opening and creating new worlds.

You can opt to create a random world instead of an empty one when a new world is created by selecting the **open with random world** checkbox. You can view an animation as the world is opened or created by selecting the **open with flythrough** checkbox, and you can opt to always open worlds in 2-D by selecting the final checkbox.



You can control the speed of animations, or switch all animations off using the **Animation** panel. The speed of animations is controlled by a slider. When the slider is set to the **Fast** end of the scale, the animations will have fewer frames, resulting in a more jerky animation which takes less time. The **Smooth** end of the scale will result in smoother, but longer, animations. You might like to play with this setting to get the effect that is just right for your computer. If nothing seems right, then you can switch all animation off.

You can choose a different effect for how new blocks are created, varying from dropping from the sky, materializing or growing in place. We think that the effects are pretty nifty. You might like to try them out.

The final world preference determines the size and shape of the block that is created when the **New Block** button is pressed. You have the option of being presented with a dialog box, always creating the same kind of block, or allowing Tarski's World to choose a size and shape for you.

The final option concerns the display of text in the sentence pane. You may opt to specify a default font size for the sentence panel.



## Using Submit

Submit is a computer program that allows you to submit your homework exercises over the Internet to the Grade Grinder, a grading server that checks your homework and returns reports to you and, if you ask, your instructor. In this chapter we describe how to use Submit.

### 6.1 Getting started

The computer you use to submit homework to the Grade Grinder must be connected to the Internet. Submit uses the same form of communication used by web browsers, so if you can access the Internet with your web browser, you should be able to submit files to the Grade Grinder.

To submit files to the Grade Grinder, you need to have all of the following ahead of time:

1. **The solution files you want to submit.** You might want to collect together all the files you want to submit in a single folder. Remember that the files must be named exactly the way you are asked to name them in the book. Submit will only send files whose names begin with **World** or **Sentences**, and that are Tarski's World files<sup>2</sup>. If you try to submit a file with an incorrect name, it will give you a chance to correct the name. If you try to submit a file with an incorrect exercise number (e.g., **World 1.1** rather than **World 10.1**), then Submit will send it but the Grade Grinder will tell you that it doesn't know how to grade it or grade it as the wrong exercise. Be careful when naming your solution files!

---

<sup>2</sup>Users of our *Language, Proof and Logic* package will realize that Submit will also allow the submission of files created by our Fitch and Boole applications provided their names begin with **Table** or **Proof** as appropriate.

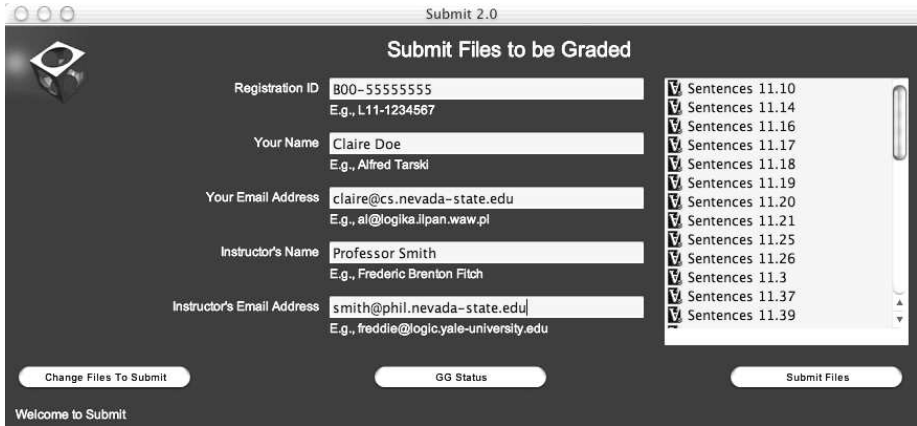


FIGURE 2 Main window in Submit.

2. **Your Registration ID number.** This is a unique ID number that is included in the Tarski's World package. It is of the form T11-1234567, that is, a letter, followed by two digits, a dash, then seven more digits. Do not let anyone else use your ID number, since the number is how the Grade Grinder associates your homework exercises with you.
3. **Your name and full email address.** The name you enter should be sufficient for your instructor to identify you. It is important that you enter your full email address, for example *claire@cs.nevada-state.edu*, not just *claire* or *claire@cs* or *claire@cs.nevada-state*—since the Grade Grinder will need the full address to send its response back to you. You must use the same email address throughout the course, so make sure you choose the right one and enter it correctly. If you don't have an email address, or don't know your full Internet email address, contact one of the computer folks at your school.
4. **Your instructor's name and full email address.** If you want your results to be sent to an instructor as well as to you, you will need his or her name and full email address. The instructor's email address must match one of the instructors in the Grade Grinder's database, so make sure you find out what it is. If you do not want results sent to an instructor, you won't need this information.

## Launching Submit

To launch Submit, double-click on the application icon, which has a blue background and shows the yellow corner of a cube (if you installed Submit on a Windows machine, you can also launch the program from the **Start** menu by choosing **Programs\TW Software\Submit**). After a moment, Submit's main window will appear on your screen. You'll know it by the twirling cube.

Your goal is simply to fill in the various parts of this window by typing in the information requested and specifying the list of files to be submitted. Once that is done, you will simply press the **Submit Files** button in the lower right of the window.

Start by filling in the information requested ( Registration ID, your name, etc.). Read about this information above if you haven't already. Remember to use your full email address and to spell it correctly. Once you have submitted files, your Registration ID will be associated with the email address you type in, so that no one can use your Registration ID to submit bogus homework in your name. In later submissions, you will have to use the exact same email address with your Registration ID, so if you have more than one email address, remember which one you used.

## 6.2 Choosing files to submit

There are several ways to choose the files you want to submit. The most common is to click on the button **Choose Files to Submit** in the lower left corner of the main Submit window. This will open another window showing two file lists. The list on the left shows all the files in the current folder (directory). The list on the right will be built by you as you choose files to submit. The goal is to find the names of your solution files on the lefthand list and move them to the righthand list.

To find your solution files, you will have to navigate around the folder structure of your computer in the lefthand list. To move to "higher" folders, those containing the folder whose contents are currently shown in the list, click on the folder name that appears above the list. A menu will pop up and show all the folders (and volume) that contain this folder. Choose the folder whose contents you want to view. To move to "lower" folders, those contained inside the folder whose contents you are viewing, choose those folder names from the list and click **Open**, or simply double-click on the folder names. Using these two techniques,

you will be able to find any file located on your computer's hard disk or on any disk inserted into one of the computer's drives.

Once you have found the file(s) you want to submit, select the file name in the lefthand list and click the **Add>>>** button to add the name to the righthand list. Keep doing this until the righthand list contains all the files you want to submit. If any of the files are of the wrong type or have names of the wrong form, Submit will let you know before putting them on the list. It will give you a chance to correct the names of files that are of the right type, but not named correctly. (This does not change the names of the files on your computer, only the name sent to the Grade Grinder.) When you are finished choosing files, click the **Done** button under the righthand file list.

Another way to specify files to submit is by choosing **Open...** from the **File** menu while you are at the main Submit window. This gives you the standard file open dialog box. If you choose a file of an appropriate type (e.g., a World file), it will be added directly to the list of files to submit. This takes longer if you have more than one file to submit.

**Macintosh only:** The fastest way to specify the files to submit is to drag the files (or a folder containing them) to the Submit application icon in the Finder. This will launch Submit (if it is not already running) and put the file names directly onto the list of files to submit.

### Submitting the files

Once you have entered all the information on the main Submit window and have constructed the list of files to submit, click the **Submit Files** button under the list of files. Submit will ask you to confirm that you want to submit the files on your list, and whether you want to send the results just to you or also to your instructor. When you are submitting finished homework exercises, you should select **Instructor Too**, but if you just want to check to see if you've done the problems right, select **Just Me**. One of these boxes must be chosen before you click the **Proceed** button, which sends your submission.

After a moment, you will get a notice back from the Grade Grinder telling you which files it received and which of them it knows how to grade. (If you misnumbered a solution, it won't know how to grade it.) You can save this notice as a receipt to prove that the files got to the Grade Grinder.

## What Submit sends

When you submit files to the Grade Grinder, Submit sends a copy of the files. The original files are still on the disk where you originally saved them. If you saved them on a public computer, it is best not to leave them lying around. You should put them on a floppy disk that you can take with you, and delete any copies from the public computer's hard disk.

### 6.3 How you know your files were received

If you receive the notice back from the Grade Grinder described above, then you know your files were received. If you receive an error message, or if nothing at all happens when you try to submit your files, then the Grade Grinder has not received them. If your submission does not get through, it is probably a problem with your Internet connection. You should try submitting them again, perhaps from another computer. There are presently two Grade Grinder servers (one in California and one in Illinois), and if Submit cannot find one, it looks for the other. If it fails both times it is probably because your computer or local network cannot access the Internet.

A second confirmation that your submission was received is the email message that the Grade Grinder will send you with the results of its grading. This will arrive shortly after you make the submission, depending on how large the submission was, how many other submissions the Grade Grinder is checking, and how long it takes email to reach you. Generally, you will receive the email message within minutes of submitting your files.

You can check on a submission by clicking on the **GG Status** button at the bottom of the Submit window. If the Grade Grinder was unable to grade your submission, it will tell you which of your submissions have been delayed and for what reasons.

### 6.4 Saving your user data

The information that you enter into the main Submit window, other than the files to submit, is known as the user data. If you would like to avoid typing your name, email address, etc., each time you submit files, you can save all of this information except the Registration ID. You do this by choosing **Save As...** from the **File** menu. This will let you save a file containing this information.

If you save this file with the default name, **Submit User Data**, and put it in the folder suggested by Submit, this information will automatically be entered into the appropriate fields when you launch the program. Alternatively, the user data file can be located elsewhere and opened from within Submit. Or, on the Macintosh, you can launch Submit by double-clicking on the user data file, and this too will enter the data into the appropriate fields. In these latter two cases, the name of the user data file does not have to be **Submit User Data**.



---

# Appendixes



## Appendix A

---

# First-order Logic

This appendix is not meant as a substitute for a logic book, though it should give you enough information to use Tarski’s World. If you are not using Tarski’s World in conjunction with a logic course, you might want to check out an introductory logic text, such as our *Language, Proof and Logic*, from the library so you can pursue further the topics we touch upon here.

### A.1 First-order languages

First-order logic is concerned with first-order languages. The adjective “first-order” signifies that the languages can talk about (i.e., quantify over) all objects in a given domain, but not about arbitrary properties of objects in the domain. So we can say things like “everything is purple,” but not things like “every property is had by something.”

All first-order languages have certain syntactic features in common: individual variables ( $u, v, w, x, \dots$ ), quantifiers ( $\exists$  and  $\forall$ ), connectives ( $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$ ) and, usually, the identity or equality symbol ( $=$ ). Sometimes, these symbols are written a bit differently, as shown in the following table, but this doesn’t affect their meaning.

Where one first-order language really differs from another is in its particular choice of predicate symbols, individual constants (names), and function symbols. Thus one can specify a first-order language by describing the predicates, names, and function symbols it uses. Tarski’s World does not consider languages with function symbols (like the addition sign), so we set these aside from now on.

Since first-order logic is concerned with properties of all first-order languages, logicians often study what is known as an “uninterpreted”

Our symbols	Common equivalents
-------------	--------------------

$\neg$	$\sim, -$
$\wedge$	$\&, \cdot$
$\rightarrow$	$\supset$
$\leftrightarrow$	$\equiv$
$\forall x$	$(x), (\forall x), \bigwedge x, \Pi x$
$\exists x$	$(\exists x), \bigvee x, \Sigma x$

first-order language, one in which the predicates are not given a fixed meaning. The language we use in Tarski's World, by contrast, is fully interpreted. Each predicate symbol has a fixed meaning. This has some important consequences which we will discuss later.

## A.2 Individual constants

Individual constants are simply symbols that are used to refer to some fixed individual object or other. They are the first-order analogue of names. For example we might use *John* as an individual constant to denote a particular person, in which case it would basically work exactly the way names work in English. The main difference is that in logic we require that each individual constant refer to exactly one object, and of course the name *John* in English can be used to refer to many different people. There are also names in English that do not refer to any actually existing object, for example *Pegasus*; we don't allow such names in first-order logic. What we do allow, though, is for one object to have more than one name; thus *John* and *Jack* might refer to the same individual. In Tarski's World the available individual constants are *a, b, c, d, e, f, n1, n2, . . .* (You are in command of the first six, in that you can use them to name objects. Tarski's World is in charge of the individual constants *n1, n2, . . .* It uses them to name objects when playing the game with you.)

## A.3 Predicate symbols

Predicate symbols are symbols used to denote some property of objects, or some relation between objects. Each such predicate symbol comes with an "arity," a number that tells you how many individual constants the predicate symbol needs in order to form a sentence. If the arity of

a predicate symbol  $P$  is 1, then  $P$  will be used to denote some property of objects, and so will require one “argument” (a name) to make a claim. For example, we might use the predicate symbol `Home` of arity 1 to denote the property of being at home. We could then combine this with the argument `John` to get the expression `Home(John)`, which expresses the claim that John is at home. If the arity of  $P$  is 2, then  $P$  will be used to represent a relation between two objects. Thus, we might use the expression `Taller(John, Mary)` to express a claim about John and Mary, the claim that John is taller than Mary. Similarly, we can have predicate symbols of any arity. However, in Tarski’s World we restrict ourselves to the arities 1, 2, and 3. Indeed, the only predicate symbols used in Tarski’s World are the following:

- Arity 1: `Cube`, `Tet`, `Dodec`,  
`Small`, `Medium`, `Large`
- Arity 2: `Smaller`, `Larger`, `SameSize`,  
`LeftOf`, `RightOf`, `SameCol`  
`BackOf`, `FrontOf`, `SameRow`  
`SameShape`, `Adjoins`, `=`
- Arity 3: `Between`

Tarski’s World assigns each of these predicates a fixed interpretation, one reasonably consistent with the English cognate. These are listed in Table 2. You can get the hang of them by working through Exercise 2.1, page 9.

#### A.4 Atomic sentences

The simplest kinds of claims that can be made in a first-order language are those made with a single predicate and the appropriate number of individual constants. A sentence formed by a predicate followed by the right number of names is called an *atomic* sentence. For example `Taller(John, Mary)` and `Cube(a)` are atomic sentences. In the case of the identity symbol, we put the two required names on either side of the predicate, as in `a = b`. This is called “infix” notation, since the predicate symbol `=` appears in between its two arguments. With the other predicates we use “prefix” notation: the arguments follow the predicate.

TABLE 2 Blocks language predicates.

Atomic Sentence	Interpretation
Tet(a)	a is a tetrahedron
Cube(a)	a is a cube
Dodec(a)	a is a dodecahedron
Small(a)	a is small
Medium(a)	a is medium
Large(a)	a is large
SameSize(a, b)	a is the same size as b
SameShape(a, b)	a is the same shape as b
Larger(a, b)	a is larger than b
Smaller(a, b)	a is smaller than b
SameCol(a, b)	a is in the same column as b
SameRow(a, b)	a is in the same row as b
Adjoins(a, b)	a and b are located on adjacent (but not diagonally) squares
LeftOf(a, b)	a is located nearer to the left edge of the grid than b
RightOf(a, b)	a is located nearer to the right edge of the grid than b
FrontOf(a, b)	a is located nearer to the front of the grid than b
BackOf(a, b)	a is located nearer to the back of the grid than b
Between(a, b, c)	a, b and c are in the same row, column, or diagonal, and a is between b and c

## A.5 Connectives

To form more complex claims from our atomic sentences, we use the connectives  $\neg$ ,  $\wedge$ ,  $\vee$ ,  $\rightarrow$ , and  $\leftrightarrow$ . The meanings of these symbols are as follows.

### Negation symbol ( $\neg$ )

This symbol is used to express negation in our language, the notion we commonly express in English using terms like *not*, *it is not the case that*, *non-* and *un-*. In first-order logic, we always apply this symbol to

the front of a sentence to be negated, while in English there is a much more subtle system for expressing negative claims. For example, the English sentences *John isn't home* and *It is not the case that John is home* have the same first-order translation:

$$\neg \text{Home}(\text{John})$$

This sentence is true if and only if  $\text{Home}(\text{John})$  isn't true, that is, just in case John isn't home. More generally, a sentence  $\neg A$  is true if and only if  $A$  is false.

### Conjunction symbol ( $\wedge$ )

This symbol is used to express conjunction in our language, the notion we normally express in English using terms like *and*, *moreover*, and *but*. In first-order logic, this connective is always placed between two sentences, whereas in English we can also conjoin nouns, verbs, and other parts of speech. For example, the English sentences *John and Mary are home* and *John is home and Mary is home* both have the same first-order translation:

$$\text{Home}(\text{John}) \wedge \text{Home}(\text{Mary})$$

A sentence  $A \wedge B$  is true if and only if both  $A$  and  $B$  are true.

### Disjunction symbol ( $\vee$ )

This symbol is used to express disjunction in our language, the notion we express in English using *or*. In first-order logic, this connective, like the conjunction sign, is always placed between two sentences, whereas in English we can also disjoin nouns, verbs, and other parts of speech. For example, the English sentences *John or Mary is home* and *John is home or Mary is home* both have the same first-order translation:

$$\text{Home}(\text{John}) \vee \text{Home}(\text{Mary})$$

Although the English *or* is sometimes used in an “exclusive” sense, to say that *exactly* one of the two disjoined sentences is true, the first-order logic  $\vee$  is always given an “inclusive” interpretation: it means that at least one and possibly both of the two disjoined sentences is true. Thus, our sample sentence is true if John is home alone, if Mary is home alone, or if both John and Mary are home. More generally, a sentence  $A \vee B$  is true if at least one of  $A$  or  $B$  is true.

If we wanted to express the exclusive sense of *or* in the above example, we could do it as follows:

$$(\text{Home}(\text{John}) \vee \text{Home}(\text{Mary})) \wedge \\ \neg(\text{Home}(\text{John}) \wedge \text{Home}(\text{Mary}))$$

As you can see, this sentence says that John or Mary is home, but they are not both home. Another important English expression that we can capture without introducing additional symbols is *neither . . . nor*. Thus *Neither John nor Mary is at home* would be expressed as:

$$\neg(\text{Home}(\text{John}) \vee \text{Home}(\text{Mary}))$$

This says that it's not the case that at least one of them is at home, i.e., that neither of them is home.

### Material conditional symbol ( $\rightarrow$ )

This symbol is used to combine two sentences  $A$  and  $B$  to form a new sentence  $A \rightarrow B$ , called a *material conditional*. The sentence  $A \rightarrow B$  is true if and only if either  $A$  is false or  $B$  is true (or both). To put it differently, this sentence is only false if the antecedent  $A$  is true and the consequent  $B$  is false. Thus,  $A \rightarrow B$  is really just another way of saying  $\neg A \vee B$ . Tarski's World in fact treats the former as an abbreviation of the latter.

We can come fairly close to an adequate English rendering of the conditional expression  $A \rightarrow B$  with the sentence *If A then B*. At any rate, it is clear that this English conditional, like the material conditional, is false if  $A$  is true and  $B$  is false. Thus, we will translate, for example, *If John is home then Mary is at the library* as:

$$\text{Home}(\text{John}) \rightarrow \text{Library}(\text{Mary})$$

Other English expressions that can frequently be translated using the material conditional  $A \rightarrow B$  include: *A only if B*, *B provided A*, and *B whenever A*. We also use  $\rightarrow$  in combination with  $\neg$  to translate sentences of the form *Unless A, B* or *B unless A*. These mean the same thing as *B if not A*, and so are translated as  $\neg A \rightarrow B$ .

While we will always translate the English *if . . . then* using  $\rightarrow$ , there are in fact many uses of the English expression that cannot be adequately expressed using the material conditional. For example, the sentence,

*If Mary had been at home then John would have been there too*

can be false even if Mary was not in fact at home. But the first-order sentence,

$$\text{Home}(\text{Mary}) \rightarrow \text{Home}(\text{John})$$



is automatically true if Mary is not at home. We will not discuss such uses further since they go beyond first-order logic.

The most important use of  $\rightarrow$  in first-order logic is not in conjunction with the above expressions, but rather with universally quantified sentences, sentences of the form *All A's are B's* and *Every A is a B*. The analogous first-order sentences have the form:

For every object  $x$  ( $A(x) \rightarrow B(x)$ )

This says that any object you pick will either fail to be an A or be a B. We will discuss such sentences in more detail later, once we have variables and the symbol  $\forall$  at our disposal.

### Biconditional symbol ( $\leftrightarrow$ )

Our final connective is the material biconditional symbol. A sentence of the form  $A \leftrightarrow B$  is true if and only if A and B have the same truth value, that is, either they are both true or both false. In English this is commonly expressed using the expression *if and only if*, and, in mathematical discourse, *just in case*. So, for example, the sentence *Mary is home if and only if John is home* would be translated as:

$\text{Home}(\text{Mary}) \leftrightarrow \text{Home}(\text{John})$

Most logic books treat a sentence of the form  $A \leftrightarrow B$  as an abbreviation of  $(A \rightarrow B) \wedge (B \rightarrow A)$ . Tarski's World also uses this abbreviation.

## A.6 Variables

Variables are a kind of auxiliary symbol. In some ways they behave like individual constants, since they can appear in the list of arguments immediately following a predicate. But in other ways they are very different from individual constants. In particular, their semantic function is not to refer to objects. Rather they are placeholders that indicate relationships between quantifiers and the argument positions of various predicates. This will become clearer with our discussion of quantifiers. First-order logic assumes an infinite list of variables so that one never runs out of variables, no matter how complex a sentence may get. But Tarski's World uses only six variables, namely,  $u, v, w, x, y,$  and  $z$ . This imposes an expressive limitation on the language used in Tarski's World, but in practice one rarely has call for more than four or five variables.<sup>3</sup>

---

<sup>3</sup>For an exploration of this expressive limitation, see Exercises 4.10–4.13, starting on page 58.

## A.7 Atomic wffs

Now that we have variables at our disposal, we can produce expressions that look like atomic sentences, except that there are variables in place of some individual constants. For example  $\text{Home}(x)$  and  $\text{Taller}(\text{John}, y)$  are such expressions. We call them *atomic well-formed formulas*, or *atomic wffs*. They are not sentences, but they will be used in conjunction with quantifier symbols to build sentences.

## A.8 Quantifiers

Our language contains two quantifier symbols,  $\forall$  and  $\exists$ . The reason these are called “quantifiers” is that they can be used to express certain rudimentary claims about the number (or quantity) of things that satisfy some condition. Specifically they allow us to say that all objects satisfy some condition, or that at least one object satisfies some condition. When used in conjunction with identity ( $=$ ) and the various connectives, they can also be used to express more complex numerical claims, say that there are exactly two things that satisfy some condition.

### Universal quantifier ( $\forall$ )

This symbol is used to express universal claims, those we express in English using such terms as *everything*, *each thing*, *all things*, and *anything*. It is always used in connection with one of the variables  $u, v, w, x, \dots$ , and so is said to be a variable binding operator. The combination  $\forall x$  is read “for every object  $x$ ,” or (somewhat misleadingly) “for all  $x$ .” If we wanted to translate the (rather unlikely) English sentence *Everything is at home* into first-order logic, we would use the expression

$$\forall x \text{Home}(x)$$

This says that every object  $x$  meets the following condition:  $x$  is at home. Or, to put it more naturally, it says that everything whatsoever is at home.

Of course we rarely make such unconditional claims about absolutely everything. More common are restricted universal claims like *Every doctor is smart*. This sentence would be translated as:

$$\forall x (\text{Doctor}(x) \rightarrow \text{Smart}(x))$$

This sentence claims that given any object at all—call it  $x$ —if  $x$  is a doctor, then  $x$  is smart. To put it another way, the sentence says that if you pick anything at all, you’ll find either that it is not a doctor or that it is smart (or perhaps both).

### Existential quantifier ( $\exists$ )

This symbol is used to express existential claims, those we express in English using such terms as *something*, *at least one thing*, *a*, and *an*. It too is always used in connection with one of the variables  $u, v, w, x, \dots$ , and so is a variable binding operator. The combination  $\exists x$  is read “for some object  $x$ ,” or (somewhat misleadingly) “for some  $x$ .” If we wanted to translate the English sentence *Something is at home* into first-order logic, we would use the expression

$$\exists x \text{ Home}(x)$$

This says that some object  $x$  meets the following condition:  $x$  is at home.

While it is possible to make such claims, it is more common to assert that something of a particular kind meets some condition, say *Some doctor is smart*. This sentence would be translated as:

$$\exists x (\text{Doctor}(x) \wedge \text{Smart}(x))$$

This sentence claims that some object, call it  $x$ , meets the complex condition:  $x$  is both a doctor and smart. Or, more colloquially, it says that there is at least one smart doctor.

## A.9 Wffs and sentences

Notice that in some of the above examples, we formed *sentences* out of complex expressions that were not themselves sentences, expressions like

$$\text{Doctor}(x) \wedge \text{Smart}(x)$$

that contain variables not bound by any quantifier. Thus, to systematically describe all the sentences of first-order logic, it is convenient to first describe a larger class, the so-called well-formed formulas, or *wffs*.

We have already explained what an atomic wff is: any  $n$ -ary predicate followed by  $n$  variables or individual constants. Using these as our atomic building blocks, we can construct more complicated wffs by repeatedly applying the following rules:

1. If  $A$  is a wff, so is  $\neg A$
2. If  $A_1, \dots, A_n$  are wffs, so is  $(A_1 \wedge \dots \wedge A_n)$
3. If  $A_1, \dots, A_n$  are wffs, so is  $(A_1 \vee \dots \vee A_n)$
4. If  $A$  and  $B$  are wffs, so is  $(A \rightarrow B)$
5. If  $A$  and  $B$  are wffs, so is  $(A \leftrightarrow B)$

6. If  $A$  is a wff and  $\nu$  is a variable (i.e., one of  $u, v, w, x, y, z$ ), then  $\forall\nu A$  is a wff, and any occurrence of  $\nu$  in  $A$  is said to be bound.
7. If  $A$  is a wff and  $\nu$  is a variable, then  $\exists\nu A$  is a wff, and any occurrence of  $\nu$  in  $A$  is said to be bound.

The way these rules work is pretty straightforward. For example, starting from the atomic wffs  $\text{Cube}(x)$  and  $\text{Small}(x)$  we can apply rule 2 to get the wff:

$$(\text{Cube}(x) \wedge \text{Small}(x))$$

Similarly, starting from the atomic wff  $\text{LeftOf}(x, y)$  we can apply rule 7 to get the wff:

$$\exists y \text{LeftOf}(x, y)$$

In this formula the variable  $y$  has been bound by the quantifier  $\exists y$ . The variable  $x$ , on the other hand, has not been bound; it is still “free.”

The rules can also be applied to complex wffs, so from the above two wffs and rule 4 we can generate the following wff:

$$((\text{Cube}(x) \wedge \text{Small}(x)) \rightarrow \exists y \text{LeftOf}(x, y))$$

A *sentence* is a wff with no unbound (free) variables. None of these wffs is a sentence, since they all contain unbound variables. To get a sentence from the last of these, we can simply apply rule 6 to produce:

$$\forall x ((\text{Cube}(x) \wedge \text{Small}(x)) \rightarrow \exists y \text{LeftOf}(x, y))$$

Here all occurrences of the variable  $x$  have been bound by the quantifier  $\forall x$ . So this wff is a sentence since it has no free variables. It claims that for every object  $x$ , if  $x$  is both a cube and small, then there is an object  $y$  such that  $x$  is to the left of  $y$ . Or, to put it more naturally, every small cube is to the left of something.

These rules can be applied over and over again to form more and more complex wffs. So, for example, repeated application of the first rule to the wff  $\text{Home}(\text{John})$  will give us all of the following wffs:

$$\begin{aligned} &\neg \text{Home}(\text{John}) \\ &\neg\neg \text{Home}(\text{John}) \\ &\neg\neg\neg \text{Home}(\text{John}) \\ &\vdots \end{aligned}$$

Since none of these contains any variables, and so no free variables, they are all sentences. They claim, respectively, that John is not home, that it is not the case that John is not home, that it is not the case that it is not the case that John is not home, and so forth.

A comment should be made about rules 2 and 3. Logic books frequently allow you to conjoin or disjoin only two wffs at a time. To make things more readable, Tarski's World allows an arbitrary number of wffs to be conjoined or disjoined in a single application of either of these rules. Thus the English sentence *John, Mary and Tom are at home* could be translated:

$$(\text{Home}(\text{John}) \wedge \text{Home}(\text{Mary}) \wedge \text{Home}(\text{Tom}))$$

If we only allowed conjunctions of two wffs at a time, we'd have to use the rule twice, and the result would have an additional set of parentheses in it somewhere. The reason we can allow disjunctions and conjunctions of any length is that with these connectives the various possible groupings make no difference: the connectives are said to be associative. Thus, for example,  $(A \wedge (B \wedge C))$  means the same thing as  $((A \wedge B) \wedge C)$ . The conditional and biconditional, by contrast, are not associative.

Tarski's World also, by the way, allows us to drop the outermost parentheses in a wff, if we want.

We have said that a sentence is a wff with no free variables. However, it can sometimes be a bit tricky deciding whether a variable is free in a wff. For example, there are no free variables in the wff,

$$\exists x (\text{Doctor}(x) \wedge \text{Smart}(x))$$

However there *is* a free variable in the deceptively similar wff,

$$\exists x \text{ Doctor}(x) \wedge \text{Smart}(x)$$

Here the last occurrence of the variable  $x$  is still free. We can see why this is the case by thinking about when the existential quantifier was applied in building up these two formulas. In the first one, the parentheses show that the quantifier was applied to the conjunction  $(\text{Doctor}(x) \wedge \text{Smart}(x))$ . As a consequence, all occurrences of  $x$  in the conjunction were bound by this quantifier. In contrast, the lack of parentheses show that in building up the second formula, the existential quantifier was applied to form  $\exists x \text{ Doctor}(x)$ , thus binding only the occurrence of  $x$  in  $\text{Doctor}(x)$ . This formula was then conjoined with  $\text{Smart}(x)$ , and so the latter's occurrence of  $x$  did not get bound.

Parentheses, as you can see from this example, make a big difference. They are the way you can tell what the "scope" of a quantifier is, that is, which variables fall under its influence and which don't.

## A.10 Satisfaction and truth

When we described the meanings of our various connectives, we told you how the truth value of a complex sentence, say  $\neg A$ , depends on the truth values of its constituents, in this case  $A$ . But we didn't give you similar rules for determining the truth value of quantified sentences. The reason is simple: the expression we apply the quantifier to in order to build a sentence is usually not itself a sentence. We could hardly tell you how the truth value of  $\exists x \text{Cube}(x)$  depends on the truth value of  $\text{Cube}(x)$ , since this latter expression is not a sentence at all: it contains a free variable. Because of this, it is *neither* true *nor* false.

To describe when quantified sentences are true, we need to introduce the auxiliary notion of *satisfaction*. The basic idea is simple, and can be illustrated with a few examples. We say that an object satisfies the atomic wff  $\text{Cube}(x)$  if and only if the object is a cube. Similarly, we say an object satisfies the complex wff  $\text{Cube}(x) \wedge \text{Small}(x)$  if it is both a cube and small. As a final example, an object satisfies the wff  $\text{Cube}(x) \vee \neg \text{Large}(x)$  if it is either a cube or not large (or both).

Different logic books treat satisfaction in somewhat different ways. We will describe the one that is built into the way that Tarski's World checks the truth of quantified sentences. Suppose  $A(x)$  is a wff containing  $x$  as its only free variable, and suppose we wanted to know whether a given object satisfies  $A(x)$ . If this object has a name, say  $b$ , then form a new *sentence*  $A(b)$  by replacing all free occurrences of  $x$  by the individual constant  $b$ . If the new sentence  $A(b)$  is true, then the object satisfies the formula  $A(x)$ ; if the sentence is not true, then the object does not satisfy the formula.

This works fine as long as the given object has a name. However, first-order logic does not require that every object have a name. How can we define satisfaction for objects that don't have names? It is for this reason that Tarski's World has, in addition to the individual constants  $a, b, c, d, e$  and  $f$ , a further list  $n1, n2, n3, \dots$  of individual constants. If we want to know of an object that does not have a name, whether it satisfies the formula  $A(x)$ , we choose the first of these individual constants not in use, say  $n7$ , temporarily name the given object with this symbol, and then check to see whether the sentence  $A(n7)$  is true. Thus any small cube satisfies  $\text{Cube}(x) \wedge \text{Small}(x)$ , because if we were to use  $n7$  as a name of such a small cube, then  $\text{Cube}(n7) \wedge \text{Small}(n7)$  would be a true sentence.

Once we have the notion of satisfaction, we can easily describe when a sentence of the form  $\exists x A(x)$  is true. It will be true if and only if there is at least one object that satisfies the constituent wff  $A(x)$ . So  $\exists x (\text{Cube}(x) \wedge \text{Small}(x))$  is true if there is at least one object that satisfies  $\text{Cube}(x) \wedge \text{Small}(x)$ , that is, if there is at least one small cube. Similarly, a sentence of the form  $\forall x A(x)$  is true if and only if every object satisfies the constituent wff  $A(x)$ . Thus  $\forall x (\text{Cube}(x) \rightarrow \text{Small}(x))$  is true if every object satisfies  $\text{Cube}(x) \rightarrow \text{Small}(x)$ , that is, if every object either isn't a cube or is small.

This approach to satisfaction is conceptually simpler than some. A more common approach is to avoid the introduction of new names by defining satisfaction for wffs with an arbitrary number of free variables. For example, one says that the *pair* of individuals John and Mary satisfy  $\text{Taller}(x, y)$  if the first of them is taller than the second. The notion is extended to complex wffs in the natural way. The only point in defining satisfaction is to be able to define truth for quantified sentences, and the two approaches are entirely equivalent for this purpose.

### A.11 Game rules

One of the main features of Tarski's World is its use of a game to help you understand just what the import of some claim is, especially when the claim does not have the truth value you expect. This game is based on simple observations that follow directly from the meanings of the logical symbols described above. The basic idea is that if you use a complex sentence to make a claim, then besides being committed to the truth of the complex sentence, you incur various commitments involving its constituents. For example, if you are committed to the truth of  $\neg A$  then you are committed to the falsity of  $A$ , and if you are committed to the truth of  $A \vee B$  then you are committed to one of  $A$  or  $B$  being true. Similarly, if you are committed to the truth of  $\exists x A(x)$ , then you are committed to there being some object satisfying the formula  $A(x)$ . The game rules, which are set out in table 3, are all based on these simple observations.

There is one somewhat subtle point that needs to be made about this way of describing the game. Sometimes you can tell that a complex sentence is true without actually knowing in advance how to play the game and win. For example, if you have a sentence of the form  $A \vee \neg A$ , then you know that it is true, no matter how the world is. But if  $A$  is

TABLE 3 Summary of the game rules

FORM	YOUR COMMITMENT	PLAYER TO MOVE	GOAL
$P \vee Q$	TRUE	you	Choose one of $P, Q$ that is true.
	FALSE	Tarski's World	
$P \wedge Q$	TRUE	Tarski's World	Choose one of $P, Q$ that is false.
	FALSE	you	
$\exists x P(x)$	TRUE	you	Choose some $b$ that satisfies the wff $P(x)$ .
	FALSE	Tarski's World	
$\forall x P(x)$	TRUE	Tarski's World	Choose some $b$ that does not satisfy $P(x)$ .
	FALSE	you	
$\neg P$	either	—	Replace $\neg P$ by $P$ and switch commitment.
$P \rightarrow Q$	either	—	Replace $P \rightarrow Q$ by $\neg P \vee Q$ and keep commitment.
$P \leftrightarrow Q$	either	—	Replace $P \leftrightarrow Q$ by $(P \rightarrow Q) \wedge (Q \rightarrow P)$ and keep commitment.

quite complex, or if you have imperfect information about the world, you may not know which of  $A$  or  $\neg A$  is true. In such a case you would be willing to commit to the truth of the disjunction without knowing just how to play the game and win. You know that there is a winning strategy for the game, but just don't know what it is.

Since there is a moral imperative to live up to one's commitments, the use of the term "commitment" in describing the game is a bit



misleading. You are perfectly justified in asserting the truth of  $A \vee \neg A$ , even if you do not happen to know your winning strategy for playing the game. Indeed, you would be worse than foolish to claim that the sentence is *not* true. But if you do claim that  $A \vee \neg A$  is true, and then play the game, you will be asked to tell which of  $A$  or  $\neg A$  you think is true. Tarski's World has been designed so you can always get complete information about the world, and so always live up to such commitments.<sup>4</sup>

## A.12 Logical equivalences

We use the abbreviation  $P \Leftrightarrow Q$  to indicate that  $P$  and  $Q$  are *logically equivalent* formulae. The symbol  $\Leftrightarrow$  is not a symbol of first order logic, but rather a shorthand way of expressing a fact about two first order formulae.

The game rules for material implication and biconditional treat the  $\rightarrow$  and  $\leftrightarrow$  connectives as abbreviations for equivalent formulae which do not use those connectives.

$$\begin{aligned} A \leftrightarrow B &\Leftrightarrow (A \rightarrow B) \wedge (B \rightarrow A) \\ A \rightarrow B &\Leftrightarrow \neg A \vee B \end{aligned}$$

This indicates a particular kind of redundancy in the language that we have described. Any sentence that uses these connectives has an equivalent (possibly longer) sentence that does not use them. The inclusion of these connectives in our language allows us to write some sentences in a form that approximates their English expression, but does not add to the expressive power of the language.

In fact we do not need all three of the connectives  $\wedge$ ,  $\vee$  and  $\neg$ , as the following equivalences show

$$\begin{aligned} A \wedge B &\Leftrightarrow \neg(\neg A \vee \neg B) \\ A \vee B &\Leftrightarrow \neg(\neg A \wedge \neg B) \end{aligned}$$

The first shows that we could eliminate all uses of  $\wedge$  in favor of  $\neg$  and  $\vee$ , and then second that we could alternatively eliminate all uses of  $\vee$  in favor of  $\neg$  and  $\wedge$ .

These previous two equivalences are possibly more familiar as the de Morgan equivalences.

$$\begin{aligned} \neg(A \wedge B) &\Leftrightarrow \neg A \vee \neg B \\ \neg(A \vee B) &\Leftrightarrow \neg A \wedge \neg B \end{aligned}$$

---

<sup>4</sup>See Exercise 4.1, page 53, for an exploration of this topic.

Other equivalences allow us to rearrange the connectives within a sentence, for example the distribution laws describe the relationship between  $\wedge$  and  $\vee$ .

$$\begin{aligned}(A \wedge B) \vee C &\Leftrightarrow (A \vee C) \wedge (B \vee C) \\ (A \vee B) \wedge C &\Leftrightarrow (A \wedge C) \vee (B \wedge C)\end{aligned}$$

The combination of these equivalences allows the definition of *normal forms* for sentences. A sentence is said to be in *negation normal form* if all of the negation symbols in the sentence apply only to atomic formulae, i.e. there are no negated conjunctions or disjunctions. The word *literal* is used to refer to atomic formulae or their negations. A sentence is said to be in *conjunctive normal form* if it is the conjunction of disjunctions of literals. Analogously, a formula is in *disjunctive normal form* if it is the disjunction of conjunctions of literals.

### A.13 Validity and logical consequence

Two of the most important notions in logic are those of *logical consequence* and *logical validity*. Suppose we have two sentences  $A$  and  $B$ . What does it mean to say that  $B$  is a logical consequence of  $A$ ? Intuitively, it means that there is no way for  $A$  to be true without  $B$  also being true. Or, to put it differently, no matter how the world is, if  $A$  is true, then so is  $B$ . For example, the sentence *Every large cube is in back of  $\mathbf{b}$*  is a logical consequence of *Every cube is in back of  $\mathbf{b}$* , since no matter how the world is, if the latter is true, so is the former. This follows simply from the meanings of the two sentences. So let's say that  $B$  is a logical consequence of  $A$  if it is impossible, simply due to their meanings, for  $A$  to be true and  $B$  to be false.

This definition can be expanded to define what it means for a sentence  $B$  to be a logical consequence of some (finite or infinite) set  $T$  of sentences. Namely,  $B$  is a logical consequence of  $T$  if and only if it is impossible for every sentence in  $T$  to be true, while  $B$  is false, due simply to the meanings of the sentences.

This definition makes sense even if the set  $T$  is empty. In this case we automatically know that all the sentences in  $T$  are true, since there are none, and so  $B$  will be a consequence of  $T$  only if it is impossible for  $B$  to be false. We then say that the sentence  $B$  is *logically valid*. In other words,  $B$  is logically valid if and only if it is impossible, simply due to its meaning, for  $B$  to be false.

As a rather obvious example of a logically valid sentence, consider

any sentence of the form  $A \vee \neg A$ . No matter what the world is like, this sentence is true, since it just claims that either  $A$  is true or it isn't true.

A word of warning is in order here. While most logic texts define these notions in very much the way we have here, there is a subtle difference that bears noting, since it could lead you astray. The difference stems from the fact, mentioned earlier, that Tarski's World deals with an *interpreted language*, whereas most logic textbooks deal with (partially) uninterpreted languages, languages where the basic predicate symbols other than  $=$  are not given fixed meanings. Consider, for example, the sentence

$$\forall x \forall y (\text{LeftOf}(x, y) \rightarrow \text{RightOf}(y, x))$$

Tarski's World treats **LeftOf** and **RightOf** as interpreted predicates, predicate symbols whose interpretation is that of being left of and being right of, respectively. Thus, for Tarski's World this is a logically valid sentence. No matter how objects are arranged in the world, if  $b$  is to the left of  $c$ , then  $c$  is to the right of  $b$ .

Most logic texts will not consider this sentence to be logically valid. Why? Because they do not treat the binary predicate symbols **LeftOf** and **RightOf** as having a fixed meaning. The only symbols most books treat as having a fixed meaning are the connectives, quantifiers, and the identity symbol. They consider worlds in which **LeftOf** is interpreted as, say, the relation of being taller than, and **RightOf** is interpreted as, say, the relation of being richer than. Then, since it is possible for one object to be taller than a second without the second being richer than it, the sentence will not be deemed valid.

To apply the above definitions (of logical consequence and logical validity) to a partially uninterpreted language, one must think of it in a somewhat different way. For example, the definition of  $B$  being a logical consequence of  $T$  would really amount to the following: no matter how the uninterpreted symbols are interpreted, and no matter how the world is, if all the sentences in  $T$  are true, then so is  $B$ .<sup>5</sup>

We do not cover the topic of demonstrating that one formula is a consequence of others in this package. For a detailed treatment, see our *Language, Proof and Logic* package. It is, however, easy to show that a formula,  $A$ , is not a consequence of some others,  $T$ , using Tarski's World. If it is possible to build a world in which the formulae in  $T$  are true while  $A$  is false, then this world demonstrates that  $A$  is not a consequence of  $T$ .

---

<sup>5</sup>For an exploration of these topics, see Exercises 4.2–4.7, starting on page 54.

We call this a *counterexample* world. We note that this demonstrates both that the sentence is not a consequence regardless of whether we treat the language as interpreted or partially-interpreted.

## Appendix B

---

# Using Tarski's World 5.x

Tarski's World lets you represent simple, three-dimensional worlds inhabited by geometric blocks of various kinds and sizes, and test first-order sentences to see whether they are true or false in those worlds. We begin with instructions on how to start and stop Tarski's World, and explain the basic layout of the screen.

### B.1 Getting started

The Tarski's World application is contained inside the folder called Tarski's World Folder. Also in this folder is a folder called TW Exercise Files, in which you will find the Tarski's World exercise files referred to in the book.

#### B.1.1 Launching Tarski's World

To run Tarski's World, double-click on the application icon, which looks like an upside-down "A" floating next to a tetrahedron (pyramid). After a moment, the Tarski's World application will appear on your screen. Pull down each of the menus that appear in the menubar (**File**, **Edit**, **Display**, ...) to see the sorts of commands they contain.

#### B.1.2 The main windows

There are three main windows on the screen (four on Windows). The *world window* is the black window in the upper left. It contains a grid on which blocks are placed and, on the left, three square buttons showing a tetrahedron, a cube, and a dodecahedron (or soccer ball). Feel free to click on these buttons to see what happens.

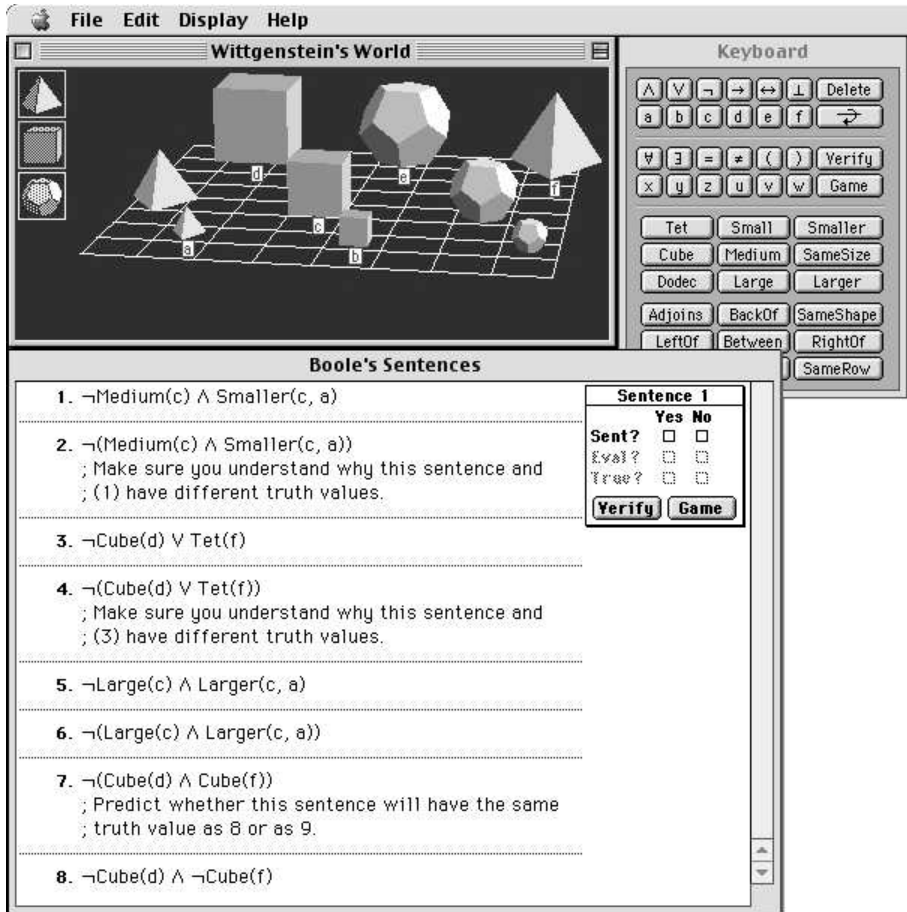


FIGURE 3 Main windows in Tarski's World 5.x.

The *sentence window* is the white window across the bottom of the screen. At first it contains only the numeral “1” inside. This is where sentences are entered and evaluated to see whether they are true or false in the world represented in the world window. Click once in the sentence window to activate it. Feel free to type something in the sentence window, say, “I’d rather be in Philadelphia.”

Finally, the *keyboard window* is the window to the right of the world window. This is the window we generally use to enter sentences of first-order logic. Click on the window to activate it. Feel free to play around by clicking on the buttons in the keyboard window.

There is one last thing to notice. On the right of the sentence window you will see the *evaluation box* (on the Macintosh) or the *inspector window* (in Microsoft Windows). This is where Tarski’s World will display the results of its evaluations of sentences and, in Windows, where you change the size and shape of blocks. We’ll say more about this later.

### B.1.3 Opening saved files

Both worlds and sentence lists can be saved as files on your disk. Indeed, many prepackaged world and sentence files come with Tarski’s World. To open a saved file, you use the **Open. . .** command on the **File** menu.

**Macintosh:** To open a file, pull down the **File** menu and choose **Open. . .**. If the world window was active when you did this, you will be presented with a list of any existing world files in the current folder. If the sentence window was active, you will be presented with a list of sentence files. You can get a list of the world files by clicking on the **World files** button at the bottom of the open dialog box. Similarly, you can get the list of sentence files by clicking on the **Sentence files** button.

**Windows:** To open a file, pull down the **File** menu and choose **Open. . .**. This will display a subsidiary menu, from which you can choose **World**, if you want to open a world file, or **Sentence**, if you want to open a sentence file. Once you have done this, you will be given a dialog box that allows you to navigate among the folders and disks on your computer.

You will have to navigate to the right folder to find the prepackaged files, which are in TW Exercise Files. Find this folder, select it, and then click **Open**, or simply double-click on the name. Feel free to open one of the files you see, say, *Ackermann’s World*, but if you make any changes to the world, don’t save them.

### B.1.4 Starting new files

If you want to start a new world or sentence file, choose **New...** from the **File** menu. You will then have to specify whether you want a world file or new sentence file.

### B.1.5 Saving a file

**Macintosh:** To save a list of sentences, choose **Save Sentences** or **Save Sentences As...** from the **File** menu. To save a world, choose **Save World** or **Save World As...** from the **File** menu.

**Windows:** To save a list of sentences, choose **Save** followed by **Sentence** from the **File** menu or choose **Save As...** followed by **Sentence**. To save a world, choose **Save** followed by **World** from the **File** menu or choose **Save As...** followed by **World**.

If the file has never been saved before, a dialog box will appear giving you the option of naming the file you are about to create. If you were to hit the return key, or click the **Save** button, the file would be saved with the default name. You should type in some other name before hitting the return key or clicking **Save**. You should also make sure you are saving the file where you want it. Check the directory name at the top of the save dialog box. If you're not in the folder where you want to save the file, navigate to the right one by clicking on this name. If you are not familiar with navigating around the disks and folders on your computer, you should ask a fellow student or younger sibling for help.

Once a file has been saved, the name of the file appears in the title bar of the corresponding window. If you are working on a named file, the **Save** and **Save As...** commands behave differently. The first will save a new version of the file under the same name, and the old version will be gone. The second gives you a chance to create a new file, with a new name, and keep the old file, with its name. For this reason, **Save As...** is the safer of the two options.

All files created by Tarski's World can be read by either the Macintosh or Windows version of the application.

### B.1.6 Quitting (Exiting) Tarski's World

Eventually you will want to leave Tarski's World. To do this, choose **Quit** (on the Macintosh) or **Exit** (in Windows) from the **File** menu. If you've made any unsaved changes to the files, Tarski's World will give you a chance to save them.



## B.2 The world window

### B.2.1 Adding blocks

To put a block on the grid, simply click the button on the left that shows the type of block you want. Try this out. On the Macintosh, also try holding down the Option key while you click on a block button. This opens up the block's "parameter window," allowing you to specify other characteristics of the block. Click **OK** when you are satisfied with the characteristics.

### B.2.2 Naming blocks

**Macintosh:** To name a block already on the grid, double-click on the block. This will open the Object Parameter window displaying the block's current shape, size, and names, if any. To give the block a name, simply choose the desired name by clicking in the box next to it. When you have chosen the block's name(s), click **OK**. If you know that you want to name a block (or change its size) when you add it to the world, you can save a bit of effort by holding down the Option key while you click the button to add the block. This immediately opens the Object Parameter window so that you can give the block a name or change its size.

**Windows:** To name a block already on the grid, click on the block. This will highlight the block and bring up the *block inspector* in the inspector window. The block inspector will display the block's current shape, size, and names, if any. To give the block a name, simply choose the desired name by clicking in the box next to it. When you have chosen the block's name(s), click **OK**.

In first-order logic, one object can have several names, but two objects cannot share the same name. Hence Tarski's World lets you give a block more than one name, but once a name is used, that name cannot be assigned to another block.

### B.2.3 Moving blocks

To move a block, position the cursor over the block and drag it to the desired position. That is, move the mouse's arrow over the block, until the arrow turns into an  $\times$  (+ on Windows). Then, with the button depressed, move the mouse until the block is where you want it. If you move the block too close to the edge, it will fall off.

### B.2.4 Sizing and shaping blocks

To change a block's size or shape, follow the instructions for naming the block (in section B.2.2). You can alter the size or shape by clicking in the appropriate circles in the parameter window (on the Macintosh) or block inspector (in Windows). Note that if two blocks are in immediately adjacent squares, then neither of them can be large. In that case, the **Large** option will be grey, and cannot be selected. When you've made your choices, click **OK**.

### B.2.5 Deleting blocks

**Macintosh:** To delete a block, click on it. It will quiver with anticipation. Then press the backspace key on the (physical) keyboard. The block will jump off the edge. You can also drag the block off the edge of the grid and drop it.

**Windows:** To delete a block, drag the block off the edge of the grid and drop it.

### B.2.6 Hiding labels

Whenever you name a block, Tarski's World labels the block with its name. Of course, in the real world we only wear name tags at unpleasant social occasions. Like us, blocks in Tarski's World can have names without wearing labels. To hide the labels, simply choose **Hide Labels** from the **Display** menu. To redisplay the labels, choose **Show Labels** from the **Display** menu.

### B.2.7 2-D view

Labels aren't the only things that can hide. Sometimes a small block can be obscured from view by another block in front of it. To get a bird's eye view of the world, choose **2-D View** from the **Display** menu. To get back to the usual perspective, choose **3-D View** from the **Display** menu.

Blocks can be moved, selected, and changed from the 2-D view in exactly the same way as the 3-D view. (You can even change to the 2-D view in the middle of playing the game; sometimes you will have to in order to pick an appropriate block, or to see what Tarski's World is referring to.)

### B.2.8 Rotating Worlds

To rotate a world by 90 degrees in either direction, choose **Rotate World Clockwise** or **Rotate World Counterclockwise** from the **Display** menu. Such a rotation counts as a change to the world and will be saved when you save the world.

## B.3 The keyboard and sentence windows

There are two ways to enter formulas into the sentence window: from the Keyboard window or from the physical keyboard. (From now on we will capitalize the word “Keyboard” whenever referring to the Keyboard window, and use lower case when referring to the computer’s physical keyboard.) Most people find it easier to use the Keyboard than the keyboard. So we will begin by describing the use of the Keyboard.

### B.3.1 Writing formulas

Tarski’s World makes writing first-order formulas quite painless. As you may have noticed while playing with the Keyboard, when you enter a predicate, like **Tet** or **BackOf**, the insertion point locates itself in the appropriate position for entering “arguments”—variables (*u, v, w, x, y, z*) or individual constants (*a, b, c, d, e, f*).

What this means is that a sentence like **BackOf(a,b)** can be entered into the sentence list with three mouse clicks in the Keyboard: first on the **BackOf** button, then on the **a** button, then on the **b** button. To enter the same thing from the physical keyboard would require 11 keystrokes.

Besides the various symbols used in the language, there are four more buttons in the Keyboard window, a **Delete** button, an **Add Sentence** button, a **Verify** button, and a **Game** button. The **Add Sentence** button is immediately below the **Delete** button. On the Macintosh, it looks like a curved arrow; on Windows it just says **Add**. These four buttons are not symbols of the language. The first allows you to delete unwanted symbols and spaces from the sentence window. It works just like the backspace key on the physical keyboard. The second allows you to add a new sentence to your sentence list after the sentence that contains the insertion point. The **Verify** and **Game** buttons do the same thing as the buttons in the evaluation box (on the Macintosh) or the inspector window (in Windows). We’ll explain these buttons later.

In order to allow you to write more readable formulas, Tarski’s World treats brackets (“[ ]”) and braces (“{ }”) as completely equivalent to

parentheses. Thus, for example, you could write  $[\text{LeftOf}(a, b) \wedge \text{Large}(a)]$  and Tarski's World will read this sentence as  $(\text{LeftOf}(a, b) \wedge \text{Large}(a))$ . But you have to type brackets and braces from the physical keyboard.

### B.3.2 Commenting your sentences

You can add comments to your sentences in a way that will be ignored by the program when it is checking to see if they are well-formed or true. You do this by prefacing each line of text you want ignored by a semicolon (;). This will cause Tarski's World to ignore anything that follows on the same line.

### B.3.3 Creating a list of sentences

To create a whole list of sentences, you first enter one sentence, and then choose **Add Sentence After** from the **Edit** menu. You are given a new, numbered line, and can then enter a new sentence. If you hit the Return key, this will *not* start a new sentence, but will simply break your existing sentence into two lines. Use **Add Sentence After!**

Instead of choosing **Add Sentence After** from the **Edit** menu, you can do this from the Keyboard window by clicking the Add Sentence button (the roundish arrow on the Macintosh, the Add button on Windows) or you can do it directly from the keyboard in two ways. You can type Shift-Return (that is, type Return while holding the shift key down) or use the keyboard equivalent shown in the menu (Command-A on the Macintosh, Control-A in Windows). In Windows, you can also click on the sentence-dividing lines to get new sentences.

In Windows, there is actually a difference between **Add Sentence After** and Shift-Return. This difference arises when the cursor is not at the right end of a sentence. In this case, **Add Sentence After** adds a blank sentence following the sentence in question, whereas Shift-Return breaks the sentence in two at the cursor's position, putting the second half in the new sentence position.

To insert a new sentence in your list *before* the current sentence, choose **Add Sentence Before** from the **Edit** menu.

### B.3.4 Moving from sentence to sentence

You will often need to move from sentence to sentence within a list of sentences. (The reason is that the evaluation box / inspector window applies only to one sentence at a time, the one in which the insertion point is present.) You can move the insertion point with the up and

TABLE 4 Keyboard equivalents for typing symbols.

Symbol	Key	Symbol	Key
$\neg$	$\sim$	$\neq$	$\#$
$\wedge$	$\&$	$\vee$	$ $
$\rightarrow$	$\$$	$\leftrightarrow$	$\%$
$\forall$	$@$	$\exists$	$/$
$\subseteq$	$—$	$\in$	$\backslash$

down arrow keys ( $\uparrow$ ,  $\downarrow$ ) on the keyboard or by clicking on the sentence of interest with the mouse. The left and right arrow keys ( $\leftarrow$ ,  $\rightarrow$ ) on the keyboard also move the insertion point, but only within a single sentence.

**Macintosh only:** If you hold down the Option key, the up arrow takes you to the first sentence of the list, the down arrow takes you to the last sentence of the list, and the left and right arrows take you to the beginning and the end of the current sentence.

### B.3.5 Deleting sentences

To delete a whole sentence and renumber the sentences that remain, choose **Delete Sentence** from the **Edit** menu. First make sure the insertion point is somewhere in the sentence you want to delete.

Note that you cannot highlight parts of two different sentences and then delete them. If you want to delete a sentence boundary, you must use the command **Delete Sentence** from the **Edit** menu.

### B.3.6 Typing symbols from the keyboard

Sentences can be entered into the sentence window by typing them on the physical keyboard. When typing predicates in the blocks language, you must be sure to spell them correctly and to capitalize the first letter (since otherwise they will be interpreted as names, not predicates). You also have to insert your own punctuation: parentheses after the predicate, and commas to separate multiple “arguments” (as in  $\text{Between}(a, x, z)$ ). To get the logical symbols, use the keyboard equivalents shown in Table 4.

Either the sentence window or the Keyboard window must be “active” before typing on the physical keyboard will have any effect. If

you type and nothing shows up, that's because the world window is currently the active window. To activate another window, just click in it somewhere.

### B.3.7 Cutting, copying, and pasting

If you want to change the order of the sentences in a list, or copy a sentence from one file to another, use the cut, copy, and paste functions.

If you highlight a string of symbols and then choose **Cut** or **Copy** from the **Edit** menu, the string of symbols is stored on the computer's clipboard. The difference between the two commands is that **Cut** deletes the highlighted symbols from their present position, while **Copy** leaves them in place. You can't see the contents of the clipboard, but the symbols will be there until you cut or copy something else to the clipboard.

Once something is on the clipboard, it can be pasted anywhere you want it. Just put the insertion point at the desired place and choose **Paste** from the **Edit** menu. A copy of the string of symbols on the clipboard will be inserted. You can paste several copies at several different points, if you want to.

### B.3.8 Printing

To print your sentences or world, choose the appropriate **Print...** command from the **File** menu. If your computer is not connected to a printer, this probably won't work.

**Macintosh only:** If your sentences print with incorrect symbols, quit Tarski's World, find the font suitcase labeled "Tarski" on your CD-ROM, drag the font suitcase onto your (closed) System Folder, and then re-launch Tarski's World. This will only be necessary if you have unusual fonts in your System Folder that violate Apple's font numbering conventions.

## B.4 The evaluation box / sentence inspector

The evaluation box (Macintosh) or sentence inspector (Windows) appears on the right of the sentence window. It is what ties the sentence and world windows together.

### B.4.1 Verifying syntax and truth

As you will learn, only some strings of symbols are grammatically correct, or well-formed, as we say in logic. These expressions are usually

called *well-formed formulas*, or *wffs*. And only some of these are appropriate for making genuine claims about the world. These are called *sentences*. Sentences are wffs with no free variables. You will learn about these concepts in the text.

To see if what you have written in the sentence window is a sentence, and if so, whether it is true in the world currently displayed, click on the Verify button (either in the Keyboard window or in the evaluation box / sentence inspector). Alternatively, on the Macintosh you can hit the Enter key on the physical keyboard, or in Windows you can type Control-Enter. If you want to check a whole list of sentences, choose **Verify All Sentences** from the **Edit** menu. Alternatively, type Option-Enter or Command-F (Macintosh), or Control-F (Windows).

When you verify a formula, the results are displayed in two places. In the evaluation box / sentence inspector, checkmarks will appear showing whether the current formula is a sentence of first-order logic, whether it can be evaluated in the current world, and finally whether it is true in the current world. (Sentences will not be evaluable in a world if they contain either predicates that Tarski's World does not understand or names that are not assigned to any block in the world.) These results are also displayed in the margin to the left of the sentence: “\*” indicates that the formula is not well-formed or not a sentence; “+” indicates that the formula is a sentence of first-order logic, but not evaluable in the current world; and “T” or “F” indicates that the sentence is true or false in the world.

## B.5 Playing the game

When you stake out a claim about a world with a complex sentence, you are committed not only to the truth of that sentence, but also to claims about its component sentences. For example, if you are committed to the truth of a conjunction  $A \wedge B$  (read “A and B”) then you are also committed both to the truth of A and to the truth of B. Similarly, if you are committed to the truth of the negation  $\neg A$  (read “not A”), then you are committed to the falsity of A.

This simple observation allows us to play a game that reduces complex commitments to more basic commitments. The latter claims are generally easier to evaluate. The rules of the game are part of what you will learn in the body of this book. Here, we will explain the kinds of moves you will make in playing the game.

To play the game, you need a guess about the truth value of the current sentence in the current world. This guess is your initial commitment. The game is of most value when this commitment is wrong, even though you won't be able to win in this case.

To start the game, click the **Game** button in the evaluation box / sentence inspector or in the Keyboard window. Tarski's World will begin by asking you to indicate your initial commitment. At this point, how the game proceeds depends on both the form of the sentence and your current commitment. A summary of the rules can be found in Table 3 on page 102.

### B.5.1 Picking blocks and sentences

As you see from the game rules, at certain points you will be asked to pick one sentence from a list of sentences. You do this by clicking on the desired sentence and then clicking **OK**.

At other points in the game, you will be asked to pick a block satisfying some formula. You do this by moving the cursor over the desired block and selecting it. Then click **OK**. Tarski's World assigns a name to the chosen block, for example *n1*, and labels it.

### B.5.2 Backing up and giving up

Tarski's World never makes a mistake in playing the game. It will win if it is possible for it to win, that is, if your initial commitment was wrong. However, *you* may make a mistake, and so lose a game you could have won. All it takes is some bad choices along the way. Tarski's World will take advantage of you. It will not tell you that you made a bad move until it has won, when it will inform you that you could have won. What this means is that there are two ways for you to lose: if you were wrong in your initial assessment, or if you make a faulty choice in the play of the game. To put this more positively, if you win a game against the computer, then you can be quite sure that your initial assessment of the sentence, as well as all subsequent choices, were correct.

To make up for the edge the computer has, Tarski's World allows you to retract any choices you have made, no matter how far into the game you've gone. So if you think your initial assessment was correct but that you've made a bad choice along the way, you can always retract some moves by clicking on the **Back** button. If your initial assessment really *was* correct, you should, by using this feature, eventually be able to win. If you can't, your initial commitment was wrong.



If, halfway through the play of the game, you realize that your assessment was wrong and understand why, you can stop the game by clicking the **End** button (on the Macintosh) or closing the game window.

### **B.5.3 When to play the game**

In general, you won't want to play the game with every sentence. The game is most illuminating when you have incorrectly assessed a sentence's truth value, but are not sure why your assessment is wrong. When this happens, you should always play the game without changing your commitment. Tarski's World will win, but in the course of winning, it will usually make clear to you exactly why your assessment was wrong. That's the real value of the game.





You might wonder what happens when you play the game with a correct assessment. In this case, if you play your cards right, you are guaranteed to win. But Tarski's World does not simply give up. At those points in the game when it needs to make choices, it will make them more or less randomly, hoping that you will blunder somewhere along the line. If you do, it will seize the opportunity and win the game. But, as we have noted, you can always renege by backing up.



---

# General Index

## Miscellaneous symbols

- , 3
  - , 3
  - |, 4
  - \*, 4, 9
  - $\forall$ , 89, 96
  - $\wedge$ , 89, 93
  - $\exists$ , 89, 97
  - =, 89, 91, 96
  - $\rightarrow$ , 89, 94
  - $\leftrightarrow$ , 89, 95
  - $\neg$ , 89, 92
  - $\vee$ , 89, 93
- Add
- Submit
    - files to list, 84
  - Tarski
    - blocks, 70, 111
    - Sentence After, 73, 114
    - Sentence Before, 73, 114
  - affirming the consequent, 56
  - ambiguity, 47
  - antecedent of a conditional
    - strengthening, 56
  - arguments, 91
  - arrow keys
    - Tarski, 73, 115
  - atomic sentence, 91
  - between, 10
  - biconditional symbol, *see*  $\leftrightarrow$
  - Choose Files to Submit
    - Submit, 83–84
  - Closing tabs
    - Tarski, 69
  - conditional
    - antecedent of
      - strengthening, 56
    - consequent of
      - affirming, 56
      - strengthening, 56
      - weakening, 56
    - symbol for, *see*  $\rightarrow$
  - conjunction symbol, *see*  $\wedge$
  - connectives, 89, 92
  - consequence, 35, 104
  - consequent of a conditional
    - affirming, 56
    - strengthening, 56
    - weakening, 56
  - consistency, 56
  - constructive dilemma, 56
  - context sensitivity, 21, 47
  - Copy
    - Tarski, 71, 75, 116
  - Cut
    - Tarski, 71, 75, 116

- definite descriptions, 38, 43
- Delete
  - Tarski
    - blocks, 71, 112
    - Sentence, 74, 115
- DeMorgan laws, 19
- disjunction symbol, *see*  $\vee$
- DNF sentence, 20
- Done
  - Submit, 84
- Edit
  - Tarski, 71, 75, 116
- elementarily equivalent worlds, 64
- Ending
  - Tarski
    - game, 77
- evaluation box
  - Tarski, 109, 116
- existential quantifier, *see*  $\exists$
- Exiting
  - Tarski, 110
- files
  - naming, 4
  - submitting, 7
- game
  - incomplete information, 53, 103
  - rules of, 101, 102
  - Tarski, 76, 118
    - backing up, 76, 77, 118
    - giving up, 76, 118
    - playing, 76, 117
- GG Status
  - Submit, 85
- Grade Grinder, 7, 81, 82, 84, 85
- Hide
  - Tarski
    - Labels, 72, 112
- identity symbol, *see*  $=$
- inconsistency, 56
- independence, 36
- individual constants, 89
- inexpressibility, 64
- inspector window
  - Tarski, 109
- Instructor Too
  - Submit, 84
- Just Me
  - Submit, 84
- keyboard window
  - Tarski, 109, 113
- language
  - blocks world, 91
- launching
  - Submit, 83
  - Tarski, 67, 107
- logical
  - consequence, 104
  - validity, 54, 55, 104
- menus
  - Tarski, 107
- mixed quantifiers, 36
- modus tollens, 56
- moving blocks
  - Tarski, 70, 111
- names, 89, 90
- Naming blocks
  - Tarski, 71, 111
- negation
  - normal form, 19
  - symbol for, *see*  $\neg$
- nonidentity symbol, *see*  $\neq$
- normal form
  - negation, 19
- numerical claim, 36

- Object Parameter window
  - Tarski, 111
- Open
  - Submit, 83, 84
  - Tarski, 68, 109
- Paste
  - Tarski, 71, 75, 116
- persistence, 60
  - under growth, 62
- predicate symbols, 89
  - Tarski's World, 91
- Print
  - Tarski
    - Sentences, 70, 116
    - World, 70, 116
- Proceed
  - Submit, 84
- quantifiers
  - mixed, 36
- Quitting
  - Tarski
    - application, 70, 110
    - game, 119
- Registration ID
  - Submit, 82, 83, 85
- reverting a file
  - Tarski, 69
- Rotating Worlds
  - Tarski, 72, 113
- satisfaction, 100
- Save
  - Submit, 85
  - Tarski, 68, 110
  - Sentences, 110
  - World, 110
- Selecting blocks
  - Tarski, 70
- sentence, 97
  - atomic, 91
  - consistent, 56
  - inconsistent, 56
  - numerical, 36
  - of first-order logic, 97
  - Tarski Sentence
    - inspector, 116
    - panel, 67
    - toolbar, 67
    - window, 109
- shaping blocks
  - Tarski, 70, 112
- Shift-Return
  - Tarski, 73, 114
- Show
  - Tarski
    - Labels, 72, 112
- sizing blocks
  - Tarski, 70, 112
- solution files
  - Submit, 81
- spatial predicates, 62
- starting
  - Submit, 83
  - Tarski, 67, 107
- strengthening the antecedent, 56
- strengthening the consequent, 56
- Submit, 7
  - submitting files, 83–85
  - User Data, 86
- summary of game rules, 102
- symbols
  - equivalents of, 89
  - predicate, 89
  - see also the beginning of this index*

- Tarski's World, 7
- transitivity
  - of  $\leftrightarrow$ , 56
- translating
  - checking, 13
  - English expressions
    - a (an)*, 28, 97
    - all*, 26, 30, 96
    - and*, 93
    - any*, 30
    - anything*, 96
    - at least one*, 97
    - but*, 93
    - each*, 30, 96
    - every*, 30
    - everything*, 96
    - if ... then*, 94
    - if and only if*, 95
    - just in case*, 95
    - moreover*, 93
    - neither ... nor*, 94
    - no*, 25, 30
    - not*, 92
    - only if*, 94
    - or*, 93
    - provided*, 94
    - some*, 28
    - something*, 97
    - the*, 38
    - two, three, ...*, 36
    - unless*, 94
    - whenever*, 94
  - extended discourse, 47
  - paraphrasing, 41
- TW Exercise Files
  - Tarski, 68, 109
  - uninterpreted language, 89, 105
  - universal quantifier, *see*  $\forall$
  - user data
    - Submit, 85–86
  - vacuous
    - generalization, 32
    - inherently, 33
  - validity, 54, 55, 104
  - variable, 95
    - bound *vs.* free, 98
    - limited number of, 59
    - reusing, 59
  - Verify
    - Tarski
      - syntax and truth, 75, 116
  - View
    - Tarski
      - 2-D, 72, 112
      - 3-D, 72, 112
  - weakening the consequent, 56
  - well-formed formula (wff), 97
  - world panel
    - Tarski, 67
  - world window
    - Tarski, 107
  - worlds
    - elementarily equivalent, 64
  - writing wffs
    - Tarski, 72–73, 113–114

---

# File Index

- Abelard's Sentences, 14
- Ackermann's World, 21
- Allan's Sentences, 29
- Arnault's Sentences, 36, 50
- Austin's Sentences, 22
  
- Barwise's Sentences, 50
- Bernays' Sentences, 21
- Bernstein's Sentences, 26
- Bolzano's World, 17, 19, 21, 41–44, 54
- Boole's Sentences, 11
- Boole's World, 12, 14, 19, 21
- Bozo's Sentences 1, 27
- Bozo's Sentences 2, 27
- Buridan's Sentences, 35, 36
  
- Carnap's Sentences, 54
- Carroll's World, 43, 47
- Claire's World, 31, 41
- CNF Sentences, 20
  
- Deckard's Sentences, 50
- DeMorgan's Sentences, 19
- Dodgson's Sentences, 32
  
- Edgar's Sentences, 27
- Edgar's World, 27
  
- Finsler's World, 49, 50
- Frege's Sentences, 34
  
- Gentzen's Other Sentences, 21
- Gentzen's Sentences, 19
- Gödel's World, 46, 58
  
- Hercule's Sentences, 43
- Hilbert's Sentences, 36
  
- König's World, 50
  
- Leibniz's Sentences, 35
- Leibniz's World, 17, 21, 27, 31, 35, 40, 49
- Löwenheim's Sentences, 48
  
- Maigret's Sentences, 26
- Maigret's World, 26
- Marple's Sentences, 50
- Marple's World, 50
- Montague's Sentences, 40
- Montague's World, 15, 29, 44
- More CNF Sentences, 20
- Mostowski's World, 53
  
- Null Quantification Sentences, 58
  
- Ockham's Sentences, 36, 61, 62
- Ockham's World, 62

- Padoa's Sentences, 57  
Peano's Sentences, 33  
Peano's World, 32, 33, 36, 38, 41, 43  
Peirce's Sentences, 25  
Peirce's World, 25, 32–34, 40, 45  
Post's Sentences, 55
- Quinn's Sentences, 12
- Ramsey's Sentences, 34  
Rebus' Sentences, 23  
Rebus' World, 23, 50  
Reichenbach's World 1, 27, 47  
Reichenbach's World 2, 47  
Robinson's Sentences, 59  
Ron's World, 40–43  
Russell's Sentences, 38
- Schönfinkel's Sentences, 26  
Sherlock's Sentences, 14  
Sherlock's World, 14  
Skolem's World, 44–46, 63
- Turing's Sentences, 19
- Venn's World, 17, 31
- Whitehead's Sentences, 36  
Wittgenstein's Sentences, 9  
Wittgenstein's World, 9–11, 13, 14, 17,  
19, 22, 31, 42, 43  
World Submit Me 1, 5, 6
- Zorn's Sentences, 49