

Practice Exam 3

Introduction to Symbolic Logic

(Real Exam 3 To Be Administered During Our Scheduled Final Exam Period)

Last 4 Digits of Your LPL Code: _____

You will have the full two hours to complete the real exam, which will be administered in class (there is no Take Home portion for Exam 3). You will be permitted to use any scratch files or “verify” options in any LPL programs that you wish. However, on the real exam you will not be permitted to consult any notes, books, or other people, or to use email or the Internet. If you do so, you will earn yourself a failing grade for the course and a trip to the Associate Provost’s office.

I encourage you to try to take this Practice Exam under these test conditions, and come prepared to discuss your questions and concerns at our last class meeting. Please answer all questions on the exam. The exam itself is worth 260 points, or 26% of your course grade. The questions on the two Parts of the exam are worth a total of 200 pts—100 pts from each Part. In order to bring the total possible points to 300, I will multiply the percentage you earned on the higher of those two parts by the 60 remaining points. So, for instance, if you earn a 88 out of 100 on Part I (Translations) and a 92 out of 100 on Part II (Proofs and Counterexamples), then your grade for the exam would be $88 + 92 + (92\% \times 60) = 235.2$ out of 260 points.

In addition to this exam booklet, the .zip archive you downloaded should include five “starter” files for use in the practice exam: `Troop.prf`, `Unit.prf`, `Vanguard.prf`, `Warrior.prf` and the TARSKI’S WORLD sentence file `Warrior.sen`.

Remember that partial credit is awarded for proofs based on the proportion of a correct proof you provide. A correct and complete proof—one with checkmarks all the way down—receives full credit, no matter how roundabout or long or inefficient it is. However, to grade an incomplete or incorrect proof, I begin by deleting any unnecessary or incorrect steps you contributed. There are two qualifications on this:

- If a step you’ve written is necessary for the proof but its justification is wrong, I leave the step and fix its justification. You then earn at least half credit for that step, depending on how much fixing the justification required.
- If I have to fix an otherwise correct necessary step by moving it into or out of a subproof, you get half credit for the step.

I then correct or complete the proof in the fewest possible steps. Points are then awarded based on the proportion of the correct proof that you contributed. For instance, if a proof you have written is worth 30 points and can be corrected and completed in 8 steps beyond the premises, and you contributed 4 of those steps, then you get half of the points (15 points) for that problem.

Part I. Quantifier Translations

Problem R. Quantifier Translations: Step by Step Method (10 pts per sentence)

Using the TARSKI'S WORLD program, create a sentence file in which you provide a translation of each of the following sentences. Translate the first sentence as sentence 1 in the file, the second sentence as sentence 2, etc. (You should be able to translate all these sentences using the "step-by-step" method, with no paraphrasing required.)

1. *Every cube with nothing behind it is large.*
2. *Nothing to the left of **b** is to the left of **a**.*
3. *Every dodec is smaller than anything of a different shape.*
4. *Some tetrahedron has something to its left that is the same size as it.*
5. *Not every cube has a dodecahedron in front of it.*

Save your file as xxxxRegiment.sen, where xxxx stands for the last 4 digits of your LPL license code.

Problem S. Quantifier Translations, Including Paraphrasing (10 pts per sentence)

Using the TARSKI'S WORLD program, create a sentence file in which you provide a translation of each of the following sentences. Translate the first sentence as sentence 1 in the file, the second sentence as sentence 2, etc. (This set of sentences will contain some sentences for which the "step-by-step" method will not work without some paraphrasing beforehand or some "fixing up" afterwards.)

1. *Every cube that adjoins a tetrahedron is larger than that tetrahedron.*
2. *Only dodecahedra are larger than every cube.*
3. *If a cube is medium, then it's in the same row as **b**.*
4. *Any tetrahedron has something that adjoins it.*
5. *If anything is in back of **a**, then **a** is medium.*

Save your file as xxxxSquadron.sen, where xxxx stands for the last 4 digits of your LPL license code.

Part II. Proofs and Counterexamples

Problem T. Quantifier Proof—“Easier” Rules Only

(20 pts)

Open the file Troop.prf in FITCH. There you should see the following valid argument:

| | |
|---|--|
| | $\forall x \forall y \text{ SameRow}(x,y)$ |
| | $\text{SameRow}(a,b) \rightarrow \neg \text{SameCol}(a,b)$ |
| | $\text{SameCol}(a,b) \vee \text{LeftOf}(a,b)$ |
| ├ | $\exists x \text{ LeftOf}(x,b)$ |

Construct a proof of this argument using FITCH. You may use the **TautCon** rule freely. You may not use, and will not need, **AnaCon** or **FO-Con**. This argument can be proved without using either of the “harder” quantifier rules that involve subproofs. Save your file as `xxxxTroop.prf`, where `xxxx` stands for the last 4 digits of your LPL license code.

Problem U. Quantifier Proof

(30 pts)

Open the file Unit.prf in FITCH. There you should see the following valid argument:

| | |
|---|--|
| | $\forall x \forall y (\neg \text{SameShape}(x,y) \vee \text{SameSize}(x,y))$ |
| | $\exists x (\text{SameShape}(x,a) \wedge x \neq a)$ |
| ├ | $\exists x (\text{SameSize}(x,a) \wedge x \neq a)$ |

Construct a proof of this argument using FITCH. You may use the **TautCon** rule freely. You may not use, and will not need, **AnaCon** or **FO-Con**. For this argument, you may have to use some of the “harder” quantifier rules that you didn’t have to use above in Problem T. Save your file as `xxxxUnit.prf`, where `xxxx` stands for the last 4 digits of your LPL license code.

Problem V. Quantifier Proofs**(30 pts)**

Open the file `Vanguard.prf` in FITCH. There you should see the following valid argument:

| | |
|---|---|
| | $\forall x (\text{Dodec}(x) \rightarrow \text{Larger}(x,b))$ |
| | $\forall x (\exists y \text{Larger}(x,y) \rightarrow \neg \text{Small}(x))$ |
| — | $\forall x (\text{Dodec}(x) \rightarrow \neg \text{Small}(x))$ |

Construct a proof of this argument using FITCH. You may use the **TautCon** rule freely. You may not use, and will not need, **AnaCon** or **FO-Con**. Again, for this argument, you may have to use some of the “harder” quantifier rules that you didn’t have to use above in Problem T. Save your file as `xxxxVanguard.prf`, where `xxxx` stands for the last 4 digits of your LPL license code.

Problem W. Validity/Invalidity**(20 pts)**

Consider the following argument:

| | |
|---|---|
| | $\text{Between}(a,b,c) \wedge (\text{Cube}(b) \vee \text{Medium}(b))$ |
| | $\text{Dodec}(a) \wedge \text{Large}(d)$ |
| | $\forall x (\text{Cube}(x) \rightarrow \text{Small}(x))$ |
| — | $\exists x (\text{Dodec}(x) \rightarrow \text{Smaller}(x,a))$ |

- If you think that this argument is valid, open the file `Warrior.prf` and construct a proof for it in FITCH. You may use the **TautCon** rule freely. You may not use, and will not need, **AnaCon** or **FO-Con**. Again, for this argument, you may have to use some of the “harder” quantifier rules that you didn’t have to use above in Problem T. Save your file as `xxxxWarrior.prf`, where `xxxx` stands for the last 4 digits of your LPL license code.
- If you think that it is invalid, open the file `Warrior.sen` in TARSKI’S WORLD and construct a counterexample. Save your file as `xxxxWarrior.wld`, where `xxxx` stands for the last 4 digits of your LPL license code.