# CPSC-354 Report

Brandon Hughes
Chapman University

September 1, 2024

**Abstract**

(Delete and Replace:) You can safely delete and replace the explanations in this file as they will remain available on the course website. For example, you should replace this abstract with your own. The abstract should be a short summary of the report. It should be written in a way that makes it possible to understand the purpose of the report without reading it.

## Contents



## 1 Introduction

(Delete and Replace): This report will document your learning throughout the course. It will be a collection of your notes, homework solutions, and critical reflections on the content of the course. Something in between a semester-long take home exam and your own lecture notes.[1]

To modify this template you need to modify the source `report.tex` which is available in the course repo. For guidance on how to do this read both the source and the pdf of `latex-example.tex` which is also available in the repo. Also check out the usual resources (Google, Stackoverflow, LLM, etc). It was never as easy as now to learn a new programming lanugage (which, btw, LaTeX is).

For writing LaTeX with VSCode use the LaTeX Workshop extension.

There will be deadlines during the semester, graded mostly for completeness. That means that you will get the points if you submit in time and are on the right track, independently of whether the solutions are technically correct. You will have the opportunity to revise your work for the final submission of the full report.

[1] One purpose of giving the report the form of lecture notes is that self-explanation is a technique proven to help with learning, see Chapter 6 of Craig Barton, How I Wish I'd Taught Maths, and references therein. In fact, the report can lead you from self-explanation (which is what you do for the weekly deadline) to explaining to others (which is what you do for the final submission). Another purpose is to help those of you who want to go on to graduate school to develop some basic writing skills. A report that you could proudly add to your application to graduate school (or a job application in industry) would give you full points.

The full report is due at the end of the finals week. It will be graded according to the following guidelines.

Grading guidelines (see also below):

- Is typesetting and layout professional?
- Is the technical content, in particular the homework, correct?
- Did the student find interesting references [BLA] and cites them throughout the report?
- Do the notes reflect understanding and critical thinking?
- Does the report contain material related to but going beyond what we do in class?
- Are the questions interesting?

Do not change the template (fontsize, width of margin, spacing of lines, etc) without asking your first.

# 2 Week by Week

## 2.1 Week 1

### Notes

During this week, there was a review of Git and being introduced to Latex and Lean. Some helpful commands include git add, commit, status, and push. Through the website "https://sudorealm.com/blog/how-to-write-latex-documents-with-visual-studio-code-on-mac", we set up latex to be able to complete the weekly report.

#### Homework

1. Finish the Natural Number Game Tutorial World.
   a) Show the completed work for levels 5 through 8.
   b) For one level, explain in detail how the Lean proof is related to its corresponding proof in mathematics.

1a. Show the completed work for levels 5 through 8.
Level 5: Prove that a+(b+0)+(c+0)=a+b+c.

```
rw[add_zero]
rw[add_zero]
rfl
```

Level 6: Prove that a+(b+0)+(c+0)=a+b+c.

```
repeat rw[add_zero]
rfl
```

Level 7: Prove that for all natural numbers a, we have succ(a)=a+1.

```
rw[one_eq_succ_zero]
rw[add_succ]
rw[add_zero]
rfl
```

Level 8: Prove that 2+2=4.

```
repeat rw[four_eq_succ_three, three_eq_succ_two, two_eq_succ_one, one_eq_succ_zero]
repeat rw[add_succ, add_succ, add_zero]
rfl
```

1b. For one level, explain in detail how the Lean proof is related to its corresponding proof in mathematics. For level 7, we had to prove the therom of the succ(n) is also equal to n+1.

Lean Proof:

```
Start: succ(n) = n + 1
  1: rw[one_eq_succ_zero]
  Result: succ n = n + succ 0
  2: rw[add_succ]
  Result: succ n = succ (n + 0)
  3: rw[add_zero]
End: succ n = succ n

Thus, proving reflexitivity.
```

Proof by Mathematics: By using the induction we are able to prove the therom of the succ(a) is equal to a+1.

```
Base Case:
  Consider n = 0,

  S(0) = 0 + 1

  0 + 1 = 1

  Thus, 0 + 1 = S(0), which holds true.

Inductive hypothesis:

  Assume for some natural number k that k + 1 = S(k).

Inductive Step:

  We need to show that k + 1 + 1 = S(k + 1).

  (k + 1) + 1 = S(k + 1), by adding parenthesis

  S(k) + 1 = S(k + 1), by using the inductive hypothesis.

  S(k + 1) = S(k + 1), by using addition of successors.

Thus, proving reflexitivity.
```

Through these steps, we can see that the end goal of proving reflexitivity on both the Lean proof and its corresponding proof in mathematics. The similarites come from the lean proof and the inductive step however, as they are similar steps in being able to prove the theorem. The lean proof is more straight forward because instead of proving it through a basis, inductive hypothesis, and inductive step, you only have to prove it through rewriting the equation so that both sides are equal. Which is done through the indutive step of the mathematics proof.

**Comments and Questions**

When looking at Formal Systems from the textbook, we are given this example of an impossible puzzle to solve. The MU problem, where you are given a set of rules and have to obtain MU from MI, however, its impossible because you can never end up without having an odd number of I's inside the string. When it comes to Formal systems, and solving them a lot of times people will look towards actually doing compared to trying to assess the logic behind this however computers, mostly AI, generally start at the logic. How might combining human intuition and AI's logical reasoning lead to more effective problem-solving strategies? Would we be able to solve problems quicker, our would AI's logical reasoning overtake the human trial and error method?

### 2.2 ...

...

## 3 Lessons from the Assignments

(Delete and Replace): Write three pages about your individual contributions to the project.

On 3 pages you describe lessons you learned from the project. Be as technical and detailed as possible. Particularly valuable are *interesting* examples where you connect concrete technical details with *interesting* general observations or where the theory discussed in the lectures helped with the design or implementation of the project.

Write this section during the semester. This is approximately a quarter of apage per week and the material should come from the work you do anyway. Just keep your eyes open for interesting lessons.

Make sure that you use LaTeX to structure your writing (eg by using subsections).

## 4 Conclusion

(Delete and Replace): (approx 400 words) A critical reflection on the content of the course. Step back from the technical details. How does the course fit into the wider world of software engineering? What did you find most interesting or useful? What improvements would you suggest?

## References

[BLA] Author, Title, Publisher, Year.