Designing and Developing a Computer Program to Assess Building Code Requirements
Nicholas Ogilvie-Thompson and James P. Skon, Ph.D.
Kenyon College Summer Science 2016

Abstract
When designing and constructing a building, careful attention must be paid so that the building meets all requisite code requirements. Building codes differ country-to-country, state-to-state, and sometimes even city-to-city, but no matter where you are, the penalties for failing to be code-compliant are stiff, ranging from hefty fines to necessitating reconstruction before the building can be used. Because of this, architects and architectural engineers take great pains ahead of construction to make sure their designs are up to code. However, there currently is no simple method for an architectural engineer to determine a building’s code requirements other than the tedious process of going over the code manually. In order to help ease these issues, I worked to create an architectural planning system that can represent architectural knowledge in such a way that it can ask the questions necessary to determine the code requirements a building would have to meet.

Background
Why do this?
- Code assessment is expensive
  - Involves architectural engineer and lawyer
- Mistakes are extremely costly
- On average buildings are “overbuilt” by ~5-10%

Project Goals
- Systematize the logic of the building code
- Use that to create a proof-of-concept of a program to determine a building’s code requirements
- Focus on usability
  - Ask users as few questions as necessary

The Building Code
The building code can be broken down into what is essentially a series of if/then statements
- Certain classifications relied on one “if”, others upon many
- Thus, our dependency system, which relied on Boolean operators, could be used to systematize the code

Usability
- Used the Turbotax model
  - Took a complicated and highly technical topic and made it accessible and easy-to-parse
- Dependencies not only determined the user’s code requirements, but also guided them through the program
  - Only asked questions that had not been made irrelevant by previous answers
- Simplified the language of the building code

Initial Approach
- On startup, database was loaded into a CLIPS file
- Dependencies read from CLIPS, answers were written to CLIPS and database
- Answers, patterns, and question sets all stored in separate CLIPS files
- Overly complex with little scalability

Revised Approach
- Eliminated CLIPS
- Answers were still written to database
- Dependencies formulated as Java functions which launched SQL queries

Conclusion
- At the end of the summer, development had been completed for Chapter 3 of the Ohio building code, “Classification of Facilities”
- The system will guide the user through a series of questions on their building’s occupancy and usage, and then report what classifications it matches
- Additionally, the project is in an excellent position going forward
- The system is scalable to the rest of the code as well as to other building codes
- Development tools are in place to assist those who might continue this project

Acknowledgments
This work was funded by the Kenyon Summer Science Scholar Program, and could not have occurred without the support and guidance of Professor Jim Skon, as well as Mr. Lance DeBenedictis, who provided architectural insight during this project’s development. Additionally, I would like to thank Joseph Schutz and Benjamin Weinberg, who worked on concurrent projects this summer, and whose work was critical to my own.