CSI 370 Computer Architecture
Research 1 - Project Proposal

Anne Konicki

The idea for my final project is a solo project that aims to dissect how classes compile down to the assembly level. The programmatic concept of a ‘class‘ data structure does not exist it low level assembly architecture, and so I wish to learn how these are created and used. As of writing this, all concepts discussed in class have implementations in assembly, and the classes have been “How does this concept one to one translate in assembly?” Learning how classes specifically translate will give insight into “How does a concept which does **NOT** exist in assembly translate to the lower level?”

In order to achieve this, I will create a simple program that uses Object Oriented Programming principles. I plan on writing a console based card game, so that classes can be created for players, a dealer(?), hands, the deck, and cards themselves. It is likely that blackjack will be used as a foundation in order to understand the class fundamentals, but the project could also be expanded to implementing Texas Hold‘em Poker if time permits. Console input and output will still be implemented in CPP. **Functions will be implemented through a mix of CPP, inline assembly, and raw .asm files in order to see all potential interactions between CPP and assembly with classes.**

Doing this project would allow me to gain insight into how CPP files are compiled down to low level languages. In my personal life I have seen reverse engineered CPP code that uses pointers to access variables, and I am curious to see how these function and would assist me in learning how other CPP programs work when decompiled.

Challenges:

- ◇ **Scope -** it is possible that even blackjack could be difficult to implement in assembly. While in best practice, a std::set collection would be used to store cards, in order to best program these functions in simple assembly, a regular array is most likely best for simplification while still providing the same functionality.
- ◇ **Scope pt.2 -** The function definitions for simple card games may be too simple. While in a card game such as Poker, functions for determining a winner could get rather complex, the game itself could very well be over-scoped. In order to get the complexity of the game, a different version of Poker other than Texas Hold‘em could be used.
- ◇ **Randomization -** A large amount of gambling based card games rely on randomization in order to make the games ‘fair.‘ Randomization has not yet beet covered in assembly, and would require calling the CPP rand() function.
- ◇ **Lack of Visuals -** By keeping the game as a simple console application, displaying the information of cards to the user in a way that is easily understandable may be difficult.