Liam Orndorff

Professor McFall

CSCI 342

Solitaire Recap

For my Solitaire project I attempted to achieve an A- level of completion and I do believe I succeeded in that. All moves except moving cards between temporary piles can be done by mouse and keyboard, my script reader can perform every combination of commands, illegal moves are cancelled, and everything is drawn correctly. The only thing that I could not get to work as described in the Solitaire packet was dragging cards from the viewport to a temporary pile. My implementation requires the view pile to be double-clicked and the second click to contain a drag to the wanted temporary pile destination.

I chose to create Enums for Denominations, Suits, and Pile Locations. This way, I could assign each card a Denomination, Suit, and where its pile is located which aided in the drawing process. In addition to my Enums, I put a lot of work into making my project much more organized and cleaner. I was not proud of how messy and cluttered my Tetris project was, so I attempted to split classes into three different packages: graphics, model, and solitaire. This helped me be able to separate the logic from the graphic and put them together in the solitaire package. Lastly, thinking of future expansion and development, I created multiple interfaces. I did this in hopes to make a well organized and more easily refactorable project.

The biggest challenge I came across was implementing the key and mouse handlers. It was often challenging to make sure my canvas had the correct listener depending on the state of the program. What helped me stay organized and was looking at McFall’s Key Mapper class from Tetris and implementing one of my own for Solitaire commands. This allowed me to let another class do the work instead of doing everything in the key handlers. Doing this project made me realize that adding little classes like a key mapper can aid in being organized and future refactoring.