Makana Valdez
AP Com Sci – Kiang
March 13, 2014

Java Independent Project

From this project I wanted to learn how to make something from scratch, learn how to connect the various classes and methods together for easy use, and learn more about blackjack. At first I really wanted to do a poker instead of blackjack. Understanding how hard it would be with so many variations of the game, it would have been very hard for me to complete that project effectively by the deadline. I decided on blackjack instead because it had basic rules, and more people in our com sci. class was doing it. This would help me connect and learn/help from my other peers in this project. After I decided this, I moved onto making my classes. I wanted a deck, calculate, player, and driver classes. This idea would very much change by the end of the project. Instead of understanding exactly what I wanted to do, I felt it would be easier to start coding something simple like a deck class. This would help increase my understanding of my own project, and understand how all the classes would connect together. At first it was very basic. The deck was an array of cards with a deck and prime deck, which once reset would just take the prime deck and fill in the regular deck (Record of Thinking #1). You could do a math random from it and deal out cards from it randomly. Then I made a small driver called blackjack driver that could deal two cards (you can see this on my first commit "Deck Class"). Kapri, Mark, Makana Tave, and Mr. Kiang were key people who helped me with my project. My general process was to do most of the coding that I understood outside of class, and then when I hit a rocky point I would come into class and try to get that problem solved. This worked almost every class period we had free time. This helped me in two ways. One, it taught me how to add thoughtful comments to my code because I would have to refresh myself on it so then I could summarize to one of peers for their help. Two, it taught me how to annunciate my ideas from code into speech, this was really useful in communication, and in general will very much help me in the future. This is really important because many times I have problems doing this, but this project helped me improve.

At the end, I really just threw away that whole basic part of my project (final commit, "Code I didn't use in finished product"). I wanted to do something more sophisticated because an array of cards with only string values was not enough to fully utilize the necessities to make a full blackjack game. Mr. Kiang helped me with starting mostly again in making a smarter deck class. I did this by making a dealer class (commit "Dealer mainly"). This class would be much easier to understand because all I had to ask myself is "what does a real life dealer do in blackjack". Well a real life dealer fills a deck, deals the cards randomly (one to a players hand and the other to itself), would be able to play multiple games, quit if the player stops playing, could stand, hit, and play its own hand. Basically the only thing else was the card value and driver I had to make outside of this class. Basically thinking in this generic way I understood all the classes that I would have to make to complete this project. It really helped me to funnel my ideas, code, and clutter work from the basics all the way into the fine details that would complete my project. This skill really helped me complete the project, and not get overwhelmed by each little detail. I will use this method in the future on large projects.

Now I just trucked along forward getting code done. I went down the lists of necessities starting from a deck array list (I changed because Kiang and Kapri convinced

me for easier efficient use) that was able to deal randomly. Then I made a card class that was able to get the value and suit. Now the cards had numeric values. This would also be easier because I could change the numeric values in this card class for the face cards (a problem before, which I didn't know how I would fix). The card class was very important because it kept the values together so that my dealer didn't have to be in charge of it making things more difficult. Basically it made smart cards in charge of their own values(You can see most of this in "dealer mainly"). Next Mr. Kiang taught me how to use an enhanced for loop that could calculate the total value of someone's hand. This would be a key instrument that I would use a lot to calculate who won a hand. Next I would make a hit and stand methods in the dealer class. This part of my code turned really into spaghetti, and it forced me to comment a lot (almost every line) in order to understand what was happening. I extensively used Mark and Kapri to help think out some of these details. Kapri was very helpful because she was pretty much ahead of me the whole time, and it helped me to ask her how she did something and why it was more affective then my code. It was also nice to play her game to see her user friendly interface compared to mine. For me user-friendly blackjack from the command line was really my big goal. I wanted it to be easy to understand because the game would be pretty basic. It made me question how I wanted to phrase some words, and what would be the most efficient way to have the player understand my game for easy use. Having someone else like Kapri who was doing the same project helped me do that. I asked myself what makes her interface good and bad, and what made my interface good and bad, then I tried to get that in between. Mark is really good at coding so he helped me many times with compiling my code. Since Mark doesn't speak that much it really taught me to ask many questions to him on why things did that, and how to read compiling errors. He was then very helpful in explaining his thoughts through the code, and also taught me a few new things. Working with others really taught me not only how helpful it is, but to be prepared and more efficient when I am able to get help (Commit "Working condition blackjack").

After I completed my game I was challenged to fix the many bugs that was occurring, It was really odd because I had to take notes on the various different ways the code was working wrong. I would choose stand immediately, or hit then stand, or hit five times just to see if the code would mess up. This in fact was true the code eventually messed up. My hit and stand methods are very long and the code was looping one too many times, or not reaching the end of the code properly, or not doing this and that. I had to basically work through these myself because my code was getting to complicated for Mark or Kapri to easily understand, and by this point they were very much busy with their own projects. One thing mark taught me though was to put testing System.out.prinln("test1"); into my code so that I could see what was going wrong. This helped me a ton and taught me how to further understand what was happening to my code. For example if it printed test1 three times to the interface I new something looped three times etc. This actually made it a lot easier for me understand what was happening and fix my code. From there it was a matter of grinding through and complete my first working code.

From that point I worked with Kapri, Mark, and Aliya and they played my game. They actually liked how simple the interface was, and thought that the game was very accurate. They liked how I also printed out the total of the dealer's hand so that instead of just saying they lost or won they actually know why. They also liked my organization of methods in my dealer class (the bulk of my work). I really enjoyed explaining my game to

them, and I think commenting really helped me understand my own work when I was summarizing to them. I liked how everything was legible and understandable at the end of my project. I think the most two most challenging parts of my project was getting started and then fixing bugs at the end. It was really hard to get my ideas going in constructing the project. I was having troubles visualizing the project as a whole, and it really helped me to learn how to outline and do a spaghetti graph drawing. Bug fixing was really tedious because I basically had to change my code to change everything to print out to fully understand the loops and exactly what the code was doing in each if statement. Overall though, the process was just about consistency: think, outline, work on code, and help in class, repeat until it needs to be cleaned up. This process will really help me in the future both making java programs and other projects in general.

I am very proud of my final project. I think everything works very will together and efficiently. I am also proud that I completed the project in general, for every piece of coding I write it is more thought out then writing or speaking, and it means a lot when I put all the pieces together. I think I would have made my project harder now that I have better understanding now of programming something. I wanted to add the Ace as 1 or 11, but I ended up running out of time, and my code didn't have a way of doing it easily. I also would have wanted to add a player, and maybe even a better interface with cards.