Phillip Dvoskin

Software Development I

Professor Juan Arias

Final Writeup

When I was first deciding a topic for my lab report, I really didn't know what to settle on. Given the three project choices – application, database and networking, with my knowledge of computer science and from the classes that I had taken, I only knew how to do application projects coding with Java using the Eclipse IDE. I thought for a long time and wanted to do something interesting for my project but I also wanted to do something that would be practical and can be applied to the real world. I didn't want to do project that would solve a specific problem and never be able to used for anything else. That's when my idea came to me, I thought maybe I could program a game that I thought was interesting. I thought of games that are many people to play and casino games to mind. I decided that was a great idea and wanted to do my project making a game with within which casino games could be played.

At first I make a planner for what I wanted to do during my project. Now that I had an idea, I needed to find some way to actually implement it. I decided to think of some casino games and I remember thinking about blackjack, poker, roulette and slot machines. I had plenty of ideas, I decided that I was going to make a game where the player started out with a certain amount of cash, perhaps $100 and they would be able to go to different casinos to play the games that I had mentioned. One casino would be low stakes and would have the lowest buy-ins to play

the games. The second casino would have medium stakes and the final casino would be for high rollers and have the highest stakes and the highest starting bids required to play any of the games there, but also potential for the biggest wins. The point of my initial game would have been to progress through the ranks of each casino and to win the game you would have had to reach some amount of money, say $1,000,000 for example. The game would have also had a save and load feature to ensure that you can come back and play starting from where you last left off.

Needless to say, I was too optimistic in my goals for the project and all my ideas that I had disappeared once I realized how hard it would have been to do all of what I had planned for. My moment of realization of the difficulty of the project was when I had started programming Blackjack. I decided that Blackjack was probably the second easiest game to program out of the ones that I wanted to do, but it would help me gauge how hard the project would be. Additionally, Blackjack is never the same and requires not only lock but lots of strategy and mathematical analysis to get good at. Because of this reason, I had wanted to program it first since I thought it would be hard to program but the aspect of difficulty would probably be what made it fun.

I decided to start programming, but before doing so I wanted to try and think out of everything that I would need to do for this project. The first thing that I decided I needed was a deck containing 52 cards to contain all the different cards and suits that could be in a deck. At first I wanted to make a array called Deck that would hold all the cards so I did so but soon realized that if I had wanted to take cards out of the deck at some point later on, I would have to copy the Deck array, make a new one with 1 less element to take out each element separately. I figured that this would have been to complicated to do and opted for using array lists as they can also store values the same way that arrays do, however, they are much easier to work with and

change out, specifically deleting and adding elements from and to the array list. I had also decided that it would probably be smart to use methods to have this code run, but at the time I didn't realize that using objects would be useful so I decided to make separate single methods for each game and call these methods from the main string to have them run the games. So I made a void method called Blackjack for my blackjack game. Now I had a string array list called deck but I needed to actually fill it with cards. To do so I made two arrays, one called "cards" which was a string array holding all the names of the cards in the deck, the other was called "cardValues" and this was was an integer array to hold all of the values of the cards.

At this point in time I ran into my first problem, in blackjack the ace card can either hold a value of 11 or 1, decided by its owner and automatically given a value of 1 if having a value of 11 for it would cause the user to bust. I decided to give it the value of 11 at first because I figured that having to change the user's hand value of cards to make it higher would have been harder than to change the user's hand value of cards to make it lower. Now I had a string array of cards and an integer array for the corresponding values of the string array. There are 4 different suits in a standard deck being the hearts, clubs, spades and diamonds. I wanted to make another string array to hold these different suits but decided against it. Since I was first making my program to run in check, it wouldn't matter what suit the cards were as long as each card in the deck was unique and there were 4 of each card. I knew that function that existed in java to add cards to the array list but to do that for all 52 cards would take up so much room so I decided to avoid doing that. I knew exactly how many times I needed each card to be added and I knew used an "for" loop would be the best for this scenario. I needed each of the 13 different types of card to loop 4 times to add each card to the array list making the deck.

Now having a deck, I needed some way to randomly draw cards from it. I wrote a statement to randomly generate a value from 0 to 12, I did this because each of my arrays had 13 spots starting from element 0. So this statement I made generated a random number from 0 to 12 and I made an array called “storeFirst” and “storeSecond” to store each of these values. I needed to store these values in arrays so I can retrieve them at a later point, since every time I called the random number statement it would generate a new random number. I made another string array list called “currentCards” and using the numbers I stored in my “storeFirst” and “storeSecond” arrays I connected to my “cards” and “cardValues” arrays to give the user 2 randomly generated cards that would be the starting cards. I also made an integer array list called “currentValues” that stored the values of the user’s current cards. Additionally, each time a random card was given to the user, it was removed from the Deck array list so that it cannot be generated again, this made the deck actually realistic to draw a random card from

Having given the user 2 random cards I needed to do my logic portion for giving the user more cards. If both the user’s cards were aces, I would assign the value of 1 to the first ace, I also made statements to check if the user card’s sum was ever over 21, if it was it would check if the user had any aces and if the answer to that was a yes, then those aces would one by one be given the values of 1 and re-checked to make sure the user’s card sum was less than or equal to 21. Following this, I needed to make a statement to see if the user wanted to stay with their current cards or hit, to draw another card randomly from the ones that still remained in the deck. I made a scanner input statement and printed out a line prompting the user to enter whether they wanted to hit or stay. I also made a check hit statement, this statement checked if any combination of uppercase or lowercase letters formed the word hit, then it would hit for the user. I wanted to do uppercase and lowercase letters so that it would work for both the words “hit” and “Hit” without

causing errors. I made an if statement to check if the user entered hit and if they had, then would be given another random card which would be removed from the deck, then they would be prompted again whether they wanted to hit or stay and this would continue until the user entered stay. After each time the user hit, it would show them, their cards and the sum of their cards so that they can gauge whether it would be more tactical for them to hit or stay with their current hand.

Once the user entered “stay” in any combination of uppercase or lowercase letters, it would stop giving the user random cards and check them against the Al’s cards for whom I would write the logic shortly after. To do the logic for the AI, I needed to write an algorithm to make sure that the win wasn’t easy for the user but not impossible. I wanted to do something challenging so I used some math and calculations. Out of the random cards that the Al can draw they all range in values from 2 to 11 depending on whether the ace is a 1 or an 11. I needed a number to check against which would help decide the Al if it should hit or not. There are 13 different types of cards the Al can draw out of the deck so I wanted the Al to be safe more than half the time. The closest split of 13 into two parts is 6 and 7. 21 – 7 = 14, so if the Al’s sum of their cards was less than or equal to 14, I decided that the Al should hit and then it would check that statement again. Just like the user all the Al’s cards that they drew would be subtracted from the Deck array list.

Finally I needed the write the last parts of my code to see whether the user or the Al won. I made two integers called “sum” and “Alsum” which were the sums of the user’s and Al’s cards. I made all the possible win conditions and had the program check for them to see who won. If the user’s sum was less than 22 and the Al’s sum was over, the user won. If the user’s and Al’s sums were less than 22 but the user’s was higher, the user won. If the user and Al sum were

equal, it was a tie, if both were over 21, it was also a tie. Lastly, if the user's sum was over 21 but the Al's wasn't then the Al sum, the Al also won if its sum was higher but both sums was less than or equal to 21.

To write my code for everything I just talked about it took me a very long time, so I dropped the casino game and just decided on doing a blackjack program. I also made a nice scoreboard that prompted the user for their name after their first loss or tie to the Al and display their name and score at the end of the program. If the user won, the program would run again and each time the user won, it incremented an integer sum which originally started with 0, by 1. I used printf, to format the print statements for the scoreboard so that it indexed the user's name up to 20 character right and also indexed the score to the right.

If I had more time I think I could have definitely cleaned up the project and made it better. Firstly, I believe working with a user interface would have probably saved me a lot of time that I used figuring out the logic to run my blackjack program on an IDE. Additionally, the scoreboard only was able to keep track of one score and I couldn't figure out how to make it store the scores and display new ones on top of them. Overall, though I would say the project was pretty challenging but really fun to do. I reinforced my knowledge of arrays, and learned about array lists. I learned more about print formatting and how the logic of different if, while and for loop statements works. I learned how to call, add, remove and do many other fancy things with array lists and how to generate, store and interact with random numbers.