Software Design & Specification

Presentation Draft

Hello, whoever’s listening, I’m Stefan from Computer Science Option 3. I’m here to talk about the program, or programs, I designed. I would present my results in these for areas. A general idea of what I’ve been doing and thinking, some details that I considered matters, my future plan for the next week which is the deadline, and at last, reflection on this assignment.

To begin with, what I’ve been doing. I believe I could say I did 2 programs, both unfinished. Chinese chess, which is what I gave up, were designed to allow 2 players to play with an interface and, if time allows, a database to store history. I finished all the preparation before coding which will be introduced later. Black jack is the program I settled at. This allows 2 to 6 players to play, and later, with an interface by TKInter which is a python’s graphical user interface package. I shifted the program because, basically, I was overconfident with myself, but it turned out I don’t have much time for this task. Yet I still want to make something cool. Hardest part I regard in the Chinese chess, because there’s various chess pieces each with different rules, is the logic to determine a valid input. When I get that done, the coding part feels not challenging and time-consuming. So I don’t have much time for the interface part. So, I decided I would make an program with easier logic and less content – hopefully less time taken, so that I can begin learning GUI soon. But it didn’t turn out so where I haven’t learnt any in weekend, even though I should due to plan.

Here’s the outline in time of my work. Mon. Dec. 5th 2016 to Sun. Dec 11th 2016, I wrote my project specification document. It includes overview about my program and central logic – program broke into procedures, and comments on functionality. Mon. Dec. 12th 2016 to Sun. Dec. 18th 2016, I wrote my software design document which includes detailed structure charts & logic. Finishing up this means all I need to do is translate the pseudocode into python grammar, theoretically. Mon. Feb. 27th 2017 to Wed. Mar. 1st 2017, like I said, I looked for alternatives, a faster program to write. Thur. Feb. 28th 2017 to Sun. Mar. 5th 2017, I started the coding of BlackJack. Ok, now the program is functional, but without a GUI at present. So, let’s run it for once.

I would introduce the rule first for those who doesn’t know. It’s actually a pretty popular game. It’s the only game in casino that players stand a chance to make money in the long run, that statistics doesn’t favor the banker. 2 to 6 players including one banker, each get 2 draws from 1 pack with banker 1 card hiding. Start from players to banker at last, each gets to “HIT” – drawing another card in times as long as they want. “STAND” means not “HIT”ting and moving on. The score is added up where “Jack”, “Queen” and “King” are counted as 10 and “Ace” can be counted as either 1 or 11. The newly drawn cards will only show then the banker finishes drawing. The aim is a score below but not exceed 21 points, where a 21 score is called “BlackJack”. The banker has to HIT if it’s score is below 16. Everyone compares to the banker for winning.

Running

So now, about details for Chinese chess, the interface is like this, and the rules are hard to explain. I don’t see the use of explaining it either. I would like to mention that I used the chess pieces for the names of the Chinese chess pieces. How it works internally? There’s a 2D Array and all the determining valid input are done with it. The structure chart includes a loop of constant inputting chess pieces’ location until a winner is found. It’s just like the “Connect 4” program in the coursebook. The logics of specific procedures were interesting when I was figuring them out but it’s not important for this presentation now.

As for BlackJack, it’s functional and only functional. Results can be shown but not in a nice and clear way. When I searched for rules, it turned out to be different and complex with all the money you put into, so I didn’t include that part. The central logic is pretty easy to understand, first, players each get draw 2 cards, then they would draw on their will in orders. Then the result would be printed. The drawing mechanism is worth mentioning. So when a card gets drawn from the pack array by random integer, the element is changed to blank, and when the next time a blank is drawn which in reality wouldn’t happen, the random integer will be generated again to draw another card. The program is still not idea. Here’s the part where the results get printed. It needs generalization, e.g. the parts I circled. May be a change in logic is needed, or a new edited format maybe.

Future plans, in the next week I would learn TKInter, modify the code to fit a GUI. Same law applies. Functional first. I’ve designed how it would look like. Pictures of cards on table are always shown, buttons include HIT, STAND, SHOW CARDS, and NEXT. The combination of these can prevent peeking of other player’s cards. After all, all players has to manipulate their cards in front of the same screen which is kind of lame.

Ok, reflection. First, Think thoroughly & Design first. Every time I don’t do this enough, every time I regret it, and I do the same again. Knowing that it would save time for e.g. modification when new ideas come into light, or maybe I don’t need a second program in the first place. Second, if you don’t know what’s happening, print it. It always helps to check your code is running as supposed and debug. To continue, keeping records is helpful, so that I can go back to the last bug-free version if I ran out of ways to fix bugs. At last, a project is way better for learning than lessons. You may forget things, grammar in different compilers can become a mess in your head. When you actually write your program, it reminds you of how exactly it works.

That’s all. Thanks for listening.