Notes: Assignment 3

Return values are good way to test, if it is significant thing to test within your context.

For instance, If you are testing, int max(int a, int b, int c), you can call the method and check if return value is correct or not.

This is not the case with dominion code. For instance, cardEffect most of the time returns 0. If you refactored any of your cards, you may be using same convenion and returning either 0,1 or void in case you decided that your refactored method shouldn't return anything. The most important thing that takes place when any of the card is played in dominion is that the "state of the game" changes. For instance, if smithy is played, the current player receives three extra cards and then he discards the current card in his hand. A good test will atleat test if these two things have taken place or not, by asserting it against the "state of the game" before and after smilty is played.

If X was my previous state of the game before smithy was played and current player's hand has A numbers of cards. If I play smithy, current player's hand now has A + 3 cards. If the player discards one of the player's card,the player's hand now has A+2 card. You can go deep and do more detailed oriented testing like, if the 3 new cards that current player received indeed come from his own pile and etc.

If you willl test something like this.

int r = MyRefactoredSmithy()

assert(r == 0) print("All is well")

then this tests for crash only. And not for the state of the game. **And you will receive only 1/3rd of full score for that unit test or card test**.

Any testing effort needs to consider (1) what to test. (2) How to test. If you know what to test, you will evantually figure out how to test it. In this course, we want to learn both. So it is very important that for remaining of assignments (3,4 & 5), you correctly determine what to test and then test it. You don't need to be very detailed oriented in "what to test", but at least test something that finds bugs.