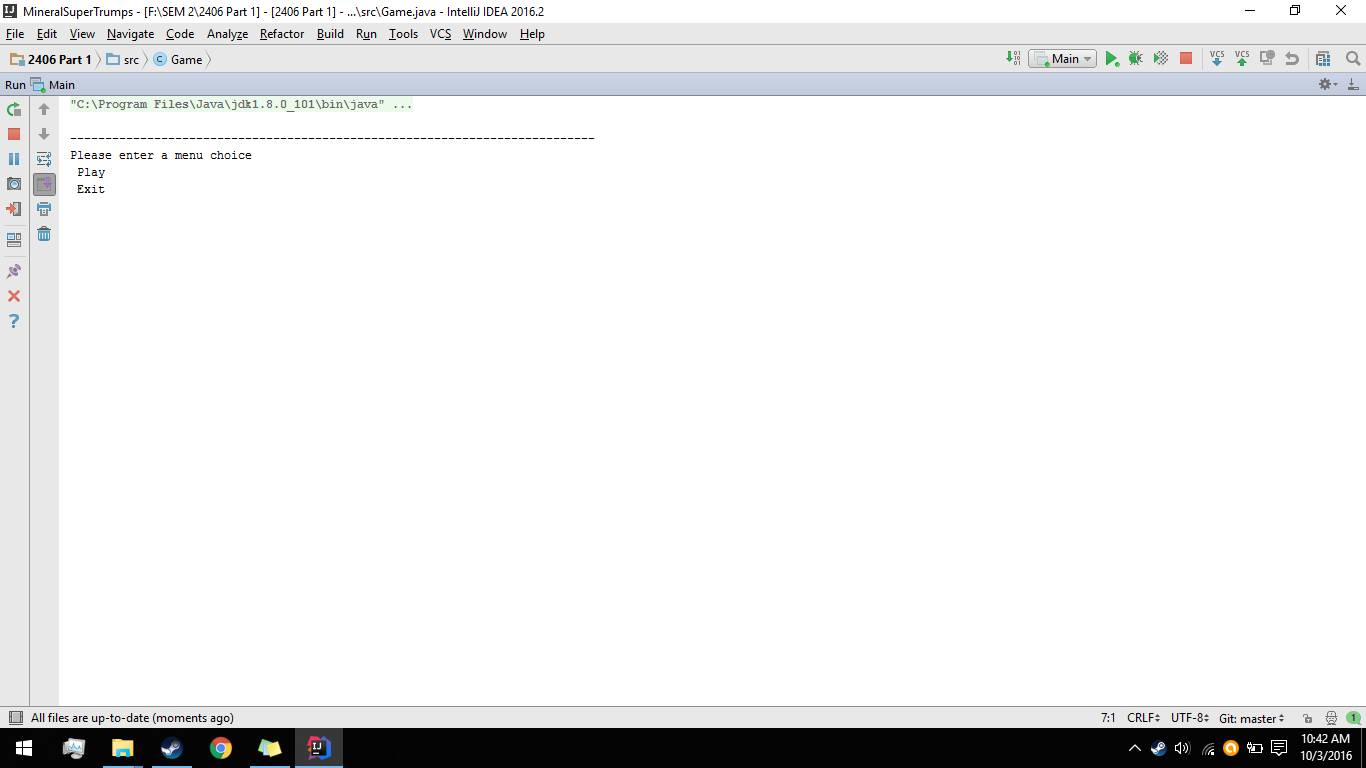
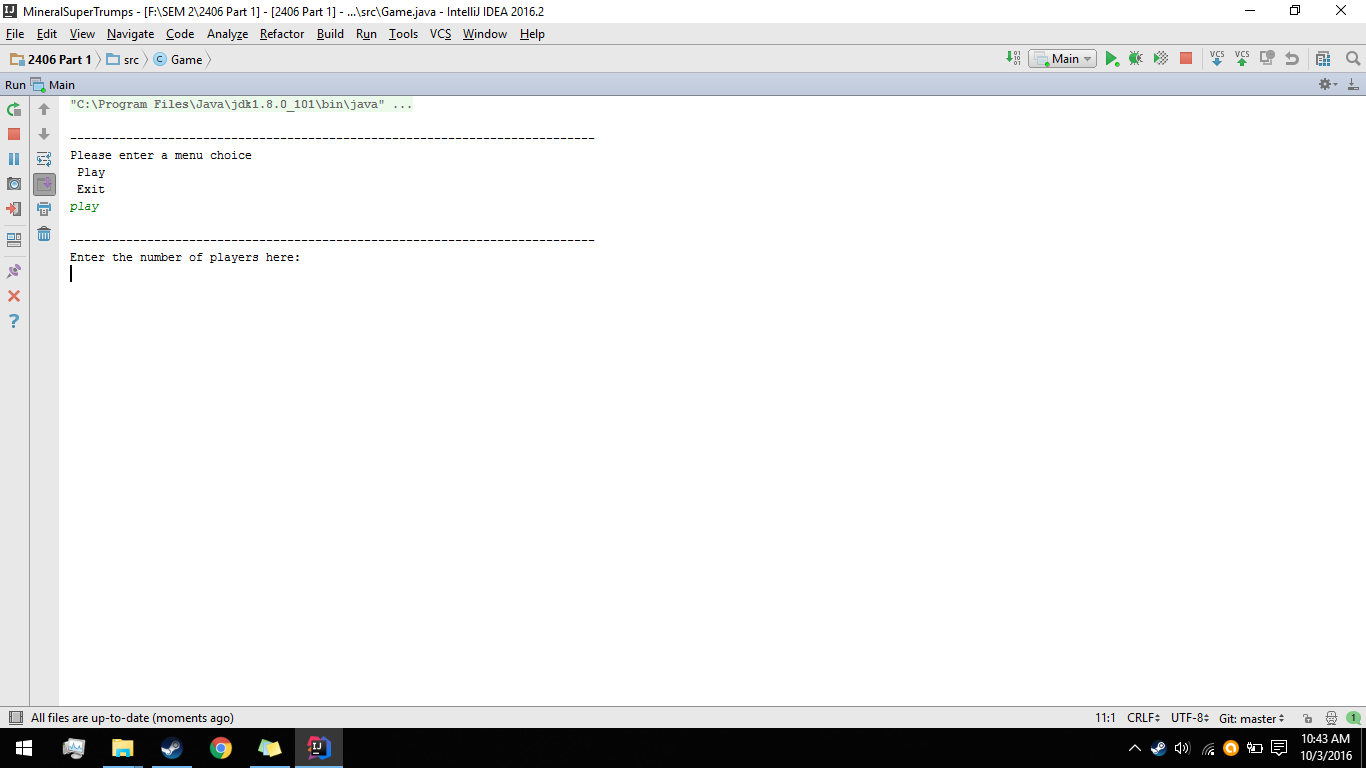
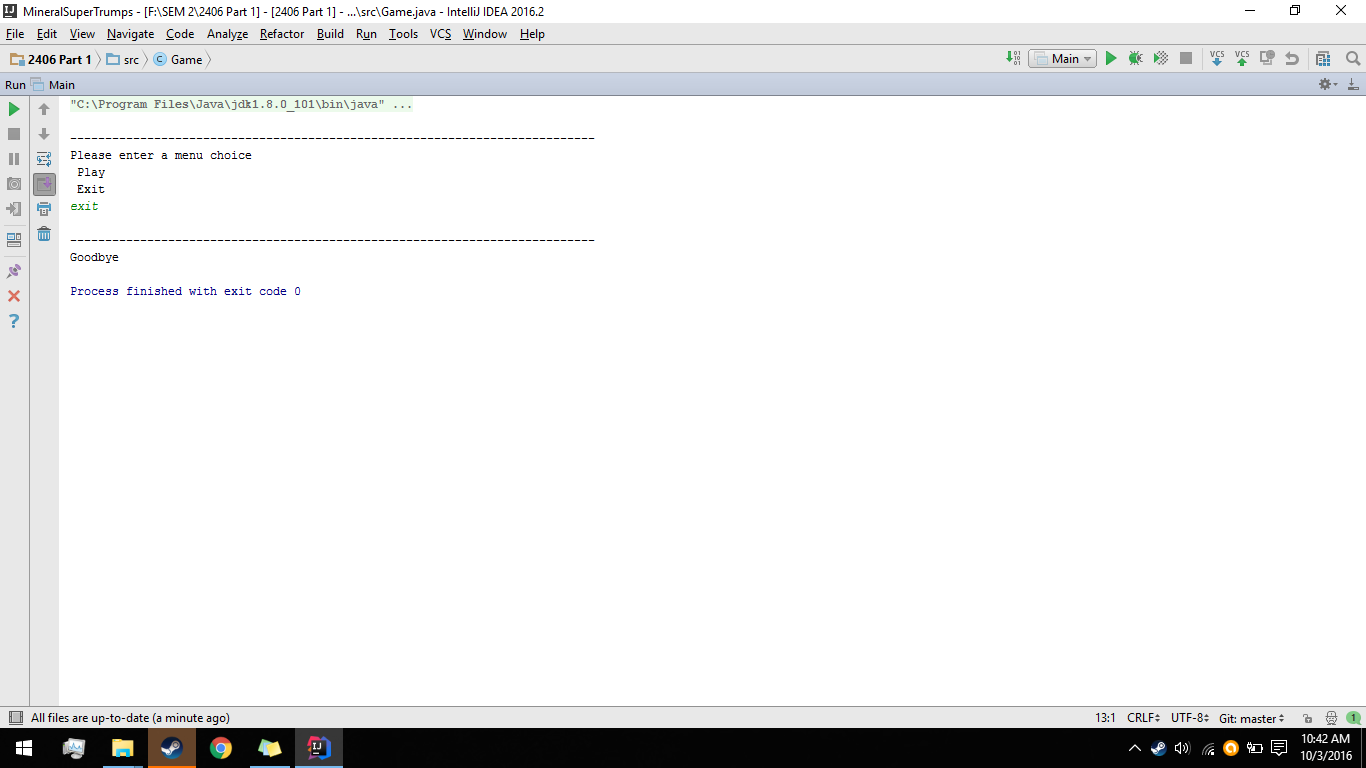
# Analysis

* As a person playing the game I want the game to support extra players so that I can other people to play against.
* As a person that is playing the game I want to be able to see all available cards that I have in my hand so that I can make a clear selection in which card I want to play next
* As a person that Is playing the game I want to be able to easily select the mineral check type, so that at the beginning of each round I and the other player will be able to have different types for each round
* As a person playing the game my hand must be able to hold many cards as well as the data of different types of cards so that I can compare cards to know which card should be played next.
* As a person playing the game I want to be notified when I have drawn a card form the deck. I would also like to know what that card is so that I can prepare my turn next round
* As a person playing the game I want to be able to have the option to pass and to be counted out of the round. This is so I don’t have to play a card every turn.
* As a person playing the game I want to know when a card has been discarded to the pile and I also want to be able to see the pile. This is so I know what card I need to play during my turn.
* As a person playing the game I wish to be notified when the deck has run out of cards. This is so that I am able to tell when the game is nearly over
* As a person playing the game when I empty all the cards out of my hand I must be declared a winner. Likewise, the game must continue until all but 1 player has not emptied their hand. This is so the game has a winning condition. Also as a person playing the game when I play the geophysicist card and the magnetite card I must also be declared a winner.
* As a person playing the game I want all the cards in the deck to be shuffled before they are dealt. This is so the game has a randomness element and cards are no longer placed in order.
* As a person playing the game I want to be notified whether the card I have played has a higher or lesser value than the card on the pile. This is so if I make a mistake I can have another go to try and place a different card.
* As a person playing the game when I or another player plays a trump card the round type should be changed and all other player that passed should be back into the round. This is so the game can operate according to its rules.
* As a person playing the game if I am the last person in the round after everyone has passed I should be allowed to choose the new type for the round. This is so the game can operate according to its rules.
* As a person playing the game if I make an input mistake I wish to be warned about it and given a chance to reenter my input. This is so I know if I have entered input incorrectly.

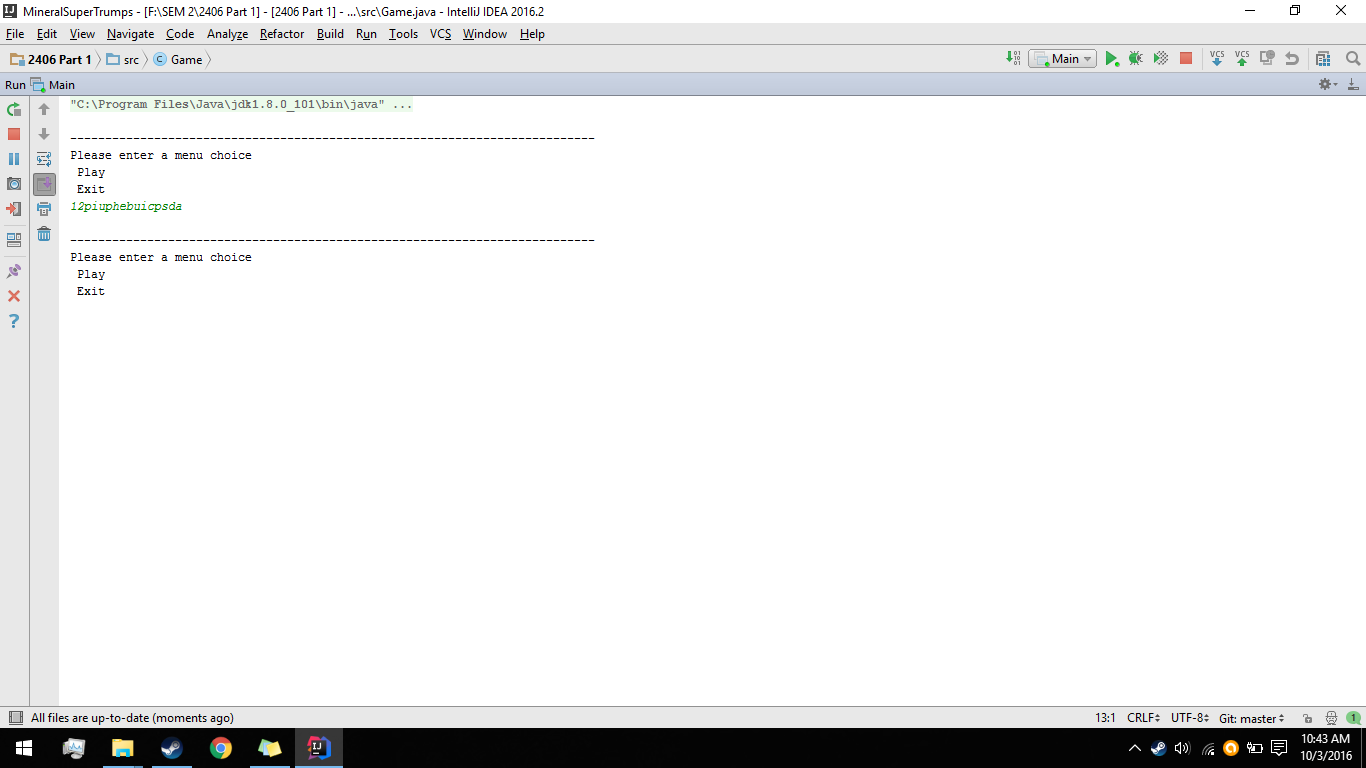
# Testing



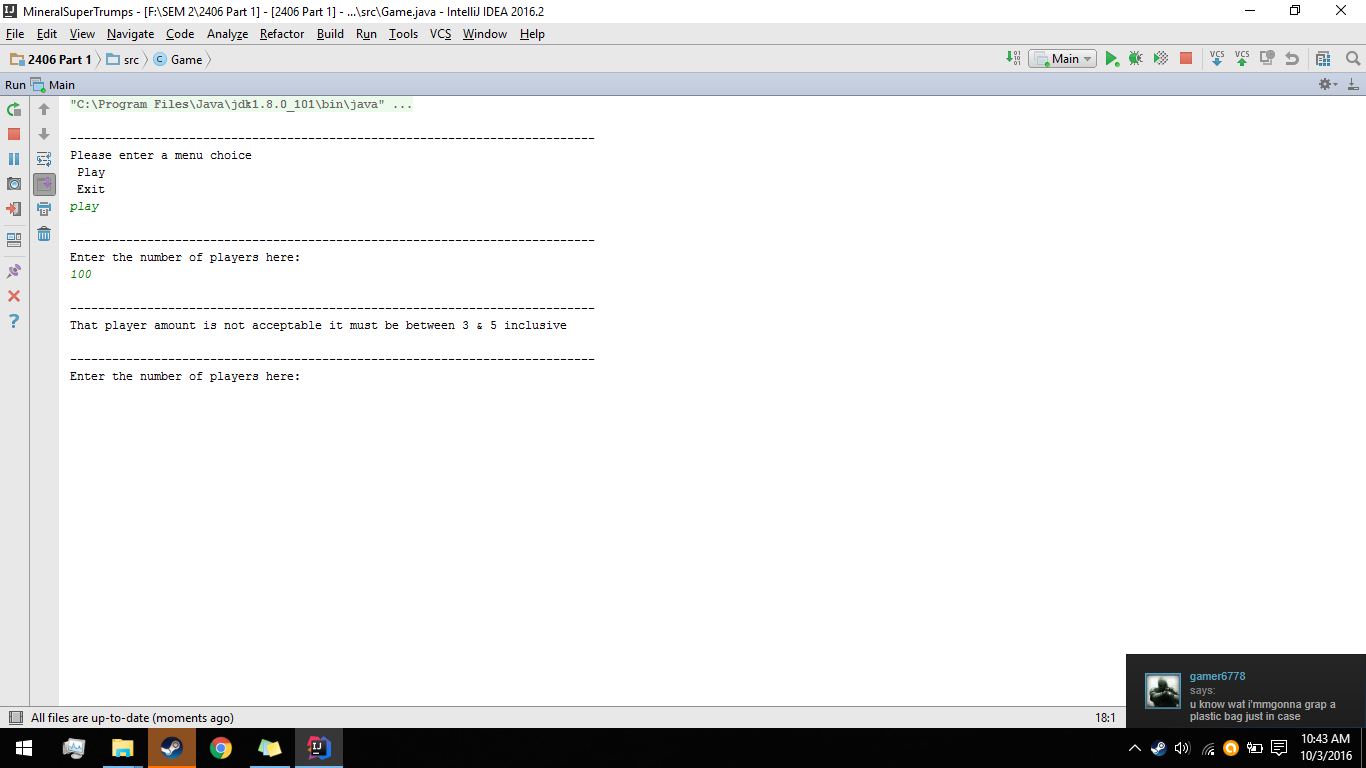
This is the starting point of the program. You either type play, to begin the game or exit to exit the game.



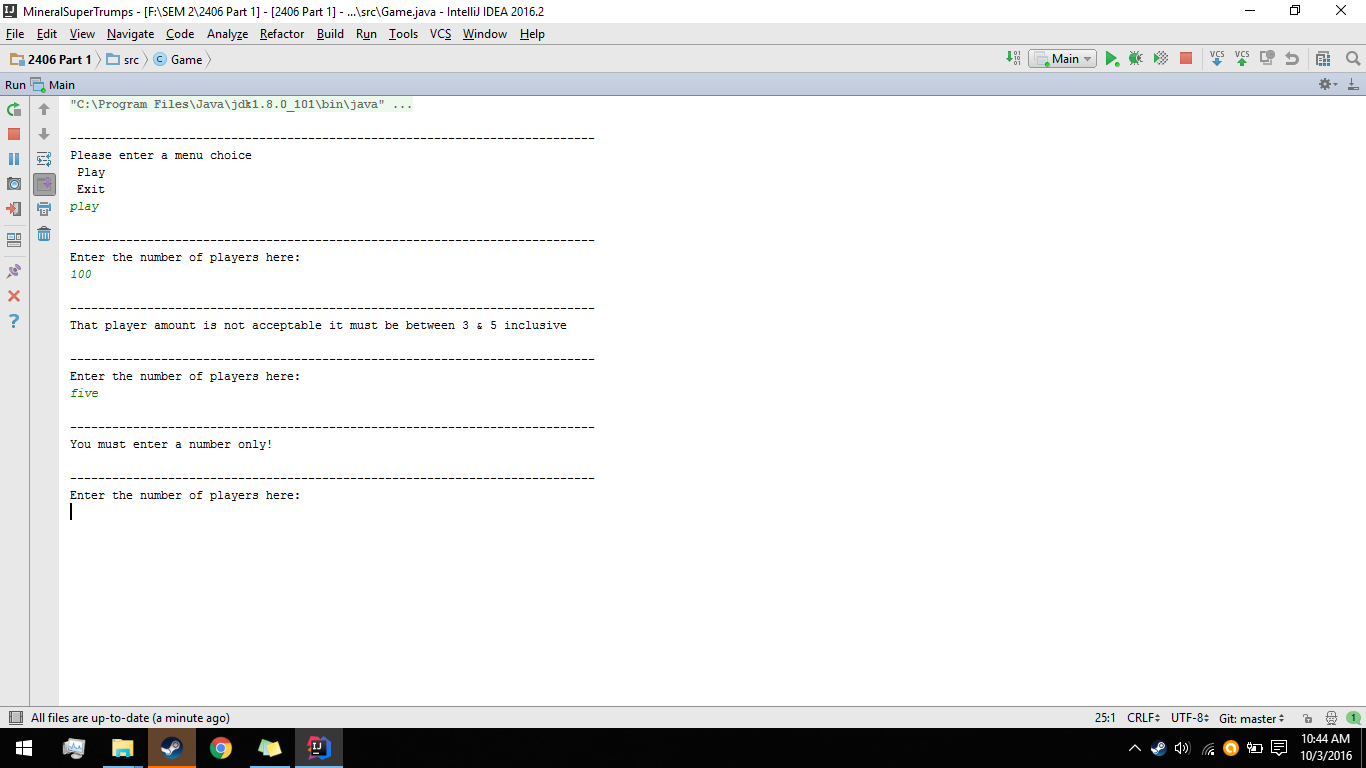
The code also has error checking in the form of a do while loop so the game will not crash due to incorrect input such as numbers or an incorrect set of characters.



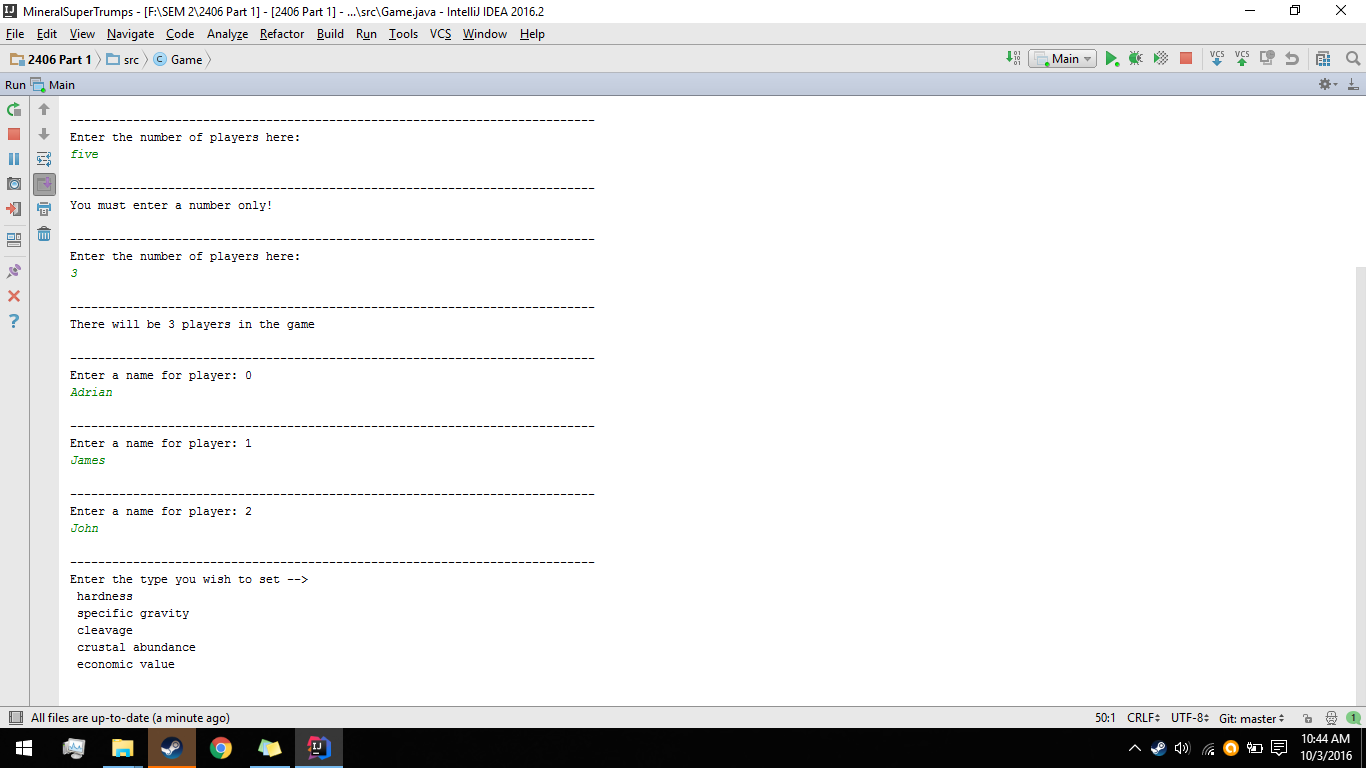
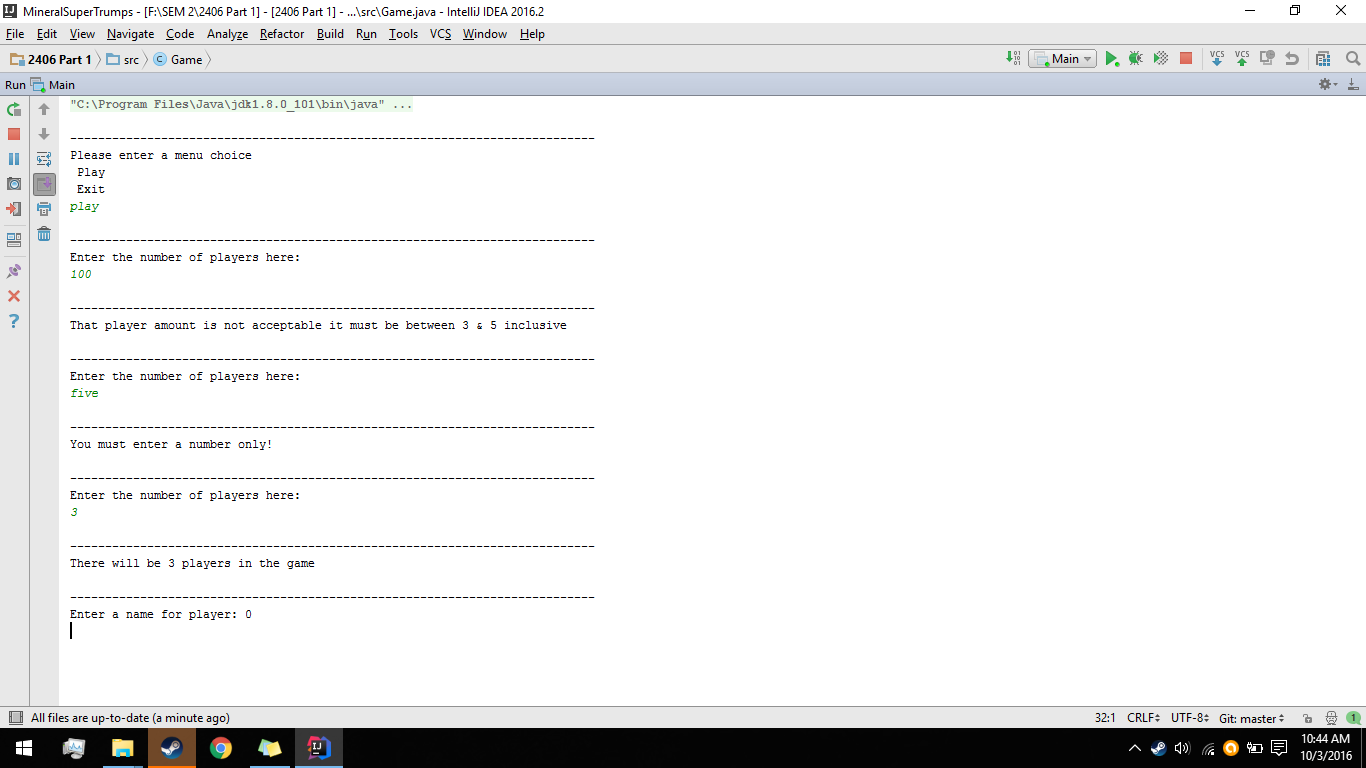
After the player has chosen to play the game it will begin the gameFlow() function which solely contains the loops and if statements needed for the game to play through. It will first ask for the player to enter in how many players will be playing. (between 3 & 5 inclusive)



This part of gameFlow() also makes use of a do while loop coupled with a try catch. The try catch handles exceptions that are not of an integer type and the loop handles integers that are not with in the required range. After this the cards that were loaded into the deck arraylist are then shuffled so they can be dealt to the player.

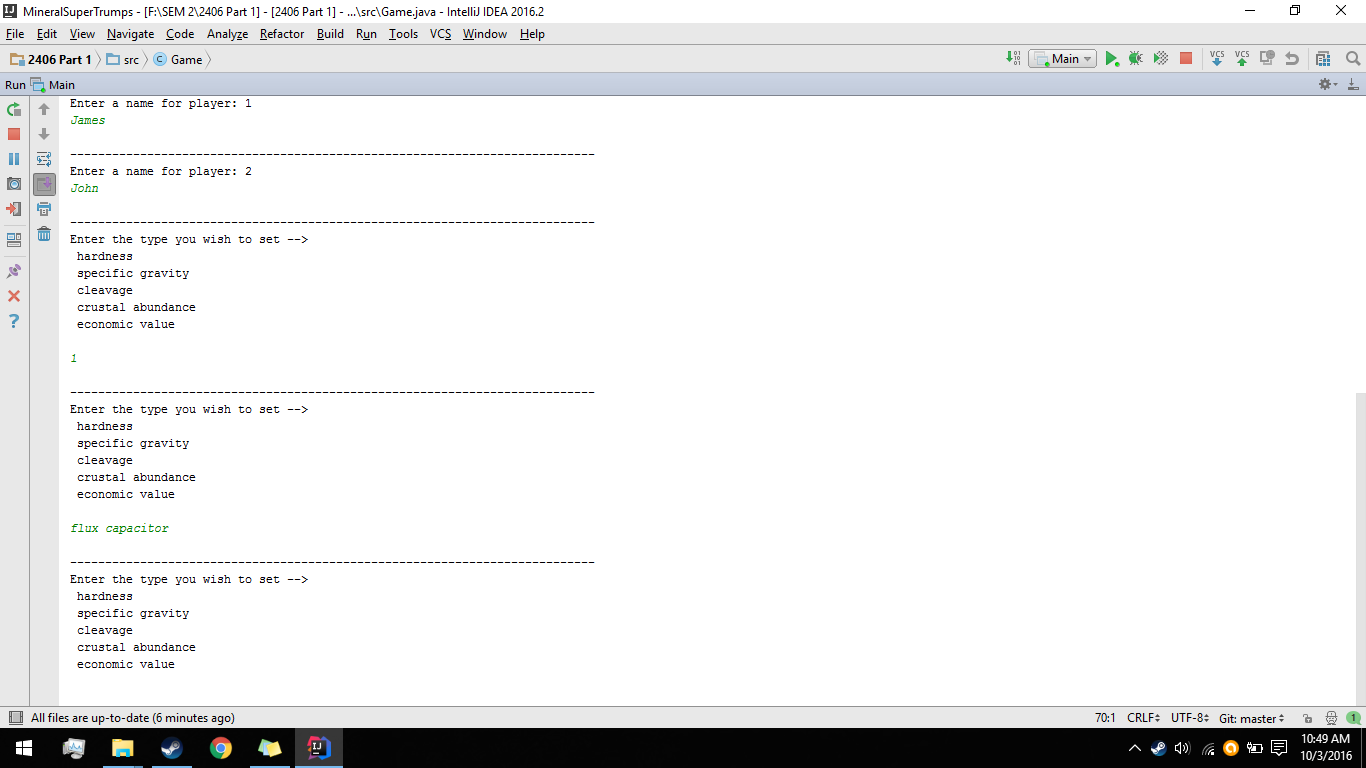


Once the player has entered a correct amount of players they will then give names to each player. Any input can be used here. After each player has received a name they will be then dealt 8 cards from the deck, which are then moved into an arraylist located in each instance of the Player object. The player is then added to another arraylist that stores current players. This arraylist is also used to keep the game flowing and determine a winning condition.

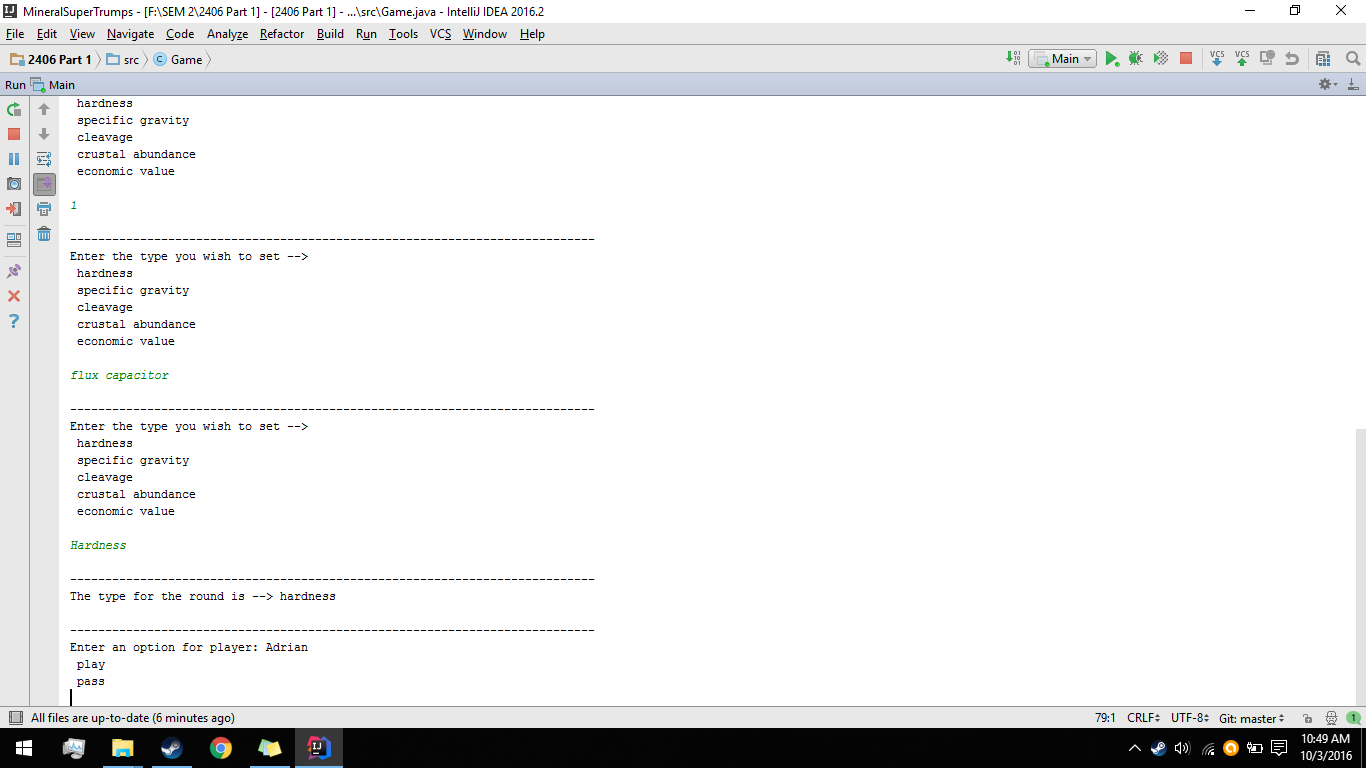


Once a player has entered all the names the first outside method located in the Game class will be called. The method called chooseType() is responsible for changing the type when a new round is started and when the Geologist trump card is called.

It consists of a do while method for the purpose of error checking user input, so the play can only input a type that is listed in the menu.

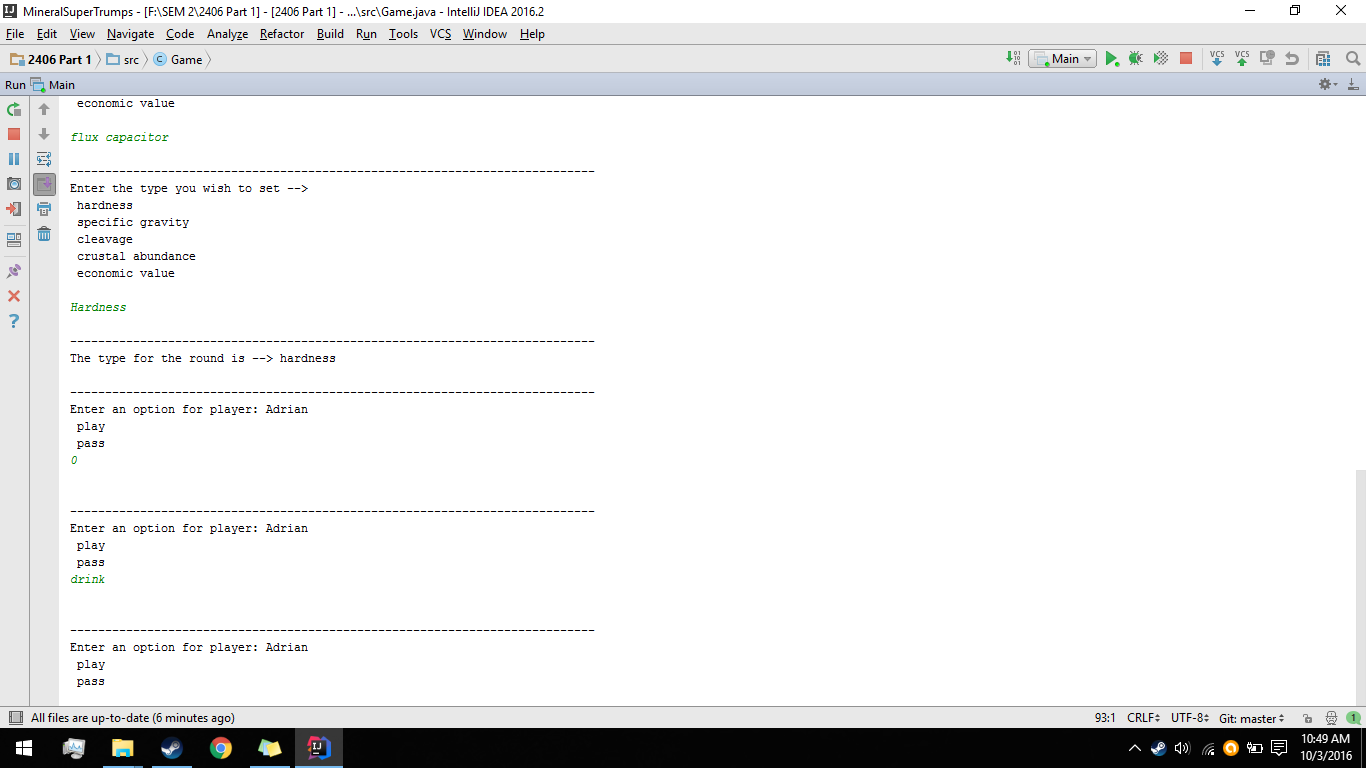


Once a correct type has been chosen the players will be prompted with the chosen type for the round and then the game can begin starting with the first player.

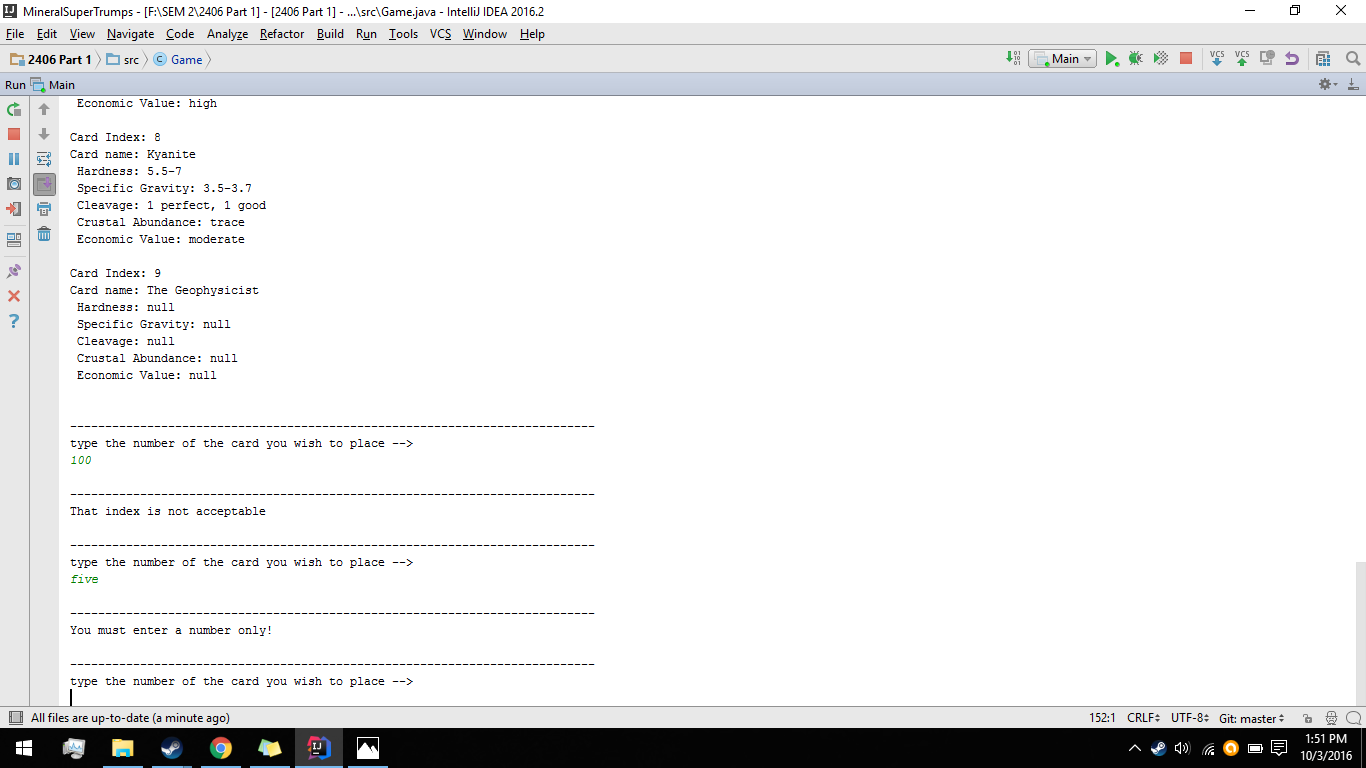
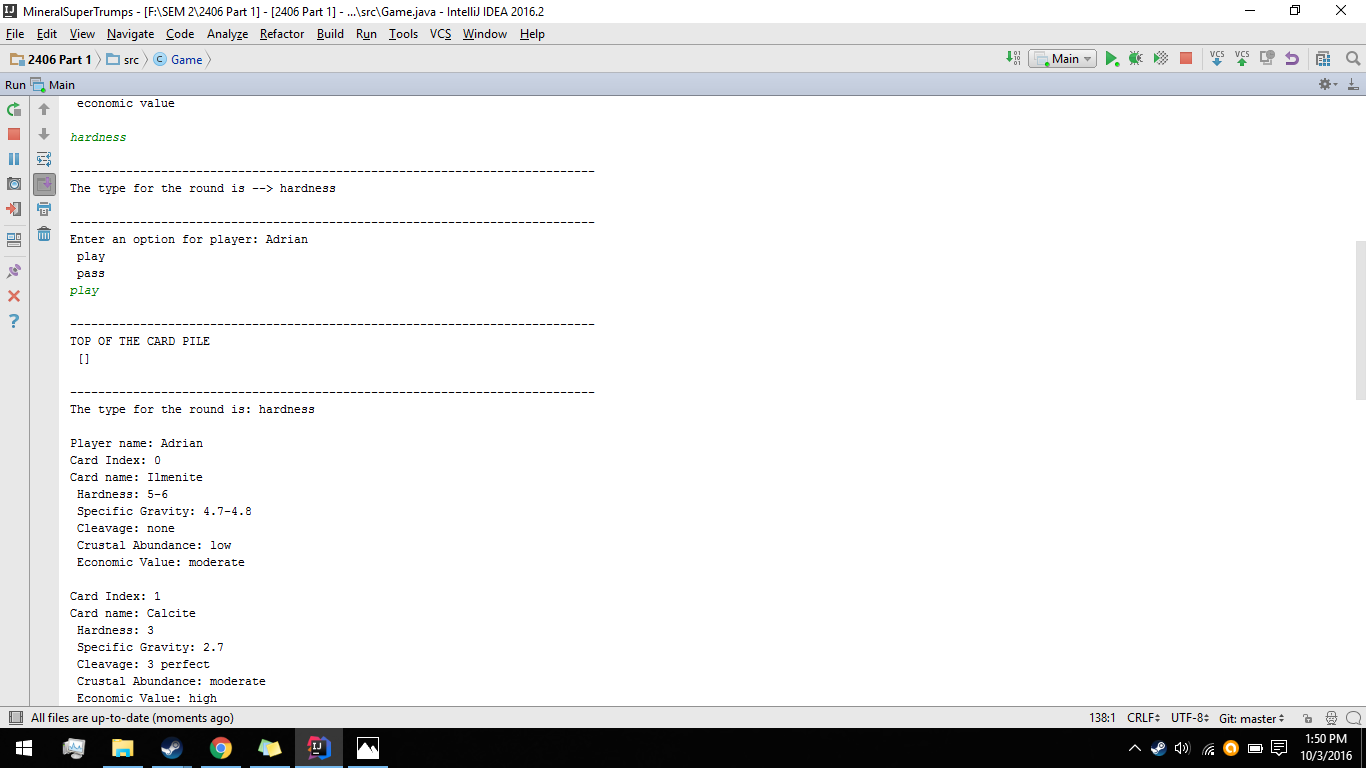


Here is where the flow of the rounds begins. Using a while loop the game will continuously cycle through players until there is all but 1 player left in the current players arraylist. With the addition of a for loop and a collection of if statements the game will cycle through each player unless that player has the inOut value in their instance set to equal false. This is how the game will know if a player has passed or not.

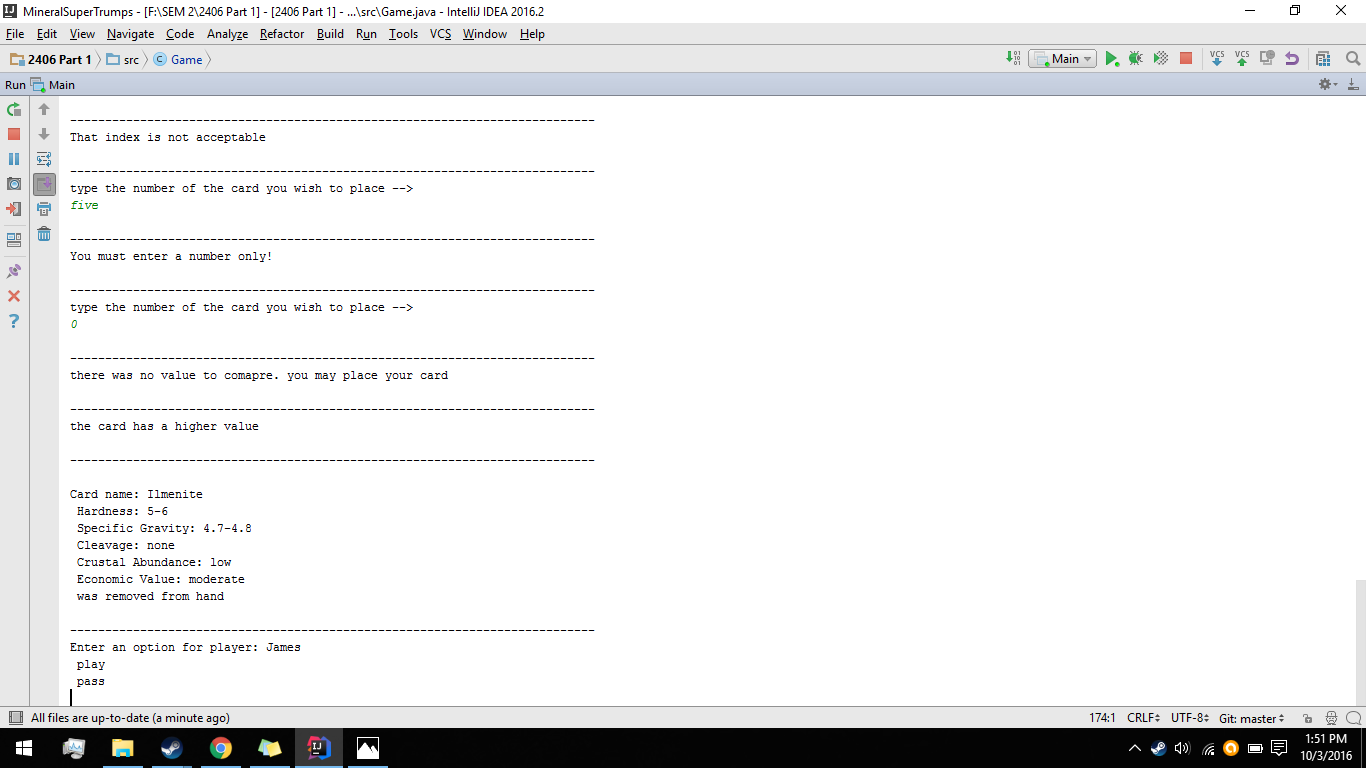
When taking control of a player you will have either 2 options to either pass or play a card. This part contains a do while loop for error checking so the player cannot enter any other options or characters.



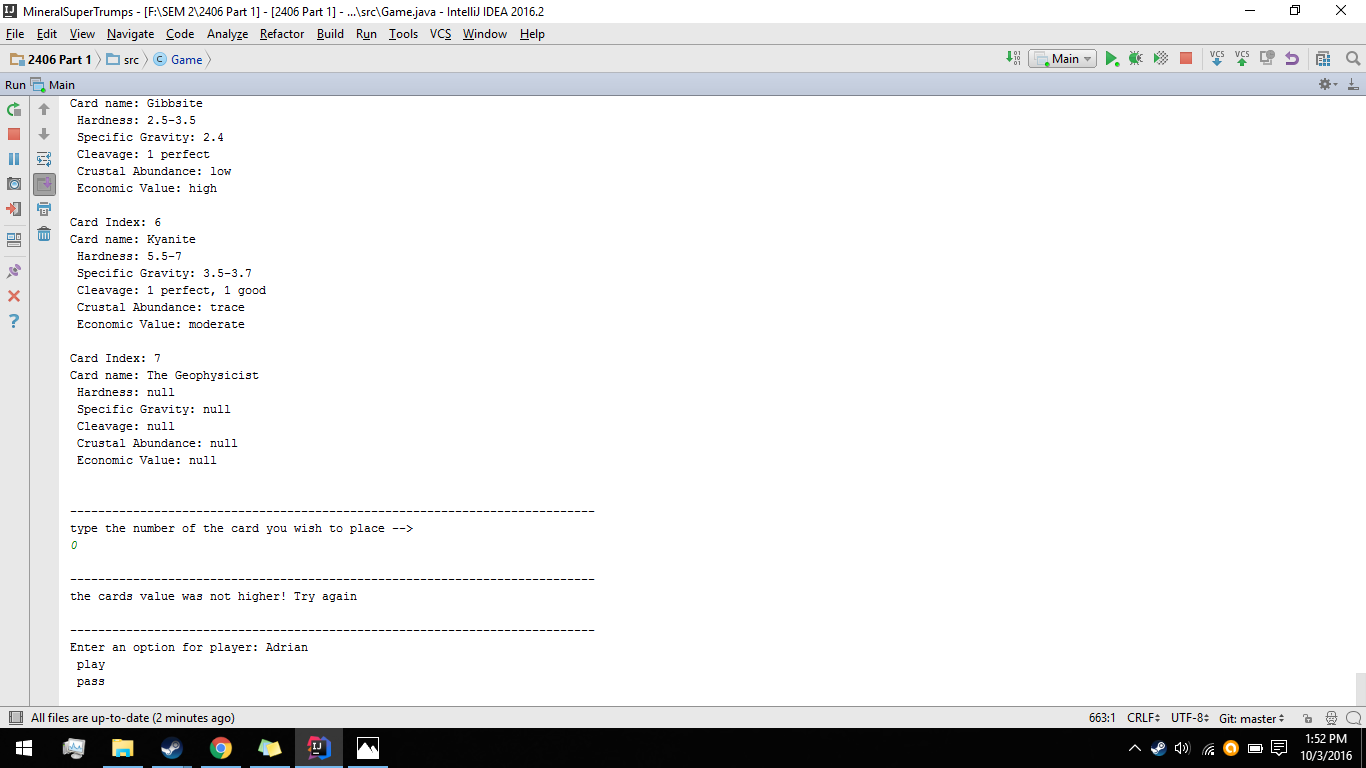
When a player chooses the play option, they will first be told the type currently set for the round (this can change however if a trump card is played). Then the player will be shown all the cards that have been previously discarded or played. Then the player will be shown all the cards they have in their hand and each card will be given an index number.



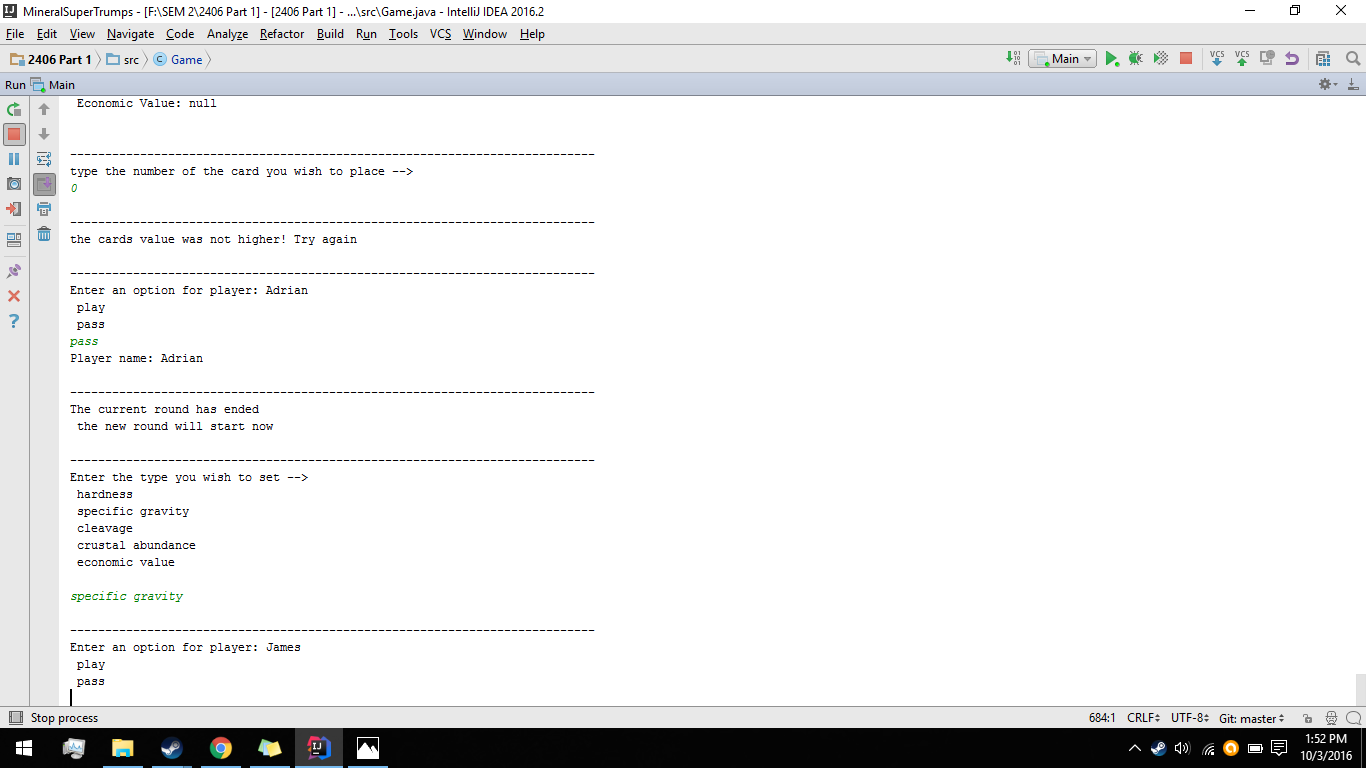
The way that a player can place a card down on the pile is by typing in the cards assigned index which it has in the hand. A try catch and do while loop have been used here to handle incorrect index inputs and incorrect character types. This is where the cardCompare() method will calculate if the card the player has played is suitable for play. If it is then a message will display that the cards value was higher on the pile. Of course if there is no card on the pile or there is no value to compare against the card the card will be placed regardless. The card will then be removed from the player’s hand and the next player will take their turn.



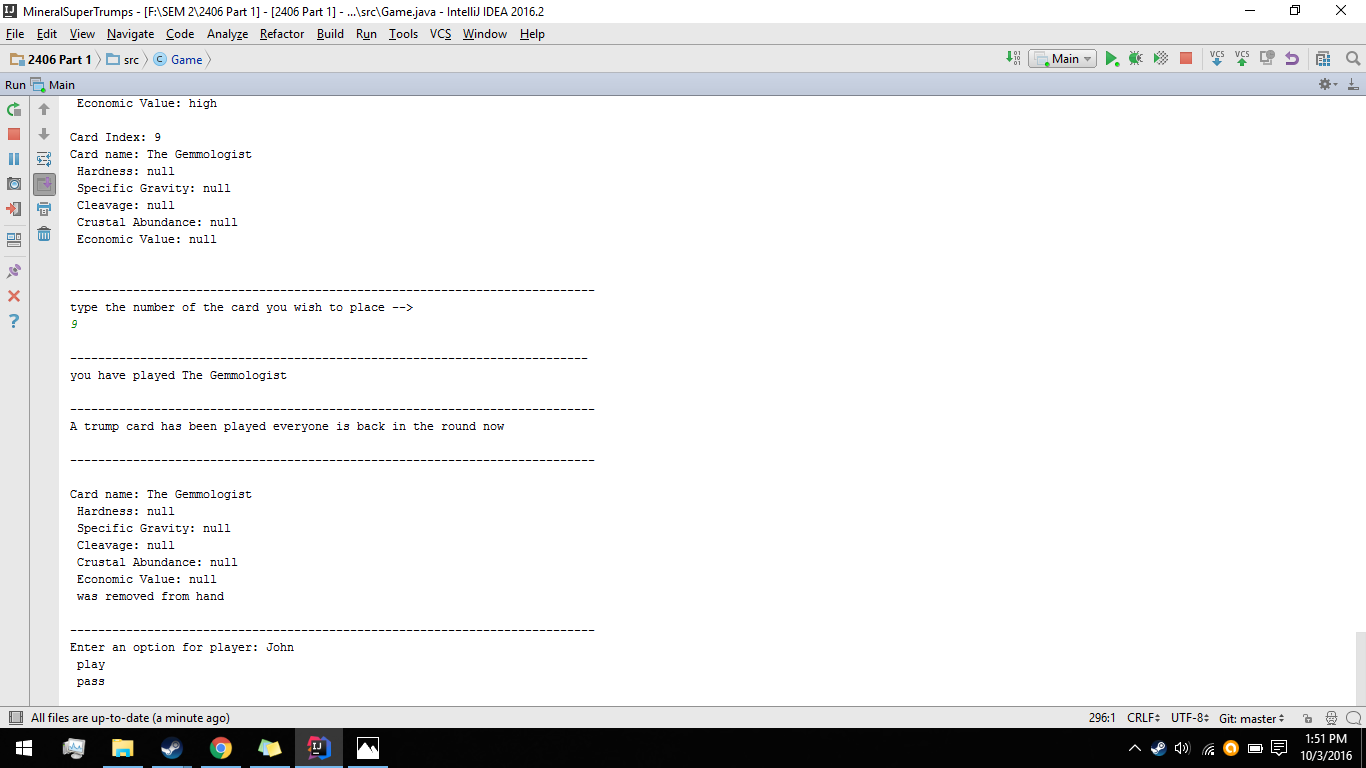
If a cards value is not higher than the cards value on the pile, then a message will be displayed and the player can either try another card or choose to pass.



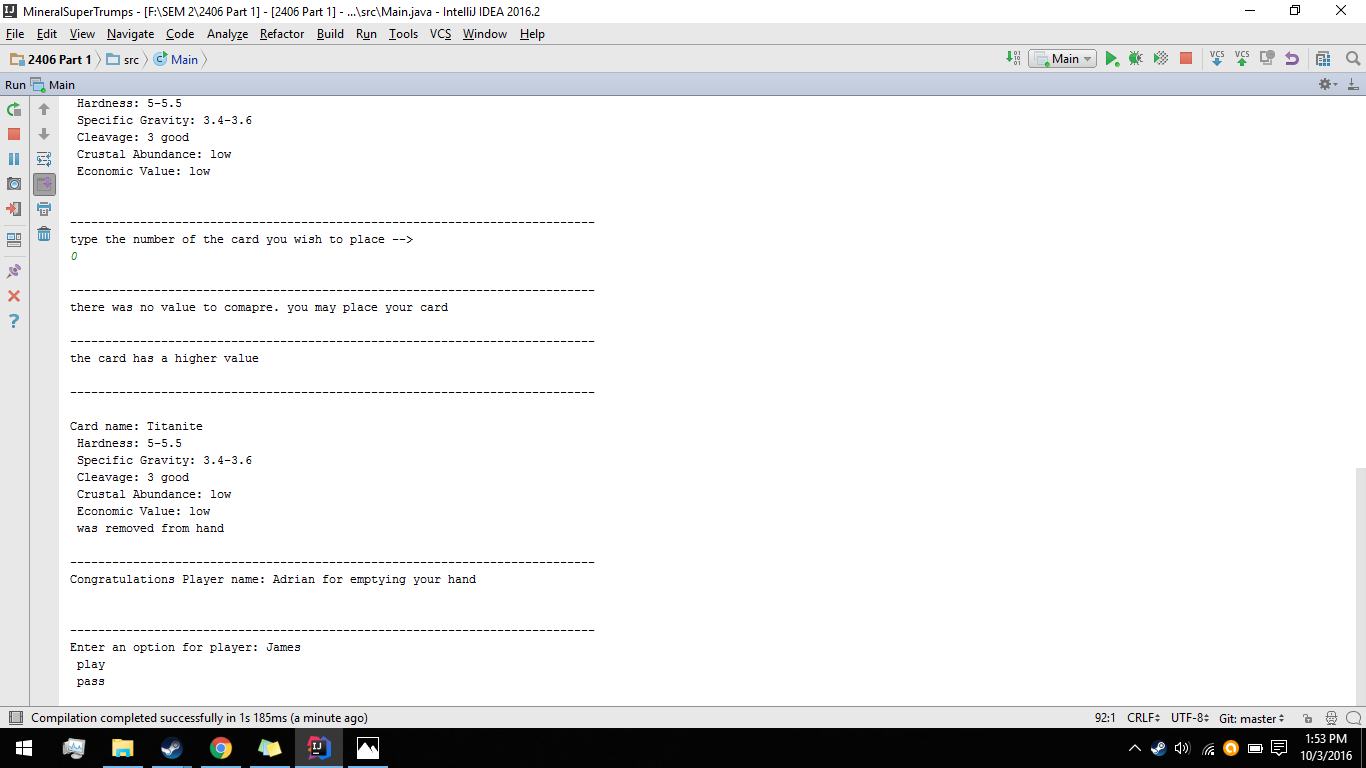
If a player chooses to pass, then their inOut counter will be changed to false and the next player will take their turn. Once everyone but 1 player has passed a new round will begin. All players inOut value will be placed to true with the help of the newRound() method. The remaining player will then be prompted to choose a new type for the round.



If a player however, happens to play a trump card then all previous people who had passed during the round will now be allowed back in due to the trumpPlayerReset() method.



In the event that a player empties their hand they will be congratulated and they will be then removed from the playerArray and then moved into the winner’s array. It is also possible for the player to win his or her hand by playing the Geophysicist trump card and by having the magnetite card in their hand at the same time.



Once everyone but a single player is left in the player array the games while loop condition will be met and the game will end by showing the winning players and the order in which they have each won.

