Design Choices-Yasin Jaamac

Dice:

This class is used for the game to generate random numbers through the roll() method. Players who role the highest go first to start the game.

Board:

This class is used to create players via an arrayList , countries via an arrayList and continents. The appropriate countries were created for the specific continents(Australia, Asia,Africa,Europe,South America and North America). This class contained a method to set the initial amount of armies for the all the cases of players entered by users. Also setting adjacencies for all countries is done here via set Adjacent Countries method in country class. Finally attack method which was done by mapping the country to the index and then as per rule checking if the passed countries are adjacent via checkAdjacentCountries method and then it went on to follow the rules for a player to attack.

Country:

This class contains the ruler with type object, with the getRuler(),setRuler() and hasRuler() method. An arraylist is used to set and get adjacent countries as a list would be the most efficient method. Increase army count and decrease army count method were put in this class and the logic for it was having a parameter numArmy for what player wants to increase army by and add that to the current number of armies, decrease Army count was same but subtraction.

Game:

This class initializes the Game via the method play, an instance of game is made and play is invoked to it. Do whiles, if statements were used to interact with the user using scanner for users to input. Users had choices to enter number of players . Players have the choice to enter a command as they already know as all the rules of game are outputted. If a command entered by a user equaled the appropriate command it would then move on with the flow. For ex once attack is entered,player would be asked to enter country to attack, country attacking from, choosing number of dice to roll for attacker and defender and would go to attack method. Same goes for fortify

Player:

We implemented an array list for the ruling Countries and ruling Continents. The rest were getter methods

Continent:

This class had methods like add country get county and dealt with the bonus army so we had a getter method for it.

Controller:

In the controller class, it acts as an interface between the model and the view. All the logic and data is separated and it intercepts all requests that come in. We had the method actionPerformed use a switch case and if a start, selection or something with mainscreen was selected it would call the specific method to deal with it. For example when startPerformed is called it goes to that method and checks if start or rules was inputed by user, the appropriate input would then be invoked by view and the appropriate method. For example if user enters start it would proceed to have view invoke method createNumOfPlayers() and if rules was inputed view would invoke method showRules(). Same goes for the other methods startperformed(),jmenubarperformed(),selectLevelPerformed() and method

mainScreenPerformed().

View:

In the view class, it represents the look of the text for our model, what the user sees. It has methods such as startmenu,which creates the startmenu panel and it is where the user is first sees when they start the game.Also method createNumOfPlayers was designed to create number of players panel by using swing components(Jbutton,JPanel etc) so user can select number of players playing. Also have other methods such as mainscreen, quit, pass etc. Furthermore it contains quit method that interacts with user if user wants to quit, JOptionOane was used to do this. Method showRules() is an option for user to click so it shows all the rules, it prints all the rules once its pressed in the menu,same goes for method help().