Name: Antreas Kasiotis
Student Number: 180355260

Testing and Debugging Template

Section 1. Issues found in Card.java

Location (e.g. constructor, method name)	**Describe the fault**	**Describe the fix**
isSameSuit	The method returns false even when compared with Card with same suit	Change card.getValue() into card.getSuit
Equals method	method Returns true when false and false when true	Comment out the first 3 if conditions in the method and left only the necessary code
toString	It returns suit=value and value= suit which is opposite	Switch value with suit and the opposite
hashCode()	Instead of this.hashCode (), variable statement should be used as a return statement	Modify the code to return the result variable
Final int	15 is set as prime number when in reality it is not	Use a valid prime number instead

Section 2. Issues found in Player.java

Location (e.g. constructor, method name)	**Describe the fault**	**Describe the fix**
isAI()	It always returns false without making any essential checking	Checking with an if statement whether the “ai” variable is true or not and then returning the equivalent value true or false
getHandAsString()	It doesn’t print the cards that the player has in his hand, but only the last card that the player got	Rewrite the current and previous output every time not just the current one
hasWon();	It returns the opposite value of what it is supposed to be returning, instead of false it returns true	Just remove the exclamation mark

Name: Antreas Kasiotis
Student Number: 180355260

Section 3. Issues found in Switch.java

Location (e.g. constructor, method name)	**Describe the fault**	**Describe the fix**
dealCards()	Cards are being shuffled before a deck is created and a null pointer exception is thrown	Replace stock=genPack(); with shuffleStock(); in order to create a deck before shuffling.
runGame()	When the user choses the option 2 the method sayGoodbye should be called instead of the method runAround	Add a new case that allows the user to exit
DiscardCard()	It is set as false instead of true	Set the variable to be true
resetFlags()	resets Draw4 to be true	Change to false
setFlag()	The cases for KING and QUEEN are opposite of what they are supposed to be	Reverse their cases

Section 4. Issues found in UserInterface.java

Location (e.g. constructor, method name)	**Describe the fault**	**Describe the fix**
getPlayerInformation()	The first println print an option for [1-4] players but in the rules of the game it is [2-4]	Replace number 1 with 2
SelectPlayer()	The check method is false because it checks if the variable ”I” which is set to be 0 is bigger than the number of players	The method should check if “I” is smaller than 0 not bigger
PrintWinnerOfGame()	The method doesn’t print the name of the winner but it prints the text before that	the method getName should be written inside the Player class to get the players name

Name: Antreas Kasiotis
Student Number: 180355260

Section 5. Issues found in Constants.java

Location (e.g. constructor, method name)	**Describe the fault**	**Describe the fix**
HAND_SIZE	The hand size cannot be 16 because based on the rules a hand that is dealt is of size 7	Change the number from 16 to 7
UI	The Ui is initialised in the wrong place. It is initialised under the For SWITCH selection	Change it's position to be under the FOR SWITCH selection

Section 6. Brief report on your testing and debugging strategies (approx. 1-2 pgs.)

Basically I just started off by looking at the instructions given for the assignment and look if the code complies with the rules of the game and whenever I would notice something wrong I would sort it out. In some cases I used the Junit method to check if a certain method would indeed return the expected value. The way that I did this is I would create a test in the Junit Test class and run it to see if the method would act appropriately, if not then I would look at the code of any inconsistencies.