|  |  |  |
| --- | --- | --- |
| **Student Ids** | Comments | Mark /30 (functionality + code) |
| 1359328  Sze Chen | **Classes**: A good selection of classes for the fully autonomous game. Your DeckOfCards class is bloated. Your discardList and DropPile objects should also be DeckOfCards objects. Aggregation is not appropriate here. A very good effort at getting a fully autonomous game working. A good use of inheritance in extending the computer player from the player class.  **Inheritance**: Fully functionality although difficult to see any difference between the basic and advanced players. Some rather untidy code submitted and you implemented the basic player functionality in the base class Player. However, you have correctly implemented polymorphism in the GUI class.  **Mini Project:** Your report is not very well presented as I can’t read some of your screenshots showing the different classes. You have presented some pseudo code but it is not very detailed. The testing description was ok as it included a test plan and some example output including screen dumps. Your decomposition and class selection are appropriate although the User class is not necessary. Quite good functionality although your GUI is awkward with an additional Update button which should not be necessary. Overall though, a good effort. | 20+8=28  15+7=22  Total=50/60  64/100 |
| 1404246  Chun Kit Lau | **Classes:** Your class structure is good with good interaction between classes. Your Deck class is rather bloated. The DiscardPile should not be aggregated into the Deck class – rather it should be an instantiation of it. A good use of container classes. Some good functionality with the interactive game implemented and automatically determining a winning hand which is a good effort.  **Inheritance**: Good functionality but you haven’t implemented polymorphism as I explained in the lecture. Rather you have passed in a parameter into a method to determine the player level. Your advanced player algorithm is a definite improvement on the basic player. Your code is nicely laid out but lacking any commenting.  **Mini Project:** Quite a well written report. You could have included some design diagrams as well as some pseudo code to explain your game logic. A good effort with testing but perhaps a more systematic approach with tabulated test results would have been a good addition to include with your screenshots. I couldn’t get the game to run. A client server connection was established but there was no indication which client’s turn it was to throw the dice. Your code is well written with a separate class to control client server connection. | 15+8=23  17+5=22  Total=45/60  57/100 |
| 1338879  Adam Robinson  1334062  Daniel Smith | **Classes:** Your class structure is good with good interaction between classes. Your Deck class is rather bloated. The discards should not be aggregated into the Deck class – rather it should be an instantiation of it. Some functionality representing an interactive game although of course it would loop until a winning hand was detected which you haven’t implemented. Nicely written and commented code.  **Inheritance:** Youhave implemented some code to implement a basic player butit doesn’t look like you have plugged it into the GUI so it is not implemented. Hence there is no observed functionality.You have implemented some code in the Draughtboard and Player classes.  **Mini Project:** Your report lacks detail in places. Not much has been written about the design process. You have a single interaction diagram which is quite simplistic. I could find very little about multithreading which is an important aspect of this exercise. You have included a bit of detail on testing but your screenshots are difficult to read. Your code has the correct set of classes including a separate class to handle client server interactions. There is not much observed functionality in your code. It seems that you have established a client server connection for multiple clients but little else. There isn’t much code in your run method of your PlayerThread class. | 13+7=20  0+5=5  Total=25/60  47/100 |
| 1383182  Weitong Yan  1347914  Shakeel Ahmed  Ah Ahmed | **Classes**: Couldn’t find any source code submitted. Looks like some functionality from the word file submitted.  **Inheritance**: Looks like full functionality although it doesn’t look like you have implemented inheritance or polymorphism as you have all of your code in the Player class. The code for calling the AdvancedPlayer method is commented out. You’ve submitted quite a bit of code which is reasonably well laid out.  **Mini Project:** Your report is reasonably well written and well presented if rather brief and lacking in some detail. Your design description including class relationships is rather superficial and I would have liked to have seen some pseudo code to perhaps describe the game logic. Your table describing testing outcomes is Ok but some screenshots showing what you did achieve might have been useful. You have some functionality. Your code establishes a client server connection for each client and displays the gui. Your server should be a console program only. You have the correct set of classes. It seems your program hasn’t been able to synchronize the clients running in turn with only one at a time being able to throw a dice. A reasonable effort though overall. | 10+0=10  14+5=19  Total 29/60  57/100 |
| 1376913  Jacob Maxwell  1364651  Raluca Sarateanu | **Classes:** Some code presented but no main method so no functionality. Your Card class is rather cumbersome and the code could have been simplified if you had used a static array.  **Draughts:** No submission  **Mini Project:** Your report is unacceptable. This should be a detailed document describing the design and implementation of your code as well as a full description of testing. Details of what should be in the report are on the course web site. It’s a shame as your program has full functionality. The design is cumbersome with too many classes – there are a lot of classes supporting the GUI which is unnecessary. I can’t find any reference to multithreading in your server code. Use of the SwingWorker extension in the Client is interesting but I am not sure if this is necessary as no lengthy computations are involved. | 0+3=3  0+0=0  Total 3/60  46/100 |
| 1394333  Sara Al-Tunaib  1364851  Timothy Rudkin | **Classes:** Some good functionality displayed with an interactive game played although it doesn’t look like you have a winning hand detection method. Your Deck class is bloated. The temp decks should not be aggregated into the Deck class – rather it should be an instantiation of it. Rather scruffy poorly indented code in places but at least it’s well commented.  **Draughts:** Looks like full functionality with the game played to completion and jumps including multiple jumps implemented. A good effort. Correct implementation of polymorphism in the GUI class. A nice use of static methods in the BasicPlayer class called from AdvancedPlayer. Not sure of the ‘added value’ in the AdvancedPLayer class as I couldn’t see much difference in the strategy when I ran it.  **Mini Project:** Your report lacks a bit of detail in places but is quite well written. You have used a diagram to illustrate your design but again it lacks detail. There is not much in your report about the multithreaded server which is an important aspect of the exercise. A good effort with describing testing using a reasonably systematic approach and including screenshots. Some of the classes your presented in your code are not mentioned in your report! I couldn’t compile your code so couldn’t check the functionality. Looking at your code, it doesn’t seem like you have implemented multi-threading. | 15 + 6=21  17+8=25  Total 46/60  35/100 |
| 1230714  Benjamin Smith | **Classes** Good functionality. You seem to have implemented a fully interactive game with automatic win detection. Your design is good but the coding is dreadful. You use incomprehensive variable names which don’t aid readability and virtually no commenting. It was very difficult for me to understand your program.  **Inheritance:** I can’t understand what you have done here. There is no GUI class so I can’t run your program. You don’t seem to have incorporated inheritance or polymorphism so you haven’t carried out the exercise. All I can give you credit for is the code submitted (which again is unreadable and not commented).  **Mini Project:** Your report is completely lacking in substance with no detail about implementation, design or testing. The required contents of a report were clearly laid out in a document on the course web site. I couldn’t find any functionality of your code. Threading has not been implemented. Your server code contains 4 infinite loops in sequence which seems bizarre. It is not clear what you have achieved here if anything. | 15 + 6=21  0+4=4  Total 25/60  12/100 |
| 1492337  Weihua Lu  1403679  Yik Ru Lui | **Classes:** Your code is somewhat complex (over-complex!) so it wasn’t clear exactly what you have done. It looks like you have implemented an interactive game to completion including winning hand detection. This represents a good effort. Good use of ArrayList container class in Deck. The code is well laid out and well commented.  **Inheritance:** More or less full functionality although multiple jumps aren’t supported (which is tricky). You have implemented polymorphism correctly although you didn’t create a separate BasicPlayer class. You advancedPlayer class is scruffy code with no commenting and not structured although it did perform better than the basic algorithm  **Mini Project:** There is not much detail about class design in your report. You have included a ‘high level’ design description which is of little value and then a list of class methods. Some formal or semi-formal design diagram would have been useful. I couldn’t find much written about the multi-threaded server which is an important aspect of the exercise. Some evidence of systematic testing including screenshots which is ok. You have some nice functionality including initiating both client and server from a single GUI. This is not strictly necessary but is convenient. You have some fairly simple client and server code but you only need 1 GUI class which should contain all the necessary panels rather than separate classes for each. | 14+8=22  14+6=20  Total 42/60  Total 70/100 |
| 1365347  Claudia Gomes Vieira  1228926  Alexander Kilhams | **Classes:** Unfortunately your code doesn’t compile so I can’t credit you with any functionality. You should make full use of the PGTA’s in the lab period if you are struggling with basic coding issues. That’s what they are paid for! A lot of the compilation errors are simple syntax. Overall your selection of classes is OK except that you do not need a separate discardDeck class as it should be an instantiation of the Deck class.  **Inheritance**: Again no functionality as compilation errors. There doesn’t seem to be a GUI class in your folder so I can’t check to see if you have implemented polymorphism. All I can find is some code in the Draughtboard class to move a piece. Again please ask in the allocated lab periods as you are clearly finding the exercises difficult  **Mini Project:** Your report has mainly focused on implementation and you have not written much about the design process. Use of pseudocode is fine but it is not necessary to then include actual code snippets after the pseudocode. Your report seems restricted to discussing how a client connects to a sever and a bit on the game logic. There is nothing in your report about multithreading which is the main point of the exercise. Your description of testing is totally inadequate. You need to have a systematic approach testing all aspects of your system. There are simple syntax errors in your code so I was not able to assess it’s functionality. | 0+6=6  0+4=10  Total 10/60  23/100 |
| 1497835  Gabriel Andrade de Souza | **Classes:** You have some basic functionality from your deck class. Shuffle and cut seem to work but a null pointer exception occurs when the deal option is selected. Your Card is fine but the Deck class is bloated with two 52-card arrays which is unnecessary. Some reasonable commented code submitted.  **Inheritance:** No submission.  **Mini Project** : There is very little functionality to your code presented. You have demonstrated a client server connection. You have also demonstrated game logic and being able to roll a dice. Your game code is inefficiently coded. You could have least have got your client to display the GUI as a further achievement. There is no multithreaded networking implemented. No detail of design or implementation in your report which consists mainly of a few screen dumps. A poor effort overall. | 10+5=15  0  Total 15/60  27/100 |
| Patel  Turner | **Classes:** Some reasonable functionality to your program. Some simple manipulations of the deck and updating a player’s hand. No implementation of a winning hand detection method but you have the basics of an interactive game. A good selection of classes but your Deck class is bloated. The discardPile should not be a member of Deck but an instantiation of Deck as it has a similar functionality.  **Inheritance:** No real functionality displayed as the board just lock. You have correctly implemented inheritance and polymorphism and you code in the BasicPlayer and AdvancedPlayer classes is well laid out (although adding some comments might help!). Not sure why the pieces don’t move as the GUI seems to suspend when you try to move a piece.  **Mini Project:** A good effort with the report. Some reasonable use of pseudo code and a very thorough description of testing. However, I couldn’t find any description of the class structure of your application, neither semi formal design diagrams nor any list of the classes you included. Your program only seems to allow a client server connection but doesn’t play the game once 4 clients have connected so I wasn’t able to verify full functionality. | 13+6=19  5+7=12  Total 31/60  49/100 |
| Russell Choudhury 1251486 and Tahir Pervez 1355444 | **Classes**: Really good work. Full automated game implemented with correct inheritance relationship for computer/human player. My only minor quibble is that an additional Discard class is redundant as it’s functionality is close to that of the Deck class.  **Inheritance:** Nicely written application. Correct implementation of polymorphism and an implementation of an AdvancedPlayer using a recursive algorithm. It seems a definite improvement on the BasicPlayer | 20+9=29  20+9=29  Total 58/60 |
| 1492116  Zhiwei Wu | **Classes:**  **Inheritance:**  **Mini Project:** Your report is not very well structured and contains no discussion about how your program was designed. It is mainly a description of how the game is played and how the clients and server connect. You have used quite a lot of code snippets in the main text which I don’t like. Please use pseudocode instead. I assume your screenshots are part of you test description showing correct operation of your program. There is no conclusion or discussion at the end of your report. Your program exhibits good functionality but please make sure you upload an up to date version of your program in future so I don’t have to edit compilation errors. | 45/100 |
|  | **Classes:**  **Inheritance:** |  |
|  | **Classes:**  **Inheritance:** |  |
|  | **Classes:**  **Inheritance:** |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |