Final Team Project Requirements

For this project you and your team will be creating a console simulation of the game *Blackjack*. Blackjack is one of the most widely-played casino games in the world[1]. The object of the game is to beat the dealer by either getting blackjack (first two cards equaling 21) without a dealer blackjack, getting a higher score than the dealer without going over 21, or the dealer going over 21. Overviews of Blackjack and its major variants are easily found on-line[2]. A detailed explanation of Blackjack is available on YouTube here[3].

Minimal Requirements (35 marks)

A project that implements ONLY the minimal requirements listed will be graded up to 70% (35/50).

	Unacceptable	Needs major improvement	Functional - significant issues	Functional - minor issues	Exemplary
OOP Concepts	0	1 – 4	5 – 6	7 – 8	9 – 10
Console UI	0	1	2 – 3	4	5
Basic Betting	0	1	2 – 3	4	5
Dealer A.I.	0	1	2 – 3	4	5
Player Options	0	1	2 – 3	4	5
Documentation	0	1	2 – 3	4	5
					/35

- ☐ **Object-Oriented Concepts.** A major component of how you will be assessed is in the degree to which you are utilizing the object-oriented concepts and techniques presented in the course. Your design should demonstrate *encapsulation*, *abstraction*, *polymorphism* and *inheritance* with a goal of ease of maintenance and reusability. Blackjack is a card game with elements that are common with many other card games (e.g. cards, decks, hands, etc). Any class that you create which could be useful in other projects should be coded in one or more header files. You are required to include at least one header file that defines these classes.
- ☐ **Basic Gameplay.**
  - o **User Interface (UI).** The project will be a console-based application written in C++ with a text/keyboard user interface (UI). You are **NOT** required to implement a graphical user interface (GUI) for your project. The UI should be simple, clean, and functional.
  - o **Basic Betting.** At a minimum, the human player should start the game with a set amount of chips. The simulation should continue until the human-player decides to stop playing, or runs out of chips to bet.
  - o **Dealer A.I.** Blackjack is always played against the dealer and no other players. For the purposes of this project you may limit the game to one human-controlled player and the computer-controlled dealer. The AI for the dealer will take hits until the dealer's cards total 17 or more. If

[1] Wikipedia, *Blackjack*, http://en.wikipedia.org/wiki/Blackjack (1 Sep 2014)
[2] Google, *Blackjack*, https://www.google.ca/search?q=blackjack (1 Sep 2014)
[3] Vegas Aces, *How to Play Blackjack FULL VIDEO*, https://www.youtube.com/watch?v=SWdPf21v5Ak (1 Sep 2014)

the dealer's first cards are an initial Ace with the remaining cards totaling six, this is known as a "soft" 17. For the purposes of this project, the dealer should hit in this situation.

- **Player Options.** The human player should have the option to hit, stand, double-down, or surrender as a minimum. You are not required to implement splitting a pair as part of the basic requirements.

- **Internal Documentation.** At a minimum, your source code should be extensively documented with opening comments and in-line commenting. Any code elements that you did not totally write yourself **MUST** be sourced ethically and legally. There are countless examples of Blackjack projects available on the Internet. If your team is using code from these examples, this **MUST** be documented with full and proper attribution of the original code author and links to the original project in the comments. Failure to do so will be considered academic dishonesty and will be dealt with as outlined in the College Student Handbook.

Recommended Features (15 marks)

A project that implements **ALL** of the minimal requirements and **three** or more of the recommended features listed will be graded up to 100% (50/50).

	Unacceptable / Not implemented	Needs major improvement	Functional - significant issues	Functional - minor issues	Exemplary
Number of Decks	0	1	2 – 3	4	5
Computer Players	0	1	2 – 3	4	5
Advanced A.I.	0	1	2 – 3	4	5
Splitting Option	0	1	2 – 3	4	5
Gameplay Log	0	1	2 – 3	4	5
Persistent Statistics	0	1	2 – 3	4	5
Approved "Other" feature(s)	0	1	2 – 3	4	5
					/15

- **Dynamic Number of Decks.** By default, Blackjack in this project should be played with a single deck of 52 cards. Include an option to play with up to eight decks of cards (2, 4, 6, or 8) at the start of the application.
- **Computer-Controlled Players.** Although the player is always playing against the dealer, it is not uncommon for several players to be playing against the same dealer at the same table. At the start of the simulation, provide the option for up to four computer-controlled players. At a minimum, the computer-controlled players should be able to bet (until they run out of chips), hit, and stand.
- **Advanced Computer-Controlled A.I.** This option requires computer-controlled players to be implemented. Implement A.I. so that the computer-controlled players have the option to hit, stand, double-down, or surrender based on basic Blackjack strategy[4].

[4] Wikipedia, *Blackjack Basic strategy*, http://en.wikipedia.org/wiki/Blackjack#Basic_strategy (1 Sep 2014)

Final Team Project Requirements

- ☐ **Splitting.** Implement the option for the human player to split if they have an initial pair.
- ☐ **Gameplay Log.** As the game is being played, record the relevant game-play actions to a text-file log. This log should record the date and time the game was started and the initial hands dealt at the game start. It should also record the actions of both the human player and any computer-controlled players (if implemented) for each game, including any cards drawn. Once the game is complete, it should record the results.
- ☐ **Persistent Statistics.** Record and update the human-player's name, the amount of chips they hold, and a summary of game results. This information should be stored in a text file and loaded every time the application is run. Provide the player the option to reset the name and statistics.
- ☐ **Other features.** If you have ideas for other features, they should be proposed to the instructor.

Evaluation Notes

Take note of the following points from the course outline regarding the final team project and in-progress team project activities:

- The final team project will be completed by teams of students only. Students will be required to self-select their own team by the deadline specified on DC Connect. The instructor may assign students to teams in exceptional circumstances. Individual final projects will not be accepted.
- Equitable participation in all final project team activities is required. Any issues regarding student participation should be brought to the instructor's attention immediately. Gross inequities identified will be dealt with on a case-by-case basis. This may include a non-participating student being removed from the team. In addition, all students will be given an opportunity to assess their own participation and that of their peers at the end of the course.
- The final team project is due by the due date assigned and posted on DC Connect and will not be accepted late.