CSC 209 Assignment 4, Summer 2003

Due by August 8, 2003, at midnight. No assignments will be accepted after this time.

Black Jack Server

For this assignment, you will implement a Black Jack server that allows people to connect to it and play the game of Black Jack over the network. The client application is provided and can be downloaded from the course web site. Before getting into the details about the assignment, let's discuss the rules of the game.

Black Jack

Black Jack is the most popular casino game of all. Black Jack's popularity is so high because if played right the odds are much higher than in any other game. In Black Jack you don't play against any other players - only against the dealer. The main objective in Black Jack is to get a hand higher than the dealer's without going over **21**. Note that the rules we play by differ slightly from the casino version. In particular, in the casino, the dealer gives every player one card, face down, including himself. This is the dealers **down card**. Then he deals a second round of cards, face down but this time the card he deals himself will be face up. This is the dealers **up card**. You now can look at both of your cards and find your total by simply adding the values of your cards (If the two cards you initially get add up to 21, then it is a ***BlackJack***). **To simplify our game, we will not bother with the dealer down card and up card. To start the game, you get no cards at all, nor does the dealer.**

The values of the cards in Black Jack from two to ten are at face value. **Jacks, Queens** and **Kings** count ten and the Ace counts eleven **or** one. The Ace always counts eleven **except** if your total **exceeds** 21 - then the value of the Ace is reduced to one. A hand with one Ace having the value of eleven is called a **soft hand** and a hand with all Aces having the value of one is called a **hard hand**. In Black Jack for instance, if you get an 8 and Ace dealt it would be a soft 19 while an 8, 10 and Ace would be a hard 19. Exceeding a total of 21, and already counting all the aces you have in your hand as one, means you are bust and lose your bet.

By turn each player will then have to make one of the following two decisions.

1. **Hit** / If you are not satisfied with your current total you can ask the dealer to hit you which means he deals you another card in addition to what you currently have. You are hit until you are satisfied with your total, or until you bust.
2. **Stand** / You stand if you don't want any more cards.

Again, this is simplified from the casino's BlackJack rules. Most casino allows the following options: **Double**,**Split**,**Insurance**,**Surrender.** For simplicity, we will not support any of these options.

After the player has made his/her decision the dealer will then play his/her hand. The playing of the dealer's hand must follow certain rules. He/she must **hit** on every total less than **17** or otherwise **stand**. Some casinos even let the dealer hit when he/she has a soft 17, but for simplicity, we will not allow this.

You win if either the dealer busts or has a total less than yours. If the total is the same it's a draw.

The Client

The client is provided and does the following:

1. It will ask you for an IP address to connect to.
2. It will ask you for the port to connect to.
3. The client will attempt to connect with the server you write at the IP and port you specified.
4. If the connection is successful, the client will read a message sent from the server and display it on screen.
5. The client will then read a line of input from you, and send it back to the server.
6. Steps 4 and 5 are repeated until the connection is terminated.

Your Job

You are responsible for implementing the Black Jack server. The game itself is run on the server. When started, the server should first prompt the user to see which port is to be used. The server then sits idle and waits for incoming connections. The server must be able to support simultaneous connections. Once a connection is established, the server will repeatedly perform two things:

1. Display a menu of options the user has by sending the text to the client.
2. Read the response from the client and parse it to understand the user's intentions.

The menu presented to the user should have the following options:

1. New Game – this will clear the current game and start a new one.
2. Hit – Randomly draw a card from the deck and add it to the player's list of cards for the current game. The card dealt and the player's current total should be printed. If the player went bust, this should be printed.
3. Stand – The player's total should be frozen, and the dealer (computer) should start to randomly deal itself cards. Each card dealt should be printed, as should the dealer's total. The server should also print out who won the game, or if it was a draw.
4. Show Hand – Display the current hand of both the player and the dealer. The dealer's hand is only printed after the game is over.
5. Debug – Select this option to toggle debug mode. If debug mode is on, then instead of randomly choosing a card from the deck, the server should prompt the user (on the client) for the card that it will deal. This is true for both the player and the dealer. The card to be dealt is specified by a pair of integers, like 0,11 – the meaning of the numbers are as follows:

   0 – hearts
   1 – Spades
   2 – Clubs
   3 – Diamonds

   The cards are numbered 1 through 13, where 1 is Ace, 2-10 are face value, and 11 is Jack, 12 is Queen, 13 is King. The first number of the pair signifies the suite, and the second number gives the card number. If the server is currently in debug mode, then it should say so at the bottom of the menu.
6. Quit

The user responds on the client by typing in the appropriate number. Once the server reads and parses the number the appropriate action is taken, and any resulting text is sent back to the client for display. Note that if the server is in debug mode, then a request to hit will result in the server responding by asking the client which card to deal by specifying the pair of integers.

Important! Your server must be named bjserv. Your submission must include an appropriate makefile, and we should be able to build your project by typing make bjserv. You also must implement the debug mode for us to grade your assignment.

To submit

Each source file ***must*** contain a prologue comment stating your name, student number, and TA or tutorial room number. Once you are satisfied with your programs, you can submit them for grading. Submit all files, including your Makefile, in the usual manner. For example:
submit -c csc209h -a A4 bjserv.cpp
You can check that your assignment has been submitted with the command
submit -l -c csc209h -a A4
You can resubmit a particular file by using the -f option, e.g.
submit -c csc209h -a A4 -f bjserv.cpp

This is the only submission method; you do not turn in any paper.