# Lab 3

CS321L Winter 2023, Professor Christopher Diggins

## Overview

This lab involves creating, using, and testing objects in the context of a Blackjack game. See: https://en.wikipedia.org/wiki/Blackjack for the rules of Blackjack.

The object of the game is to win money by creating card totals higher than those of the dealer's hand but not exceeding 21, or by stopping at a total in the hope that the dealer will bust. On their turn, players choose to "hit" (take a card) or "stand" (end their turn and stop without taking a card).

Face cards are worth 10, and Aces are worth either 1 or 11, depending on what is more advantageous for the hand evaluation. A hand evaluated with an Ace worth 11 is called a “soft” hand.

In our version of Blackjack:

- We will use a single deck of cards.
- Players cannot split, double, or surrender.
- Dealer stands on a soft 17 or higher, or hits automatically.
- After the bet the player receives two initial cards
- Next the dealer receives one card face up.
- Then the user can choose what to do

Payout:

- If the player busts, they lose their bet.
- If the dealer gets blackjack, the player loses their bet.
- If the player gets blackjack, the player gets 2.5 * bet.
- If the player hand has a higher value than the dealer, they get 2 * bet
- If the player hand is the same as the dealer, they keep their bet
- If the player hand is less than the dealer, they lose their bet

## Tasks

Create the following classes:

- Card
- Deck
- Hand
- Player
- Game
- Options

Create the following enumerations:

- Suit – Hearts, Clubs, Diamonds, Spades
- Rank – Ace, Two, …, King
- Decision – Hit, Stand

Create and use the following methods:

- Decision Game.GetUserDecision()
- Card Deck.GetNextCard()
- Bool Deck.IsEmpty()
- void Deck.Reset()
- void Deck.Shuffle()
- bool Game.DealerStands(Hand)
- int Hand.GetValue()
- bool Hand.IsBust(Hand)
- bool Hand.IsValueSoftOrHard()
- void Hand.AddCard(Card)
- void Hand.ToString()
- void Game.ToString()
- int Options.NumberOfDecks()

Submission:

- The project file and source code of the game.
- Play three rounds of the game. Put the output in a text-file: output.txt.

Grading

- **4 points**: the demonstrated output of the game respects the rules outlined above.
- **2 points**: the code implements and uses the requested methods
- **4 points**: code follows the coding guidelines and is easy to read and understand.