


Assignment 2

[Submit Assignment](#)

Due Nov 16 by 11:59pm **Points** 10 **Submitting** a file upload **File Types** zip
Available until Nov 21 at 11:59pm

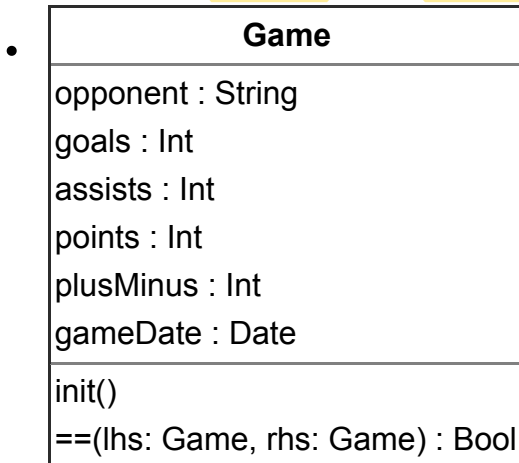
Assignment to be done individually. Write an iPhone app that allows users to track hockey games they play. The user will be able to add game records that keep track of the opponent, goals, assists, plus/minus, and the date the game was played. The app will use a navigation controller, a table view controller, and a view controller. The app must persist the user's data between runs. The app must be properly commented. Each method you create must have a brief description preceding it. When complete, compress your project folder into a single zip file and submit it.

Hockey Tracker

- Need a Mac? See [Student Resources](#).
- Create an iOS Application using the Single View Application template.
 - Product Name: Hockey Tracker
 - Organization Name: Mohawk College
 - Organization Identifier: ca.mohawkcollege
 - Language: Swift
 - User Interface: Storyboard
 - Use Core Data: Unchecked
 - Include Unit Tests: Unchecked
 - Include UI Tests: Unchecked
-  Source Control: Create Git repository on my Mac: Unchecked
- You will need to delete the existing view controller (and in the storyboard) and add a new one named **GamesViewController**.
- Add the following statement of authorship as a comment to the top of GamesViewController.swift replacing Student Name and 123456789 with your name and student number. **Failure to include a statement of authorship will result in a grade of 0.** These two lines must be the first two lines of the file.
 - I, Student Name, student number 123456789, certify that this material is my original work. No other person's work has been used without due acknowledgement and I have not made my work available to anyone else.
- Images for app: [02_hockey_tracker_images.zip](#). Add the images to the image asset catalog.

Game

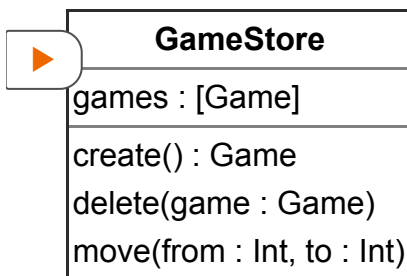
- Add a Swift class named **Game**.
- The Game class implementation must adhere to the class diagram.
- It also must conform to the **Equatable** protocol.



- Add a property observer to both goals and assists that ensures they don't go below 0. If either goes below 0, set them to 0.
- points is a computed property that returns the sum of goals and assists.
- Set opponent to an empty string, goals, assists, and plusMinus to 0 and gameDate to the current date in the initializer.
- Not a follower of hockey? Wondering what plus/minus is? If a player on team A is on the ice and team A scores a goal, the player is assigned +1; if team B scores, the player is assigned -1. At the end of the game, plus and minuses are summed and this yields the plus/minus for the game. This explanation is slightly simplified and doesn't consider penalties. The point for this assignment is that this value can be negative, 0, or positive. Whereas, goals, assists, and points cannot be negative.

GameStore

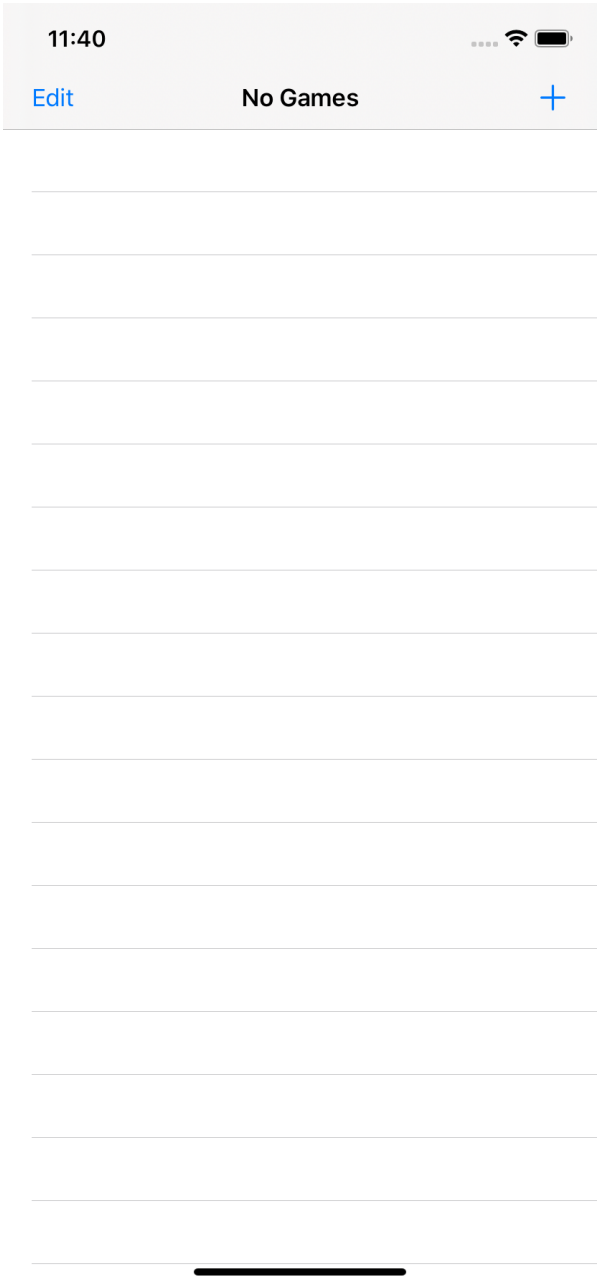
- Add a Swift class named **GameStore**.
- The GameStore class implementation must adhere to the class diagram.



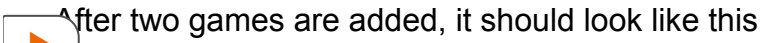
- When games are created, they must be added to the games array.

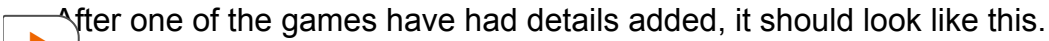
GamesViewController

- When the app initially loads, it should look like this

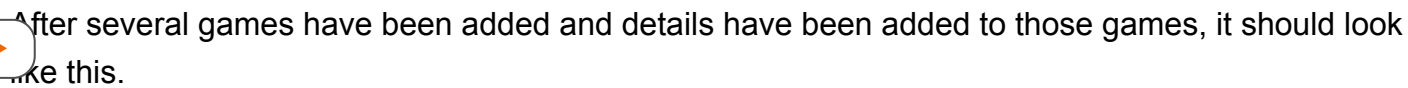


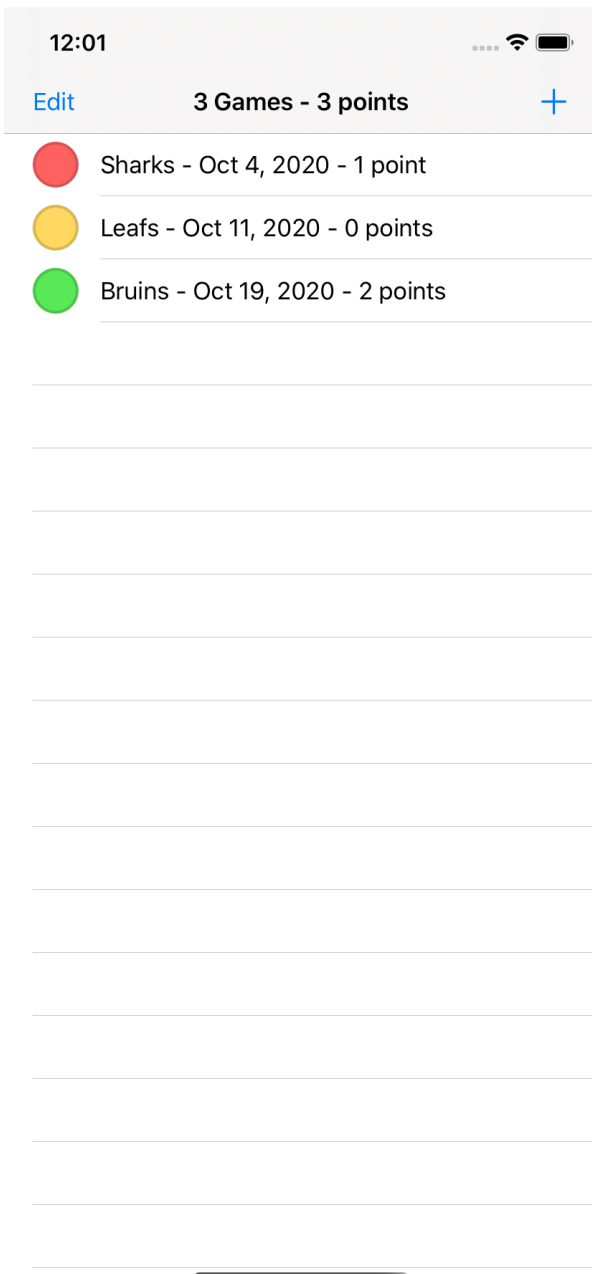
When the plus button is tapped, have the game store add a new game.





●








- ▶ The table must support deleting and rearranging rows.
- ▶ Add a method named `updateTitle` that updates the navigation title as necessary. It will indicate how many games are present and the total number of points scored in those games. Call this method when appropriate.
- In the title the word games and points will appear. Whenever there is only 1 (game or point), the word should be game or point. Example: 0 Games, 1 Game, 2 Games, 3 Games, etc.
 - Set the Table View Cell Style to Basic.
 - Each cell has an `imageView` property that further has an `image` property.
 - This image property can be set to a `UIImage` object created with a named file, using `UIImage's init(named:)` initializer.
 - If a game opponent hasn't been entered yet, show the blank circle in the cell along with the text seen in the screenshots.
 - If an opponent has been entered, show the circle as below, show the opponent, the game date formatted as shown and the number of point(s) as shown in the screenshots.


- Games that have a plusMinus below 0 show the red circle in the cell.
- Games that have a plusMinus equal to 0 show the yellow circle in the cell.
- Games that have a plusMinus above 0 show the green circle in the cell.
- Once an opponent has been entered, the cell should show the image, the opponent's name, the formatted date of the game and the number of points scored, as shown in the screenshots.
- When an existing row is tapped, show the game on the DetailViewController.

DetailViewController

- Stack views are the recommended approach to layout the view controller, the text fields and the date picker should line up.
- If the current game doesn't have an opponent, the navigationItem title should look like this.

11:57

•••••

 Back


New Game

Opponent

Goals

Assists

Plus/Minus



Game Date

April	1	2013
May	2	2014
June	3	2015
July	4	2016
August	5	2017
September	6	2018
October	7	2019
November	8	2020
December	9	2021
January	10	2022
February	11	2023
March	12	2024
April	13	2025
May	14	2026
June	15	2027

- Entering a new game should look like this.

11:58

BackNew Game

Opponent

Sharks

Goals

Assists

Plus/Minus

April	1	2013
May	2	2014
June	3	2015
July	4	2016
August	5	2017
September	6	2018
October	7	2019

Game Date

November82020

December92021

123

456

789

0



If the current game has an opponent, the navigationItem title should look like this.

11:59

< 2 Games - 1 point Sharks

Opponent

Goals

Assists

Plus/Minus

March	28	2013
April	29	2014
May	30	2015
June	31	2016
July	1	2017
August	2	2018
September	3	2019

▶ If the return key or background are tapped, the keyboard should be dismissed.

▶ Centre the text input for goals, assists and plus/minus.

- Investigate how the date picker works and add it to your code. Ensure that it looks as shown.
- When the "back" button is tapped, "save" the changes to the game.
 - Only "save" a game if an opponent has been entered.
 - Ensure that if goals, assists or plus/minus are entered, that they are valid integers.

Persisting Data

- Once the app is functional to this point, modify it so that it persists its data.
- Modify the Game class to conform to the Codable protocol.

GameStore

- Modify the GameStore to persist the data.

- Updated GameStore class diagram.

GameStore
games : [Game] itemArchiveURL : URL
init() create() : Game delete(game : Game) move(from : Int, to : Int) saveChanges()

- The archiveURL closure property will determine the location of the file which will be named games.plist.
- The initializer will attempt to unarchive the game data and put them into the array. It will also register as an observer for when the app enters background.
- Save the game data into the plist.

