Assignment I: Concentration (Концентрация)

Цель:

Цель задания состоит в воспроизведении демонстрационного примера, представленного на Лекции, и небольшое его усовершенствование. Очень важно, чтобы вы понимали, что вы делаете на каждом шаге воспроизведения демонстрационного примера Лекции, это даст вам возможность подготовить себя к расширению возможностей игры Концентрация.

Другая цель задания состоит в приобретении опыта создания проектов в Xcode и набора кода с нуля. **Не используйте copy / paste любого кода откуда угодно**. Печатайте и смотрите, что Xcode делает, когда вы что-то вводите.

Обязательно посмотрите ниже раздел подсказок **Подсказки (Hints)**!
Посмотрите также раздел **Оценка (Evaluation)**.

Материалы

- Вам необходимо установить (бесплатно) программу Xcode 9 с App Store на вашем Mac (предыдущая версия Xcode НЕ будет работать). Настоятельно рекомендуется, чтобы вы сделали это незамедлительно, так как в случае возникновения каких-то проблем, связанных с работой Xcode, вы могли бы получить помощь на Piazza или от ассистентов профессора в их офисные часы.
- Ссылку на видео лекций вы можете найти там, где вы нашли этот документ, то есть на iTunes U.

Обязательные пункты задания

1. Заставьте работать игру Концентрация, демонстрируемую на Лекциях 1 и 2. Печатайте весь код, не пользуйтесь копированием и вставкой кода откуда-то.

2. Добавьте больше карт в вашу игру.

3. Добавьте на ваш UI кнопку “New Game”, которая заканчивает текущую игру и начинает новую.

4. В данный момент карты в Моделе не рандомизированы (именно поэтому в вашем UI парные карты всегда лежат на тех же самых местах). Перетасуйте карты в методе init() класса Concentration.

5. Введите в игру концепцию “Тема” (“theme”). Тема **theme** определяет множество эмоджи, из которого выбираются эмоджи для карт. Все эмоджи в определенной теме **theme** должны иметь отношение к этой теме. Смотри **Подсказки (Hints)**, в которых есть примеры таких тем. Ваша игра должна, по крайней мере, иметь 6 различных тем, и темы должны выбираться случайно каждый раз при старте новой игры.

6. Ваша архитектура должна давать возможность добавлять новую тему одной строкой кода.

7. Добавьте на ваш UI метку для счета в игре (score label). Счет в игре формируется добавлением 2-х очков за каждое совпадение и штрафом в 1 очко за каждое несовпадение ранее увиденной карты.

8. Отслеживание числа переворотов карт flipCount определенно НЕ принадлежит вашему Controller в правильной MVC архитектуре. Исправьте это.

9. Весь новый добавленный UI должен правильно располагаться и выглядеть хорошо в портретном режиме на iPhone X.

Подсказки (Hints)

1. Экономия очень ценна в кодировании: самый легкий путь избежать ошибок при кодировании - это вообще не писать ни одной строчки кода. Это задание требует очень, очень мало строк кода, так что если вы начинаете писать более 100 строк кода, то вы явно движетесь не в том направлении.

2. Примеры тем: animals (🐼🐓🐴), sports (🏈🏉⚾), faces (😀😦😑).

3. Если вы перевернули 🐧 + 👻, затем перевернули ✏ + 🎈, затем перевернули 2-х 👻, ваш счет будет 2, потому что вы получаете очки за совпадение (и никаких штрафов вам не будет начислено за переворот 🐧, ✏ или 🎈, так как они еще не были вовлечены в процесс проверки на совпадение, а также 👻 не участвовал в несовпадении). Если затем вы перевернули 🐧 + 🐼, затем перевернули 🎈 + 🐧, ваш счет упадет на 3 очка до -1, потому что 🐧 мы уже видели (при самом первом перевороте) и она последовательно участвовала в двух отдельных несовпадениях (уменьшаем счет на -1 за каждое несовпадение) и 🎈 также участвовал в несовпадении после того, как мы уже видели его (-1). Если мы перевернем затем 🐧 + 🐧, мы получим 2 очка за совпадение и вернемся к общему числу очков 1.

4. Если вы хотите получить массив Array, содержащий все ключи в словаре Dictionary (имеющем имя foo в данном примере), используйте ...

```
let fooKeys = Array(foo.keys)
```

Что нужно изучать

Это частичный список концепций, в которых это задание увеличивает ваш опыт работы или демонстрирует ваши знания.

1. Xcode 9
2. Swift
3. MVC
4. UIViewController subclass
5. UILabel and UIButton
6. Target/Action (@IBAction)
7. Outlets (@IBOutlet) и Outlet Collections
8. functions и properties (instance variables)
9. let против var
10. Value type (struct, enum) против reference type (class)
11. Strong typing и type inference (вывод типа из контекста)
12. didSet
13. **for in** (и ..< CountableRange синтаксис)
14. Array<Element> и Dictionary<Key, Value>
15. [Element] и [Key:Value] синтаксис
16. Инициализация struct и class
17. viewDidLoad
18. Optionals (включая неявно “развернутые” Optionals)
19. ?? Optional оператор для значения по умолчанию
20. // TODO
21. arc4random()
22. Преобразование типов (например, из UInt в Int)
23. Stack View и (простой) autolayout

Оценка (Evaluation)

Во всех заданиях этого семестра требуется написание качественного кода, на основе которого строится приложение без ошибок и предупреждений, следовательно вы должны тестировать полученное приложение до тех пор, пока оно не начнет функционировать правильно согласно поставленной задачи.

Приведем наиболее общие соображения, по которым задание может быть отклонено:

- Приложение не создается.
- Один или более пунктов в разделе **Обязательные задания (Required Tasks)** не выполнены.
- Не понята фундаментальная концепция задания.
- Приложение не создается без предупреждений.
- Код - небрежный или тяжелый для чтения (например, нет отступов и т.д.).
- Ваше решение тяжело (или невозможно) прочитать и понять из-за отсутствия комментариев, из-за плохого наименования методов и переменных, из-за непонятной структуры и т.д.

Часто студенты спрашивают: “Сколько комментариев кода нужно сделать?” Ответ - ваш код должен легко и полностью быть понятным любому, кто его читает. Вы можете предполагать, что читатель знает iOS API и знает, как работает представленный на Лекциях 1 и 2 код игры **Concentration**, но вам не следует предполагать, что он уже знает решение проблемы в этом Задании.

Дополнительные пункты (Extra Credit)

Мы постарались создать Дополнительные задания так, чтобы они расширили ваши познания в том, что мы проходили на этой неделе. Попытка выполнения по крайней мере некоторых из них приветствуется в плане получения максимального эффекта от этого курса. В этом Задании дополнительные пункты (Extra Credit) очень легкие и их немного (это скорее похоже на “разминку” для будущих Заданий).

1. Измените цвет фона (background) и цвет “обратной” стороны карты (“рубашки”) так, чтобы они соответствовали теме theme. Например, у нашей темы **Хэллоуин черный** цвет фона и **оранжевый** цвет “рубашки” карт. Возможно, у темы **Зима** будет **голубой** фон и **белая** “рубашка” карт. Тема **Строительство** будет **черной** и **желтой**. У UIViewController есть свойство с именем view, которое подсоединено к топовому **view** в экранном фрагменте (**scene**) (то есть **view**, которое было черным в Лекции)..

2. Вы можете определять время с помощью структуры Date. Почитайте документацию о том, как она работает и используйте ее при вычислении счета в игре (**score**) таким образом, что чем быстрее мы выбираем карты, тем лучше счет пользователя. Вы можете модифицировать Обязательное Задание 3, связанное с вычислением счета в игре, но счет все равно должен отражать вознаграждение от совпадений и штраф от несовпадений ранее виденных карт (а дополнительно основываться на времени). Совершенно нормально, если “хороший счет” будет меньшим числом, а “плохой счет” - большим числом.