```swift
//
//  OrganizationalLeadership.swift
//  FBLA-QuizME
//
//  Created by Udit Garg on 11/28/18.
//  Copyright © 2018 Udit Garg. All rights reserved.
//

import Foundation
import UIKit
import MessageUI


class OrganizationalLeadership: UIViewController {

    // Set up variables that represent labels and buttons on the storyboard
    @IBOutlet weak var QuestionLabel: UILabel!
    @IBOutlet weak var Answer1: UIButton!
    @IBOutlet weak var Answer2: UIButton!
    @IBOutlet weak var Answer3: UIButton!
    @IBOutlet weak var Answer4: UIButton!
    @IBOutlet weak var NextQuestion: UIButton!
    @IBOutlet weak var ScoreLabel: UILabel!

    // Create an array of integers that represent the number of questions
    var randomQuestionArray:[Int] = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]

    override func viewDidLoad() {
        super.viewDidLoad()

        // Hide Initial Next Question Buttons
        NextQuestion.isHidden = true

        //Format the buttons
        Answer1.layer.borderWidth=1
        Answer1.layer.borderColor=UIColor.darkGray.cgColor
        Answer1.layer.cornerRadius=5
        Answer2.layer.borderWidth=1
        Answer2.layer.borderColor=UIColor.darkGray.cgColor
        Answer2.layer.cornerRadius=5
        Answer3.layer.borderWidth=1
        Answer3.layer.borderColor=UIColor.darkGray.cgColor
        Answer3.layer.cornerRadius=5
        Answer4.layer.borderWidth=1
        Answer4.layer.borderColor=UIColor.darkGray.cgColor
        Answer4.layer.cornerRadius=5

        // Set the answers to be incorrect
        Answer1Correct = false
        Answer2Correct = false
        Answer3Correct = false
        Answer4Correct = false

        // As soon as the view loads start generating the questions
        RandomQuestions()

        ScoreNumber = 0
    }

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }

    // If the answer is right then the Next Question button is enabled
    func rightAnswer() {
        NextQuestion.isHidden = false
        ScoreNumber = Int(ScoreNumber) + 2
        ScoreLabel.text = String(format: "%i", ScoreNumber)
    }

    // If the answer is wrong then the Next Question button is hidden
    func wrongAnswer() {
        ScoreNumber = Int(ScoreNumber) - 1
        ScoreLabel.text = String(format: "%i", ScoreNumber)
    }

    // This function randomly generates questions without repeat
    func RandomQuestions(){

        Answer1.isEnabled = true
        Answer2.isEnabled = true
        Answer3.isEnabled = true
        Answer4.isEnabled = true


        // This makes randomIndex represent the number of questions available for this
question
        let randomIndex = Int(arc4random_uniform(UInt32(randomQuestionArray.count)))

        // Generates questions until all of the questions for this topic have been
answered
        if randomQuestionArray.count > -1 {

            switch (randomQuestionArray[randomIndex]) {
                case 0:
                    QuestionLabel.text = "SWOT analysis does NOT look at a leader's"
                    Answer1.setTitle("opportunities", for: .normal)
                    Answer2.setTitle("weaknesses", for: .normal)
                    Answer3.setTitle("strengths", for: .normal)
                    Answer4.setTitle("timeline", for: .normal)
                    Answer4Correct = true
                case 1:
                    QuestionLabel.text = "Which of the following is an example of a
figurehead role?"
                    Answer1.setTitle("serving on committees", for: .normal)
                    Answer2.setTitle("scheduling when employees will use resources",
for: .normal)
                    Answer3.setTitle("signing official documents", for: .normal)
                    Answer4.setTitle("answering letters", for: .normal)
                    Answer3Correct = true
                case 2:
                    QuestionLabel.text = "Two-factor theory proposes that employees
are motivated by"
                    Answer1.setTitle("poor counties and industrializing countries.",
for: .normal)
                    Answer2.setTitle("rich countries and poor countries.", for:
.normal)
                    Answer3.setTitle("poor counties and other poor countries.", for:
.normal)
                    Answer4.setTitle("rich countries and other rich countries.", for:
.normal)
```

```swift
                    Answer3Correct = true
                case 3:
                    QuestionLabel.text = "The neutral third party who helps resolve a
conflict is the"
                    Answer1.setTitle("arbitrator", for: .normal)
                    Answer2.setTitle("negotiator", for: .normal)
                    Answer3.setTitle("motivator", for: .normal)
                    Answer4.setTitle("mediator", for: .normal)
                    Answer4Correct = true
                case 4:
                    QuestionLabel.text = "Which of the following is NOT a benefit of
self-managed teams?"
                    Answer1.setTitle("a decreased likelihood of social loafing", for:
.normal)
                    Answer2.setTitle("a sense of belonging and ownership in one's
work", for: .normal)
                    Answer3.setTitle("greater employee participation", for: .normal)
                    Answer4.setTitle("reduced operational costs", for: .normal)
                    Answer1Correct = true
                case 5:
                    QuestionLabel.text = "Power based on the user's personal
relationship with others is"
                    Answer1.setTitle("referent", for: .normal)
                    Answer2.setTitle("reinforcement", for: .normal)
                    Answer3.setTitle("relationship", for: .normal)
                    Answer4.setTitle("reward", for: .normal)
                    Answer1Correct = true
                case 6:
                    QuestionLabel.text = "According to path-goal theory, which of
these behaviors encourage employees to reach their peak performance? "
                    Answer1.setTitle("achievement oriented", for: .normal)
                    Answer2.setTitle("competitive", for: .normal)
                    Answer3.setTitle("supportive", for: .normal)
                    Answer4.setTitle("participative", for: .normal)
                    Answer1Correct = true
                case 7:
                    QuestionLabel.text = "The ___ leadership model is used to
determine if a person's leadership style is task- or relationship-oriented, and if the
situation matches the leader's style to maximize performance."
                    Answer1.setTitle("behavioral", for: .normal)
                    Answer2.setTitle("contingency", for: .normal)
                    Answer3.setTitle("path-goal", for: .normal)
                    Answer4.setTitle("normative", for: .normal)
                    Answer2Correct = true
                case 8:
                    QuestionLabel.text = "Shirking of individual responsibility is
also known as"
                    Answer1.setTitle("groupthink", for: .normal)
                    Answer2.setTitle("social loafing", for: .normal)
                    Answer3.setTitle("synergy", for: .normal)
                    Answer4.setTitle("task facilitation", for: .normal)
                    Answer2Correct = true
                case 9:
                    QuestionLabel.text = "The form of coaching in which a more
experienced manager helps a less experienced protege is"
                    Answer1.setTitle("networking", for: .normal)
                    Answer2.setTitle("mentoring", for: .normal)
                    Answer3.setTitle("relegating", for: .normal)
                    Answer4.setTitle("delegating", for: .normal)
                    Answer2Correct = true
                case 10:
                    QuestionLabel.text = "Charisma is"
                    Answer1.setTitle("found in all transformational leaders", for:
.normal)
                    Answer2.setTitle("relational in nature", for: .normal)
                    Answer3.setTitle("totally determined by the situation", for:
.normal)
                    Answer4.setTitle("a psychological phenomenon", for: .normal)
                    Answer2Correct = true
                case 11:
                    QuestionLabel.text = "____ means that the leader works with
followers to help them identify andlearn the behaviors that will lead to successful
task accomplishment and organizational rewards."
                    Answer1.setTitle("Diagnostic leadership", for: .normal)
                    Answer2.setTitle("Directive leadership", for: .normal)
                    Answer3.setTitle("Prescriptive leadership", for: .normal)
                    Answer4.setTitle("Path clarification", for: .normal)
                    Answer1Correct = true
                case 12:
                    QuestionLabel.text = "Gem Agency uses social events to reinforce
examples of its values, sees employees as their top asset, and has a strong link
between performance and rewards What type of culture do they have?"
                    Answer1.setTitle("hierarchical", for: .normal)
                    Answer2.setTitle("weak", for: .normal)
                    Answer3.setTitle("strong", for: .normal)
                    Answer4.setTitle("national", for: .normal)
                    Answer3Correct = true
                default:
                    break
            }
            // Removes the possibility of the question that was just shown to be shown
again
            randomQuestionArray.remove(at: randomIndex)
        }

        // If the user is on the last question then show them that they have reached
the last question
        if (randomQuestionArray.count < 1) {
            let alert = UIAlertController(title: "Wow!", message: "You have reached
the last question for Organizational Leadership! Nice Job! Complete this question and
then click on 'Your Score' for a rating!", preferredStyle: .alert)
            alert.addAction(UIAlertAction(title: "Continue", style: .default, handler:
{ action in
                switch action.style{
                case .default:
                    print("default")
                case .cancel:
                    print("cancel")
                case .destructive:
                    print("destructive")
                }}))
            self.present(alert, animated: true, completion: nil)
            NextQuestion.isEnabled = false
        }

        if (randomQuestionArray.count == 0) {
            let alert = UIAlertController(title: "Wow!", message: "You got
\(ScoreNumber) out of 13 questions correct nice job! To see a detailed breakdown of
your score, click-Your Score-next to your score number.", preferredStyle: .alert)
            alert.addAction(UIAlertAction(title: "Continue", style: .default, handler:
{ action in
                switch action.style{
```

```
                case .default:
                    print("default")
                case .cancel:
                    print("cancel")
                case .destructive:
                    print("destructive")
                }}))
            self.present(alert, animated: true, completion: nil)
        }
    }

    // These 4 functions tell the user if they got the correct answer or if they got
the incorrect answer
    @IBAction func Answer1(_ sender: Any) {
        if Bool(Answer1Correct) == true {
            rightAnswer()
            Answer1.layer.backgroundColor = UIColor.green.cgColor
            Answer1.isEnabled = false
            Answer2.isEnabled = false
            Answer3.isEnabled = false
            Answer4.isEnabled = false
        } else {
            wrongAnswer()
            Answer1.layer.backgroundColor = UIColor.red.cgColor
            Answer1.isEnabled = false
        }
    }

    @IBAction func Answer2(_ sender: Any) {
        if Bool(Answer2Correct) == true {
            rightAnswer()
            Answer2.layer.backgroundColor = UIColor.green.cgColor
            Answer2.isEnabled = false
            Answer1.isEnabled = false
            Answer3.isEnabled = false
            Answer4.isEnabled = false
        } else {
            wrongAnswer()
            Answer2.layer.backgroundColor = UIColor.red.cgColor
            Answer2.isEnabled = false
        }
    }

    @IBAction func Answer3(_ sender: Any) {
        if Bool(Answer3Correct) == true {
            rightAnswer()
            Answer3.layer.backgroundColor = UIColor.green.cgColor
            Answer3.isEnabled = false
            Answer1.isEnabled = false
            Answer2.isEnabled = false
            Answer4.isEnabled = false
        } else {
            wrongAnswer()
            Answer3.layer.backgroundColor = UIColor.red.cgColor
            Answer3.isEnabled = false
        }
    }

    @IBAction func Answer4(_ sender: Any) {
        if Bool(Answer4Correct) == true {
            rightAnswer()
```

```swift
            Answer4.layer.backgroundColor = UIColor.green.cgColor
            Answer4.isEnabled = false
            Answer1.isEnabled = false
            Answer2.isEnabled = false
            Answer3.isEnabled = false
        } else {
            wrongAnswer()
            Answer4.layer.backgroundColor = UIColor.red.cgColor
            Answer4.isEnabled = false
        }
    }

    // Resets the colors and answers and generates another question
    @IBAction func NextQuestion(_ sender: Any) {
        Answer1.layer.backgroundColor = UIColor.white.cgColor
        Answer2.layer.backgroundColor = UIColor.white.cgColor
        Answer3.layer.backgroundColor = UIColor.white.cgColor
        Answer4.layer.backgroundColor = UIColor.white.cgColor
        NextQuestion.isHidden = true
        Answer1Correct = false
        Answer2Correct = false
        Answer3Correct = false
        Answer4Correct = false
        RandomQuestions()
    }
}
```