```swift
//
//  Item.swift
//  Homepowner2
//
//  Created by Chelsea Irizarry on 10/20/16.
//  Copyright © 2016 Chelsea Irizarry. All rights reserved.
//

import UIKit

class Item: NSObject {
    var name: String
    var valueInDollars: Int
    var serialNumber: String?
    let dateCreated: NSDate

    //Designated initializer
    init(name: String, serialNumber: String?, valueInDollars: Int) {
        self.name = name
        self.valueInDollars = valueInDollars
        self.serialNumber = serialNumber
        self.dateCreated = NSDate()

        super.init()
    }

    //Convenience initializer
    convenience init(random: Bool = false) {
        if random {
            let adjectives = ["Fluffy", "Rusty", "Shiny"]
            let nouns = ["Bear", "Spork", "Mac"]

            var idx = arc4random_uniform(UInt32(adjectives.count))
            let randomAdjective = adjectives[Int(idx)]

            idx = arc4random_uniform(UInt32(nouns.count))
            let randomNoun = nouns[Int(idx)]

            let randomName = "\(randomAdjective) \(randomNoun)"
            let randomValue = Int(arc4random_uniform(100))
            let randomSerialNumber =
NSUUID().uuidString.components(separatedBy: "_").first!

            self.init(name: randomName, serialNumber:
randomSerialNumber, valueInDollars: randomValue)
        }

        else {
            self.init(name: "", serialNumber: nil, valueInDollars: 0)
        }
```

```
    }

}



//
//  ItemStore.swift
//  Homepowner2
//
//  Created by Chelsea Irizarry on 10/20/16.
//  Copyright © 2016 Chelsea Irizarry. All rights reserved.
//

import UIKit

class ItemStore {
    var allItems = [Item]()

    func createItem() -> Item {
        let newItem = Item(random: true)

        allItems.append(newItem)

        return newItem
    }

    //Deleting Rows
    func removeItem(item: Item){
        if let index = allItems.index(of: item) {
            allItems.remove(at: index)
        }
    }

    //Moving Rows
    func moveItemAtIndex(fromIndex: Int, toIndex: Int){
        if fromIndex == toIndex {
            return
        }

        //Get reference to object being moved so you can reinsert it
        let movedItem = allItems[fromIndex]

        //Remove item from array
        allItems.remove(at: fromIndex)

        //Insert item in array at new location
        allItems.insert(movedItem, at: toIndex)
    }
}
```

```
//
//  ItemsViewController.swift
//  Homepowner2
//
//  Created by Chelsea Irizarry on 10/20/16.
//  Copyright © 2016 Chelsea Irizarry. All rights reserved.
//

import UIKit

//Define a UITableViewController subclass named ItemsViewController
class ItemsViewController: UITableViewController {

    var itemStore: ItemStore!

    //Adding Rows
    @IBAction func addNewItem(sender: AnyObject){

        //Create a new item and add it to the store
        let newItem = itemStore.createItem()

        //Figure out where that item is in the array
        if let index = itemStore.allItems.index(of: newItem) {
            let indexPath = NSIndexPath(row: index, section: 0)

            //Insert this new row into the table
            tableView.insertRows(at: [indexPath as IndexPath],
with: .automatic)

        }
    }

    //Deleting Rows
    override func tableView(_ tableView: UITableView, commit
editingStyle: UITableViewCellEditingStyle, forRowAt indexPath:
IndexPath) {

        //If the table view is asking to commit a delete command...
        if editingStyle == .delete {
            let item = itemStore.allItems[indexPath.row]

            let title = "Delete \(item.name)?"
            let message = "Are you sure you want to delete this item?"

            let ac = UIAlertController(title: title, message: message,
preferredStyle: .actionSheet)

            let cancelAction = UIAlertAction(title: "Cancel",
```

```
style: .cancel, handler: nil)
            ac.addAction(cancelAction)

            let deleteAction = UIAlertAction(title: "Delete",
style: .destructive, handler: { (action)-> Void in
                //Remove the item from the store
                self.itemStore.removeItem(item: item)

                //Also remove that row from the table view with an
animation
                self.tableView.deleteRows(at: [indexPath],
with: .automatic)
            })
            ac.addAction(deleteAction)

            //Present the alert controller
            present(ac, animated: true, completion: nil)

        }
    }

    //Moving Rows
    override func tableView(_ tableView: UITableView, moveRowAt
sourceIndexPath: IndexPath, to destinationIndexPath: IndexPath) {
        //Update the model
        itemStore.moveItemAtIndex(fromIndex: sourceIndexPath.row,
toIndex: destinationIndexPath.row)
    }

    //Editing Mode
    @IBAction func toggleEditingMode(sender: AnyObject){
        //If your are currently in editing mode...
        if isEditing {
            //Change text of button to inform user of state
            sender.setTitle("Edit", for: .normal)

            //Turn off editing mode
            setEditing(false, animated: true)
        }

        else{
            //Change text of button to inform user of state
            sender.setTitle("Done", for: .normal)

            //Enter editing mode
            setEditing(true, animated: true)
        }
    }

    override func tableView(_ tableView: UITableView,
```

```
numberOfRowsInSection section: Int) -> Int {
        return itemStore.allItems.count
    }

    override func tableView(_ tableView: UITableView, cellForRowAt
indexPath: IndexPath) -> UITableViewCell {
        //Get a new or recycled cell
        let cell = tableView.dequeueReusableCell(withIdentifier:
"UITableViewCell", for: indexPath)

        //Set the tex on the cell with the description of the item
        //that is at the nth index of items, where n = row this cell
        //will appear in on the tableview
        let item = itemStore.allItems[indexPath.row]
        cell.textLabel?.text = item.name
        cell.detailTextLabel?.text = "$\(item.valueInDollars)"

        return cell
    }

    override func viewDidLoad() {
        super.viewDidLoad()

        //Get the height of the status bar
        let statusBarHeight =
UIApplication.shared.statusBarFrame.height
        let insets = UIEdgeInsets(top: statusBarHeight, left: 0,
bottom: 0, right: 0)
        tableView.contentInset = insets
        tableView.scrollIndicatorInsets = insets

    }

}
```