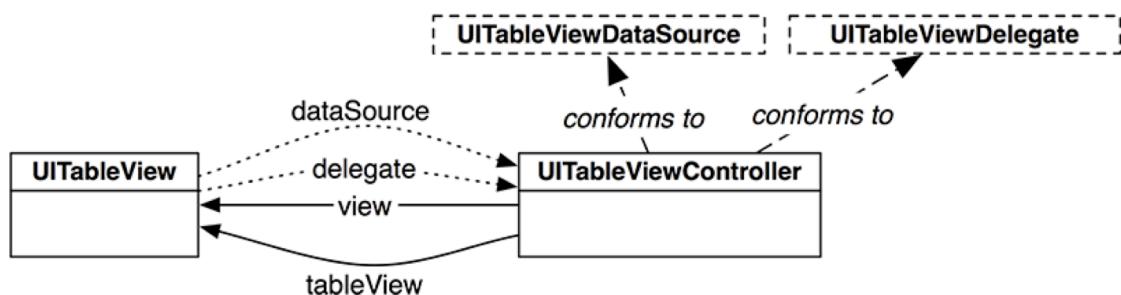


Table View Exercise

Application that we are going to make in this exercise.



UITableView Class Diagram



Task 1. สร้าง Project StudentTableView based on Single View Application

Task 2. สร้าง Student class ที่เป็น Subclass ของ NSObject เพื่อกีบ ชื่อ, นามสกุล, ชั้นปี
สร้าง Constructor (init function) ที่เหมาะสม และ convenience init เพื่อใช้สร้าง Object
ด้วย data สุ่มๆ เพื่อทดสอบ (เฉลย ท้ายเอกสาร)

https://developer.apple.com/library/content/documentation/Swift/Conceptual/Swift_Programming_Language/Initialization.html

Question: convenience init ต่างกับ init ธรรมดายังไง

ตัวอย่าง:

```

class MyPet: NSObject {
    var name:String
    init(name:String) {
        self.name=name
    }
    convenience init(random: Bool = false) {
        if random {
            let n:[String] = ["Pii", "Mike", "Momo"]

            var idx = arc4random_uniform( UInt32(n.count))
            let name = n[Int(idx)]

            self.init(name: name)
        }else{
            self.init(name: "-")
        }
    }
}

```

Task 3. สร้าง StudentData class ที่เป็น Subclass ของ NSObject เพื่อเก็บ Array of Student. แล้วสร้าง method

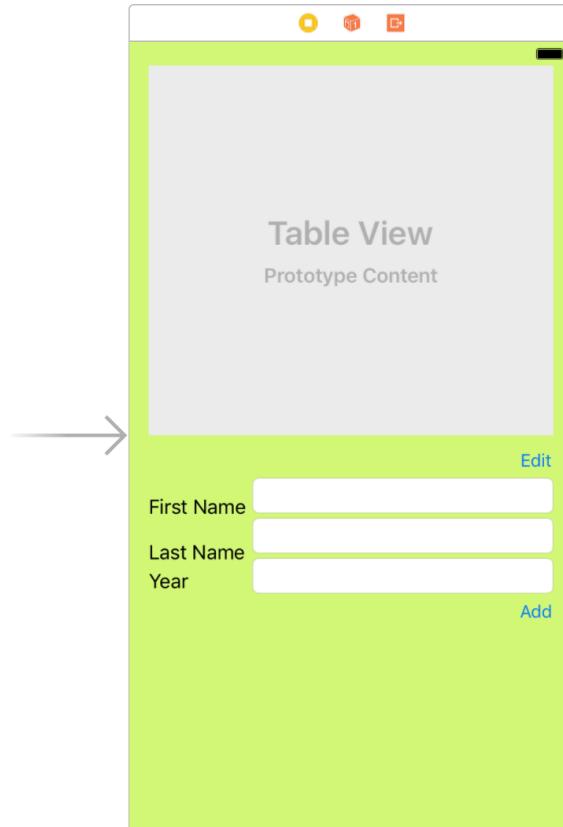
- add(student:Student)
- add(firstName:String, lastName:String, year:Int)
- addAt(index:Int,student:Student)
- getAt(index:Int) -> Student
- deleteAt(index:Int) -> Student
- total()->Int

(เฉลย ท้ายเอกสาร)

Question: Why this is better than just use Array? (ทำไมสร้าง Object ห่อ Array ดีกว่า Array เป็นๆๆ)

Task 4. สร้าง StudentData class ที่เป็น Subclass ของ NSObject เพื่อเก็บ Array of Student. แล้วสร้าง method

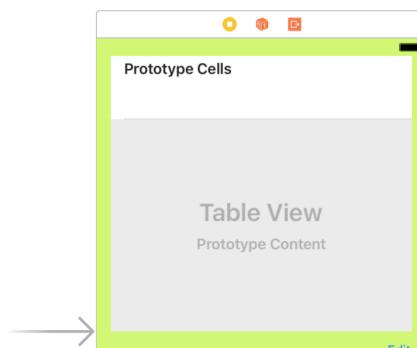
Task 5. Create a page as shown. (If you don't want trouble later, make sure that the green part is a view on top of main view with all constraints. You will need it for moving view from under a keyboard later.)



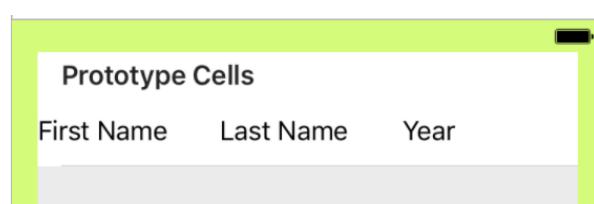
Task 6. Create all button event handlers and outlets for all text fields and Table View.

Task 7. Put one Table View Cell in Table View.

Table View Cell = ຕົນແບບຂອງແຄວໃນຕາರັງ ຈ້າຍກາ ມີລາຍແບບກໍ່ໃຫ້ເພີ່ມຕາມຈຳນວນຮູບແບບ ໄນໃຊ້ສື່
ຕາມຈຳນວນແຄວທີ່ຢາກໄດ້

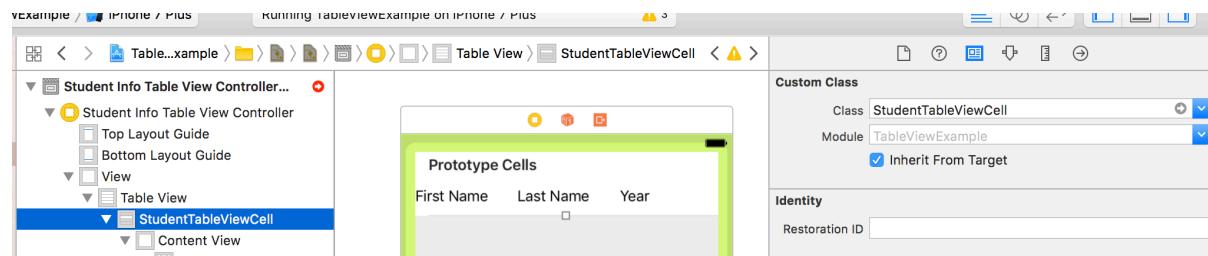


Task 8. Put 3 Labels in Table View Cell with appropriate constrains.

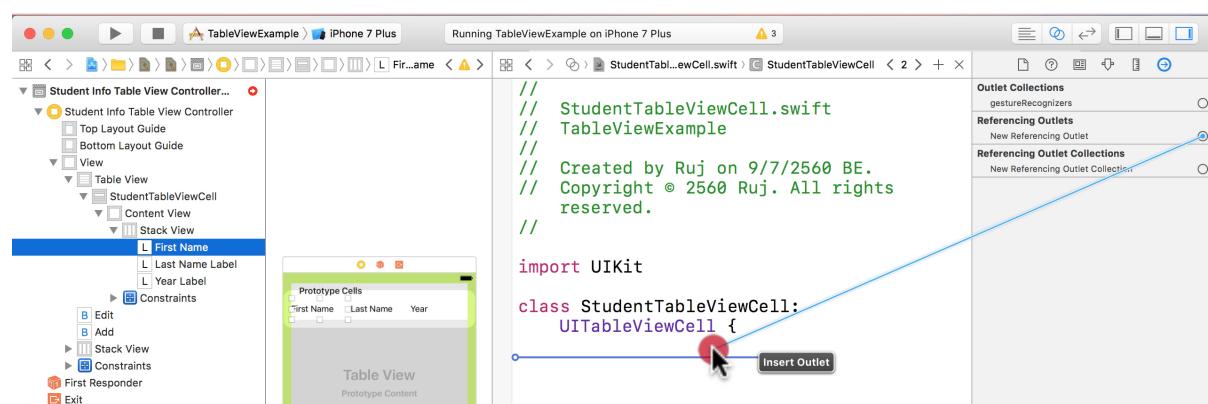
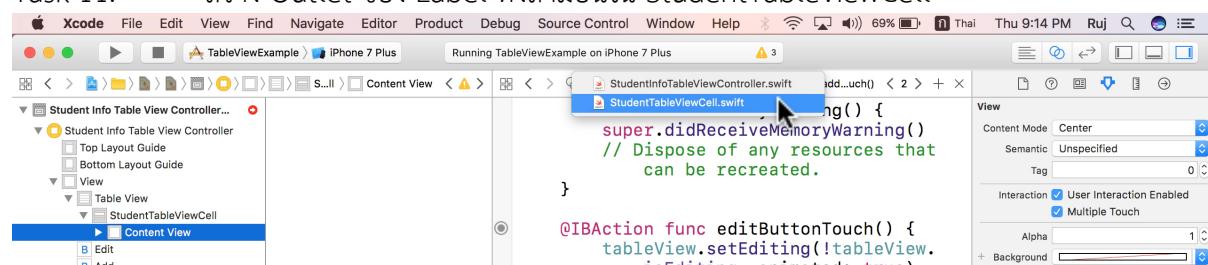


Task 9. Create StudentTableViewCell ที่เป็น subclass ของ UITableViewCell

Task 10. เปลี่ยน Class ของ Table View Cell เป็น StudentTableViewCell



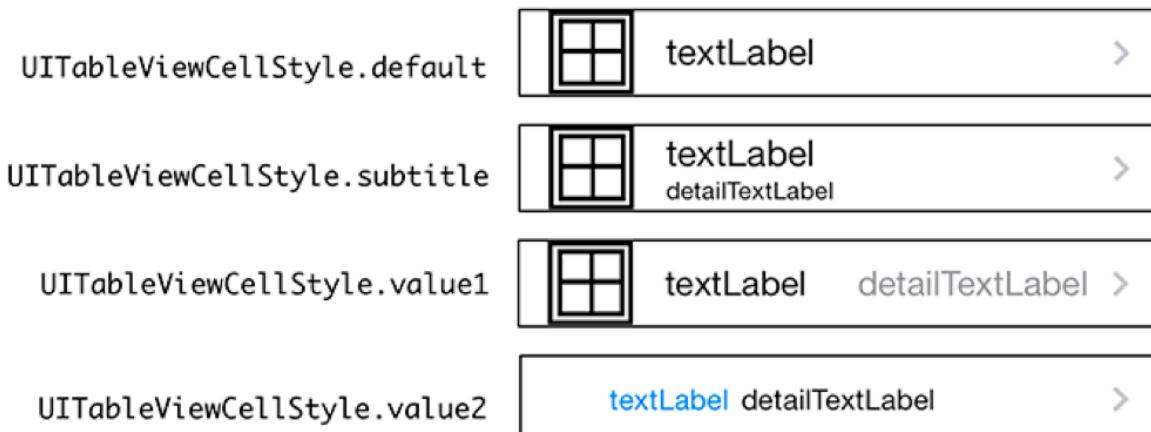
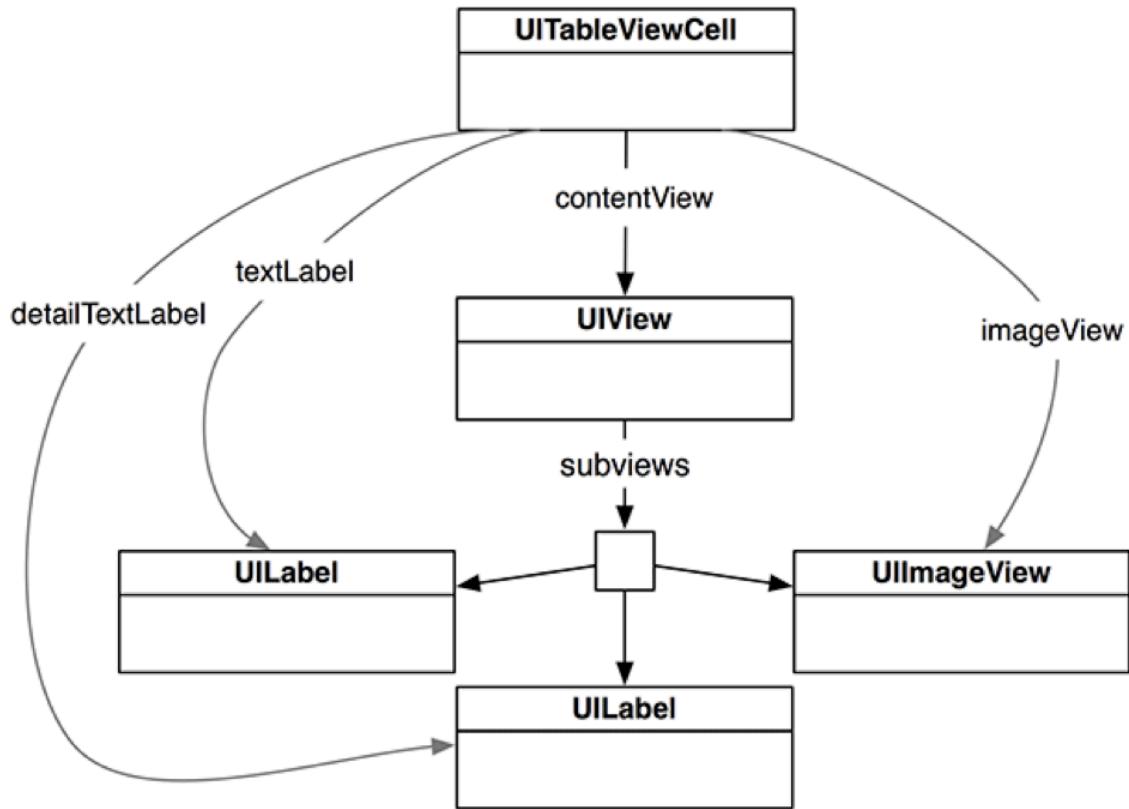
Task 11. สร้าง Outlet ของ Label ทั้งสามอันใน StudentTableViewCell



Task 12. สร้าง Outlet ของ Label ทั้งสามอันใน StudentTableViewCell

Task 13. สร้าง methods ใน StudentTableViewCell เพื่อใช้ตั้งค่าของ Label โดยง่าย
(เฉลย ท้ายเอกสาร)

โดยปกติถ้าไม่อยากออกแบบสามารถใช้ Table View Cell แบบสำเร็จรูปได้ มีให้เลือก 4 แบบ: Basic, Left Detail, Right Detail, Subtitle และ มี Fields ดังแสดงในรูป



Task 14. สร้าง Field ใน View Controller เพื่อเก็บ StudentData (ส่วนนี้คือ Model ใน MVC) และเรียก self.ชื่อที่ตั้ง.add(student: Student(random: true)) ใน viewDidLoad() สักหลายๆครั้งตามขอบใจเพื่อใส่ข้อมูลสุ่มลงใน StudentData

Task 15. Extends your View Controller with UITableViewDelegate and UITableViewDataSource using extension. ใช้ extension จะทำให้ Code อ่านได้ง่ายกว่า ใช้วิธีอื่น ใส่ extension

(โดย ท้ายเอกสาร)

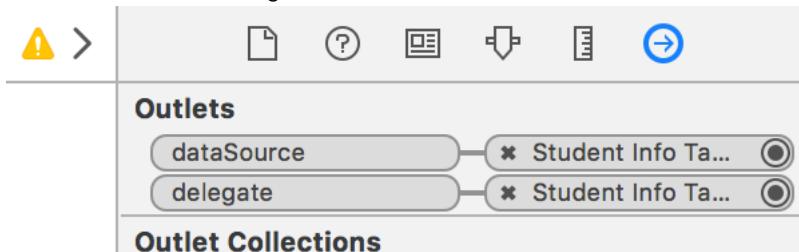
ตัวอย่าง (View Controller ในตัวอย่าง ชื่อ StudentInfoTableViewController)

```
class StudentInfoTableViewController: UIViewController {
    ...
}
```

```
}
```

```
extension StudentInfoTableViewController: UITableViewDelegate{  
}
```

Task 16. Link dataSource and delegate ของ UITableView ไปที่ ViewController ของเรา

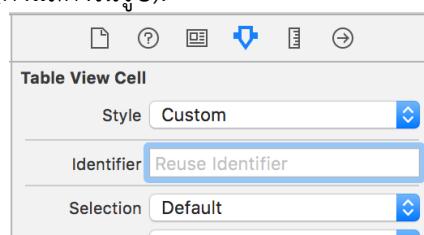


ในตัวอย่าง StudentInfoTableViewController เป็น ViewController class

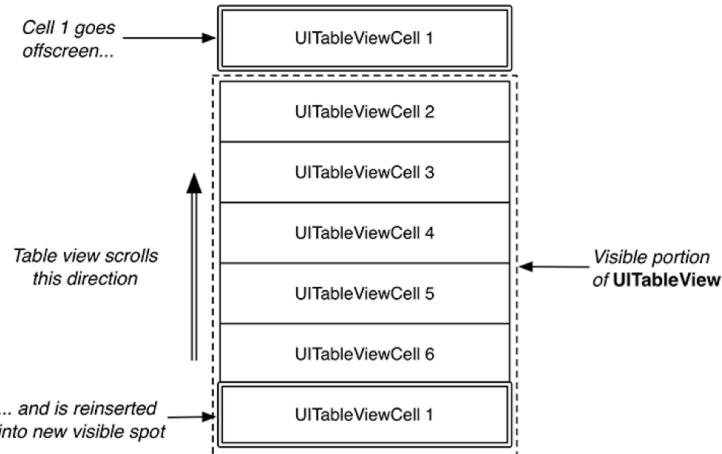
Task 17. UITableViewDataSource ต้อง implements อย่างน้อย 3 methods นี้ (ใช้ Autocomplete อย่าพยายามพิมพ์เอง จะผิดได้ง่าย) งานของเรา มี Section เดียว

```
func numberOfSections(in tableView: UITableView) -> Int  
  
func tableView(_ tableView: UITableView,  
              numberOfRowsInSection section: Int) -> Int  
  
func tableView(_ tableView: UITableView,  
              cellForRowAt indexPath: IndexPath) -> UITableViewCell {  
    let cell = tableView.dequeueReusableCell(  
       (withIdentifier: "StudentTableViewCell", for: indexPath)  
    as! StudentTableViewCell  
  
    // ใส่ code สำหรับดึงค่า StudentTableViewCell ตรงนี้  
  
    return cell  
}
```

Function ล่างสุด (func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath)) มีหน้าที่ recycle และสร้าง Table View Cell ตามต้องการโดยใช้คำสั่ง tableView.dequeueReusableCell เพื่อสร้าง Table View Cell ตามชนิดที่ระบุโดย withIdentifier. String ที่ระบุจะเป็นอะไรก็ได้แต่โดยทั่วไปจะใช้ชื่อ class จะได้เข้าใจง่าย ซึ่งนี่จะต้องตรงกับ reuse identifier ที่ระบุใน Storyboard (ดังแสดงในรูป).



การทำงานของ recycle ของ Table View Cell เป็นดังในรูป Cell ที่หลุดหน้าจอจะถูกเอามา recycle มาแสดงอีกด้านหนึ่งของ Table



เรามาระบุ Template ของ Table View Cell หลายแบบได้แต่ต้องตั้ง reuse identifier ให้ต่างกัน และเรียกมาใช้ ตามค่าที่ใส่ใน reuseIdentifier

Task 18. ลอง Run ดู โดยยังไม่ตั้ง reuse identifier ของ Table View Cell ใน Storyboard

Question: ทำไง Error

Task 19. ตั้ง reuse identifier ของ Table View Cell ใน Storyboard ให้ตรงกับ reuseIdentifier ใน code แล้ว run ใหม่

Task 20. เติมเต็ม event listener ของ ปุ่ม Add ให้เพิ่ม Row ใน Table

(เฉลย ท้ายเอกสาร)

การเพิ่ม Row ทำได้โดยใช้ method ต่อไปนี้ของ UITableView

```
insertRows(at: [IndexPath], with: UITableViewRowAnimation)
```

Note: ต้องใส่ข้อมูลใน Model (ในที่นี่คือ StudentData) ก่อนเรียก method insertRows

การทดลอง: ให้ลอง เรียก method insertRows และค่อยใส่ข้อมูลใน StudentData เปรียบเทียบ กับ ใส่ข้อมูลใน StudentData และค่อยเรียก method insertRows

Task 21. เติมเต็ม event listener ของ ปุ่ม Edit ให้เปลี่ยนไปกลับ mode ของ TableView ระหว่าง edit mode กับ ธรรมดា (ให้เปลี่ยน Text ของ Button ระหว่าง Edit <-> Done ได้ด้วยตาม Mode) การเปลี่ยน Mode ใช้ method setEditing(_ editing: Bool, animated: Bool) ของ TableView (เฉลย ท้ายเอกสาร)
Mode Editing จะทำให้สามารถ ลบและย้าย Row ได้

Note: ปุ่มใช้ method setTitle เพื่อเปลี่ยน คำ

<https://developer.apple.com/documentation/uikit/uicontrolstate>

Task 22. ลอง Run และสังเกตว่า ลบได้หรือไม่

Task 23. การลบจำเป็นต้อง Implement method ของ UITableViewDataSource

```
tableView(_ tableView: UITableView,  
         commit editingStyle: UITableViewCellEditingStyle,  
         forRowAtIndexPath indexPath: IndexPath)
```

โดย argument editingStyle จะเป็นตัวบอกรว่าเป็น event ไหน

<https://developer.apple.com/documentation/uikit/uitableviewcelleditingstyle>

จะ implement method สำหรับ delete

Question: ควรใช้ method ของ UITableView ชื่ออะไรในการลบ และควรลบจาก StudentData
ก่อนหรือจาก TableView ก่อน

(เฉลย ท้ายเอกสาร)

Task 24. เราสามารถถ่าย Row ได้ใน edit mode (Feature จะไม่แสดงใน UI จนกว่าจะ implement
method) โดย implement

```
tableView(_ tableView: UITableView,  
         moveRowAtIndexPath sourceIndexPath: IndexPath,  
         to destinationViewControllerIndexPath: IndexPath)
```

จะ implement method สำหรับถ่าย Row

(เฉลย ท้ายเอกสาร)

Homework

Task 25. Implement: ให้ลบ Text Field หลังกด add.

Task 26. Make keyboard not blocking text fields and complete keyboard dismissing

Task 27. เพิ่ม Section ที่สอง ให้ Table เพื่อบันทึก ราคาอาหารในโรงอาหาร โดยที่ Row ใน section ที่
สอง เป็นคุณลักษณะแบบ กับ Section เก่า ให้ใส่ UI เสริมตามสมควร

Answer

Task 2:

```
class Student: NSObject {
    var firstName:String
    var lastName:String
    var year:Int
    init(firstName:String, lastName:String, year:Int) {
        self.firstName = firstName
        self.lastName = lastName
        self.year = year
    }

    convenience init( random: Bool = false) {
        if random {
            let fn = ["ปิติ", "มานี", "ชุใจ"]
            let ln = ["รักເພົາໄທຍ່", "ພິທິກອບນີ້ນ", "ເລີສຄ້າ"]

            var idx = arc4random_uniform( UInt32( fn.count))
            let firstName = fn[ Int(idx)]

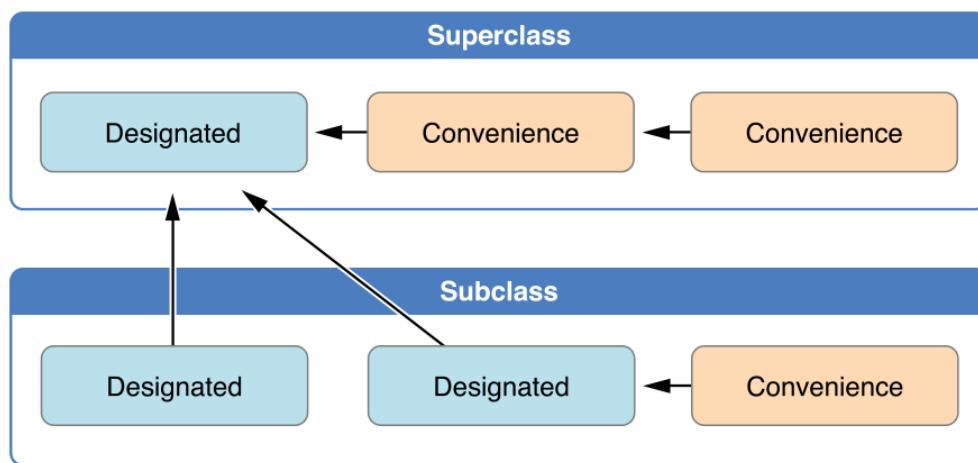
            idx = arc4random_uniform( UInt32( ln.count))
            let lastName = ln[ Int(idx)]

            let year = Int(arc4random_uniform( UInt32( 4)))+1

            self.init(firstName: firstName, lastName: lastName, year: year)
        }else {
            self.init(firstName: "-", lastName: "-", year: 0)
        }
    }
}
```

Swift subclasses do not inherit their superclass initializers by default.

Convenience initializers must call another initializer on the same type, whereas designated initializers must call a designated initializer on its superclass.



Task 3:

```
class StudentData: NSObject {

    private var data:[Student] = []

    func add(student:Student) {
        data.append(student)
    }

    func add(firstName:String, lastName:String, year:Int) {
        data.append(
            Student(firstName: firstName,
                    lastName: lastName,
                    year: year))
    }

    func addAt(index:Int,student:Student){
        data.insert(student, at: index)
    }

    func getAt(index:Int) -> Student{
        return data[index]
    }

    func deleteAt(index:Int) -> Student {
        return data.remove(at: index)
    }

    func total()->Int{
        return data.count
    }
}
```

Task 13

```
import UIKit

class StudentTableViewCell: UITableViewCell {

    @IBOutlet weak var firstNameLabel: UILabel!
    @IBOutlet weak var lastNameLabel: UILabel!
    @IBOutlet weak var yearLabel: UILabel!

    override func awakeFromNib() {
        super.awakeFromNib()
        // Initialization code
    }

    override func setSelected(_ selected: Bool, animated: Bool) {
        super.setSelected(selected, animated: animated)

        // Configure the view for the selected state
    }

    func set(student:Student){
        self.firstNameLabel.text = student.firstName
        self.lastNameLabel.text = student.lastName
        self.year = student.year
    }

    var firstName:String{
        set(name){
            self.firstNameLabel.text = name
        }
        get{
            return self.firstNameLabel.text!
        }
    }

    var lastName:String{
        set(name){
            self.lastNameLabel.text = name
        }
        get{
            return self.lastNameLabel.text!
        }
    }

    var year:Int{
        set(year){
            self.yearLabel.text = String(year)
        }
        get{
            return Int(self.yearLabel.text!)!
        }
    }
}
```

Task 16

Note: studentData เป็น field ที่เก็บ StudentData

```
func numberOfSections(in tableView: UITableView) -> Int {
    // #warning Incomplete implementation, return the number
    // of sections
    return 1
}

func tableView(_ tableView: UITableView,
    numberOfRowsInSection section: Int) -> Int {
    // #warning Incomplete implementation, return the number
    // of rows
    return self.studentData.total()
}

func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {
    let cell = tableView.dequeueReusableCell(withIdentifier: "StudentTableViewCellCell", for: indexPath) as!
    StudentTableViewCell
    cell.set(student: self.studentData.getAt(index: indexPath.row))
    return cell
}
```

Task 20

```
@IBAction func addButtonTouch() {
    let indexPath = IndexPath(row: self.studentData.total(), section: 0)
    self.studentData.add(
        firstName: self.firstNameTextField.text!,
        lastName: self.lastNameTextField.text!,
        year: Int(self.yearTextField.text!)!)
    self.tableView.insertRows(at: [indexPath], with: .automatic)
}
```

Task 21

```
@IBAction func editButtonTouch(_ sender: UIButton) {
    if(self.tableView.isEditing){
        sender.setTitle("Edit", for: UIControlState.normal)
    }else{
        sender.setTitle("Done", for: UIControlState.normal)
    }
    tableView.setEditing(!tableView.isEditing, animated: true)
}
```

Task 23

```
func tableView(_ tableView: UITableView, commit editingStyle: UITableViewCellEditingStyle, forRowAt indexPath: IndexPath) {
    switch editingStyle {
    case .delete:
        self.studentData.deleteAt(index: indexPath.row)
        self.tableView.deleteRows(at: [indexPath], with: .automatic)
        break
    default:
        break
    }
}
```

Task 24

```
func tableView(_ tableView: UITableView,
    moveRowAt sourceIndexPath: IndexPath,
    to destinationIndexPath: IndexPath) {
    let data = self.studentData.deleteAt(index: sourceIndexPath.row)
    self.studentData.addAt(index: destinationIndexPath.row, student: data)
}
```