Основы Swift

Управление памятью

Ссылки на объект

```
class TestClass {
    init() {
        print("TestClass создан")
    }

    deinit {
        print("TestClass удален")
    }
}
```

```
var reference1: TestClass? =
TestClass()                          // 1

var reference2 = reference1 // 2

print("Две ссылки")

reference1 = nil                     // 1
print("Одна ссылка")

reference2 = nil                     // 0
print("Ссылок нет")
```

Reference cycle

```
class User {
    let name: String
    var devices: [Device]

    init(name: String) {
        self.name = name
        devices = [ ]

        print("Создан
        пользователь
        \(name)")
    }
}
```

```
class Device {
    let model: String
    let owner: User

    init(model: String, owner:
    User) {
        self.model = model
        self.owner = owner

        print("Создано
        устройство \(model).
        Его владелец
        \(owner.name)")
    }
}
```

Reference cycle

```
var bob: User? = User(name: "Bob")
var iPhone7: Device? = Device(model:
                              "iPhone 7", owner: bob!)
bob!.devices.append(iPhone7!)

bob = nil
iPhone7 = nil
```

Weak reference

```
class User {
    let name: String
    var devices: [Device]

    init(name: String) {
        self.name = name
        devices = [ ]
        print("Создан
        пользователь \(name)")
    }

    deinit {
        print("Удален
        пользователь \(name)")
    }
}
```

```
class Device {
    let model: String
    weak var owner: User?

    init(model: String, owner:
    User) {
        self.model = model
        self.owner = owner
        print("Создано
        устройство \(model).
        Его владелец
        \(owner.name)")
    }
    deinit {
        print("Удалено
        устройство \(model)")
    }
}
```

Weak reference

```
var bob: User? = User(name: "Bob")
var iPhone7: Device? = Device(model:
                                "iPhone 7", owner: bob!)
bob!.devices.append(iPhone7!)

bob = nil
iPhone7 = nil
```

Reference cycle

```
class Counter {
    var value: Int = 0
    lazy var valueChanger = {
        self.value += 1
    }

    deinit {
        print("Counter удален")
    }
}

var counter: Counter? = Counter()
counter = nil
```

Weak reference

```
class Counter {
    var value: Int = 0
    lazy var valueChanger = {
        [unowned self] in
        self.value += 1
    }
    deinit {
        print("Counter удален")
    }
}

var counter: Counter? =  Counter()
counter = nil
```