Class

```
class Vehicle {
    var speed: Float = Float(0)
    static let className: String = "Vehicle"

    var description: String {
       return "traveling at \(speed) miles
per hour"
    }

    func makeNoise() {
       // do nothing - an arbitrary vehicle
doesn't make a noise
    }

    final class func blah() {
    }
}
```

Simple Car

```
class Car: Vehicle {
    var color: UIColor

    override init() {
       color = UIColor.blackColor()
    }

    deinit {
       // Not necessary to implement
```

```
        // use to invalidate timers, async
operations
        // and other things that can go
horribly wrong
    }

    func accelerateToCitySpeedLimit() {
        speed = Float(50)
    }

    func accelerateBySmallAmount() {
        speed += 5
    }

    func emergencyBreak() {
        speed = Float(0)
    }

    func repaint(toColor color: UIColor) {
        self.color = color
    }

    override func makeNoise() {
        print("Vroom vroom")
    }

//  override class func blah() {
//      print("Blah")
//  }
}

print(Car.className)
Car.blah()
```

```swift
let car1: Car = Car()
car1.makeNoise()
print(car1.speed)

car1.accelerateToCitySpeedLimit()
print(car1.speed)
print(car1.description)

car1.emergencyBreak()
print(car1.speed)

for _ in 0..<5 {
    car1.accelerateBySmallAmount()
}
print(car1.speed)
car1.emergencyBreak()

print(car1.color.colorDescription)
car1.repaint(toColor: UIColor.yellowColor())
print(car1.color.colorDescription)

let car2 = car1
print(car1.speed)
print(car2.speed)

car2.accelerateToCitySpeedLimit()
print(car1.speed)
print(car2.speed)
```

Struct

```
struct Point {
    var x, y: Float
}

struct Size {
    var width, height: Float
}

struct Rectangle {
    var origin: Point
    var size: Size

    var description: String {
       return "{\(origin.x), \(origin.y), \
(size.width) \(size.height)}"
    }
}

var rectangle1 = Rectangle(origin: Point(x:
0, y: 0),
                           size: Size(width:
100, height: 100))
var rectangle2 = rectangle1

print(rectangle1.description)
print(rectangle2.description)

rectangle2.origin.y = 88

print(rectangle1.description)
print(rectangle2.description)
```

Protocol

```
protocol Person {
    var name: String { get set }
    var idNumber: Int { get set }

    func sayHello()
    func goToSleep()
}

class Janusz: Person {
    var name: String = ""
    var idNumber: Int = 0

    func sayHello() {

    }

    func goToSleep() {

    }
}

extension Janusz {
    func hasMustache() -> Bool {
       return true
    }
}

let januszPerson = Janusz()
januszPerson.hasMustache()
```

Generics

```
func swapTwoInts(inout a: Int, inout b: Int)
{
    let temporaryA = a
    a = b
    b = temporaryA
}

func swapTwoDoubles(inout a: Double, inout
b: Double) {
    let temporaryA = a
    a = b
    b = temporaryA
}

func swapTwoStrings(inout a: String, inout
b: String) {
    let temporaryA = a
    a = b
    b = temporaryA
}

var two = 2
var four = 4
swapTwoInts(&two, b: &four)
print(two, four)

func swapTwoValues<T>(inout a:  T, inout b:
T) {
    let temporaryA = a
    a = b
    b = temporaryA
}
```

```
swapTwoValues(&two, b: &four)
print(two, four)

var letter1 = "a"
var letter2 = "b"
swapTwoValues(&letter1, b: &letter2)
print(letter1, letter2)
```