Основы Swift

Проверка типов

Классы

```
class Device {
    var name: String

    init(name: String) {
        self.name = name
    }
}

class Phone: Device {
    var capacity: Int

    init(capacity: Int, name: String) {
        self.capacity = capacity

        super.init(name: name)
    }
}
```

```
class Headphones: Device {
    var resistance: Int

    init(resistance: Int, name: String) {
        self.resistance = resistance

        super.init(name: name)
    }
}

func printDescription(device: Device) {
    print("Device: \(device.name)")
}
```

Неявное приведение

```
let phone = Phone(capacity: 256, name: "iPhone X")
let headphones = Headphones(resistance: 32,
                            name: "Beats Studio3")

printDescription(device: phone)
printDescription(device: headphones)
```

Ошибка

```
func doSomethingWithPhone(phone: Phone) {
    print("\(phone.capacity)")
    // ...
}

let device = Device(name: "Some device")
doSomethingWithPhone(phone: device) // Error
```

Массив

```
let arrayOfDevices = [phone, headphones]
                                                    // [Device]

arrayOfDevices[0] is Headphones     // false
arrayOfDevices[1] is Device         // true

let result = arrayOfDevices.first { $0 is Phone }
type(of: result)                                    // Device?
```

as? as!

```
let optionalHeadprones = result as? Headphones
optionalHeadprones                  // nil
type(of: optionalHeadprones)        // Headphones?

let nonOptionalPhone = result as! Phone
nonOptionalPhone.capacity           // 256
type(of: nonOptionalPhone)          // Phone
```

Использование if let

```
func printDetailedDescription(device: Device) {
    if let phone = device as? Phone {
        print("Phone: \(device.name) with capacity
        \(phone.capacity)")
    } else if let headphones = device as?
    Headphones {
        print("Headphones: \(device.name) with
        resistance \(headphones.resistance)")
    } else {
        print("Device: \(device.name)")
    }
}
printDetailedDescription(device: phone)
                    // Phone: iPhone X with capacity 256
printDetailedDescription(device: headphones)
  // Headphones: Beats Studio3 with resistance 32
printDetailedDescription(device: device)
                                    // Device: Some device
```

Upcasting

```
func process(_ device: Device) {
    print("process as Device")
}

func process(_ phone: Phone) {
    print("process as Phone")
}

process(phone) // process as Phone
process(phone as Device) // process as Device
```

Bridging

```
let swiftString = "This is string"
let objcString: NSString = swiftString
                                    as NSString
let sameSwiftString: String = objcString as String
```

Использование протокола

```
protocol Printable {
    func detailedDescription() -> String
}

class Device: Printable {
    var name: String

    init(name: String) {
        self.name = name
    }

    func detailedDescription() -> String {
        return "Device: \(device.name)"
    }
}
```

Использование протокола

```
class Phone: Device {
    var capacity: Int
    init(capacity: Int, name: String) {
        self.capacity = capacity
        super.init(name: name)
    }
    override func
    detailedDescription() -> String {
        return "Phone: \(device.name)
        with capacity
        \(phone.capacity)"
    }
}
```

```
class Headphones: Device {
    var resistance: Int

    init(resistance: Int, name: String) {
        self.resistance = resistance

        super.init(name: name)
    }

    override func
    detailedDescription() -> String {
        return "Headphones:
        \(device.name) with
        resistance
        \(headphones.resistance)"
    }
}
```

Универсальная функция

```
func printDetailedDescription(_ printable: Printable) {
    print(printable.detailedDescription())
}

printDetailedDescription(phone)
        // Phone: Some device with capacity 256

printDetailedDescription(headphones)
// Headphones: Some device with resistance 32

printDetailedDescription(device)
                // Device: Some device
```

Расширение UIView

```
extension UIView: Printable {
    func detailedDescription() -> String {
        return self.description
    }
}

let view = UIView(frame: CGRect(x: 10, y: 20,
                                width: 4, height: 5))

printDetailedDescription(view)
// <UIView: 0x7fac1950b390; frame = (10 20; 4
5); layer = <CALayer: 0x60800002f380>>
```