Основы Swift

Optionals

Определение Optional

```swift
let intFromString = Int("5")
print(intFromString)              // Optional(5)

let failedIntFromString = Int("Five")
print(failedIntFromString)        // nil
```

Объявление Optional

```swift
let long: Optional<Int> = Int("5")
let short: Int? = Int("Five")
```

Устройство Optional

```
enum Optional<T> {
    case some(T)
    case none
}
```

nil значение

```swift
let counter: Int?                    // nil
if counter != nil {
    counter!.increase()
}
```

Optional binding

```swift
if let nonOptionalCounter = counter {
    nonOptionalCounter.increase()
}

if let firstCounter = firstOptionalCounter, let
secondCounter = secondOptionalCounter,
firstCounter < secondCounter {
    print("Counters not nil. first counter <
    secondCounter")
}
```

Использование guard let

```
guard let nonOptionalCounter = counter {
    return
}

nonOptionalCounter.increase()
```

nil-coalescing оператор

```
nonOptionalCounter = counter != nil ?
                        counter! : anotherCounter

nonOptionalCounter = counter ??
                        anotherCounter


let optional5: Int? = 5
let optionalNil: Int? = nil


var a = optional5 ?? 10 // 5
var b = optionalNil ?? 20 // 20
```

Optional Chaining

```swift
class SomeClass {
    struct InnerStruct {
        var variable: Int
    }

    var innerStruct: InnerStruct?
}
let myClass = SomeClass()
var anotherOptional =
                    myClass.innerStruct?.variable

myClass.innerStruct?.variable = 55

func getSomeValue() -> Int { return 5 }
s.innerStruct?.variable = getSomeValue()
```

Вызов методов

```swift
class SomeClass {
    struct InnerStruct {
        func printSomething() {
            print("Что-то!")
        }
    }

    var innerStruct: InnerStruct?
}

let myClass = SomeClass()

if myClass.innerStruct?.printSomething() != nil {
}
```

Subscript

```
class SomeClass {
    var arrayOfInts: [Int]?
}

let myClass = SomeClass()

myClass.arrayOfInts?[1] = 5

if let value = myClass.arrayOfInts?[1] {
    print("Some Value: \(value)")
}
```