Strings and Functions
Collections and Loops

Strings

```
let greeting = "Hello"
var otherGreeting = "Salutations"
```

```
let joke = """
  Q: Why did the chicken cross the road?
  A: To get to the other side!
  """
print(joke)
```

```
Q: Why did the chicken cross the road?
A: To get to the other side!
```

String basics

Escaping

```
let greeting = "It is traditional in programming to print \"Hello, world!\""
```

Escape	Description
`\"`	Double quote
`\\`	Backslash
`\t`	Tab
`\r`	Carriage return (return to beginning of the next line)

String basics

Empty strings

```
var myString = ""

if myString.isEmpty {
  print("The string is empty")
}
```

Concatenation

```
let string1 = "Hello"
let string2 = ", world!"
var myString = string1 + string2 // "Hello, world!"
```

```
myString += " Hello!" // "Hello, world! Hello!"
```

Interpolation

```swift
let name = "Rick"
let age = 30
print("\(name) is \(age) years old")
```

```
Rick is 30 years old
```

String equality and comparison

```
let month = "January"
let otherMonth = "January"
let lowercaseMonth = "january"

if month == otherMonth {
  print("They are the same")
}

if month != lowercaseMonth {
  print("They are not the same.")
}
```

```
They are the same.
They are not the same.
```

String equality and comparison

Ignoring case

```
let name = "Johnny Appleseed"
if name.lowercased() == "joHnnY aPPleseeD".lowercased() {
  print("The two names are equal.")
}
```

```
The two names are equal.
```

String equality and comparison

Prefix and suffix

```
let greeting = "Hello, world!"

print(greeting.hasPrefix("Hello"))
print(greeting.hasSuffix("world!"))
print(greeting.hasSuffix("World!"))
```

```
true
true
false
```

String equality and comparison

Finding substrings

```swift
let greeting = "Hi Rick, my name is Amy."
if greeting.contains("my name is") {
  print("Making an introduction")
}
```

```
Making an introduction
```

String equality and comparison

Checking length

```swift
let name = "Ryan Mears"
let count = name.count
let newPassword = "1234"

if newPassword.count < 8 {
    print("This password is too short. Passwords should have at least 8 characters.")
}
```

```
This password is too short. Passwords should have at least 8 characters.
```

Functions

Functions

Defining a function

```
func functionName (parameters) -> ReturnType {
    // Body of the function
}
```

```
func displayPi() {
  print("3.1415926535")
}

displayPi()
```

```
3.1415926535
```

Parameters

```
func triple(value: Int) {
  let result = value * 3
  print("If you multiply \(value) by 3, you'll get \(result).")
}

triple(value: 10)
```

```
If you multiply 10 by 3, you'll get 30.
```

Parameters

Multiple parameters

```swift
func multiply(firstNumber: Int, secondNumber: Int) {
  let result = firstNumber * secondNumber
  print("The result is \(result).")
}

multiply(firstNumber: 10, secondNumber: 5)
```

```
The result is 50.
```

Return values

```
func multiply(firstNumber: Int, secondNumber: Int) -> Int {
  let result = firstNumber * secondNumber
  return result
}
```

Return values

```
func multiply(firstNumber: Int, secondNumber: Int) -> Int {
 return firstNumber * secondNumber
}
```

```
let myResult = multiply(firstNumber: 10, secondNumber: 5)
print("10 * 5 is \(myResult)")
```

```
print("10 * 5 is \(multiply(firstNumber: 10, secondNumber: 5))")
```

Argument labels

```swift
func sayHello(firstName: String) {
  print("Hello, \(firstName)!")
}

sayHello(firstName: "Amy")
```

Argument labels

```
func sayHello(to: String, and: String) {
  print("Hello \(to) and \(and)")
}

sayHello(to: "Luke", and: "Dave")
```

Argument labels

External names

```swift
func sayHello(to person: String, and anotherPerson: String) {
  print("Hello \(person) and \(anotherPerson)")
}

sayHello(to: "Luke", and: "Dave")
```

Argument labels

Omitting labels

```swift
print("Hello, world!")
```

```swift
func add(_ firstNumber: Int, to secondNumber: Int) -> Int {
    return firstNumber + secondNumber
}

let total = add(14, to: 6)
```

Default parameter values

```swift
func display(teamName: String, score: Int = 0) {
  print("\(teamName): \(score)")
}

display(teamName: "Wombats", score: 100)
display(teamName: "Wombats")
```

```
Wombats: 100
Wombats: 0
```

Lab 3a

Open playground Lab for 3a.playground

Try pages 1 and 3.

Collections

Collection types

Array

Dictionary

Arrays

Defining

```
[value1, value2, value3]
```

```
var names: [String] = ["Anne", "Gary", "Keith"]
```

Arrays

Defining

```
[value1, value2, value3]
```

```
var names = ["Anne", "Gary", "Keith"]
```

Arrays

Defining

```
var numbers = [1, -3, 50, 72, -95, 115]
```

Arrays

Defining

```swift
var numbers: [Int8] = [1, -3, 50, 72, -95, 115]
```

Arrays

contains

```
let numbers = [4, 5, 6]
if numbers.contains(5) {
  print("There is a 5")
}
```

```
There is a 5
```

Array types

```
var myArray: [Int] = []
```

```
var myArray: Array<Int> = []
```

```
var myArray = [Int]()
```

Working with arrays

repeating

```
var myArray = [Int](repeating: 0, count: 100)


let count = myArray.count


if myArray.isEmpty { }
```

Working with arrays

Accessing or setting a specific item

```
var names = ["Anne", "Gary", "Keith"]
let firstName = names[0]
print(firstName)
```

```
Anne
```

```
names[1] = "Paul"
print(names)
```

```
["Anne", "Paul", "Keith"]
```

Working with arrays

Appending

```
var names = ["Amy"]
names.append("Joe")
names += ["Keith", "Jane"]
print(names)
```

```
["Amy", "Joe", "Keith", "Jane"]
```

Working with arrays

Inserting

```
var names = ["Amy", "Brad", "Chelsea", "Dan"]
names.insert("Bob", at: 0)
print(names)
```

```
["Bob", "Amy", "Brad", "Chelsea", "Dan"]
```

Working with arrays

Removing

```
var names = ["Amy", "Brad", "Chelsea", "Dan"]
let chelsea = names.remove(at:2)
let dan = names.removeLast()
print(names)
```

```
["Amy", "Brad"]
```

```
names.removeAll()
print(names)
```

```
[]
```

Working with arrays

```
var myNewArray = firstArray + secondArray
```

Working with arrays

Arrays within arrays

```
let array1 = [1,2,3]
let array2 = [4,5,6]
let containerArray = [array1, array2]
let firstArray = containerArray[0]
let firstElement = containerArray[0][0]
print(containerArray)
print(firstArray)
print(firstElement)
```

```
[[1, 2, 3], [4, 5, 6]]
[1, 2, 3]
1
```

Dictionaries

```
[key1 : value1, key2: value2, key3: value3]
```

```swift
var scores = ["Richard": 500, "Luke": 400, "Cheryl": 800]
```

```swift
var myDictionary = [String: Int]()
var myDictionary = Dictionary<String, Int>()
var myDictionary: [String: Int] = [:]
```

Add/remove/modify a dictionary

Adding or modifying

```
var scores = ["Richard": 500, "Luke": 400, "Cheryl": 800]

scores["Oli"] = 399

let oldValue = scores.updateValue(100, forKey: "Richard")
```

Add/remove/modify a dictionary

Adding or modifying

```
var scores = ["Richard": 500, "Luke": 400, "Cheryl": 800]

scores["Oli"] = 399

if let oldValue = scores.updateValue(100, forKey: "Richard") {
  print("Richard's old value was \(oldValue)")
}
```

```
Richard's old value was 500
```

Add/remove/modify a dictionary

Removing

```
var scores = ["Richard": 100, "Luke": 400, "Cheryl": 800]
scores["Richard"] = nil
print(scores)

if let oldValue = scores.removeValue(forKey: "Luke") {
  print("Luke's score was \(oldValue) before he stopped playing")
}
print(scores)
```

```
["Cheryl": 800, "Luke": 400]
Luke's score was 400 before he stopped playing
["Cheryl": 800]
```

Accessing a dictionary

```
var scores = ["Richard": 500, "Luke": 400, "Cheryl": 800]

let players = Array(scores.keys) //["Richard", "Luke", "Cheryl"]
let points = Array(scores.values) //[500, 400, 800]

if let myScore = scores["Luke"] {
  print(myScore)
}
```

```
400
```

```
if let henrysScore = scores["Henry"] {
  print(henrysScore)
}
```

Loops

Loops

```
for
while
```

for loops

```
for index in 1...5 {
  print("This is number \(index)")
}
```

for loops

```
for _ in 1...5 {
  print("Hello!")
}
```

for loops

```
let names = ["Joseph", "Cathy", "Winston"]
for name in names {
  print("Hello \(name)")
}
```

```
for letter in "ABCDEFG".characters {
  print("The letter is \(letter)")
}
```

for loops

```
for (index, letter) in "ABCDEFG".characters.enumerated() {
  print("\(index): \(letter)")
}
```

for loops

```swift
let vehicles = ["unicycle" : 1, "bicycle" : 2, "tricycle" : 3, "quad bike" : 4]
for (vehicleName, wheelCount) in vehicles {
  print("A \(vehicleName) has \(wheelCount) wheels")
}
```

while loops

```swift
var numberOfLives = 3

while numberOfLives > 0 {
  playMove()
  updateLivesCount()
}
```

while loops

```swift
var numberOfLives = 3

while numberOfLives > 0 {
  print("I still have \(numberOfLives) lives.")
}
```

while loops

```swift
var numberOfLives = 3
var stillAlive = true

while stillAlive {
  print("I still have \(numberOfLives) lives.")
  numberOfLives -= 1
  if numberOfLives == 0 {
    stillAlive = false
  }
}
```

Control transfer statements

```
for counter in -10...10 {
  print(counter)
  if counter == 0 {
    break
  }
}
```

```
-10
-9
...
0
```

Lab 3 Collections and Lab 3 Loops

Open and complete exercises 1 and 3 in `Lab3 Collections.playground`.

Open and complete exercise 1 in Lab 3 Loops.playground.