Control Flow

swift 提供了多种控制流语句，包括较为复杂的 switch

For-In Loops

可以用 for-in 遍历一个序列，包括基本的容器，range，或者 string

for-in 循环的变量是隐式声明的常量，如果循环中不需要的话可以用下划线作为占位符

```
for _ in 1...power {
    answer *= base
}
```

如果想要带步进的 range，可以使用 `stride(from:to:by:)`（或者闭区间版的 `stride(from:through:by:)`）

```
let minuteInterval = 5
for tickMark in stride(from: 0, to: minutes, by: minuteInterval) {
    // render the tick mark every 5 minutes (0, 5, 10, 15 ... 45, 50, 55)
}
```

While Loops

有两种循环 `while` 和 `repeat-while`

`repeat-while` 会执行一次循环体，无论循环变量的真假

Conditional Statements

swift 提供两种条件语句：`if` 和 `switch`

`switch` 可以处理较复杂的情况

switch 的 case 必须是全面的，即所有输入都有一个满足的 case。swift 的 switch 没有 fall through，所以不需要 case 之后的 break。每个 case 必须有至少一条可执行的语句，如果需要一个 case 体匹配多种情况，使用逗号分隔每种情况

除去最简单的匹配是否相等，也可以匹配是否在区间

```
case 0:
    naturalCount = "no"
case 1..<5:
    naturalCount = "a few"
```

tuple 也可以用作 switch 的条件，可以单独处理每个分量

```
let somePoint = (1, 1)
switch somePoint {
case (0, 0):
    print("\(somePoint) is at the origin")
case (_, 0):
    print("\(somePoint) is on the x-axis")
case (0, _):
    print("\(somePoint) is on the y-axis")
case (-2...2, -2...2):
    print("\(somePoint) is inside the box")
default:
    print("\(somePoint) is outside of the box")
}
```

switch 可以将被匹配的值绑定到常量/变量用作在 case 体内使用

```
let anotherPoint = (2, 0)
switch anotherPoint {
case (let x, 0):
    print("on the x-axis with an x value of \(x)")
case (0, let y):
    print("on the y-axis with a y value of \(y)")
case let (x, y):
    print("somewhere else at (\(x), \(y))")
}
```

在 binding 时可以用 `where` 来添加额外的条件

```
case let (x, y) where x == y:
    print("(\(x), \(y)) is on the line x == y")
```

Control Transfer Statement

swift 提供了五种控制转移语句

- `continue`：中断当前循环，开始下一次循环

- `break`：跳出当前循环，如果在 switch 中使用可以直接结束 switch，一般用于处理匹配但不继续处理的值
- `fallthrough`：用 fallthrough 提供 C 风格的 switch
- `return`：从函数调用中返回
- `throw`：抛出异常

可以在控制流语句前加上 label，用于显式指定 continue 等语句的目标

```
label name: while condition {
    statements
}
```

Early Exit

`guard` 用于确保执行某些语句前条件必须为真，`guard` 必须有一个 else 子句

```
guard let name = person["name"] else {
    return
}
```

如果条件满足，绑定的变量可以在后续使用，如果失败，应当采取措施避免执行后面的语句（需要被绑定的值）

相比 if 可读性更高

Checking API Availability

swift 内置了检查 API 是否可用的功能

```
if #available(iOS 10, macOS 10.12, *) {
    // Use iOS 10 APIs on iOS, and use macOS 10.12 APIs on macOS
} else {
    // Fall back to earlier iOS and macOS APIs
}
```

可以检查的平台包括 iOS，macOS，watchOS，tvOS 等，可以检查大版本或者小版本