



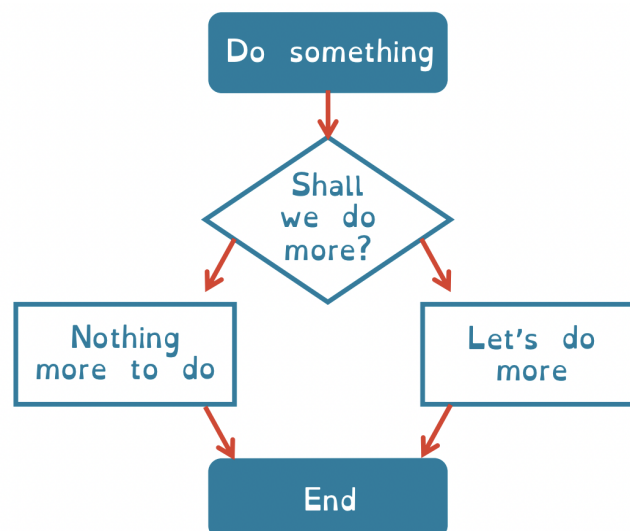
# Lesson 2

## PDF Slides Lesson 2

 [Lesson2\\_slides.https://drive.google.com/file/d/15GQAKEqWODAaheZBMtSDHy8FNiMz3bf5/view?usp=drivesdk](https://drive.google.com/file/d/15GQAKEqWODAaheZBMtSDHy8FNiMz3bf5/view?usp=drivesdk)  
pdf

## If-else Statements

- This diagram below explains If statements



1. Launch Xcode by going to **Applications > Xcode**.
2. Select **File > New > Playground > iOS > Blank** and save your Playground with the name '**IfElsePG**'.
3. Open up the text file and copy and paste the code from it into your new Playground.



4. Your Playground should now look like this:

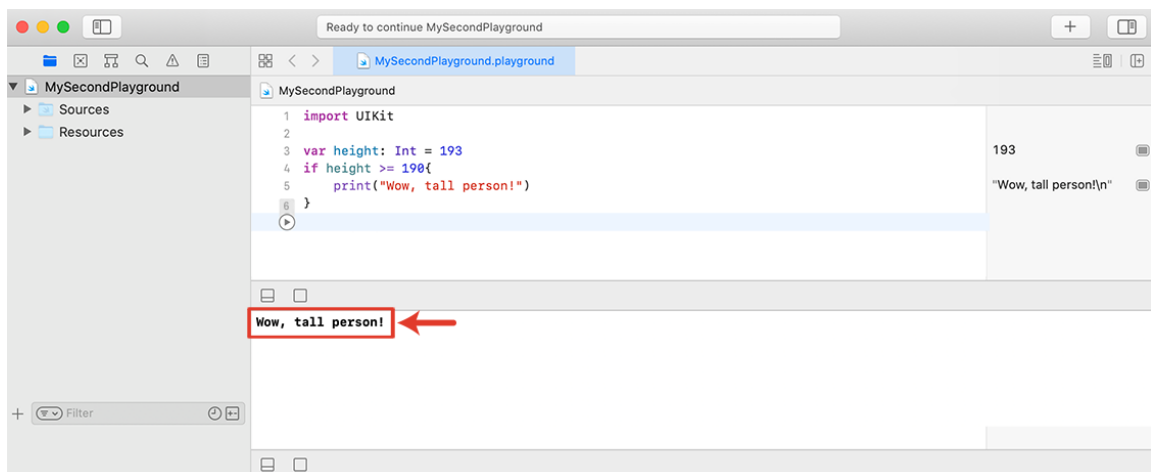
```
import UIKit

var height: Int = 193
if height >= 190 {
    print ("Wow, tall person!")
}
```

5. Click on the **Play** button to see what happens.



The text should appear in the results area down the bottom of your screen.



6. Change the height value from 193 to 200. Click on the **Play** button to run it again. The text 'Wow, tall person!' should appear again.



7. Now let's change the height value from 193 to 190. Click on the **Play** button to run it again. The text 'Wow, tall person!' will still appear. This is because in our code we're checking that the height is **greater than or equal to** 190. If it is, the text string appears.



8. Change the height value from 190 to 183 and again run the program to see what happens. What happens? In this case – nothing happens. This is because we've set the height value to 183 and it doesn't come into the if statement as the height value is **not** greater than or equal to 190. The program checks the height against the conditions specified and simply drops out of the code and comes to an end.
9. We're now going to build on this piece of code and turn our **if statement** into an **if-else statement**.

Enter the following lines of code to the bottom of your Playground so your Playground now looks like this.



```
import UIKit

var height: Int = 183
if height >= 190 {
    print ("Wow, tall person!")
}
else {
    print ("Another short person!")
}
```

10. In this piece of code we are testing whether the height of a person is greater than or equal to 190. Click on the **Play** button to test your code. The text "person is short" should appear in the results window.
- If height is greater than or equal to 190 we say: "Wow, tall person!"
  - If height is less than 190 we move into the else statement and say: "Another short person!"

## Logical Operators

### Types of Logical Operators

|  Operator |  Description |
|--|---|
| <code>==</code>  | The items must be equal to each other   |
| <code>!=</code>  | The values must not be equal to each other  |
| <code>&gt;</code>  | The value on the left is greater than the value on the right                                    |
| <code>&gt;=</code>   | The value on the left is greater than or equal to the value on the right                        |

| <u>Aa</u><br>Operator | Description  |
|-----------------------|--|
| $\leq$                | The value on the left is smaller than the value on the right                             |
| $\leq\equiv$          | The value on the left is smaller than or equal to the value on the right                 |
| <u>&amp;&amp;</u>     | AND - the condition on the left must be true AND the condition on the right              |
| <u>  </u>             | OR - the condition on the left must be true OR the condition on the right must be true   |
| !                     | The opposite of the conditional statement immediately following the operator is returned |

1. Launch Xcode by going to **Applications > Xcode** and open up the Playgrounds file you've just downloaded. It should look like this.

```
import UIKit

var studentAge: Int = 17
var birthMonth: String = "June"

if studentAge >= 10 && birthMonth == "June" {print("10 or older, born in June!")}
else{
    print("Perhaps less than 10")
    print("Also possibly not born in June!")
}
```

2. In this Playground we have two variables and an if-else statement:
  - A **variable** called studentAge which is initially set to 17.
  - A **variable** called birthMonth which is initially set to June.
  - An **if-else** statement where the following conditions are checked:
    - **if** the studentAge is greater than or equal to 10 **AND** the birthMonth equals June then the program prints out "10 or older, born in June!".
    - **else** the program prints out "Perhaps less than 10" and "Also possibly not born in June!".
3. Click on the **Play** button to see what happens. The text "10 or older, born in June!" should appear in your results window.



4. We're now going to make some changes to the if statement.

Change 'birthMonth == June' (birthMonth **equals** June) to 'birthMonth != June' (birthMonth **not equal** to June).

Your Playground should now look like this:

```
import UIKit

var studentAge: Int = 17
var birthMonth: String = "June"

if studentAge >= 10 && birthMonth != "June" {print("10 or older, born in June!")}
else{
    print("Perhaps less than 10")
    print("Also possibly not born in June!")
}
```

5. Click **Play** to run the code again. The text "Perhaps less than 10" and "Also possibly not born in June!" should appear in your results window. This is because we have indicated that if our birthMonth does not equal June then we print the first statement ("10 or older, born in June!"). However as the birthMonth does equal June this means that we enter into second part of the if-else statement.



6. Let's make some more changes to the programming logic. Your Playground should now look like this:

- Change && to ||
  - this changes the **AND** statement to an **OR** statement
- Change != to ==
  - this changes **does not equal** back to **equals**
- Change the birthMonth variable from "June" to "July"
  - this changes the birthMonth from June to July.

```
import UIKit

var studentAge: Int = 17
var birthMonth: String = "July"
```

```
if studentAge >= 10 || birthMonth == "June" {print("10 or older, born in June!")}  
else{  
    print("Perhaps less than 10")  
    print("Also possibly not born in June!")  
}
```

7. Click **Play** to run the code again. The text “10 or older, born in June!” will appear in your results window. This is because our if statement is asking IF the code on the right hand side is true **OR** the code on the left hand side is true, then print (“10 or older, born in June!”). As our studentAge = 17, when the code runs and checks (if studentAge >= 10 **OR** birthMonth == “June”) the answer is TRUE and the code executes. **However; one important thing for app developers to note** is that if you update your programming logic, you may also need to update any corresponding text feedback. In this example the print out says “10 or older, born in June!” **even though** our birthMonth value was set to July. We neglected to update our text string when we updated our logic.

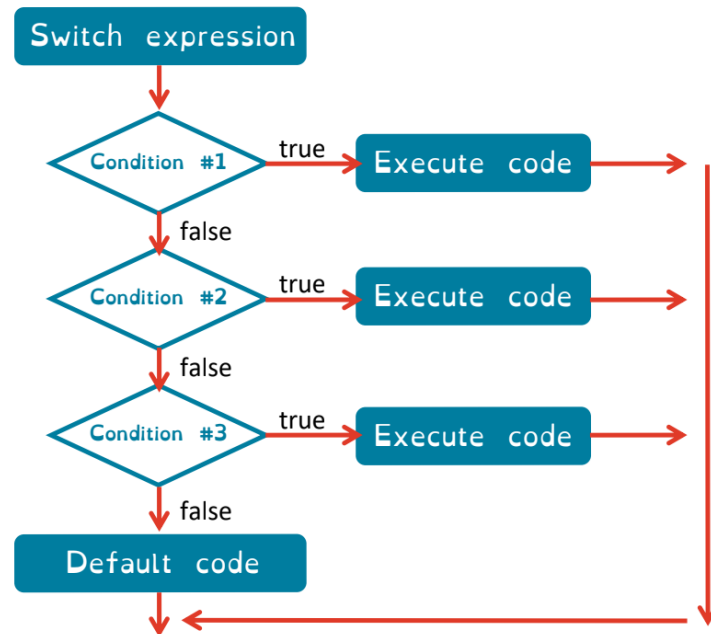


8. Change the value of studentAge to 9 and run the code again. The text “Perhaps less than 10” and “Also possibly not born in June!” will appear in your results window.
9. Change the value of birthMonth to “June” and run the code again. The text “10 or older, born in June!” will appear. Which doesn’t make sense! Similar to before, as the studentAge >= 10 is not true, because this is an OR statement, the program goes on to check the code on the right hand side. In this case it checks to see that the birthMonth equals June, which it is, so the statement is still true. Therefore the program continues and it executes the next line of code. However as we’ve just noted; this feedback is very confusing and this is because we neglected to make changes to the feedback (print statement) when we changed our logic.

That concludes the end of this exercise where we’ve demonstrated the use of logical operators and shown how to apply them in Swift.

## Switch Statements

- Basically a chained up If statement



- Example with number of sibling

```
var numOfSyblings: Int = 2
switch numOfSyblings{
case 0:
    print("Only Child")
case 1:
    print("A single sibling, possibly twins... Or not")
case 2:
    print("A brother or two, or, a sister or two")
default:
    print("Wow....")
}
```

## Collection in Swift

### Array

- An Array is an ordered list of data of the same type
- Similar to a variable, it could be created without defining data type

```
var integerList: [int] = [1, 2, 3, 4]
var numList = [38, 31, 12, 10]
```

- Index starts at position 0 `print(numList[0])`

- The table below lists operators that could be used on arrays

## Operations

| -               | Operation Name                                  | Description   |
|-----------------|---|---|
| <u>Untitled</u> | <code>&lt;array_name&gt;.contains(x)</code>     | If x is in the array, returns true else false           |
| <u>Untitled</u> | <code>&lt;array_name&gt;.isEmpty</code>         | Returns true if array is empty, otherwise returns false |
| <u>Untitled</u> | <code>&lt;array_name&gt;.append(x)</code>       | Appends x to the end of the array                       |
| <u>Untitled</u> | <code>&lt;array_name&gt;.insert(x, at:y)</code> | Inserts x into the array at position y                  |
| <u>Untitled</u> | <code>&lt;array_name&gt;.remove(at:y)</code>    | Removes the item at position y in the array             |
| <u>Untitled</u> | <code>&lt;array_name&gt;.removeLast()</code>    | Removes the last item in the array                      |
| <u>Untitled</u> | <code>&lt;array_name&gt;.removeAll()</code>     | Removes everything form the array                       |
| <u>Untitled</u> | <code>&lt;array_name&gt;.insert(x, at:y)</code> |   |
| <u>Untitled</u> | <code>&lt;array_name&gt;.remove(at:y)</code>    |   |
| <u>Untitled</u> | <code>&lt;array_name&gt;.removeLast()</code>    |   |
| <u>Untitled</u> | <code>&lt;array_name&gt;.removeAll()</code>     |   |

- Example code:

```
var numList = [10, 11, 12, 13]

print(numList.count)

numList.append(69)

print(numList)

numList.removeLast()

print(numList)
```

## Dictionary

- dictionary contains a word and a definition

```
var myDictionary = ["David": 28, "Tristan": 23]
```

- Will replace the value if the key exists:

```
nameAge["David"] = 44
```



- Will update the value for key if it exists:

```
nameAge.updateValue(26, forKey: "David")
```

- Will remove the value for key if it exists:

```
nameAge.removeValue(forKey: "David")
```

- This ensures that the key exists before attempting to do anything with a value.

```
if let davidAge = nameAge["David"]{  
    print(davidAge)  
}
```

## Loops

- Two kinds of loops, for loop and while loop

## For Loops

- Allow the code to repeat in a sequence or range or collection
- For Loop vs non for loop

```
print(10)  
print(11)  
print(12)  
print(13)  
print(14)  
pprint(15)
```

```
for num in 10...15{  
    print("Number is:\(num)")  
}
```

- Looping through arrays

```
var lectures = ["David", "Tristan", "Steve"]  
for lecture in lectures{  
    print("Lecture is: \(lecturer)")  
}
```

- Looping through dictionary

```
var ages = ["David":28, "Tristan":22, "Steve":45]
for person in ages{
    print(person)
}
for person in ages{
    print(person.key)
}
for person in ages{
    print(person.value)
}
```

## While Loop

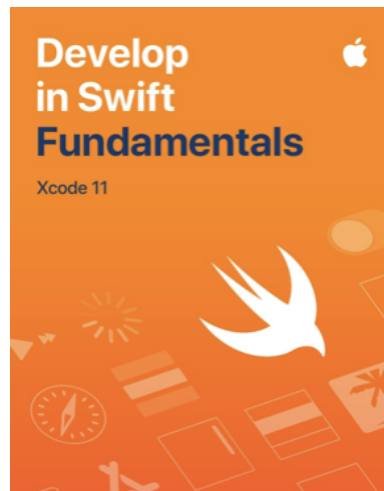
- Keep looping until the condition is met

```
var myScore = 5
while myScore < 10{
    print("Not there yet, go again?")
    myScore += 1
}
```

```
var points = 5
var numOfServes = 1
while points < 50 && numOfServes < 3{
    print("Playton, point = \(points)")
    points = points * 2
    numOfServes += 1
}
```

## Extra Resources


- In Apple Books:



## Source Code:

Ace5584/IOS-Dev-Notes

Contribute to Ace5584/IOS-Dev-Notes development by creating an account on GitHub.

 <https://github.com/Ace5584/IOS-Dev-Notes/tree/main/Course%201/L2>

Ace5584/**IOS-Dev-Notes**



 1  
Contributors

 0  
Issues

 0  
Stars

 0  
Forks

