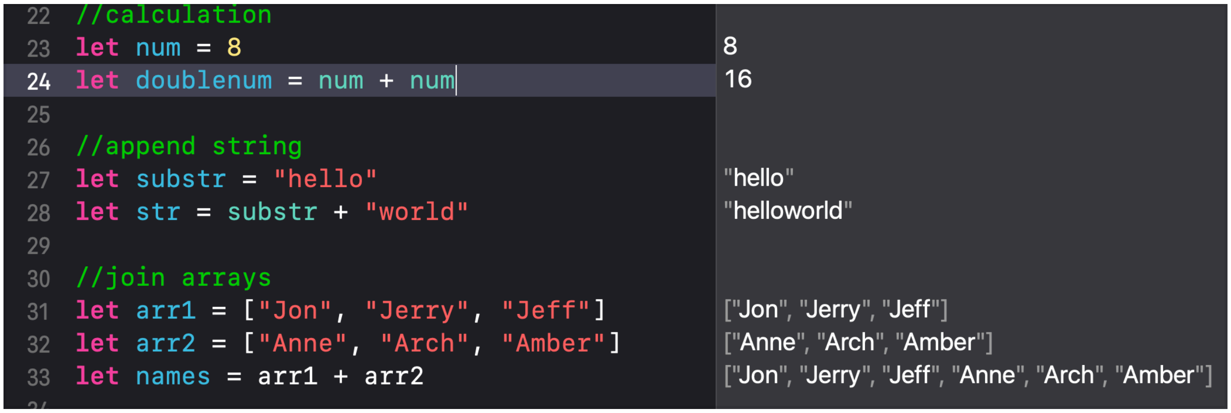
# SWIFT INTRO :
# OPERATORS & CONDITIONS & LOOPS

✓ Arithmetic Operator : * / + - %

✓ Operator Overloading:

+ We can do -= or +=
+ An operator does depends on the values you use it with
+ For example : with + , we can do sum , append string , join arrays



```swift
//calculation
let num = 8
let doublenum = num + num

//append string
let substr = "hello"
let str = substr + "world"

//join arrays
let arr1 = ["Jon", "Jerry", "Jeff"]
let arr2 = ["Anne", "Arch", "Amber"]
let names = arr1 + arr2
```

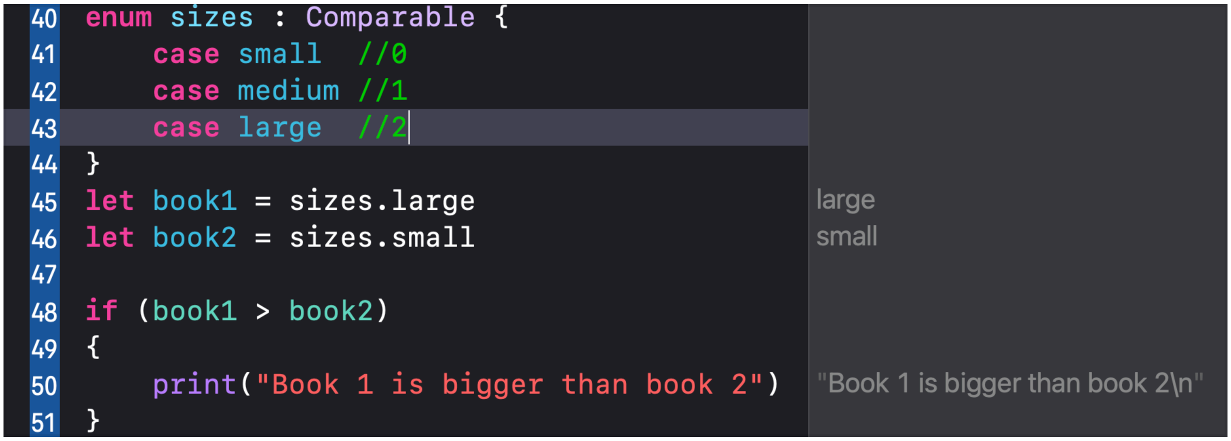
```
8
16


"hello"
"helloworld"


["Jon", "Jerry", "Jeff"]
["Anne", "Arch", "Amber"]
["Jon", "Jerry", "Jeff", "Anne", "Arch", "Amber"]
```

✓ Comparison Operators

== !=

<= >=

- We can have comparable enum as well

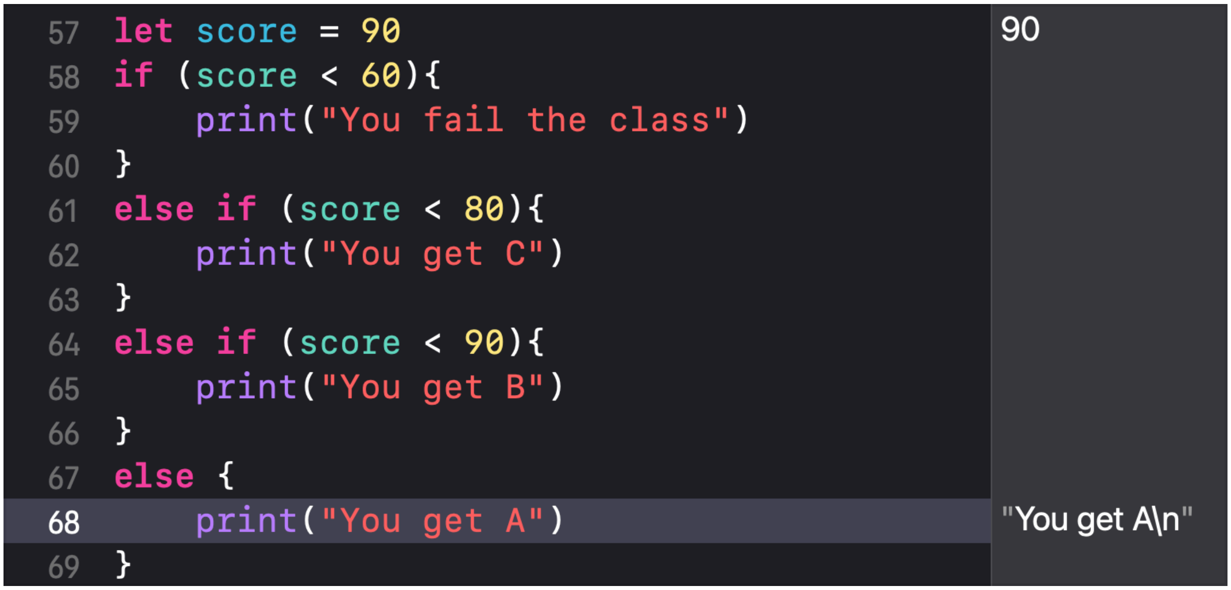


```swift
enum sizes : Comparable {
    case small  //0
    case medium //1
    case large  //2
}
let book1 = sizes.large
let book2 = sizes.small

if (book1 > book2)
{
    print("Book 1 is bigger than book 2")
}
```

```
large
small



"Book 1 is bigger than book 2\n"
```

That will print “true”, because **small** comes before **large** in the enum case list

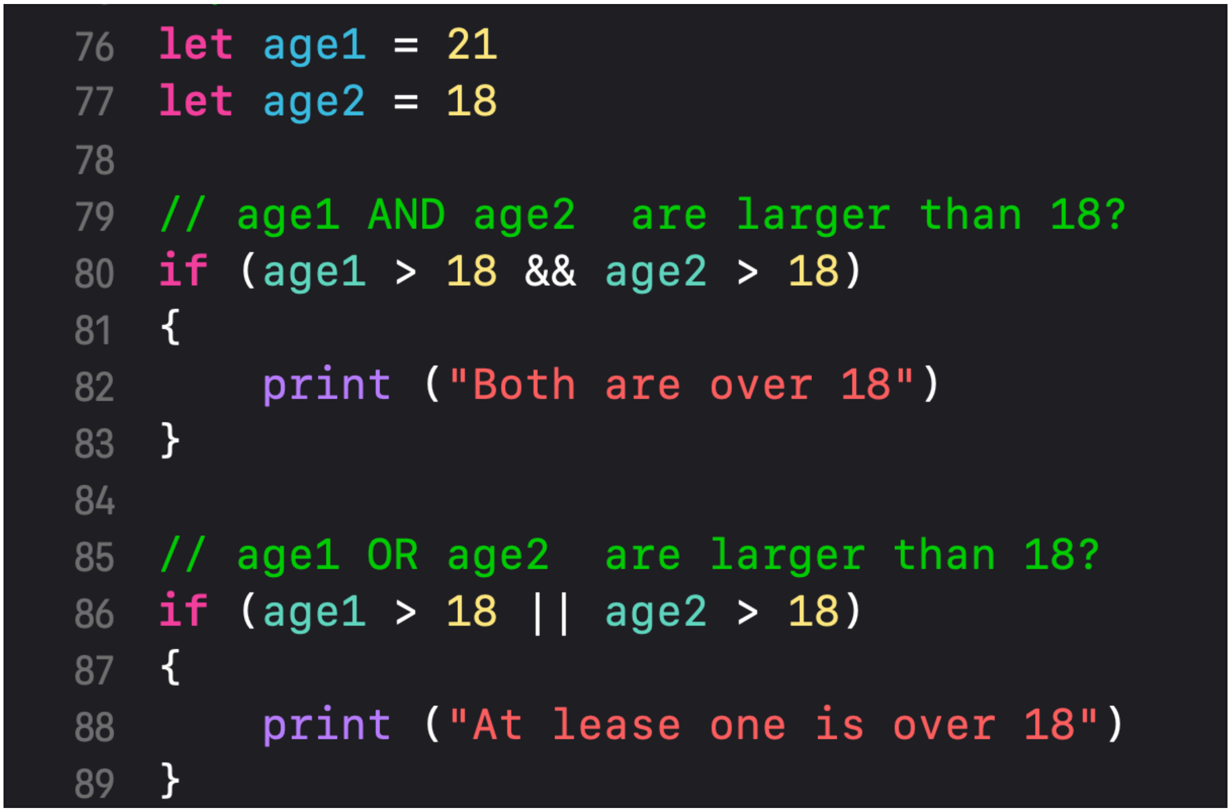
## ✓ Conditions (if / else if / else )



```swift
let score = 90
if (score < 60){
    print("You fail the class")
}
else if (score < 80){
    print("You get C")
}
else if (score < 90){
    print("You get B")
}
else {
    print("You get A")
}
```

```
90
"You get A\n"
```

## ✓ Combining operators (  &&   | | )

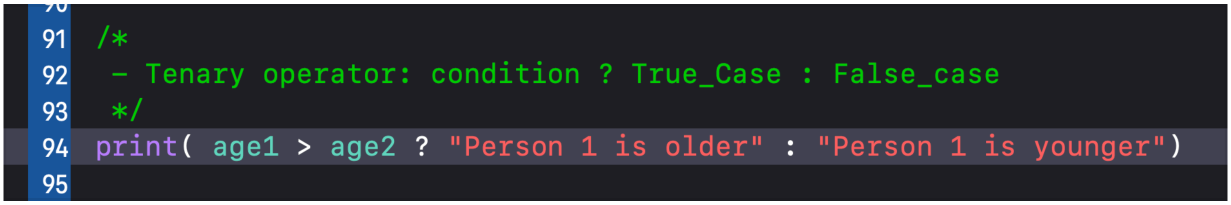


```swift
let age1 = 21
let age2 = 18

// age1 AND age2  are larger than 18?
if (age1 > 18 && age2 > 18)
{
    print ("Both are over 18")
}

// age1 OR age2  are larger than 18?
if (age1 > 18 || age2 > 18)
{
    print ("At lease one is over 18")
}
```

## ✓ Ternary Operator

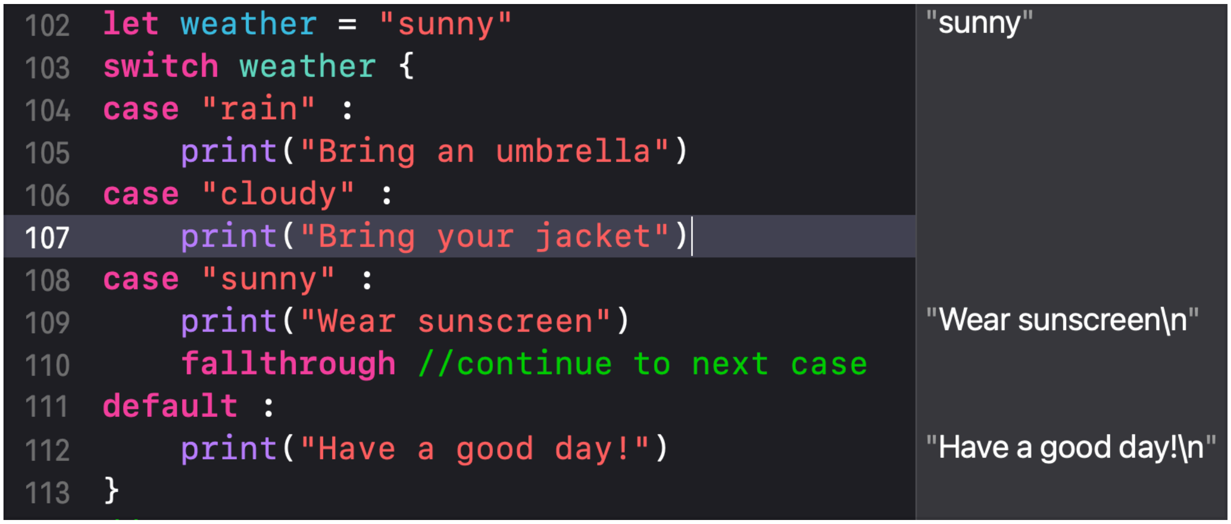
The ternary operator is a condition plus true or false blocks all in one, split up by a question mark and a colon

It works with three values at once, which is where its name comes from: it checks a condition specified in the first value, and if it's true returns the second value, but if it's false returns the third value.



```
/*
 - Tenary operator: condition ? True_Case : False_case
 */
print( age1 > age2 ? "Person 1 is older" : "Person 1 is younger")
```

### Switch condition



```
let weather = "sunny"
switch weather {
case "rain" :
    print("Bring an umbrella")
case "cloudy" :
    print("Bring your jacket")
case "sunny" :
    print("Wear sunscreen")
    fallthrough //continue to next case
default :
    print("Have a good day!")
}
```

```
"sunny"
"Wear sunscreen\n"
"Have a good day!\n"
```

The last case – **default** – is required because Swift makes sure you cover all possible cases so that no eventuality is missed off.

Swift will only run the code inside each case. If you want execution **to continue on to the next case**, use the **fallthrough** keyword

### Range operators

- Half-open range operator ( . . >)

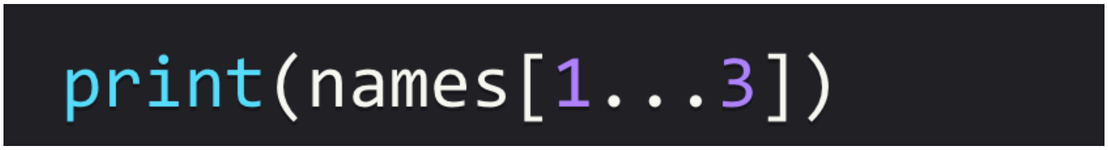
creates ranges up to but **excluding the final value**

- Closed range operator ( . . . )

creates ranges up to and ***including* the final value**

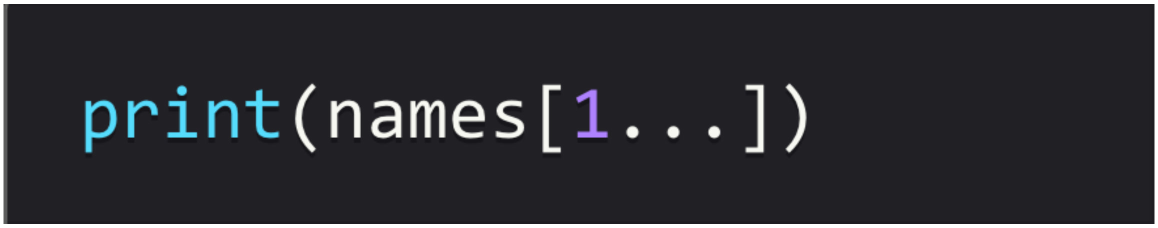
```
let names = ["Piper", "Alex", "Suzanne", "Gloria"]
```

+ To print a range of values



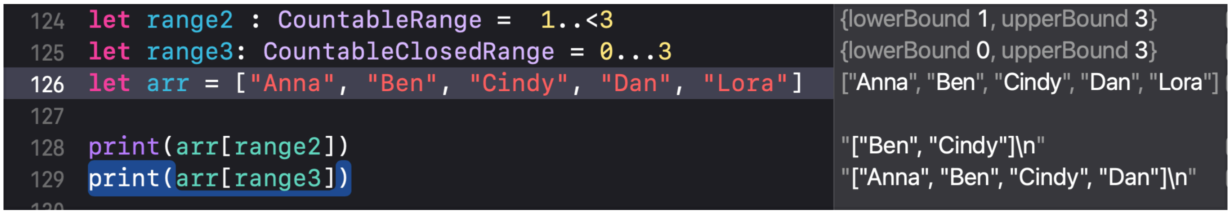
```swift
print(names[1...3])
```

+ A ***one-sided range*** to say “give me 1 to the end of the array”



```swift
print(names[1...])
```

+ We can also define a range as a var

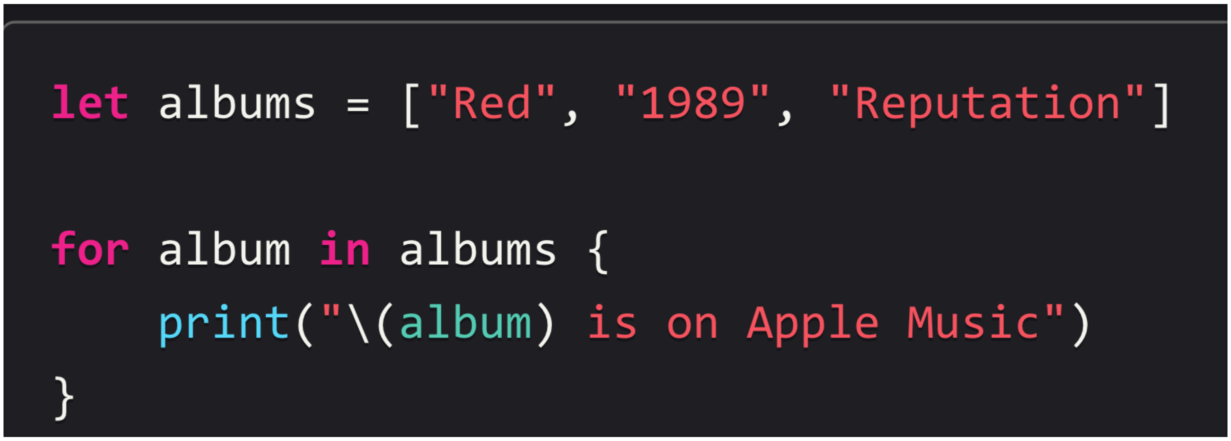


```swift
let range2 : CountableRange =  1..<3
let range3: CountableClosedRange = 0...3
let arr = ["Anna", "Ben", "Cindy", "Dan", "Lora"]

print(arr[range2])
print(arr[range3])
```

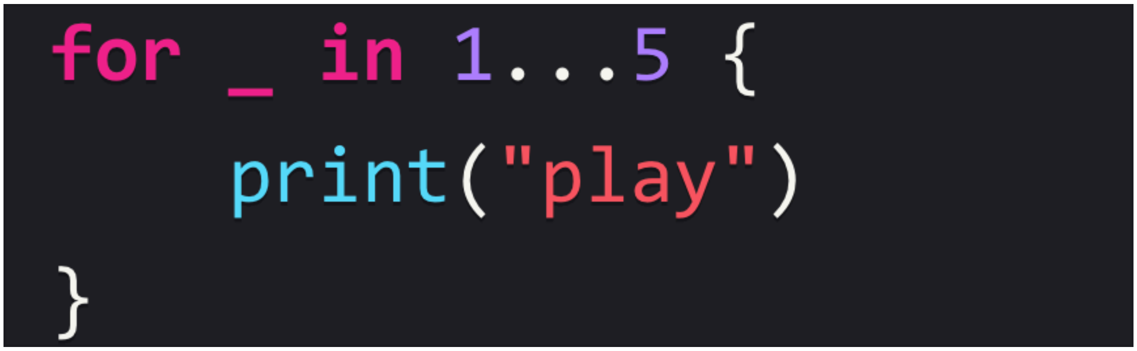
```
{lowerBound 1, upperBound 3}
{lowerBound 0, upperBound 3}
["Anna", "Ben", "Cindy", "Dan", "Lora"]

"["Ben", "Cindy"]\n"
"["Anna", "Ben", "Cindy", "Dan"]\n"
```

✓ Loops

– For Loop:

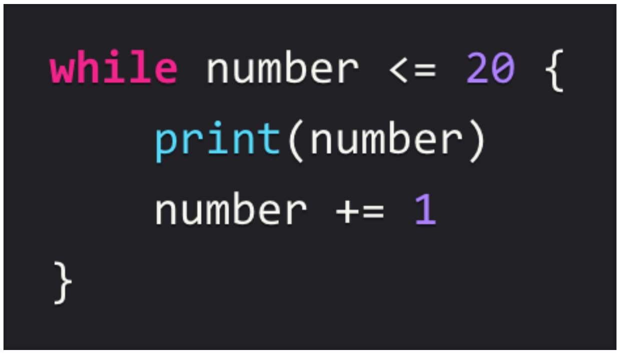


```swift
let albums = ["Red", "1989", "Reputation"]

for album in albums {
    print("\(album) is on Apple Music")
}
```



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

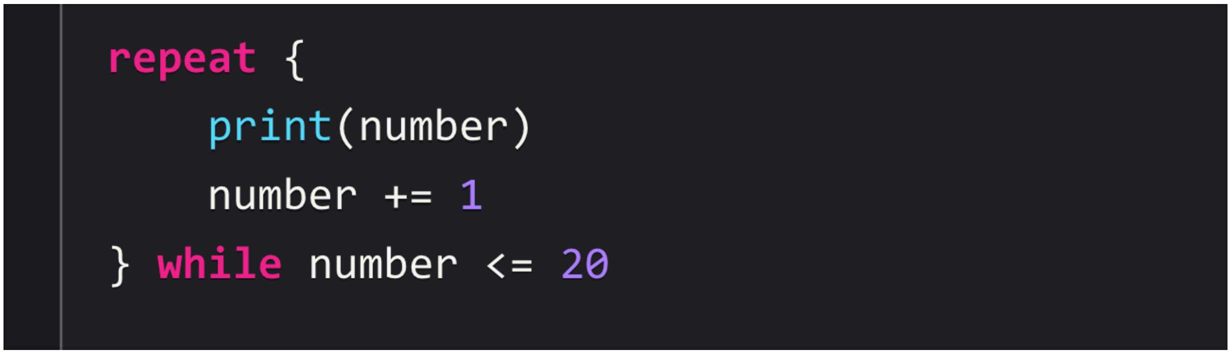
- While loop:



```
while number <= 20 {
    print(number)
    number += 1
}
```

- Repeat loop (Similar to DO - WHILE)

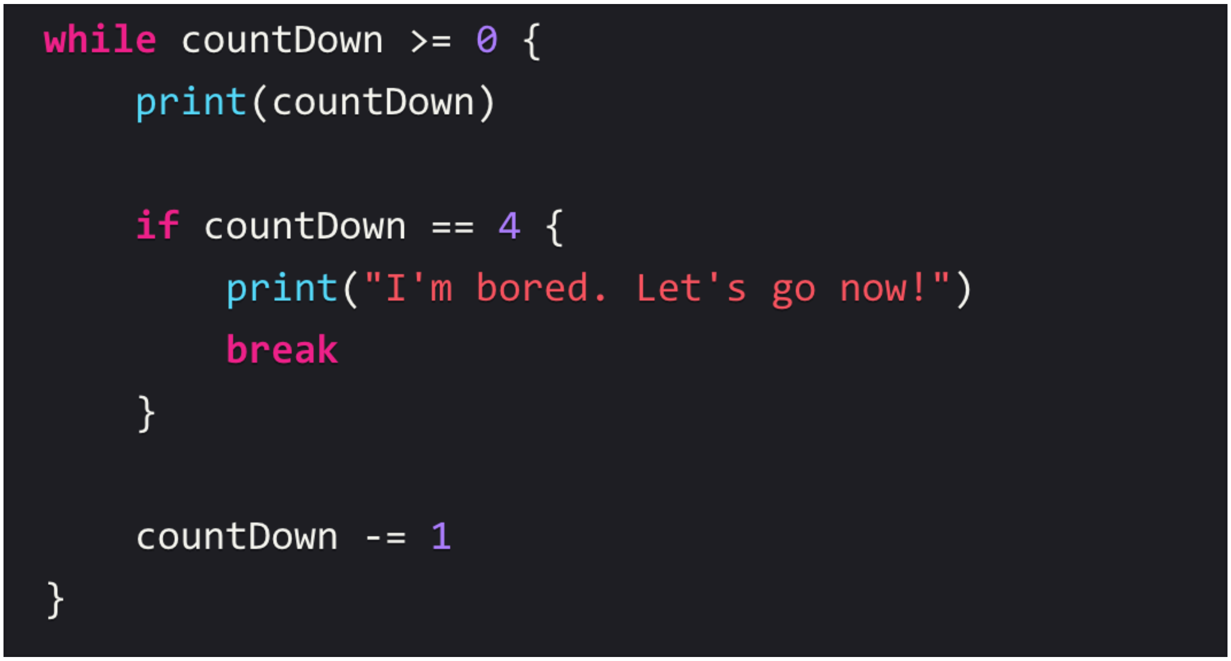
It’s identical to a **while** loop except the condition to check comes at the end.



```
repeat {
    print(number)
    number += 1
} while number <= 20
```

- Exiting LOOP  using BREAK

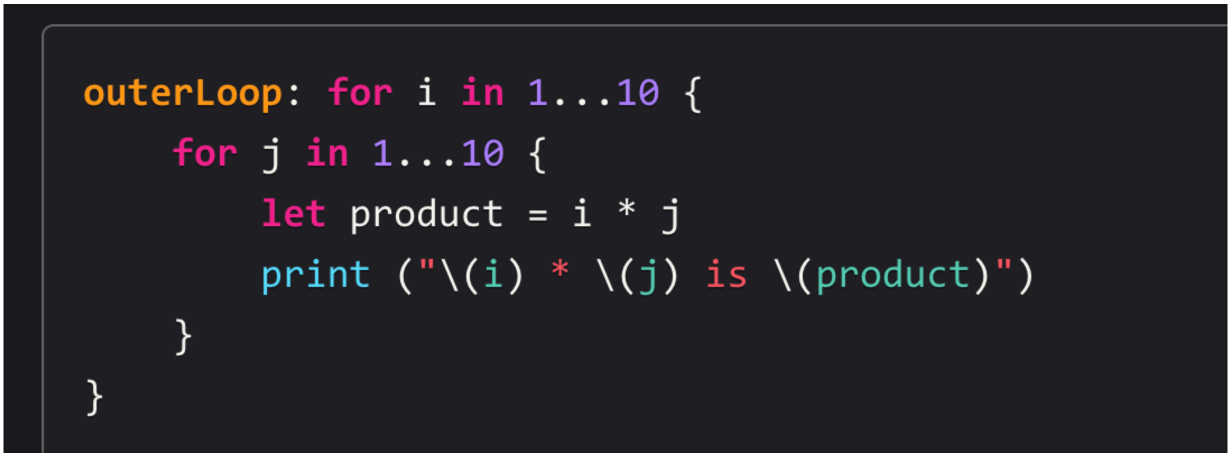
You can exit a loop at any time using the **break** keyword



```
while countDown >= 0 {
    print(countDown)

    if countDown == 4 {
        print("I'm bored. Let's go now!")
        break
    }

    countDown -= 1
}
```

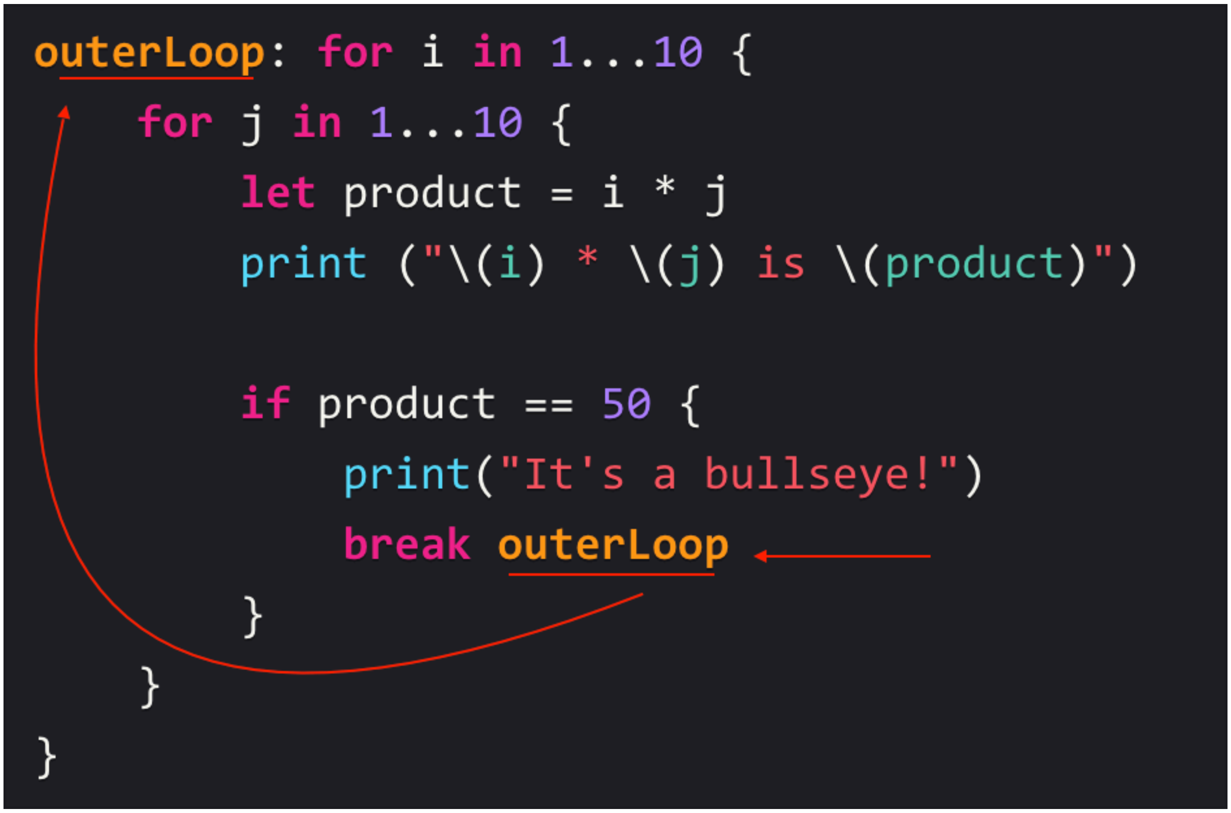
– Exiting Multiple loops

1. Label outer loop (you can label as many as you need)



```
outerLoop: for i in 1...10 {
    for j in 1...10 {
        let product = i * j
        print ("\(i) * \(j) is \(product)")
    }
}
```

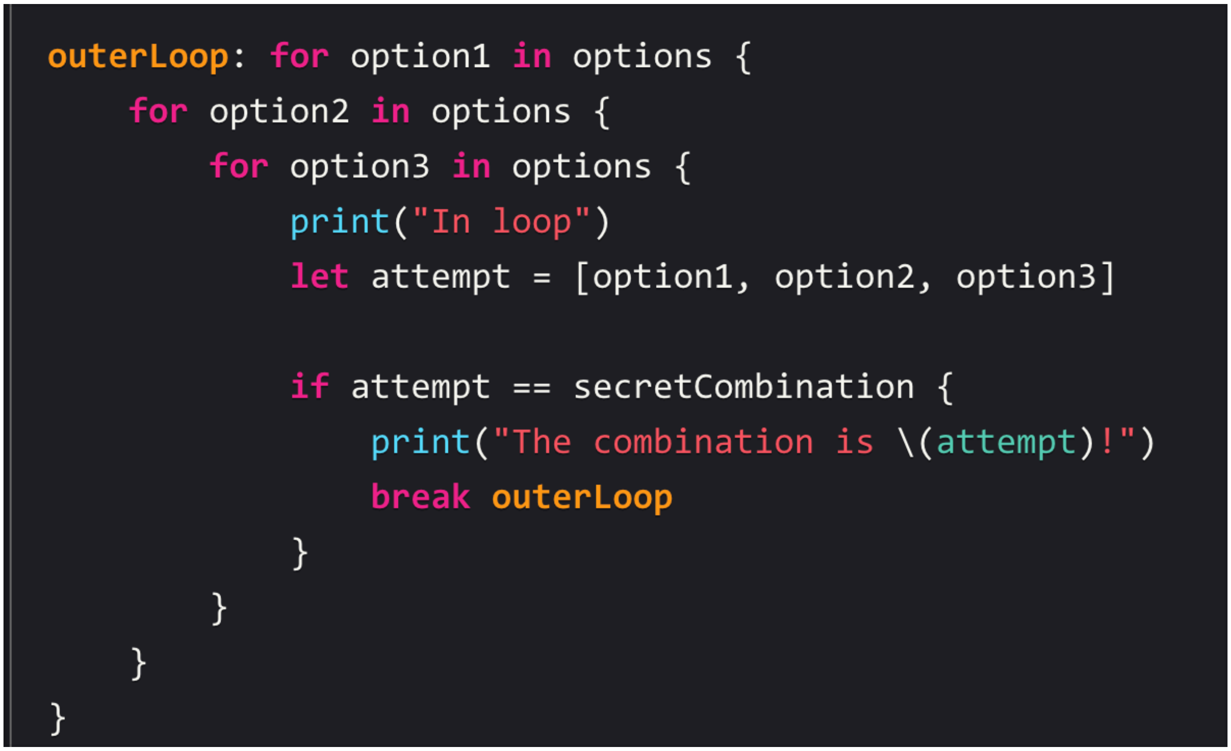
2. Add condition inside inner loop and use '**break outerLoop**' to exit both loops at the same time



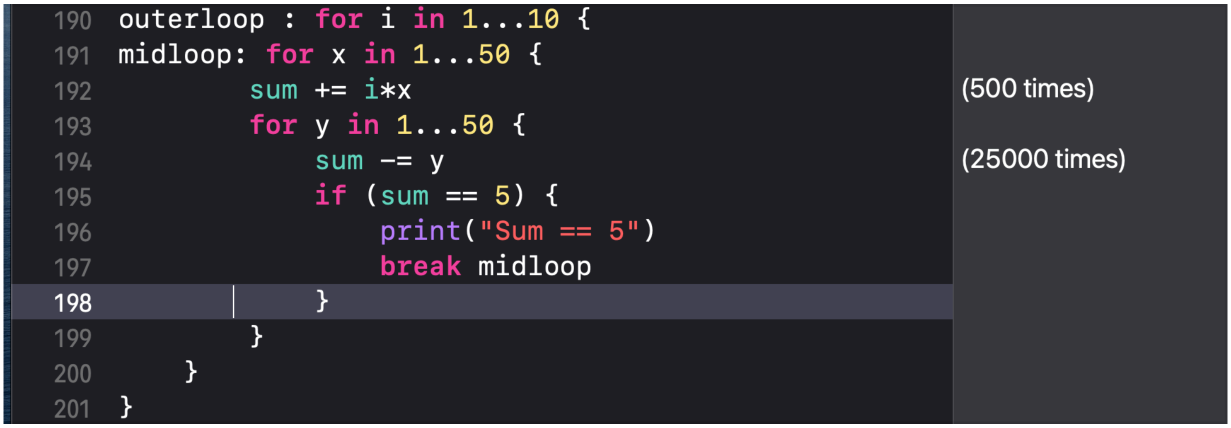
```
outerLoop: for i in 1...10 {
    for j in 1...10 {
        let product = i * j
        print ("\(i) * \(j) is \(product)")

        if product == 50 {
            print("It's a bullseye!")
            break outerLoop
        }
    }
}
```

** With a regular **break**, only the inner loop would be exited – the outer loop would continue where it left off.

Another example of exiting multiple loops



```
outerLoop: for option1 in options {
    for option2 in options {
        for option3 in options {
            print("In loop")
            let attempt = [option1, option2, option3]

            if attempt == secretCombination {
                print("The combination is \(attempt)!")
                break outerLoop
            }
        }
    }
}
```

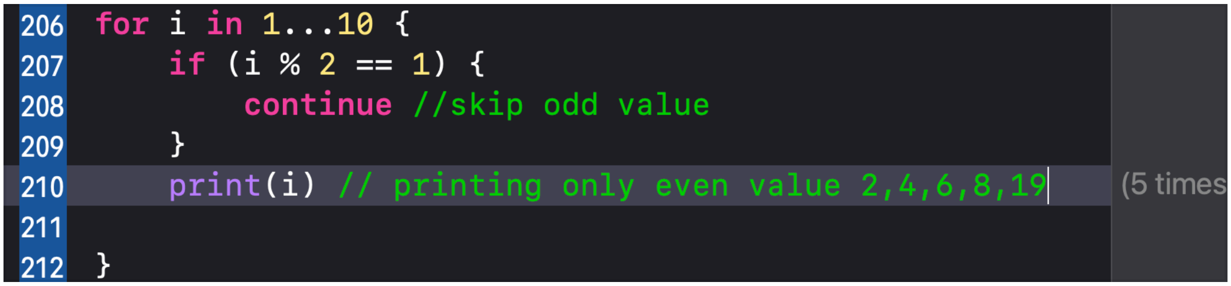


```
outerloop : for i in 1...10 {
midloop: for x in 1...50 {
        sum += i*x
        for y in 1...50 {
            sum -= y
            if (sum == 5) {
                print("Sum == 5")
                break midloop
            }
        }
    }
}
```

(500 times)

(25000 times)

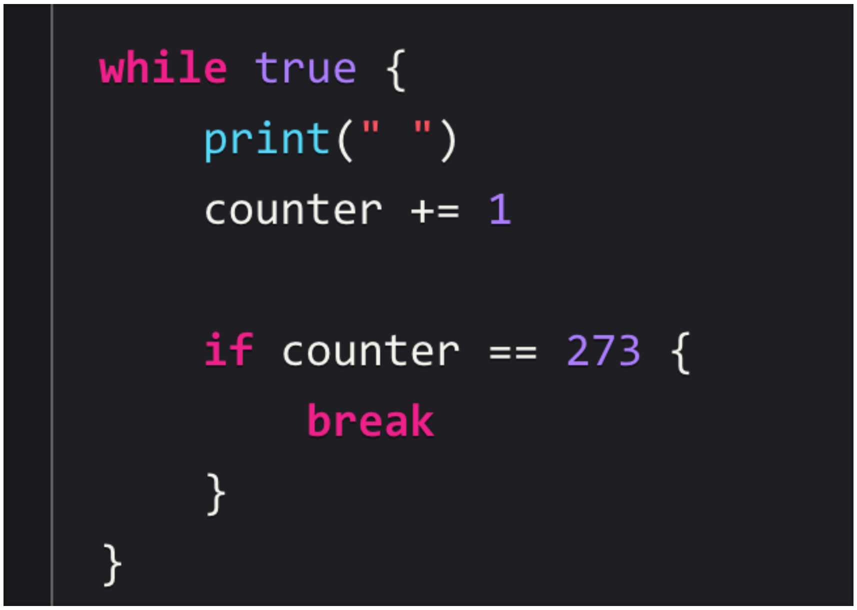
- Skipping items:



```
for i in 1...10 {
    if (i % 2 == 1) {
        continue //skip odd value
    }
    print(i) // printing only even value 2,4,6,8,19

}
```

(5 times

- Infinite loops using **while true**:



```
while true {
    print(" ")
    counter += 1

    if counter == 273 {
        break
    }
}
```