

# SWIFT TRAINING - WEEK 2-3

## Strings

### Strings

A string is an ordered collection of characters, such as "We Love Swift" or "eat, sleep, code, repeat!". In Swift strings are represented by the String type which is a collection of values of Character type.

### Creating strings

You can include predefined String values within your code as **string literals**. A **string literal** is a fixed sequence of textual characters surrounded by a pair of double quotes ("").

```
let aString = "Hello"
```

Take note of a special case of string, the empty string.

```
var emptyString = "" // empty string literal
```

You can create a new string by concatenating two strings using the + operator.

```
let newString = "Hello" + " swift lovers" // "Hello swift lovers"
```

String interpolation can also be used to combine strings. By using the \(<value>) syntax inside a string literal.

```
let bat = "BAT"
```

```
let man = "MAN"
```

```
let batman = "\bat)\man)" // "BATMAN" - \bat) will be replaced with "BAT" and \man) with "MAN"
```

```
println("\bat) + \man) = \batman)")
```

```
// "BAT + MAN = BATMAN"
```

### Characters

In some cases you will want to work with the individual characters that make up the string. To do that you can use the for-in syntax. A string exposes the list of characters with the unicodeScalars property.

```
var someString = "this string has 29 characters"
```

```
for scalar in someString.unicodeScalars {
```

```
    // to convert a UnicodeScalar into a String use string interpolation
```

```
    // you will need to do this in order to compare characters
```

```
    var characterAsString = "\scalar)"
```

```
    println(characterAsString)
```

```
}
```

### Counting String Characters

To count the number of characters in a string use the countElements function.

```
var string = "This string has 29 characters"
```

```
println(countElements(string)) // 29
```

**Comparing**

You can compare string using the same operators as numbers.

```
"We Love Swift" == "We Love Swift" // true
```

```
"We Love Swift" == "we Love swift" // false - string comparison is case sensitive
```

```
"We Love Swift" != "We Love Swift" // false
```

```
"We Love Swift" != "we Love swift" // true
```

```
"a" < "b" // true
```

```
"c" > "b" // true
```

```
"b" > "c" // false
```

```
"z" > "hello" // true
```

You can also test if a string has a certain suffix or prefix.

```
let aString = "Hello world!"
```

```
if aString.hasPrefix("Hello") {
    println("\(aString) starts with Hello")
}
```

```
if aString.hasSuffix("!") {
    println("\(aString) ends with !")
}
```

**4.1 Full name**

You are given the `firstName` and `lastName` of a user. Create a string that contains the `fullName`.

```
var firstName = "Andrei"
```

```
var lastName = "Puni"
```

```
// your code here
```

**4.2 Sum**

You are given two numbers `a` and `b`. Compute the sum of `a` and `b` and create a string that contains the sum written like bellow:

For `a = 2` and `b = 5`

```
"2 + 5 = 7"
```

For `a = 12` and `b = 19`

```
"12 + 19 = 31"
```

```
var a = 14
```

```
var b = 23
```

```
// your code here
```

### 4.3 Replace

Replace all the occurrences of the character "e" in aString with "\*".

```
var aString = "Replace the letter e with *"
```

```
var result = ""
```

```
// your code here
```

### 4.4 Reverse

Reverse aString and print the result.

"Hello" -> "olleH"

"We Love Swift" -> "tfiW evoL eW"

```
var aString = "this string has 29 characters"
```

```
// your code here
```

### 4.5 Palindrome

Print true if aString is a palindrome, and false otherwise. A palindrome is a string which reads the same backward or forward.

```
let aString = "anutforajaroftuna"
```

```
// your code here
```

### 4.6 Word Slicer - Split the sentence into words and display them.

```
var problem = "split this string into words and print them on separate lines"
```

```
// your code here
```

## 4.7 Longest Word

**CREATE COMMENT LINES DETAILING HOW THIS WORKS.**

```
var problem = "find the longest word in the problem description"
```

```
// this will help the algorithm see the last word  
problem += " "
```

```
var word = ""  
var length = 0
```

```
var max = 0  
var longestWord = ""
```

```
for character in problem.characters {  
    if character == " " {  
        if length > max {  
            max = length  
            longestWord = word  
        }  
        word = ""  
        length = 0  
    } else {  
        word += "\(character)"  
        length += 1  
    }  
}
```

```
print(longestWord)
```