	**Labo Apps for iOS** Dirk Hostens
Opdracht 1	

Een eerste stap in het programmeren van apps voor het iOS OS is het aanleren van de nieuwe programmeertaal SWIFT. In deze opdracht zullen we in de XCode Playground kennismaken met de IDE en de SWIFT programmeertaal.

In XCode selecteren we File – New - Playground Hier kan je experimenteren om zo de SWIFT taal onder de knie te krijgen. Experimenteer eventueel eerst zelf om zo de syntax en IDE onder de knie te krijgen. Daarna voer je in een nieuwe Playground volgende taken uit.

Deel 1: The basics
https://docs.swift.org/swift-book/documentation/the-swift-programming-language/thebasics

- definieer een variabele text met als waarde “SWIFT”, een Integer constante max met waarde 10 en een Double variabele average met waarde 0. Doe dit zodat je gebruik maakt van **Type inference.**
- print elk karakter van de tekst variabele op een nieuwe lijn (for statement)
- definieer een variabele naam als **Tuple** die voornaam en familienaam verzamelt
    - print voornaam en familienaam op een aparte lijn
- definieer een var reversedString, type String **optional** met als waarde nil.
    - Indien de reversedString nil is dan print je de tekst “no result” en anders de waarde van de variabele. Gebruik hiervoor Optional Binding.
    - Ken nu een waarde toe aan de variabele en test het opnieuw

Deel 2: Functions
https://docs.swift.org/swift-book/documentation/the-swift-programming-language/functions

- definieer een functie reverseString die een String als parameter meekrijgt en als retunwaarde een String heeft. De functie geeft een willigkeurige tekst van achter naar voor terug. Zo is de returnwaarde van reverseString(str: “Hello”) de String “olleH”
    - print het resultaat van de functie reverseString met als parameter “Vives”

    - test het oproepen van de functie wanneer je “nil” als parameter meegeeft. Wat gebeurt er?
        - Los dit op door de String parameter tekst als **optional** te definiëren. Bij een nil waarde return je een lege String.
- Schrijf een functie getUpperLowerCount die een String meekrijgt en als returnwaarde een **tuple** met dezelfde string in hoofdletter (uppercase), in kleine letters (lowercase) en het aantal characters (charCount) teruggeeft.
    - Roep de functie op met de String “iOS 26” en bewaar resultaat in nieuw variable result
    - Print op 3 verschillende lijnen het resultaat in hoofdletter, in kleine letter en het aantal characters
    - Print op een nieuwe lijn de var result. Bekijk het resultaat.
- Schrijf een functie divide die 2 parameter meekrijgt van type Int maar die een double teruggeeft (resultaat van deling).
    - Roep de functie op en print het resultaat op een nieuwe lijn, test ook de deling door 0 en bekijk het resultaat.
- Schrijf een functie calculate die een x aantal parameters (***variadic parameter***) van type Double meekrijgt en als returnwaarde het gemiddeld (avg) teruggeeft, de kleinste (min) waarde, grootste waarde (max) en het aantal elementen (count). Indien geen getallen worden meegegeven als parameter dan geef je één nil waarde terug.
    - Tip: Gebruik … om een 0 of meerdere parameters van een bepaald type mee te geven in een function
      `vb: func calculate(numbers: Int...)`
    - Roep de functie aan met volgende getallen en print het resultaat telkens op een nieuwe lijn
        - `calculate(numbers:10,0,5)`
        - `calculate(numbers:4,5,6,-3)`
        - `calculate(numbers:-3)`
        - `calculate()`
- Definieer 2 variabelen, x met als waarde 10.0 en y met als waarde 3
- Schrijf een functie increment die deze 2 variabelen als parameter meekrijgt en ze met 1 verhoogt. De functie heeft geen returnwaarde. Maak gebruik van **inout** parameters…
    - Roep de functie op met de x en y waarden
    - Print de nieuwe waarden van x en y (11.0 en 4) op een nieuwe lijn.

Deel 3: Error handling

https://docs.swift.org/swift-book/documentation/the-swift-programming-language/errorhandling

- Neem een copy van de functie getUpperLowerCount en maak de parameter een **optional** String. Throw een custom exception (**error handling**) StringConversionError.nilParamater als de parameter een nil waarde heeft en een StringConversionError.emptyParameter als de parameter een lege string is. Gebruik hiervoor **guard**.

- Roep de functie op incl. exception handling voor nil, lege String en “iOS 26” String value. Print het resultaat of bij de exception telkens een aangepaste foutboodschap mee: “Nil value parameter not allowed” & “Empty String parameter not allowed”.

Deel 4: Structures and Classes
https://docs.swift.org/swift-book/documentation/the-swift-programming-language/classesandstructures
https://docs.swift.org/swift-book/documentation/the-swift-programming-language/properties

- Maak een struct Iphone met volgende velden:
  - supplier, een constante met waarde “Apple”
  - type, String
  - dimension met een height en een width
- Defineer een constructor waarbij de dimension height en width variabele 0.0 als waarde krijgen en type een lege String waarde krijgt
- Definieer een constructor waarbij ook de waarden van de dimension variabele en type worden meegegeven als parameter.
  - Maak een object iPhoneAir aan met behulp van de empty constructor
  - Maak een object iPhoneAir2 aan met behulp van de andere constructor en geef volgende waarden mee: hoogte is 15.62, breedte 7.47 en type “iPhone Air”
- Definieer een ENUM met de verschillende iPhone types: iPhoneAir, iPhone17Pro, iPhone17ProMax en iPhone17.
- Pas de code aan zodat Type vervangen wordt door deze ENUM in de klasse iPhone. Maak de variabele type optional zodat bij de empty constructor de variabele type de waarde nil kan krijgen
- Zorg voor een veld description (Computed Property) die een String teruggeeft met een beschrijving van het type. Maak gebruik van een switch statement om de enum type te evalueren.
  - Print de description() van de objecten iPhone Air op een nieuwe lijn

Deel 5: Closures
https://docs.swift.org/swift-book/documentation/the-swift-programming-language/closures

- Definieer een array met volgende elementen: `"Dirk", "Els", "Marc", "Eline", "Dominiek"`
- Bekijk in de Array documentatie de syntax om een Filter Closure functie toe te passen
- Pas een filter toe op de array zodat enkel een array wordt teruggegeven met de namen die beginnen met een “D”. Gebruik hiervoor de Closure function filter.
- Doe nu hetzelfde maar plaats je Closure in een aparte functie `filterArr`. De functie krijgt een String al parameter en een Bool als returnwaarde. Je roept dit dan als volgt aan:

```
var filtered2 = arr.filter(filterArr)
```

- Het nadeel is dat je de waarde waarop je filtert ,“D” in ons geval, niet kan meegeven in je filter functie. Pas je code aan zodat je de filter functie als volgt kunt aanroepen. In vb wordt er gefilterd op alle namen die beginnen met “M”. filterClosure is de functie die je zelf moet schrijven. Deze functie zal dus een functie als returnwaarde hebben.
  ```
  var filtered3 = arr.filter(filterArrayExtended(letter: "E"))
  ```
- Bekijk in de Array documentatie de syntax om een Map Closure functie toe te passen
- Pas dit toe op de array en zorg ervoor dat je een array terugkrijgt met alle elementen in hoofdletters.
- Doe hetzelfde maar creëer een array met de lengte van elke String uit de array gesorteerd van laag naar hoog.

Succes!