**Data types(waardes):**

**“String” =** tekst gebaseerde data aangegeven door het tussen haakjes te zetten “…”

**char(character) =** een tekst gebaseerde waarde die niet per woord maar per letter kijkt.

**Int(Integers) =** voor hele getallen

**Double =** alle getallen en tot 2 decimaal

**Float =** alle getallen tot oneindig decimaal

**Bool (boolean)= heeft** maar twee waardes “waar” of “niet waar”/ 0 of 1

(true of false)

**Constant of variabele:**

**Let:** geeft een constante aan

**Var:** geeft een variabele aan

Let of var geven een datatype aan de verschillende namen waar jij die in wilt bewaren bv: (let “jijMoetNuEchtLeren”(String”))

De waarde van een Let kan niet meer woorden veranderd, bij een variable kan dat later in je code nog wel.

**let maximumNumberOfLoginAttempts = 10**

**var currentLoginAttempt = 0**

Let of var moeten aangegeven worden voor ze gebruikt kunnen worden, je kan ook meerde waardes geven op de zelfde codeLine van een datatype, mits je ze met een komma split.

BV:

Var X = 0**,** Y = 1**,** Z = 2

Als je een var of let aangeeft kan je ook duidelijk maken welke data type je er aan wilt geven, hierdoor kunnen verschillende waardes aangegeven woorden binnen dat type in het zelfde woord, dit doe je door een dubbele punt en het data type in de code te vermelden

BV:

**Var: welcomeMessage: string**

Hierdoor kan je aan welcomeMessage nu een String waarde geven zoals

BV:

**welcomeMessage = “Hey dude”**

String wordt moet wel nog steeds tussen haakjes aangegeven worden.

Alhoewel let niet meer van waarde veranderd kan worden, kan je dit bij var later wel doen door dit

BV:

**Var: welcomeMessage: string**

**welcomeMessage = “Hey dude”**

15 regels later kan je het veranderen

**welcomeMessage = “Hey man”**

**Printing value’s**

Waardes kan je printen door **print** te schrijven gevolgd door je waarde tussen haakjes.

BV:

**Print(welcomeMessage)**

Dit zal nu “Hey man” printen.

**Basic operators:**

Een basic operator is een symbool of zin die je gebruikt om de waardes van een variabele te checken, samen te voegen of te veranderen.

**Lijst van basic operators:**

**+ plus**

**- minus**

**\* keer**

**/ gedeeld door**

**% remainder operator: kijkt naar hoe vaak waarde in waarde past en laat de overgebleven waarde zien. BV**

9 % 4 == 1

want 4 past twee keer in 9 en dan houdt je 1 over

som = 9 = (4 x 2) + 1

**comparison operator:**

(a == b) **gelijk aan**

(a != b) **niet gelijk aan**

(a > b) **groter dan**

(a < b) **minder dan**

(a >= b) **groter dan of gelijk aan**

(a <= b) **minder dan of gelijk aan**

**logical operator:**

gebaseerd op waardes van booleans:

(!a) **niet logisch = keert waardes van Bool om false wordt true en andersom BV:**

let waarheid = false

if !waarheid{

print("ACCESS DENIED")

}// Dit print acces denied

(a && b) **logisch en = deze operator creeerd een som waarbij alle twee de waardes true moeten zijn om de true statement eruit te krijgen, als er eentje false is worden ze allebei false**

**BV:**

let ikWilSlapen = true

let ikHoefNietNaarSchool = false

if ikWilslapen && ikHoefNietNaarSchool {

print("Ga lekker slapen!")

} else {

print("Gast ga is wat doen")

}

// dit print "Gast ga is wat doen")

(a || b) **logisch of = bij deze operator hoeft maar een van de twee waardes true te zijn om de de true statement eruit te krijgen.**

**BV:**

let ikHoefNietNaarSchool = false

let ikWilSlapen = true

if ikHoefNietNaarSchool || ikWilSlapen {

print("gaLekkerSlapen")

} else {

print("Gast ga is wat doen")

}

// dit print "Ga lekker slapen"

**Je kan deze operators ook combineren.**

**BV: (van internet geplukt)**

**Let** enteredDoorCode = false

**Let** passedRetinaScan = false

**Let** hasDoorKey = true

**Let** knowsOverridePassword = true

if enteredDoorCode && passedRetinaScan || hasDoorKey || knowsOverridePassword {

print("Welcome!")

} else {

print("ACCESS DENIED")

}

// Prints "Welcome!"

**In dit voorbeeld gebruikt je meerdere operators zodat je meerdere waardes aan een statement kan hangen. && en || gebruiken beide nog steeds maar twee waardes. Door deze te combineren hangt de statement nu af van drie waardes en niet twee.**

**Je kan deze code op de volgende manier lezen:**

Als we de correcte deur code hebben en door de retinaScan komen of als we de sleutel hebben of de overrideCode hebben, komen we nog steeds binnen.

Zelfs al zijn de eerste false zijn hebben we de andere twee waarde wel waardoor we toch binnen komen.

**String and Characters:**

In swift is String een data type die bestaat uit meerde characters.

Dit type kan gezien worden als String a.k.a een stuk tekst in je code of als een collectie van characters a.k.a meerdere letters die een tekst vormen.

**Syntaxis/syntax:** ook vaak zinsleer genoemd, is een deelgebied binnen de theoretische taalkunde dat in ruime zin de studie omvat van alles wat met de opbouw en structuur van zinsdelen en zinnen te maken heeft, ofwel van de zinsbouw.

De syntax om een String te creëren in Swift is makkelijk en de structuur en waarde kunnnen, mits je het aangeeft als een **var** op elk moment in je code veranderd worden.

Je kan twee Strings zelfs combineren met een operator als +.

Strings kunnen ook gebruikt worden om let, var, literals en expressions in andere (langere) Strings te implementeren, dit heet **String interpolation.**

Dit maakt het makkelijk om meerdere Strings waar je zelf verschillende waarde aan hebt gegeven te printen of in andere String op te slaan.

**String literals:**

Je kan Strings die je een waarde hebt gegeven in je code weergeven als String literal.

Dit doe je door de desbetreffende String tussen aanhalingstekens te zetten (“String”) en het een begin waarde te geven zoals let of var.

**BV zo geef je een String aan met een waarde er aan vast hier een Constante:**

let eenString = "waarde van de String "

**Als je meerdere zinnen in een String wilt zetten als waarde kan dit door de waarde aan te geven met drie aanhalingstekens aan het begin en het eind van de gehele zin(“””)**

**BV:**

let meerdereZinString = “”” Ik ben nu al twee en een half uur bezig. “ik heb al vier grote koffies op en mijn beeld begint een beetje te trillen” misschien heb ik een pauze nodig?”””

**Je kan ook een lege String waarde creëren om later daarmee een langere String mee te bouwen of meerdere waarde erin op te slaan.**

**Dit kan je doen door een variabele te koppelen aan een lege String1 of een nieuwe String syntax initialiseren (nieuwe opslag medium te koppelen aan String data type)2**

1. var simpeleString = “”
2. var syntaxString = String()

**Je kan de eigenschap van een String checken met een Boolean(waarde) met de volgende formule/code .isEmpty:**

if simpeleString.isEmpty {

print("niks te zien")

}

Dit print "niks te zien"

**Function’s**

**// function calling en hoe het werkt**

**// var aanmaken waar je waarde van rsultaat in called**

var y: Double = 0

**// func naam(parameter:type, parameter:type) -> returnType**

func math(X: Double,A: Double) -> Double {

**// code van je func: nu een gemakkelijke som**

y = A\*X

**// altijd een return nodig om het resultaat in de returnType te zien**

return y

}

//print(math(X: 1, A: 3))

**// het toevoegen van vrschillende variabele**

var Z: Double = 5

var y1: Double = 0

var r: Double = 2

var s: Double = 3

**// de verschillende variable door de bovengenoemde functie callen**

y1 = math(X:r, A:s)

**// en dan uitprinten**

print(Z\*y1)

**array’s and dictionaires**

**Logboek: swift definitions**

**Instance:** an occurence of a class or object

**Syntax:** de juiste grammatica die je moet gbruiken om tegen de computer te kunnen praten.

**Init/initialization:** het initialiseren van programma’s is nodig om