פונקציות

בשיעור זה נכיר

- פונקציות בשפת Swift - תחביר ושימוש

פונקציות

ראשית, הגדרה.

פונקציה היא קטע קוד מוגדר, שאפשר לקרוא לביצועו על ידי קטע קוד.

ניתן אף לתת שם לפונקציה

פונקציות - דוגמאות

נראה דוגמאות של פונקציות

דוגמא לפונקציה שאינה מקבלת אף ארגומנט ואינה מחזירה ערך.

```
func doNothing(){
    //doing nothing
}
```

והנה עוד אחת

```
func doSomething(){
    var n1 = 4;
    var n2 = 3;
    println("\(n1) + \(n2) = \(n1+n2)")
}
```

נקרא לפונקציות

```
doNothing()
doSomething()
```

פונקציות - דוגמאות

פונקציות שמקבלות ארגומנט, וקריאה להן.

```
func getArgumnetFunc(num : Int){
    println("hey, I have this num argumnet:\(num)")
}

getArgumnetFunc(4)
```

```
func sayHelloFunc(name : String){
    println("hello \(name)")
}

sayHelloFunc("Class")
```

פונקציות - דוגמאות

```
func giveMeRandomNumber() -> Int{
    var randNum : Int = Int(rand())
    return randNum
}

var randomNumber : Int = giveMeRandomNumber()
```

פונקציות שמחזירות ערך.

```
func isTodaySunday() -> Bool{
    var today : NSDate = NSDate()
    var calendar : NSCalendar = NSCalendar(calendarIdentifier: NSGregorianCalendar)
    var dateComponents : NSDateComponents! = calendar.components(NSCalendarUnit.CalendarUnitWeekday, fromDate: today)
    var day : Int = dateComponents.weekday
    return day==1
}

if isTodaySunday(){
    println("today is sunday")
} else {
    println("today isn't sunday")
}
```

פונקציות - דוגמאות

```
func checkIfNumberGreateThen5(num : Int) -> Bool{
    return num > 5
}

var num : Int = 6
var result : Bool = checkIfNumberGreateThen5(num)
if result{
    println("\(num) is greater then 5")
} else {
    println("\(num) isn't greater then 5")
}
```

פונקציות שמקבלות
ארגומנט ומחזירות ערך.

```
func powByTen(num : Int) -> Int{
    var result = 1
    for _ in 1..10{
        result *= num
    }
    return result
}

println("10 pow of 2 is \(powByTen(2))")
```

פונקציות - דוגמאות

```
func sayHelloToPerson(firstname s1:String,lastname s2 : String){
    println("hello \(s1) of the known and respected family \(s2)")
}

sayHelloToPerson(firstname: "Benny", lastname: "Davidovitz")
```

פונקציות שמקבלות יותר מארגומנט יחיד.

```
func getAvarage(arg1 : Float, arg2 : Float) -> Float{
    return (arg1 + arg2)/2
}

let arg1 = rand()
let arg2 = rand()

println("avarage of \(arg1) and \(arg2) is \(getAvarage(Float(arg1),Float(arg2)))")
```

פונקציות - דוגמאות

המשך פונקציות שמקבלות יותר מארגומנט יחיד.

```
func isPitagorized(a : Float, b : Float, c : Float) -> String{
    if (powf(a,2) + powf(b,2)) == powf(c,2){
        return "pitagorized"
    } else {
        return "not pitagorized"
    }
}

let n1 : Float = 4.5
let n2 : Float = 6.5
let n3 : Float = 9

var result : String = isPitagorized(n1, n2, n3)
```

פונקציות - דוגמאות

פונקציות המקבלות כמות בלתי מוגבלת של ארגומנטים

```
func findAvarageOfNumber(#numbers : Int...) -> Float{
    var sum : Int = 0
    for item in numbers{
        sum += item
    }
    var avarage = Float(sum)/Float(numbers.count);
    return avarage
}

let n1 = 3
let n2 = 45
let n3 = 76
let n4 = -100

var avg = findAvarageOfNumber(numbers: n1,n2,n3,n4)
```

הערת ביניים - tuple

```
let card1 = (13,"King")

var card1Value = card1.0
var card1Name = card1.1

let card2 = (value : 12,name : "Queen")

var card2Value = card2.value
card2Value = card2.0

var card2Name = card2.name
card2Name = card2.name
```

Tuple הוא אובייקט שמכיל יותר מאובייקט אחד
על כן הוא למעשה data-collection
אפשר להתייחס אליו כמערך, ואפשר להתייחס כמילון

כתוצאה מכך
פונקציה תוכל להחזיר יותר מערך יחיד כתשובה.
הפונקציה פשוט מחזירה tuple
בדיוק כמו שניתן להחזיר מערך או מילון...

והנה כמה דוגמאות ל tuple ולהתייחסות לערכיו

פונקציות - דוגמאות

פונקציות המחזירות יותר מערך אחד

```
func getFloorAndTop(num : Float) -> (floor : Int, top : Int){
    var intVal = Int(num)
    if (num >= 0){
        var floor = intVal
        var top = intVal + 1
        return (floor,top)
    } else {
        var top = intVal
        var floor = top-1
        return (floor, top)
    }
}

let num : Float = -3.5
let result = getFloorAndTop(num)
println("\(num) floor is \(result.floor) and top is \(result.top)")
```

פונקציות - דוגמאות

קצת רקורסיה...

מה היא רקורסיה?

פונקציה הקוראת לעצמה שוב ושוב עד לתנאי עצירה כלשהוא.
כל רקורסיה ניתן לכתוב באמצעות לולאה ולהפך, הוכח מתמטית.

דוגמת המחשה : מיהו יהודי, פרק 12

תרגיל: סדרת פיבונצ׳י

פונקציות - דוגמאות

קצת רקורסיה...

```
func howManyPIs(var num : Double) -> Int{
    let pi = M_PI
    if (num < pi){
        return 0
    }

    return 1 + howManyPIs(num-pi)
}

let someNumber = Double(rand())
let count = howManyPIs(someNumber)
```

פונקציות - דוגמאות

פונקציה המחזירות פונקציה

```
class Student{
    var level : Int
    init(){
        level = 0
    }

    func getCheckFunc() -> (Int -> Bool){
        var targetGrade = 0
        switch level{
        case 0:
            targetGrade = 60
        case 1:
            targetGrade = 70
        case 2:
            targetGrade = 80
        default:
            targetGrade = 56
        }


        func checkFunc(#score : Int) -> Bool{
            return (score > targetGrade)
        }
        return checkFunc;

    }
}
```

```
var student = Student()
student.level = 2
var checkFucntion = student.getCheckFunc()
var didPass = checkFucntion(85)
```

פונקציות - דוגמאות

פונקציה המקבלת פונקציה

```
func norAction(b1 : Bool, b2: Bool) -> Bool{
    return !(b1 || b2)
}

func nandAction(b1 : Bool, b2: Bool) -> Bool{
    return !(b1 && b2)
}

func performBinaryAction(action : (Bool, Bool) -> Bool, b1 : Bool, b2 : Bool) -> Bool{
    return action(b1,b2)
}

var answer = performBinaryAction(nandAction, true, false)
```

פונקציות - דוגמאות

פונקציה המקבלת פונקציה - מיון מערך

```
func comparator(n1 : Int, n2 : Int) -> Bool{
    return n1 < n2
}

var numbersArray = [5,2123,2435,231,123,56,33,999,1000,0,5000,3456]
var sortedArray = sort(numbersArray,comparator)
```

1

```
func stringComperator(s1 : String, s2 : String) -> Bool{
    return s1 < s2
}

var namesArray = ["Zelda","Ofra","Zehava","Rachel","Nancy","Rotem","Michal"]
namesArray = sort(namesArray,stringComperator)
```

2

```
var namesArray = ["Zelda","Ofra","Zehava","Rachel","Nancy","Rotem","Michal"]
namesArray = sort(namesArray, >)
```

3

פונקציות - דוגמאות

שינוי ערך חיצוני לפונקציה

```
func appendPrefix(inout str : String) -> Void{
    str = "Simon Said: " + str
}

var string : String = "wake up!"
appendPrefix(&string)
string
```

ואיך אפשר בלי swap הקלאסי

```
func mySwap(inout #num1 : Int, inout #num2 : Int){
    num1 = num1 + num2
    num2 = num1 - num2
    num1 = num1 - num2
}

var number1 = 3
var number2 = 5
mySwap(num1: &number1, num2: &number2)
```

פונקציות - דוגמאות

```
func doSomething(){
    func doNothing(){
        //doing nothing
    }

    for _ in 1..100{
        doNothing()
    }

    println("I'm actually doing nothing 100 times")
}
```

פונקציה המוגדרת בתוך פונקציה

פונקציות - סיכום

לסיכומו של עניין

מגוון הערכים שפונקציה יכולה לקבל ולהחזיר או רחב
כל מבנה של פונקציה נדרש על פי העניין

בעת שיעלה הצורך ביצירת פונקציה, חשבו היטב מהו המבנה הנכון ביותר, הגנרי ביותר, והניתן לשימוש חוזר ביותר.
ובו תשתמשו.

וכמובן, לתרגל, לתרגל, ושוב לתרגל.