**44643 Mobile Computing-iOS**

**Note: while invoking the function, the parameter should be given using the argument label that is given while defining the function.**

1. **func greetUser(){**

**print("Welcome User")**

**}**

**Function Call:** **greetUser()**

1. **func sayHello()->String{**

**var name="Swift"**

**return "Hello "+name+"!"**

**}**

**print(sayHello())**

1. **func favoriteProgram(name:String)->String {**

**var name="My favorite programming language is \(name)"**

**return name**

**}**

**print(favoriteProgram(name:"Java"))**

1. **func addNumbers(number1:Int,number2:Int)->Int{**

**return number1+number2**

**}**

**print("The sum of given numbers is \(addNumbers(number1: 10, number2: 20))")**

1. **func manipulateNumber(input: Int,mode:Bool)->String{**

**var counter=input**

**if(mode){**

**counter+=2**

**}else{**

**counter-=1**

**}**

**return "The \(mode ? "incremented" : "decremented" ) value is \(counter)"**

**}**

**var val=19**

**print(manipulateNumber(input:val,mode: false))**

1. **func login(sid username:String, password:String)->Bool{**

**if(username=="swift5.5" && password=="uikit") {**

**return true**

**}**

**return false;**

**}**

**var loggedIn:Bool = login(sid: "swift5.5", password: "uiki")**

**if loggedIn {**

**print("User login success")**

**}**

**else**

**{**

**print("Username or Password is incorrect")**

**}**

Note: The use of argument labels can allow a function to be called in an expressive, sentence-like manner, while still providing a function body that’s readable and clear in intent.

Here **sid** is the argument label and can be used when function is invoked instead of Parameter Name(username)

1. **func fullName(\_ firstName:String,\_ lastName:String)->String{**

**return lastName+","+firstName**

**}**

**print(fullName("Susan","Connar"))**

1. **func sumAndAvg(\_ numbers:Double...)->(sum:Double,avg:Double){**

**var total:Double=0**

**var avg:Double**

**for number in numbers {**

**total+=number**

**}**

**avg=total/Double(numbers.count)**

**return (total,avg)**

**}**

**let result=sumAndAvg(1,20.5,3,4.9,10)**

**print("Sum = \(result.sum)")**

**print("Average = \(result.avg)")**

1. **func calculate(symbol: String) -> (Int, Int) -> Int {**

**// inner function to add two numbers**

**func add(num1:Int, num2:Int) -> Int {**

**return num1 + num2**

**}**

**// inner function to subtract two numbers**

**func subtract(num1:Int, num2:Int) -> Int {**

**return num1 - num2**

**}**

**let operation = (symbol == "+") ? add : subtract**

**return operation**

**}**

**let operation = calculate(symbol: "+")**

**let result = operation(10, 7)**

**print("Result:", result)**

1. **func nextLeapYear(\_ year:inout Int)->Int {**

**year+=1**

**if ((year % 4 == 0) && (year % 100 != 0)) ||**

**(year % 400 == 0){**

**return year**

**}**

**else{**

**nextLeapYear(&year)**

**}**

**return year;**

**}**

**var year=2000**

**print("Upcoming leap year is: \(nextLeapYear(&year))")**