实验编号： 二 **四川师大iOS高级开发技术 实验报告 2018** 年 **10**月 **5** 日

**计算机科学学院**2016 级4班 实验名称：  **iOS实验** \_

姓名：\_樊琳\_\_ 学号：2016110408 指导老师：李贵洋 实验成绩:\_\_\_\_\_

**实验二\_ \_\_\_ iOS实验\_\_\_\_**

1. 实验目的及要求

熟练掌握闭包,扩展,泛型, 枚举,类,派生,协议的内容

1. 实验内容

作业1（闭包、扩展、泛型）：

1. 给定一个Dictionary，Dictionary包含key值name和age，用map函数返回age字符串数组;
2. 给定一个String数组，用filter函数选出能被转成Int的字符串
3. 用reduce函数把String数组中元素连接成一个字符串，以逗号分隔
4. 用 reduce 方法一次求出整数数组的最大值、最小值、总数和
5. 新建一个函数数组，函数数组里面保存了不同函数类型的函数，要求从数组里找出参数为一个整数，返回值为一个整数的所有函数；
6. 扩展Int，增加sqrt方法，可以计算Int的Sqrt值并返回浮点数，进行验证；
7. 实现一个支持泛型的函数，该函数接受任意个变量并返回最大和最小值，分别传入整数值、浮点数值、字符串进行验证。
8. 掌握版本控制git的进阶使用
   1. git reset 恢复到之前修改的版本；
   2. git log 看提交记录；
   3. git branch 新建分支；
   4. git checkout 切换分支；
   5. git branch -d 删除分支；

作业2:（枚举、类、派生、协议）(红色字体为新增内容)

1. 实现Person类：
   1. 要求具有firstName, lastName，age，gender等存储属性,fullName计算属性；其中gender是枚举类型（male，female）；
   2. 具有指定构造函数和便利构造函数；
   3. 两个Person实例对象可以用==和!=进行比较；
   4. Person实例可以直接用print输出；
   5. Person增加run方法(方法里面直接print输出Person XXX is running;
2. 从Person分别派生Teacher类和Student类：
   1. Teacher类增加属性title，实例可以直接用print输出；
   2. Student类增加属性stuNo，实例可以直接用print输出；
   3. Teacher和Student重载run方法(方法里面直接print输出Teacher XXX is running和Student XXX is running)
3. 分别构造多个Person、Teacher和Student对象，并将这些对象存入同一个数组中；
4. 新建一个协议SchoolProtocol，协议包括一个department属性(Enum，自己实现enum的定义)和lendBook方法（随便写点内容，能区隔即可）；
5. 修改Teacher和Student，让这两个类实现该协议；
6. 对数组执行以下要求：
   1. 分别统计Person、Teacher和Student对象的个数并放入一字典中，统计完后输出字典内容；
   2. 对数组按以下要求排序并输出：age、fullName、gender+age；
   3. 对数组进行穷举，调用每个对象的run方法，同时调用满足协议SchoolProtocol对象的lendBook方法；
7. 实验主要流程、基本操作或核心代码、算法片段（该部分如不够填写，请另加附页）

作业1（闭包、扩展、泛型）：

1. 给定一个Dictionary，Dictionary包含key值name和age，用map函数返回age字符串数组;
2. 给定一个String数组，用filter函数选出能被转成Int的字符串
3. 用reduce函数把String数组中元素连接成一个字符串，以逗号分隔
4. 用 reduce 方法一次求出整数数组的最大值、最小值、总数和
5. 新建一个函数数组，函数数组里面保存了不同函数类型的函数，要求从数组里找出参数为一个整数，返回值为一个整数的所有函数；
6. 扩展Int，增加sqrt方法，可以计算Int的Sqrt值并返回浮点数，进行验证；
7. 实现一个支持泛型的函数，该函数接受任意个变量并返回最大和最小值，分别传入整数值、浮点数值、字符串进行验证。
8. 掌握版本控制git的进阶使用
   1. git reset 恢复到之前修改的版本；
   2. git log 看提交记录；
   3. git branch 新建分支；
   4. git checkout 切换分支；
   5. git branch -d 删除分支；

**作业一**

1.

var arr=["fanlin":18,"xhh":17]

var arr2 = arr.map({"\($0.value)"})

print("arr = \(arr)")

print("arr2 = \(arr2)")

2.

var arr=["1","5","ds","8","."]

var arr2 = arr.flatMap({Int($0)})

print("arr = \(arr)")

print("arr2 = \(arr2)")

3.

var stringArray=["1","2","3"]

stringArray.reduce("", {(string1, string2) -> String in

return string1 == "" ? string2 : string1 + "," + string2

})

4.

var arr=[3,5,2,7,7,9]

var arr2=arr.reduce(0,{max($0,$1)})

print(arr2)

var arr3=arr.reduce(0,{min($0,$1)})

print(arr3)

var arr4=arr.reduce(0,+)

print(arr4)

5.

6.

#if os(Linux)

import Glibc

#else

import Darwin

#endif

extension Int{

mutating func Sqrt() {

print("the number after sqrt is \(sqrt(Double(self)))")

}

}

var test = 4

test.Sqrt()

7.

func minAndmax<T:Comparable>(a: T, b: T) -> T {

return max(a,b)

}

print(minAndmax(a:1, b:2))

print(minAndmax(a:2.50, b:2.50))

print(minAndmax(a:"abcd", b:"abcde"))

作业2:（枚举、类、派生、协议）(红色字体为新增内容)

1. 实现Person类：
   1. 要求具有firstName, lastName，age，gender等存储属性,fullName计算属性；其中gender是枚举类型（male，female）；
   2. 具有指定构造函数和便利构造函数；
   3. 两个Person实例对象可以用==和!=进行比较；
   4. Person实例可以直接用print输出；
   5. Person增加run方法(方法里面直接print输出Person XXX is running;
2. 从Person分别派生Teacher类和Student类：
   1. Teacher类增加属性title，实例可以直接用print输出；
   2. Student类增加属性stuNo，实例可以直接用print输出；
   3. Teacher和Student重载run方法(方法里面直接print输出Teacher XXX is running和Student XXX is running)
3. 分别构造多个Person、Teacher和Student对象，并将这些对象存入同一个数组中；
4. 新建一个协议SchoolProtocol，协议包括一个department属性(Enum，自己实现enum的定义)和lendBook方法（随便写点内容，能区隔即可）；
5. 修改Teacher和Student，让这两个类实现该协议；
6. 对数组执行以下要求：
   1. 分别统计Person、Teacher和Student对象的个数并放入一字典中，统计完后输出字典内容；
   2. 对数组按以下要求排序并输出：age、fullName、gender+age；
   3. 对数组进行穷举，调用每个对象的run方法，同时调用满足协议SchoolProtocol对象的lendBook方法；

**作业二**

import Foundation

import Glibc

enum Gender{

case male

case female

}

enum credit

{

case good

case normal

case bad

}

protocol SchoolProtocol

{

var department : credit{get set}

func lendBook() -> Bool

}

class Person {

var firstName: String

var lastName: String

var age: Int

var gender: Gender

var fullName: String{ return firstName + lastName }

init(firstName: String, lastName: String, age: Int, gender: Gender){

self.firstName = firstName

self.lastName = lastName

self.age = age

self.gender = gender

}

convenience init(firstName: String){

self.init(firstName: firstName, lastName: "", age: 43, gender: Gender.male)

}

func description() -> String{

return "name:\(fullName) age:\(age) gender:\(gender)"

}

static func ==(x:Person,y:Person) ->Bool{

return x.fullName==y.fullName&&x.age==y.age&&x.gender==y.gender

}

static func != (x:Person,y:Person) ->Bool

{

return !(x == y)

}

func run()

{

print("Person \(firstName+lastName) is running;")

}

}

class Teacher: Person,SchoolProtocol{

var department : credit

var title : String

init(title: String, firstName: String, lastName: String, age: Int, gender: Gender,d:credit){

self.title = title

self.department=d

super.init(firstName: firstName, lastName: lastName, age: age, gender: gender)

}

override func description() -> String{

return "title:\(title) name:\(fullName) age:\(age) gender:\(gender) department:\(department)"

}

override func run()

{

print("Teacher \(firstName+lastName) is running")

}

func lendBook() -> Bool

{

if(self.department == credit.good || self.department == credit.normal)

{

return true;

}else{

return false;

}

}

}

class Student: Person,SchoolProtocol{

var department : credit

var stuNo: String

init(stuNo:String, firstName: String,

lastName: String, age: Int, gender: Gender,d:credit){

self.department=d

self.stuNo = stuNo

super.init(firstName: firstName, lastName: lastName, age: age, gender: gender)

}

override func description() -> String{

return "stuNo:\(stuNo) name:\(fullName) age:\(age) gender:\(gender) department:\(department)"

}

override func run()

{

print("Student \(firstName+lastName) is running")

}

func lendBook() -> Bool

{

if(self.department == credit.good || self.department == credit.normal)

{

return true;

}else{

return false;

}

}

}

var person1 = Person(firstName: "Li", lastName: " guiyang", age: 18, gender: Gender.female)

print(person1.description())

var person2 = Person(firstName: "gao")

print(person2.description())

var teacher = Teacher(title: "Math",firstName: "1", lastName: "2", age: 23, gender: Gender.male)

print(teacher.description())

var student = Student(stuNo:"2016110001", firstName: "han",lastName: "meimei", age: 32, gender: Gender.female)

print(student.description())

var teacher1 = Teacher(title:"Chinese",firstName:"jiang ",lastName:"hong",age:19,gender:Gender.female)

print(teacher1.description())

var student1 = Student(stuNo: "2016110002",firstName: "Wang ",lastName: "qian", age: 20,gender: Gender.male)

print(student1.description())

var arry = [person1,person2,teacher,student,teacher1,student1]

设置了协议后的构造：

var person1 = Person(firstName: "Li", lastName: " guiyang", age: 18, gender: Gender.female)

print(person1.description())

var person2 = Person(firstName: "gao")

print(person2.description())

var teacher = Teacher(title: "Math",firstName: "1", lastName: "2", age: 23, gender: Gender.male,d:credit.good)

print(teacher.description())

var student = Student(stuNo:"2016110001", firstName: "han",lastName: "meimei", age: 32, gender: Gender.female,d:credit.good)

print(student.description())

var teacher1 = Teacher(title:"Chinese",firstName:"jiang ",lastName:"hong",age:19,gender:Gender.female,d:credit.good)

print(teacher1.description())

var student1 = Student(stuNo: "2016110002",firstName: "Wang ",lastName: "qian", age: 20,gender: Gender.male,d:credit.good)

print(student1.description())

var arry = [person1,person2,teacher,student,teacher1,student1]

6

var person1 = Person(firstName: "Li", lastName: " guiyang", age: 18, gender: Gender.female)

var person2 = Person(firstName: "gao")

var teacher = Teacher(title: "Math",firstName: "1", lastName: "2", age: 23, gender: Gender.male,d:credit.good)

var student = Student(stuNo:"2016110001", firstName: "han",lastName: "meimei", age: 32, gender: Gender.female,d:credit.good)

var teacher1 = Teacher(title:"Chinese",firstName:"jiang ",lastName:"hong",age:19,gender:Gender.female,d:credit.good)

var student1 = Student(stuNo: "2016110002",firstName: "Wang ",lastName: "qian", age: 20,gender: Gender.male,d:credit.good)

var arry = [person1,person2,teacher,student,teacher1,student1]

var number:Dictionary<String,Int>=["Person": 0,"Student": 0,"Teacher": 0]

for i in arry

{

if i is Student {

number["Student"]!=number["Student"]!+1

}

else if i is Teacher {

number["Teacher"]!=1+number["Teacher"]!

}

else

{

number["Person"]!=1+number["Person"]!

}

}

print(number)

6.b

arry.sort(){$0.age>$1.age}

print(arry)

arry.sort(){$0.fullName>$1.fullName}

print(arry)

func compareGender(x:Gender,y:Gender) ->Bool

{

return x.hashValue > y.hashValue

}

func compare(x:Person,y:Person) ->Bool {

return compareGender(x:x.gender,y:y.gender) && x.age>y.age

}

arry.sort(by:compare)

print(arry)

6.c

for obj in arry

{

obj.run()

if let teacher = item as? Teacher {

teacher.lendBook()

} else if let student = item as? Student {

student.lendBook()

}

1. 实验结果的分析与评价（该部分如不够填写，请另加附页）

这次实验对我们的要求是熟练掌握闭包,扩展,泛型, 枚举,类,派生,协议的内容.在这些内容里面拓展和协议对于我们来说是新的内容,在以前的学习中我们也没有学习过类似的内容.需要更加努力学习这方面的知识.闭包在上一次的实验中也有涉及到,但是这次是关于那三个函数的运用.

注：实验成绩等级分为（90－100分）优，（80－89分）良，(70-79分)中，（60－69分）及格，（59分）不及格。