实验编号： 1 **四川师大《IOS》实验报告 2018** 年 **9** 月 **5** 日

### **计算机科学学院** 级 4 班 实验名称： Swift类和子类 \_

姓名： 郭周倩 学号： 2016110413 指导老师：\_\_李贵洋\_\_ 实验成绩:\_\_\_\_\_

**实验 一 Swift类和子类\_\_\_\_\_\_\_\_**

1. 实验目的及要求
2. 用 Swift语言编写类和子类
3. 掌握类和子类的定义及使用
4. 掌握git的基本使用命令
5. 实验要求
6. 认真填写实验报告，要求附加部分运行界面和主要代码；
7. 对设计好的程序，检查输出是否符合预期，如有错请分析错误原因并解决；
8. 实验内容

* 作业1:（重点：闭包）

1. 找出从1-10000的所有质数，将找出的质数存入可变数组，对数组进行正反排序并输出结果(采用5种以上的方法实现)；
2. 掌握版本控制git的基本使用
   1. git init 初始化版本库
   2. git add . 添加当前目录下的所有文件到暂存区
   3. git commit -m 提交暂存区的文件到版本库

* 作业2:（枚举、类、派生）

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；
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. 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. 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；
5. 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:（重点：闭包）

1. 找出从1-10000的所有质数，将找出的质数存入可变数组，对数组进行正反排序并输出结果(采用5种以上的方法实现)；

* 程序代码：

var primes = [Int]()

for i in 2...10000{

var isPrime = true

for j in 2..<i{

if i % j == 0{

isPrime = false

}

}

if isPrime {

primes.append(i)

}

}

primes.sort() //直接排序

print(primes) //1、编写函数来逆序输出

func compare1(one:Int,two:Int) -> Bool{

return one > two

}

primes.sort(by: compare1(one:two:)) //1.调用函数

primes.sort(by: {(one:Int,two:Int)->(Bool) in return one>two}) //2、匿名闭包

primes.sort(by: {(one,two)->(Bool) in return one>two}) //3、匿名闭包

primes.sort(by: { $0 > $1}) //4、匿名闭包

primes.sort (by: > ) //5、匿名闭包

primes.sorted(); //建立新数组排序

print(primes)

* 运行结果：

1. 掌握版本控制git的基本使用
   1. git init 初始化版本库
   2. git add . 添加当前目录下的所有文件到暂存区
   3. git commit -m 提交暂存区的文件到版本库

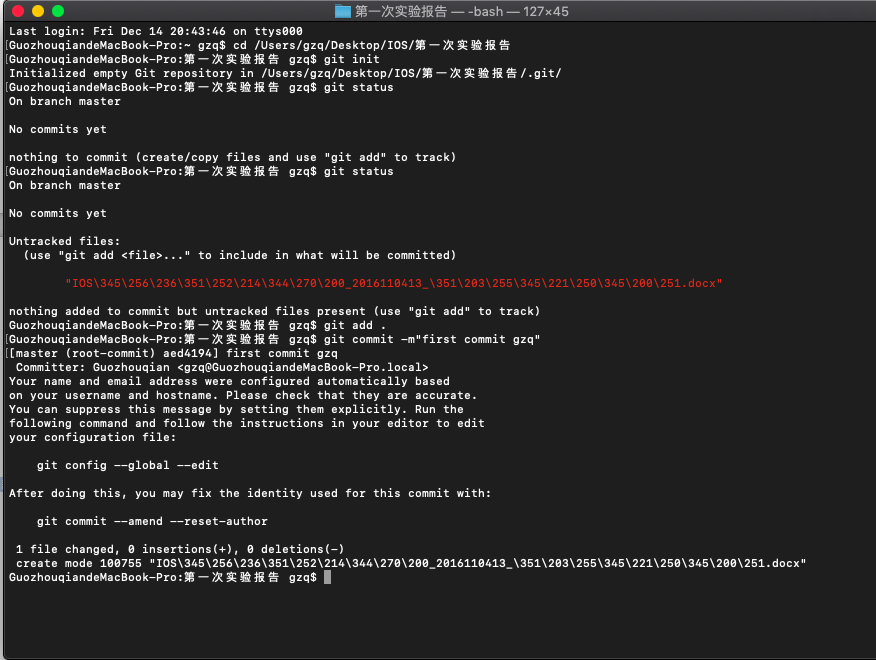
* 程序代码：

git init

git add .

git commit -m “first commit gzq”

* 运行结果：



* 作业2:（枚举、类、派生）

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；

* 程序代码：

enum Gender:String{

case female

case male

}

class Person{

//存储属性

var firstName : String

var lastName : String

var age : Int

var gender:Gender

//计算属性

var fullName : String{

get{

return firstName+""+lastName

}

}

//指定构造函数

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

self.firstName = firstName

self.lastName=lastName

self.age = age

self.gender = gender

}

//便利构造函数

convenience init(name:String){

self.init(firstName:name,lastName:"",age:18,gender:Gender.female)

}

var description:String{

return"Name:\(fullName) Age:\(age) Gender:\(gender)"

}

}

let p1 = Person(firstName: "guo", lastName: "zhouqian", age: 20, gender: Gender.female)// 使用指定构造函数构造的对象

let p2 = Person(name: "guozhouqian")//使用便利构造函数构造的对象

print(p1.description)

print(p2.description)

//重新方法如下：

func == (left:Person,right:Person)->Bool{

return left.firstName==right.firstName && left.lastName==right.lastName && left.age == right.age && left.gender == right.gender

}

func != (left:Person,right:Person)->Bool{

return left.firstName != right.firstName || left.lastName != right.lastName || left.age != right.age || left.gender != right.gender

}

//比较一

if p1 == p2 {

print("两个对象比较相等")

}else{

print("两个对象比较不相等")

}

//比较二

if p1 != p1 {

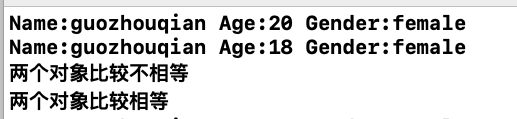
print("两个对象比较不相等")

}else{

print("两个对象比较相等")

}

* 运行结果：



1. 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；
2. 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；
3. 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；

* 程序代码：

class Student:Person{

var stuNo:String

//指定构造函数

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

self.stuNo = stuNo

super.init(firstName: firstName, lastName: lastName, age: age, gender: gender)//调用的是指定构造函数

}

//便利构造函数

convenience init(name:String){

self.init(firstName: name,lastName: "",age: 18,gender: Gender.female,stuNo: "001")

}

override var description: String{

return super.description + " stuNo:\(stuNo)"

}

}

class Teacher:Person{

var title:String

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

self.title = title

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

}

convenience init(name:String){

self.init(firstName: name, lastName: "", age: 18, gender: Gender.female, title: "teacher")

}

override var description: String{

return super.description+" title:\(title)"

}

}

let s1 = Student(firstName: "guo", lastName: "zhouqian", age: 20, gender: Gender.female, stuNo: "013")

let s2 = Student(name: "guozhouqian")

print(s1.description)

print(s2.description)

let t1 = Teacher(firstName: "li", lastName: "guiyang", age: 40, gender: Gender.male, title: "headTeacher")

let t2 = Teacher(name: "liaoxuehua")

print(t1.description)

print(t2.description)

var persons = [Person]()

persons.append(p1)

persons.append(p2)

persons.append(s1)

persons.append(s2)

persons.append(t1)

persons.append(t2)

var stuNum = 0

var teaNum = 0

var perNum = 0

for i in 0..<persons.count{

if persons[i] is Student{

stuNum += 1

}else if persons[i] is Teacher{

teaNum += 1

}else{

perNum += 1

}

}

//字典

var perDic: [String:Int] = ["Person":perNum,"Teacher":teaNum,"Student":stuNum]

for (key, value) in perDic {

print("字典 key \(key) - 字典 value \(value)")

}

//数组排序

persons.sort(by: { $0.age > $1.age}) //age排序

persons.sort(by: { $0.fullName > $1.fullName}) //fullName排序

//persons.sort(by: {$0.age > $1.age && $0.gender.rawValue > $1.gender.rawValue }) //gender+age排序

persons.sort { //gender+age排序

if $0.age == $1.age{

return $0.gender.rawValue > $1.gender.rawValue

}

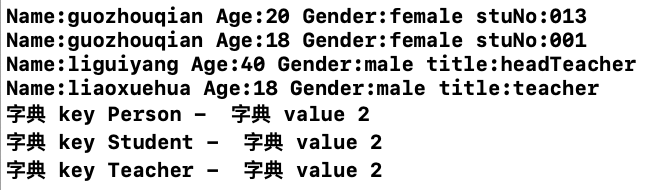
else{

return $0.age > $1.age

}

}

* 运行结果：



1. 实验结果的分析与评价（该部分如不够填写，请另加附页）

文件上传GitHub链接：<https://github.com/gzqxyz/swift-test>

由于没有苹果机，完成实验内容就显得较为麻烦，上课的时间不够完成作业，只有在网上在线编辑器上面进行编写而且上传到个GitHub上也并没有使用命令语句上传，是自己创建到仓库，然后将在电脑上的编辑好的文件之间拖拽到上传位置，也可以进行修改和版本的对比，但是并没有实现老师期望的功能，可能只能当作是一个存储工具，希望以后听老师讲解以后能够正真的将GitHub运用起来。

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。