窗体顶端

enum Gender:Int{

case male

case female

case unknow

//重载>操作符，方便后面排序使用

static func >(lhs: Gender, rhs: Gender) -> Bool {

return lhs.rawValue < rhs.rawValue

}

}

class Person: CustomStringConvertible {

var firstName: String

var lastName: String

var age: Int

var gender: Gender

var fullName: String {

get {

return firstName + lastName

}

}

//构造方法

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, age: Int, gender: Gender) {

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

}

convenience init(firstName: String) {

self.init(firstName: firstName, age: 0, gender: Gender.unknow)

}

required convenience init() {

self.init(firstName: "")

}

//重载==

static func ==(lhs: Person, rhs: Person) -> Bool {

return lhs.fullName == rhs.fullName && lhs.age == rhs.age && lhs.gender == rhs.gender

}

//重载!=

static func !=(lhs: Person, rhs: Person) -> Bool {

return !(lhs == rhs)

}

//实现CustomStringConvertible协议中的计算属性，可以使用print直接输出对象内容

var description: String {

return "fullName: \(self.fullName), age: \(self.age), gender: \(self.gender)"

}

}

var p1 = Person(firstName: "彭")

var p2 = Person(firstName: "彭", age: 20, gender: .female)

print(p1)

print(p1 == p2)

print(p1 != p2)

//教师类

class Teacher: Person {

var title: String //标题

//构造方法

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

self.title = title

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

}

init(title: String) {

self.title = title

super.init(firstName: "", lastName: "", age: 0, gender: .unknow)

}

convenience required init() {

self.init(title: "")

}

//重写父类的计算属性

override var description: String {

return "title: \(self.title), fullName: \(self.fullName), age: \(self.age), gender: \(self.gender)"

}

}

var t1 = Teacher(title: "hello")

print(t1) //输出title: hello, fullName: , age: 0, gender: unknow

//学生类

class Student: Person {

var stuNo: Int //学号

//构造方法

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

self.stuNo = stuNo

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

}

init(stuNo: Int) {

self.stuNo = stuNo

super.init(firstName: "", lastName: "", age: 0, gender: Gender.unknow)

}

required convenience init() {

self.init(stuNo: 0)

}

//重写父类的计算属性

override var description: String {

return "stuNo: \(self.stuNo), fullName: \(self.fullName), age: \(self.age), gender: \(self.gender)"

}

}

var s1 = Student(stuNo: 2016110328)

print(s1)

//初始化一个空的Person数组

var array = [Person]()

//生成5个Person对象

for i in 1...5 {

let temp = Person(firstName: "彭", lastName: "\(i)", age: 20, gender: .male)

array.append(temp)

}

//生成3个Teacher对象

for i in 1...3 {

let temp = Teacher(title: "hello", firstName: "陈", lastName: "\(i)", age: 21, gender: .male)

array.append(temp)

}

//生成4个Student对象

for i in 1..<5 {

let temp = Student(stuNo: 2015110100 + i, firstName: "王", lastName: "\(i)", age: 19, gender: .male)

array.append(temp)

}

//定义一个字典，用于统计每个类的对象个数

var dict = ["Person": 0, "Teacher": 0, "Student": 0]

for item in array {

if item is Teacher { //是否是Teacher类

dict["Teacher"]! += 1

} else if item is Student { //是否是Student

dict["Student"]! += 1

} else { //Person类

dict["Person"]! += 1

}

}

//输出字典值

for (key, value) in dict {

print("\(key) has \(value) items")

}

//原始数组

print("------------------------------")

for item in array {

print(item)

}

//根据age从大到小排序

print("------------------------------")

array.sort { return $0.age > $1.age}

for item in array {

print(item)

}

//根据age从大到小排序

print("------------------------------")

array.sort { return $0.age > $1.age}

for item in array {

print(item)

}

//根据全名从前往后排序

print("------------------------------")

array.sort { return $0.fullName < $1.fullName}

for item in array {

print(item)

}

//根据gender和age从大往小排序

print("------------------------------")

array.sort { return ($0.gender > $1.gender) && ($0.age > $1.age) }

for item in array {

print(item)

}