enum Gender: Int {

case male

case female

case unknow

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

return lhs.rawValue < rhs.rawValue

}

}

enum Department {

case one, two, three

}

protocol SchoolProtocol {

var department: Department { get set }

func lendBook()

}

class Person: CustomStringConvertible {

var firstName: String

var lastName: String

var age: Int

var gender: Gender

var fullName: String {

get {

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, 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: "")

}

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

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

}

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

return !(lhs == rhs)

}

var description: String {

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

}

func run() {

print("Person \(self.fullName) is running")

}

}

var p1 = Person(firstName: "彭")

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

print(p1)

print(p1 == p2)

print(p1 != p2)

p1.run()

class Teacher: Person, SchoolProtocol {

var title: String

var department: Department

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

self.title = title

self.department = department

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

}

init(title: String, department: Department) {

self.title = title

self.department = department

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

}

convenience required init() {

self.init(title: "", department: Department.one)

}

override var description: String {

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

}

override func run() {

print("Teacher \(self.fullName) is running")

}

func lendBook() {

print("Teacher \(self.fullName) lend a book")

}

}

var t1 = Teacher(title: "hello", department: .one)

print(t1)

class Student: Person, SchoolProtocol {

var stuNo: Int

var department: Department

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

self.stuNo = stuNo

self.department = department

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

}

init(stuNo: Int, department: Department) {

self.stuNo = stuNo

self.department = department

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

}

required convenience init() {

self.init(stuNo: 0, department: .one)

}

override var description: String {

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

}

override func run() {

print("Student \(self.fullName) is running")

}

func lendBook() {

print("Teacher \(self.fullName) lend a book")

}

}

var s1 = Student(stuNo: 2016110328, department: .two)

print(s1)

var array = [Person]()

for i in 1...5 {

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

array.append(temp)

}

for i in 1...3 {

let temp = Teacher(title: "hello", firstName: "李", lastName: "\(i)", age: 21, gender: .female, department: .one)

array.append(temp)

}

for i in 1..<5 {

let temp = Student(stuNo: 2016110300 + i, firstName: "王", lastName: "\(i)", age: 19, gender: .male, department: .two)

array.append(temp)

}

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

for item in array {

if item is Teacher {

dict["Teacher"]! += 1

} else if item is Student {

dict["Student"]! += 1

} else {

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)

}

//穷举，调用run方法和lendBook方法

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

for item in array {

item.run()

if let teacher = item as? Teacher {

teacher.lendBook()

} else if let student = item as? Student {

student.lendBook()

}

}