**上机：（考察知识点Swift中的属性、属性观察器、方法和下标脚本的继承）**

**题目：**

按照一下要求完成继承的相关练习，要求如下：

1. 首先定义一个Person类，接下来定义一个Student类继承自Person类。
2. 在Person类中定义姓名和年龄属性，并给年龄属性加上属性观察器。
3. 在Person类中定义一个下标脚本。
4. 在Person类中定义一个方法。
5. 实例Student类，并调用它父类中的属性、下标脚本和方法。

在Student类中增加新的属性和方法并调用。

class Person {

    var age = 20 {

        willSet{

            print("新的年龄是\(newValue)")

        }

        didSet{

            if (age > oldValue) {

                print("比原来多了\(age - oldValue)岁")

            } else {

                print("比原来少了\(oldValue - age)岁")

            }

        }

    }

}

class Student: Person {

}

let newStudent = Student()

newStudent.age = 18

newStudent.age = 22



**上机：（考察知识点Swift中的继承）**

**题目：**

1. 创建playground文件。
2. 创建一个类Student，类中有 name（姓名）和age（年龄）属性，一个包含两个参数的构造函数，用于给name和age属性赋值，一个show（）方法打印Student的属性信息。
3. 创建一个类UnderGraduate，它继承自类Student，在类中定义一个major（专业）属性，有一个包含三个参数的构造函数，前两个参数用于给继承的name和age属性赋值，第三个参数给major专业赋值，一个重写的show方法用于打印UnderGraduate的属性信息。

新建一个UnderGraduate类的对象，姓名为“小明”，年龄为19，专业是“信息工程”，并调用该对象的show方法。

class Student {

    var name:String

    var age:Int

    init(*name*:String, *age*:Int){

        self.name = name

        self.age = age

    }

    func show(){

        print("该学生的名字是\(name),年龄是\(age)")

    }

}

class UnderGraduate: Student {

    var major:String

    init(*name*:String, *age*:Int, *major*:String){

        self.major = major

        super.init(name: name, age: age)

    }

    override func show() {

        print("该本科生的名字是\(name)，年龄是\(age)，专业是\(major)")

    }

}

let undergraduate = UnderGraduate(name: "小明", age: 19, major: "信息工程")

undergraduate.show()

