

微光招新

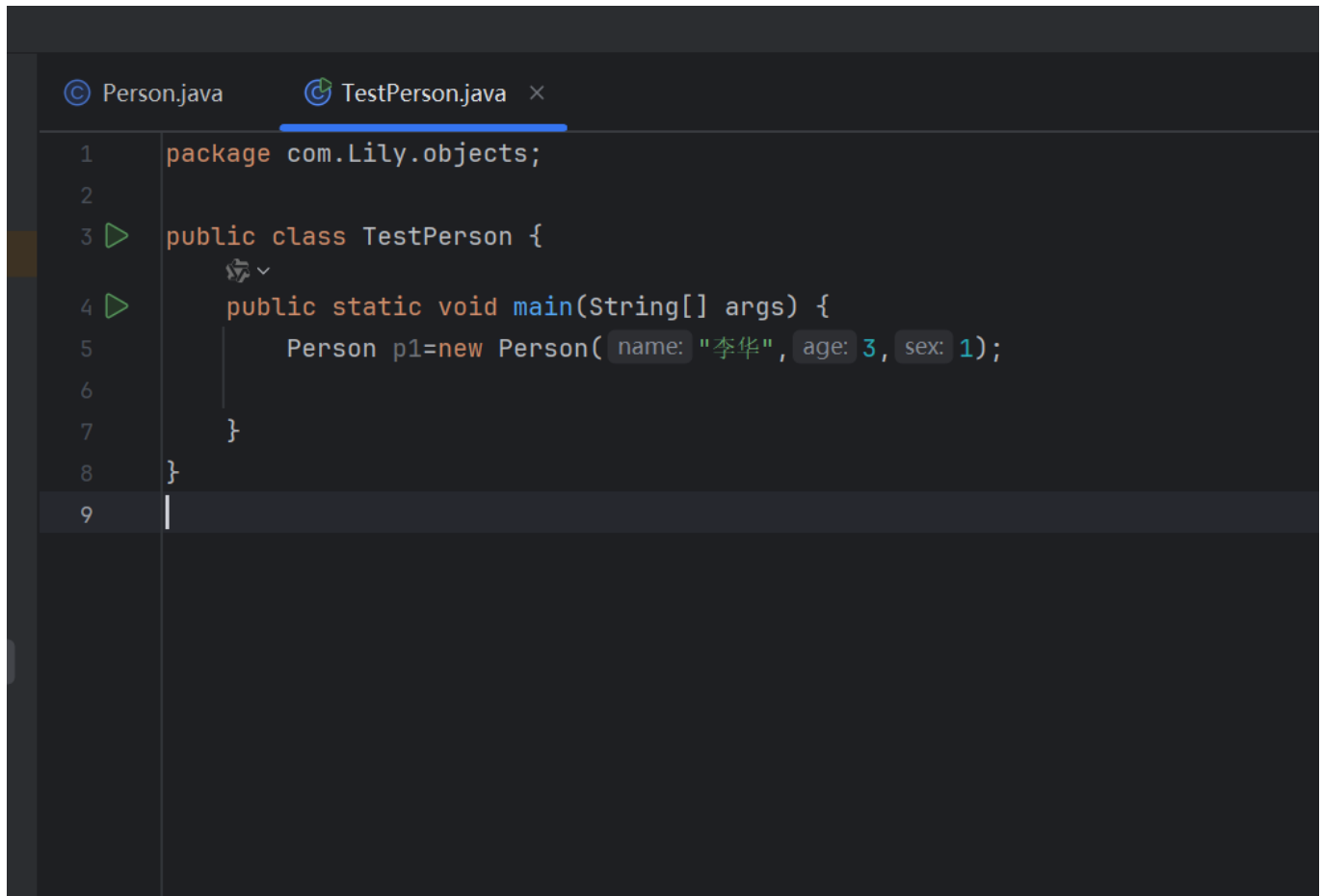
Java 05

Task1

```
public Person(Person p){//创建一个构造函数，用一个person 对象来当参数
    this.name=p.name;//将引入的对象的name, age, sex等属性赋给新对象
    this.age=p.age;
    this.sex=p.sex;
}
```

1. this关键字：代表当前对象（有点像C的指针），this.name=p.name;this会自动指向调用自己的对象,(p.可以算表示person对象的地址)
- 2.

```
public Person(String name,int age,int sex){ 1个用法
    this.name=name;
    this.age=age;
    this.sex=sex;
}
public Person(){//因为前面加了一个有参构造方法导致默认的无参构造方法没有了，这里多建一个无参构造方法
}
```



```
© Person.java    TestPerson.java ×
1 package com.Lily.objects;
2
3 public class TestPerson {
4     public static void main(String[] args) {
5         Person p1=new Person( name: "李华", age: 3, sex: 1);
6     }
7 }
8
9
```



对象和类的关系：类是模板（设计图：对象里该有什么），对象是具体的实例，对象存储在堆内存中，但是对象中会存储其类的地址

3. 访问修饰符：public, private, protected, default（1）public：public 修饰的变量或方法是公有的，可以被自己及其他包中的任何类访问（2）private：private 修饰的变量或方法是私有的，只能在本类中访问（3）protected：protected 修饰的变量或方法只能在本包和其他包的子类中被访问，不能被别的包中的非子类访问（4）default：不写修饰符，默认为default，只能在本包中访问

```
public class Person { 11 个用法 1 个相关问题
    //私有化成员变量
    public String name;
    int weight; 1 个用法
    private int age; 3 个用法
    protected int sex;//为什么这里的性别要用int啊? 6 个

    public Person(String name,int weight,int age,int sex){
        this.name=name;
        this.age=age;
        this.sex=sex;
    }
}
```

```
1 package com.Lily.objects;
2
3 ▶ public class TestPerson {
4     ▶ public static void main(String[] args) {
5         Person p1=new Person( name: "李华", weight: 49, age: 3, sex: 1);
6         ⚡ Person p2=new Person(p1);|
7         System.out.println(p1.name);
8         System.out.println(p1.age);
9         System.out.println(p1.sex);
10        System.out.println(p1.weight);
11
12
13    }
```

```
package com.LLY.testthe modify;  
import com.Lily.objects.Person;  
  
public class TestSonClass extends Person { 0 个用法  
       
    public void test() 0 个用法  
    {  
        System.out.println(name);  
        System.out.println(age);  
        System.out.println(sex);  
        System.out.println(weight);  
    }  
}
```

Task2

4.

