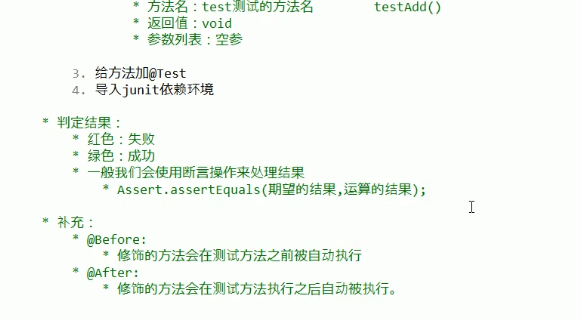
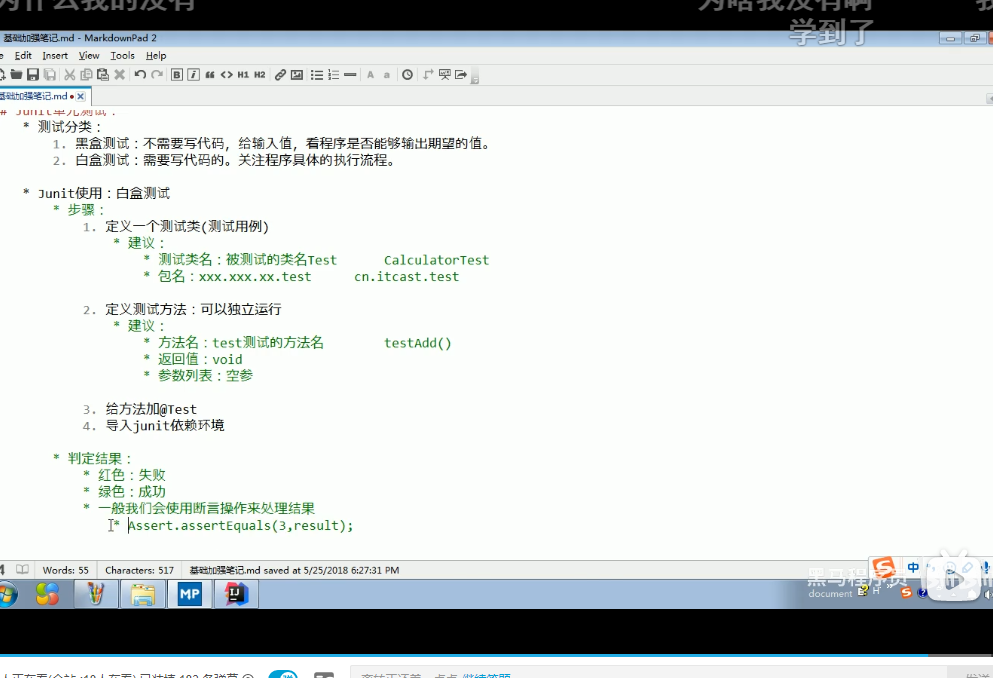


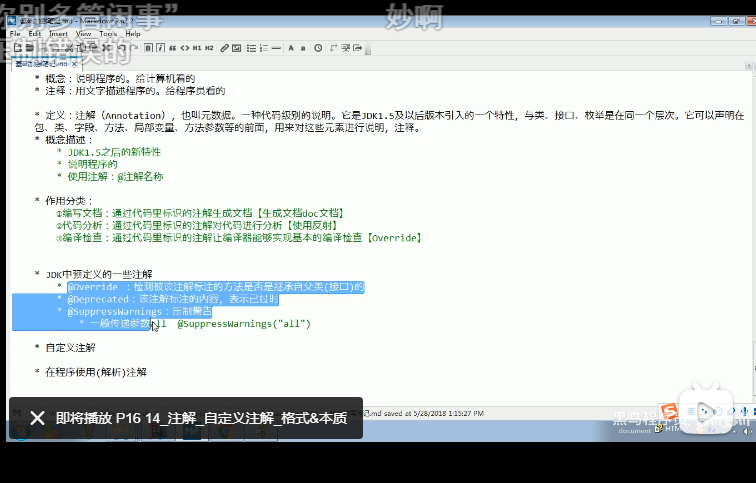
断言 判断结果与我们期望的是否一致

public class Test1Test {  
 @Test  
 public void sumTest(){  
 System.*out*.println("我被调用了");  
  
 Test1 test1 = new Test1();  
 System.*out*.println(test1.sum(7,6));  
 //断言  
 Assert.*assertEquals*(1,test1.sum(7,6));  
  
 }  
}

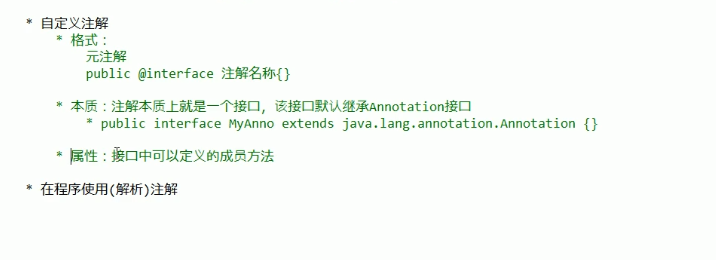
//断言  
Assert.*assertEquals*(1,test1.sum(7,6));



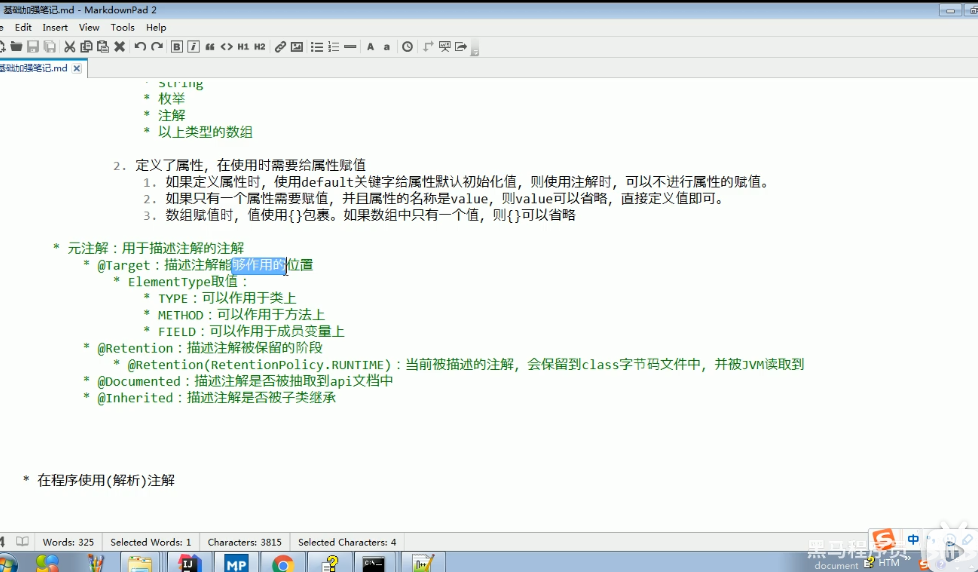




编译和反编译







编译 PS C:\Users\16784\Desktop\ce> javac Test3.java

反编译 PS C:\Users\16784\Desktop\ce> javap Test3.class

public interface org.om.Test3 extends java.lang.annotation.Annotation {

}



@Pro(className = "org.orm.Te.Demo1",methodName = "show")  
public class Demo3 {  
 public static void main(String[] args) throws Exception{  
// 获取该类的字节码文件  
 Class<Demo3> demo = Demo3.class;  
// 自动生成一个和接口复制 子类实现对象  
// 获取该类的上边的注解独享  
  
 Class<Pro> tt = Pro.class;  
 Pro an = demo.getAnnotation(tt);  
// 调用注解对象中的抽象方法,获取返回值

// 调用注解对象中的抽象方法,获取返回值  
 String classname = an.className();  
 String methodName = an.methodName();  
  
 System.*out*.println(methodName);