Code.java

```
public class Code {

	// describing whether the code can run correctly,
	private boolean canRun;
	//describing whether the code can be compiled correctly
	private boolean canCompile;
	//describing the number of lines in the code.
	private int lines;

	public Code() {
		//the default value is false.
		this.canRun=false;
		//the default value is false.
		this.canCompile=false;
		//the default value is 0.
		this.lines=0;
	}

	public Code(boolean canCompile, boolean canRun, int lines) {
		this.canRun=canRun;
		this.canCompile=canCompile;
		this.lines = lines;
	}
	//return whether the code can be compiled correctly.
	public boolean compile() {
		return canCompile;
	}
	//return whether the code can run correctly.
	public boolean run() {
		return canRun;
	}

	//makes the code compile and run correctly.
	public void debug() {
		this.canRun=true;
		this.canCompile=true;

	}

	//write the code to a certain number of lines.
	public void coding(int lines) {
		this.lines=lines;
		System.out.println("This code now have "+lines+" codes.");
	}

	//returns the number of lines in the code.
	public int countLines() {
		return lines;
	}

	//tests
	public static void testCode() {
```

```
            // Test for the first constructor,the default values both should be
false.
            Code code1=new Code();
            System.out.println(code1.compile()==false );
            System.out.println(code1.run()==false );

            //Test for the second constructor,set the values
            Code code2=new Code(true,false, 100);
            System.out.println(code2.compile()==true );
            System.out.println(code2.run()==false );
            System.out.println(code2.countLines()==100);
            // Test Debug.
            code2.debug();
            //Then code should can both compile and run.
            System.out.println(code2.compile()==true );
            System.out.println(code2.run()==true );

            // Test Coding & countLines
            code2.coding(15);
            System.out.println(code2.countLines()==15);
      }
}
```

Assignment.java

```
public class Assignment {
      //a Code object
      private Code myCode;
      //describing the score of the current assignment.
      private int score;
      //describing whether the assignment is submitted on time before deadline.
      private boolean submitted;
      //describing the student's name who finished this assignment.
      private String name;

      //constructor
      public Assignment(Code myCode,Boolean submitted,String name) {
            this.myCode =myCode;
            this.submitted =submitted;
            this.name=name;
      }

      //submits the assignment on time.
      public void submit() {
            this.submitted =true;
            //add an output to test
            if (this.submitted ==true){
                  System.out.println("This assignment has been submitted! :) ");
                  }

      }
```

```java
	//returns whether the assignment is submitted
	public boolean isSubmitted() {
		return submitted;
	}

	//set the score of the current assignment
	public void setScore(int score) {
		this.score=score;
	}

	//get the score of the current assignment
	public int getScore() {
		return score;
	}

	//returns the name variable.
	public String getName() {
		return name;
	}

	//get the code of the current assignment
	public Code getCode() {
		return myCode;
	}
	//tests
	public static void testAssignment() {
		// Test for the constructor
		Code code0=new Code();
		Assignment a0=new Assignment(code0,false,"xiaoyu");

		System.out.println(a0.getName()=="xiaoyu");
		System.out.println(a0.isSubmitted()==false);

		//Test for code
		//the default code can not be run
		System.out.println(a0.getCode().run()==false);

		//Test for submit
		a0.submit();
		System.out.println(a0.isSubmitted()==true);

		//Test for score
		a0.setScore(15);
		System.out.println(a0.getScore()==15);


	}
}
```

Student.java

```
public class Student {

	private Assignment myAssignment;
	//describing whether the student is honest or not.
	//An honest student will write assignment by him/herself,otherwise will copy
from others.
	//The default value for honest is true.
	private boolean honest=true;
	private String name;

	public Student(String name,Boolean honest) {

		this.name=name;
		//The default value for honest is true.
		this.honest=honest;
	}

	//return the student’s assignment
	public Assignment getAssignment() {
		return myAssignment;
	}

	//write an assignment
	public void writeAssignment(Assignment a) {
		this.myAssignment=a;
	}

	//copy an existing assignment
	public void copyAssignment(Assignment a) {
		this.myAssignment=a;

	}
	//return the student's name
	public String getName() {
		return name;
	}
	public static void testStudent() {
		Assignment a1=new Assignment(null,true,"xiaoxue");
		//create 2 students to test
		Student stu1=new Student("xiaoxue",true);
		Student stu2=new Student("xiaoyu",false);

		System.out.println(stu1.getName()=="xiaoxue");
		System.out.println(stu2.getName()=="xiaoyu");

		//Student1 Write assignment
		stu1.writeAssignment(a1);

		//get Student1's assignment
```

```
            Assignment assignment1=stu1.getAssignment();
            //Student2 Copy that assignment
            stu2.copyAssignment(assignment1);
            //stu2's name should NOT be equal with the name variable in the copied
assignment object
            System.out.println((stu2.getName()==assignment1.getName())==false);



      }
}
```

Teacher.java

```
public class Teacher {

      private String name;

      public Teacher(String name) {
            this.name=name;
      }

      //gives score to the assignment
      public int grading(Student s) {
            //get the student's assignment
            Assignment hw1=s.getAssignment();
            //If the assignment is not submit on time, the score is 0;
            if(hw1.isSubmitted()==false) {
                  hw1.setScore(0);
                  return hw1.getScore();
            }

            //For copying assignments
            //the assignment's name (the student who finished the assignment) is not
equal to the name of the student who submitted the assignment,
            //the score is 0
            if (hw1.getName()!=s.getName()) {
                  hw1.setScore(0);
            }

            //If the assignment is submitted on time
            //but cannot compile, the score is 0;
            if (hw1.getCode().compile()==false) {
                  hw1.setScore(0);

                  //can compile, but cannot run, the score is 50;
            }else if (hw1.getCode().compile()==true && hw1.getCode().run()==false) {
                  hw1.setScore(50);
                  //can compile, can run,
                  //but the line number is less than 100, the score is 80;
            }else if (hw1.getCode().compile()==true && hw1.getCode().run()==true &&
hw1.getCode().countLines()<100) {
```

```
                hw1.setScore(80);

            }else if (hw1.getCode().compile()==true && hw1.getCode().run()==true &&
hw1.getCode().countLines()>=100) {
                hw1.setScore(100);
            }


            return hw1.getScore();

      }

      //Tests
      public static void testTeacher() {
            //set one teacher named xyz
            Teacher t1=new Teacher("xyz");

            //tests for grading

            //4 kinds of code
            Code code1=new Code(true,true, 50);
            Code code2=new Code(true,false,120);
            Code code3=new Code();
            Code code4=new Code(true,true, 120);


            //set 6 different students
            //not submit,score 0
            Assignment a1=new Assignment(code1,false,"abin");
            Student stu1=new Student("abin",true);
            stu1.writeAssignment(a1);
            System.out.println(t1.grading(stu1)==0);

            //Copying ,score 0
            Student stu2=new Student("huanfeng",false);
            stu2.copyAssignment(a1);
            System.out.println(t1.grading(stu1)==0);

            //submitted , cannot compile, score 0
            Assignment a2=new Assignment(code3,true,"Potter");
            Student stu3=new Student("potter",true);
            stu3.writeAssignment(a2);
            System.out.println(t1.grading(stu3)==0);

            //submitted , can compile, cannot run ,score 50
            Assignment a3=new Assignment(code2,true,"Ron");
            Student stu4=new Student("Ron",true);
            stu4.writeAssignment(a3);
            System.out.println(t1.grading(stu4)==50);

            //submitted , can compile, can run, code lines <100 ,score 80
            Assignment a4=new Assignment(code1,true,"Lupin");
            Student stu5=new Student("Lupin",true);
            stu5.writeAssignment(a4);
            System.out.println(t1.grading(stu5)==80);
```

```
            //submitted , can compile, can run, code lines >100 ,score 100
            Assignment a5=new Assignment(code4,true,"Hermione");
            Student stu6=new Student("Hermione",true);
            stu6.writeAssignment(a5);
            System.out.println(t1.grading(stu6)==100);


      }
}
```

Start.java

```
public class Start {
      public static void main(String[] args) {
            //run the tests;
            Code.testCode();
            Assignment.testAssignment();
            Student.testStudent();
            Teacher.testTeacher();
      }
}
```