

Knowledge Reasoning (I) - RDFS

一、RDFox配置

导入License文件

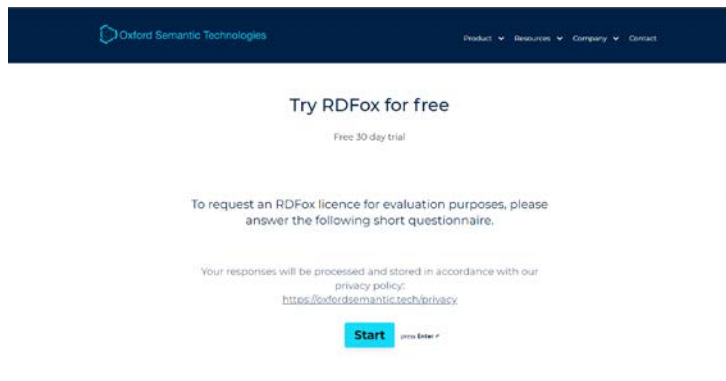
有Licence文件？

- Mac/linux: 将licence文件(RDFox.lic)放置入~/.RDFox/文件夹下
- Windows: 将licence文件(RDFox.lic)放置入C:\Users\用户名\AppData\Local\RDFox\文件夹下，下图以Windows路径为例

此电脑 > 系统 (C:) > 用户 > hbunz > AppData > Local > RDFox

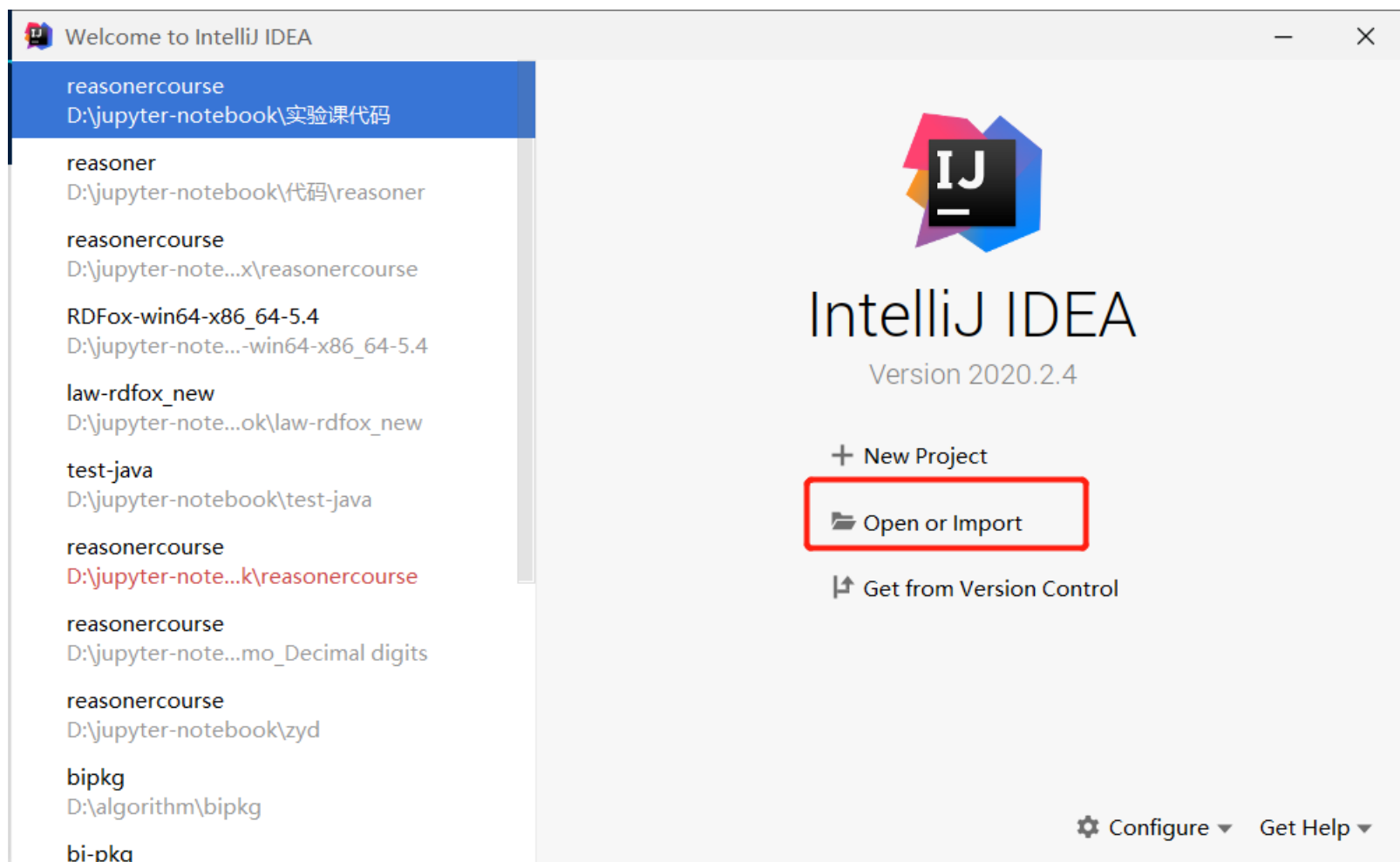
无Licence文件？

- 申请地址：<https://www.oxfordsemantic.tech/tryrdfoxforfree>
- 填写**组织邮箱**（如zhangsan@seu.edu.cn）并申请通过，可获得30天免费试用期



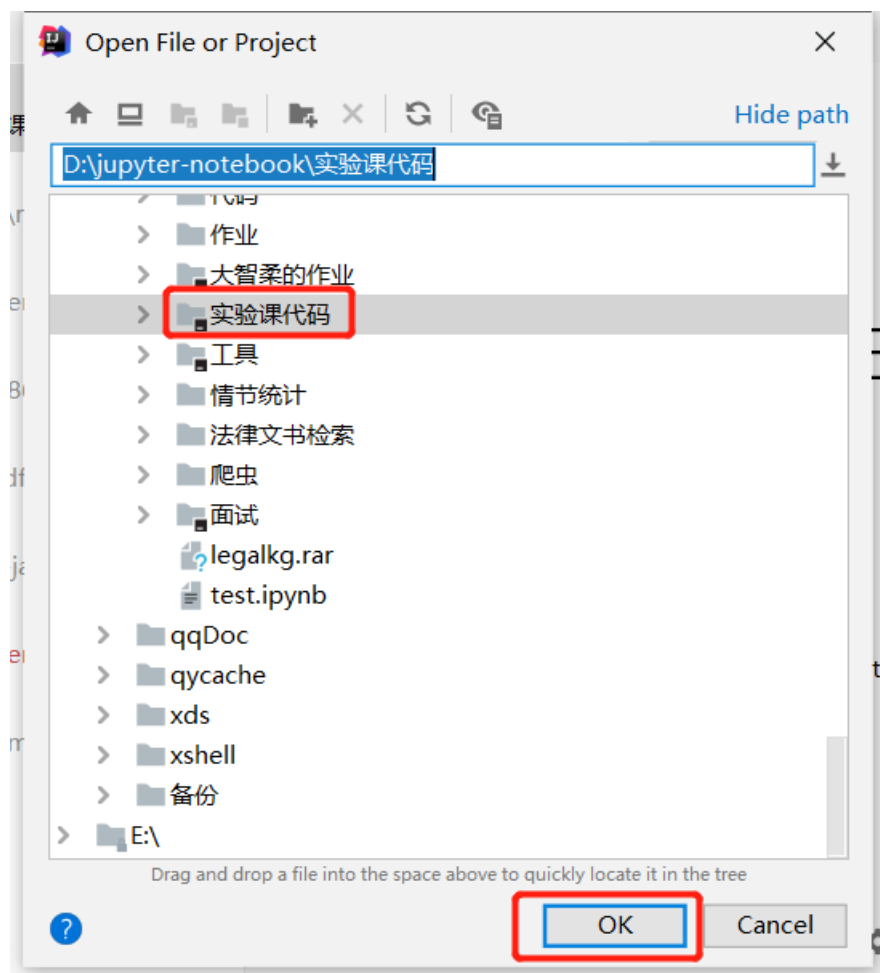
导入java项目

以idea为例，点击“Open or Import”；



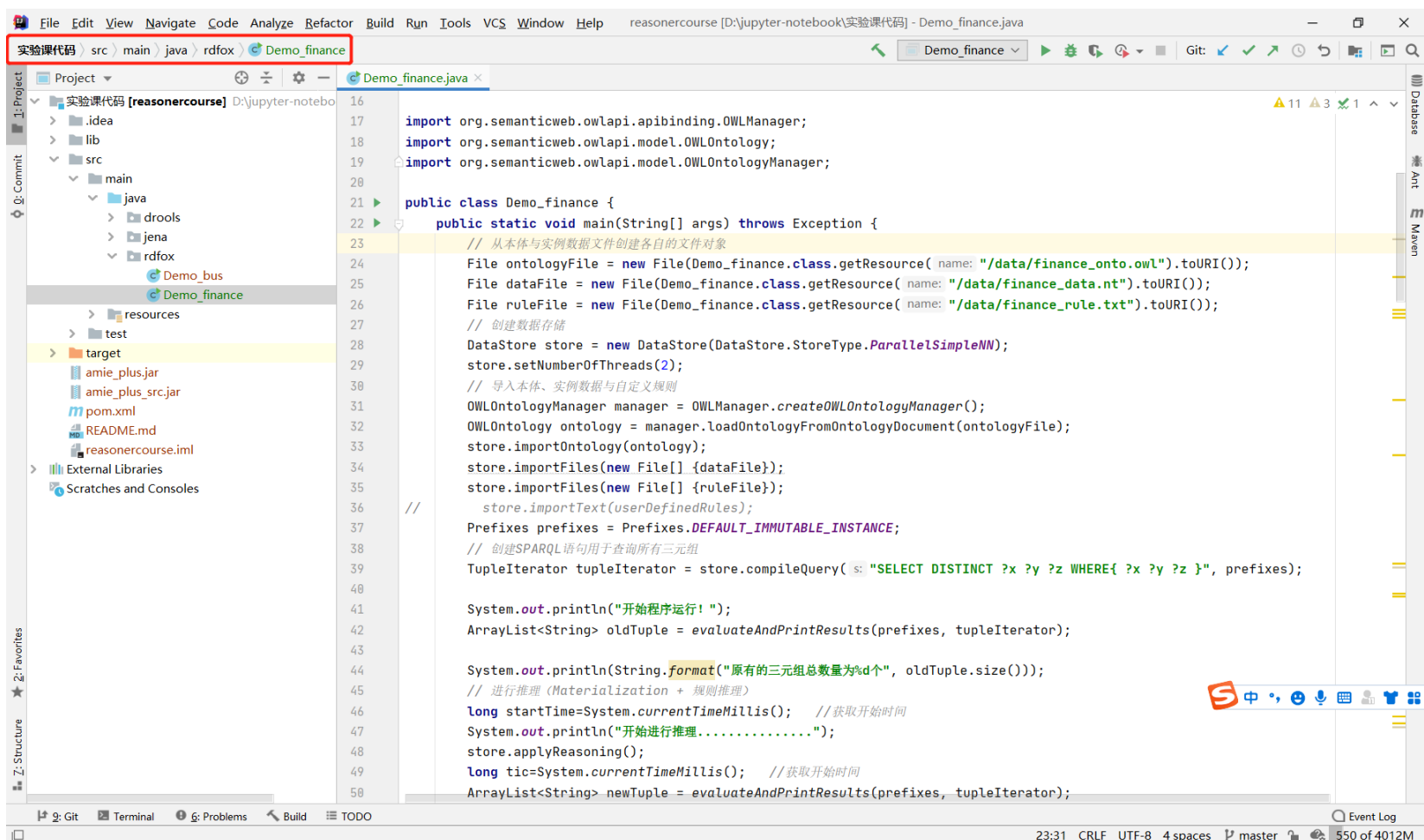
导入java项目

点击java项目“实验课代码”，点击“OK”确认；



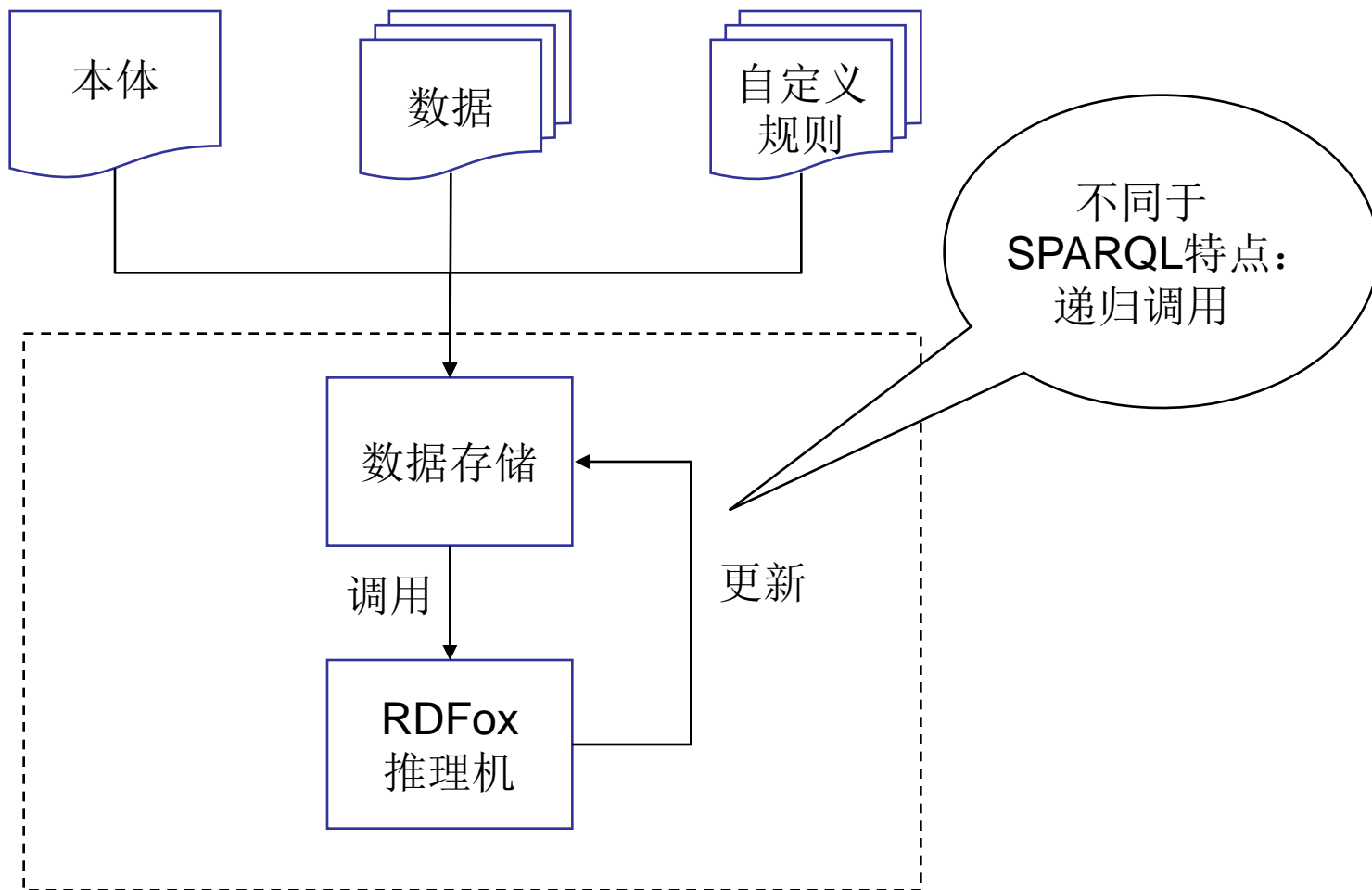
导入java项目

主程序文件位于“实验课代码/src/main/java/rdfox”路径下。

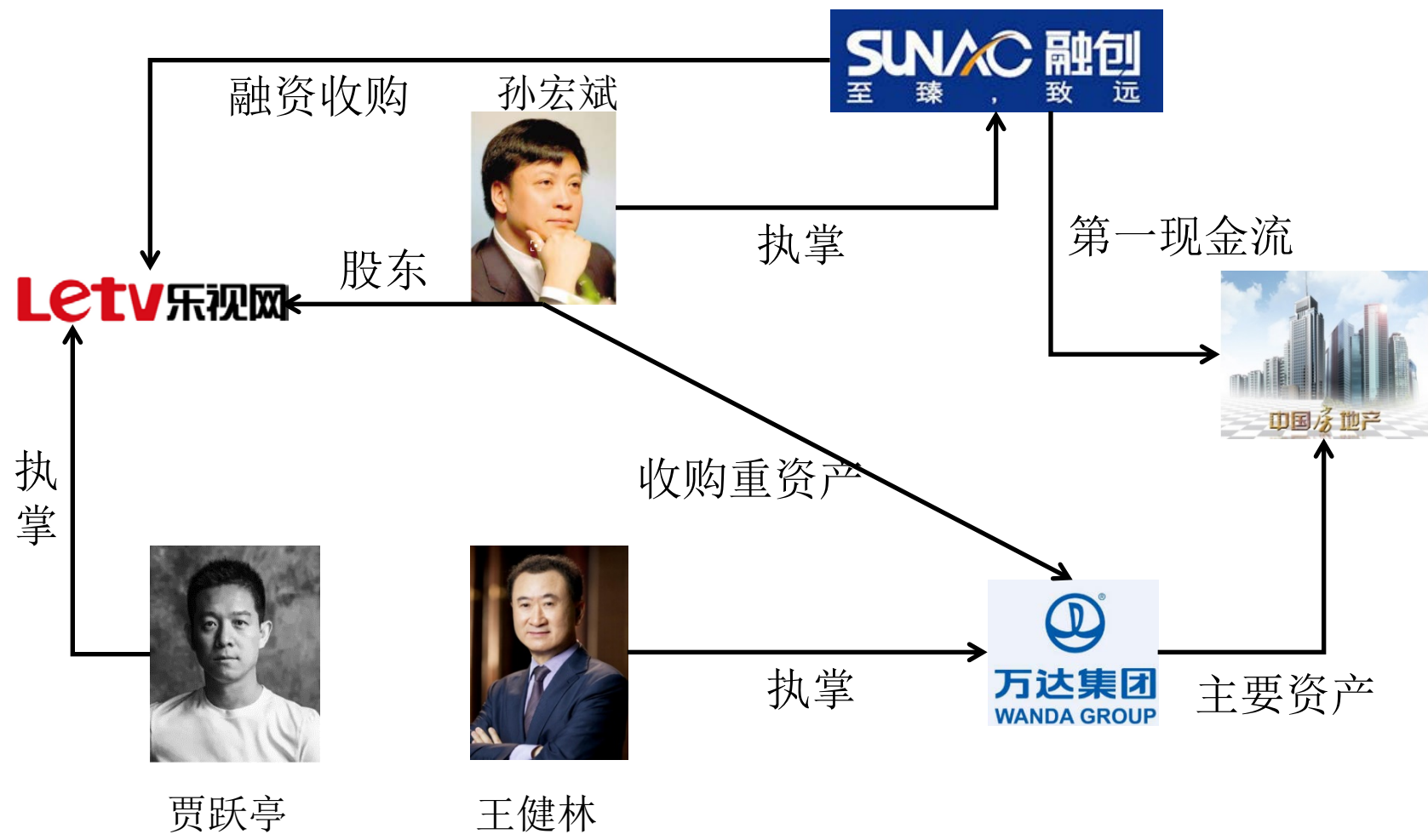


二、知识推理示例

RDFox推理过程



示例一：金融领域知识推理

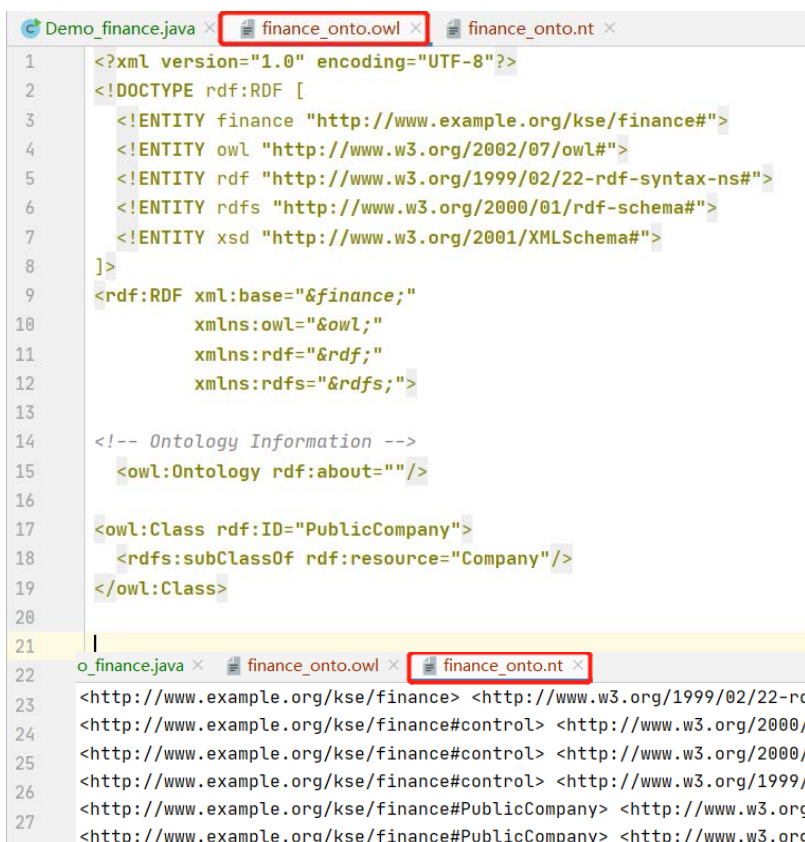


RDFox实践-程序输入

本体：以XML或Turtle的形式给出；

XML形式对应“实验课代码；

\src\main\resources\data\finance_onto.owl”。



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE rdf:RDF [
3   <!ENTITY finance "http://www.example.org/kse/finance#">
4   <!ENTITY owl "http://www.w3.org/2002/07/owl#">
5   <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#">
6   <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#">
7   <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">
8 ]>
9 <rdf:RDF xml:base="&finance;"
10   xmlns:owl="&owl;"
11   xmlns:rdf="&rdf;"
12   xmlns:rdfs="&rdfs;">
13
14 <!-- Ontology Information -->
15   <owl:Ontology rdf:about=""/>
16
17   <owl:Class rdf:ID="PublicCompany">
18     <rdfs:subClassOf rdf:resource="Company"/>
19   </owl:Class>
20
21
22
23 <http://www.example.org/kse/finance> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#Ontology> .
24 <http://www.example.org/kse/finance#control> <http://www.w3.org/2000/01/rdf-schema#range> <http://www.example.org/kse/Company> .
25 <http://www.example.org/kse/finance#control> <http://www.w3.org/2000/01/rdf-schema#domain> <http://www.example.org/kse/Person> .
26 <http://www.example.org/kse/finance#control> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#ObjectProperty> .
27 <http://www.example.org/kse/finance#PublicCompany> <http://www.w3.org/2000/01/rdf-schema#subClassOf> <http://www.example.org/kse/Company> .
28 <http://www.example.org/kse/finance#PublicCompany> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.w3.org/2002/07/owl#Class> .
```

例如：

- 谓词“control”的定义域是Person，值域是Company，形式化规则为：
 $\text{control}(X,Y) \rightarrow \text{Person}(X),$
 $\text{control}(X,Y) \rightarrow \text{Company}(Y)$
- PublicCompany是Company的子类，形式化规则为：
 $\text{PublicCompany}(X) \rightarrow \text{Company}(X)$

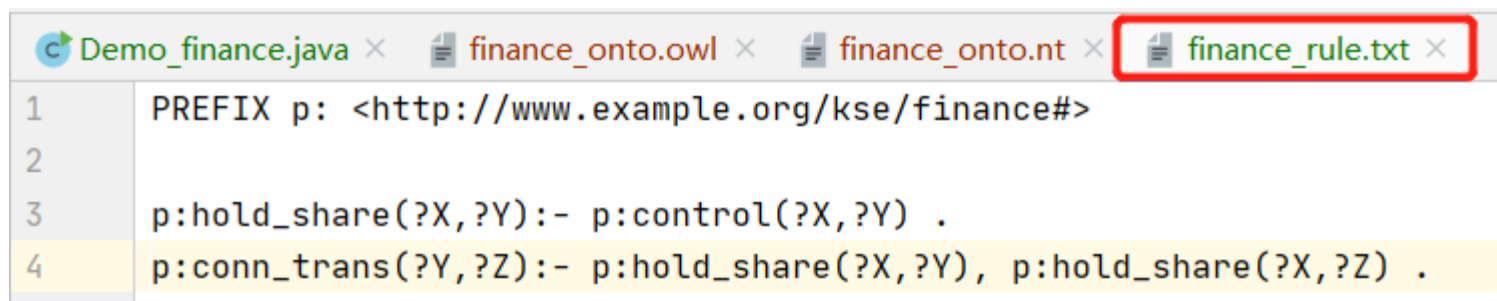
RDFox实践-程序输入

规则：描述具体情况，以三元组的形式给出；

对应“实验课代码\src\main\resources\data\finance_rule.txt”。

例如

- 执掌一家公司就一定是这家公司的股东，形式化规则为：
`control(X,Y)->hold_share(X,Y)`
- 某人同时是两家公司的股东，那么这两家公司一定有关联交易,形式化规则为：
`hold_share(X,Y),hold_share(X,Z) ->conn_trans(Y,Z)`



The screenshot shows a code editor with four tabs: Demo_finance.java, finance_onto.owl, finance_onto.nt, and finance_rule.txt (which is highlighted with a red box). The code in the active tab is as follows:

```
1 PREFIX p: <http://www.example.org/kse/finance#>
2
3 p:hold_share(?X,?Y):- p:control(?X,?Y) .
4 p:conn_trans(?Y,?Z):- p:hold_share(?X,?Y), p:hold_share(?X,?Z) .
```

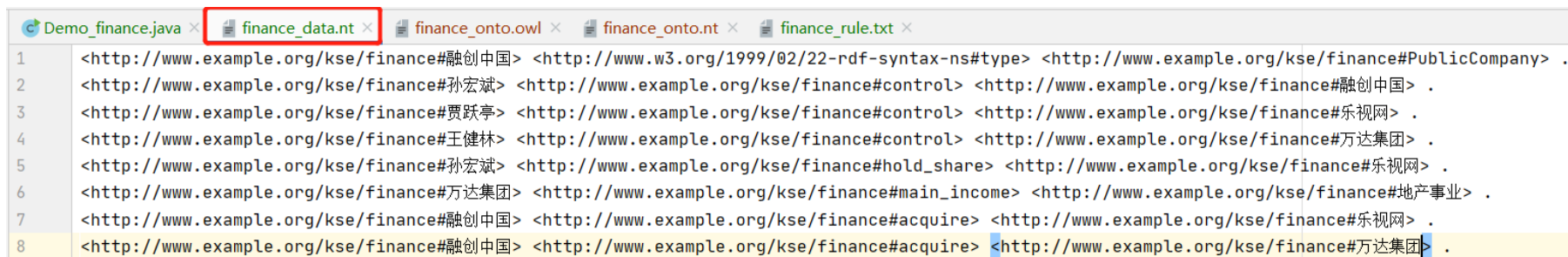
Datalog 规则可以看作IF THEN 语句，THEN部分首先被写入，并通过符号 :- 与IF部分分开。IF部分和 THEN部分由若干原子组成，原子之间用逗号分隔，每个原子都是可能出现变量或常量的三元组。规则忠实的描述了关系的传递性质，从而发挥推理作用。

RDFox实践-程序输入

数据：记录事实信息，以三元组的形式给出；

对应“实验课代码\src\main\resources\data\finance_data.nt”；

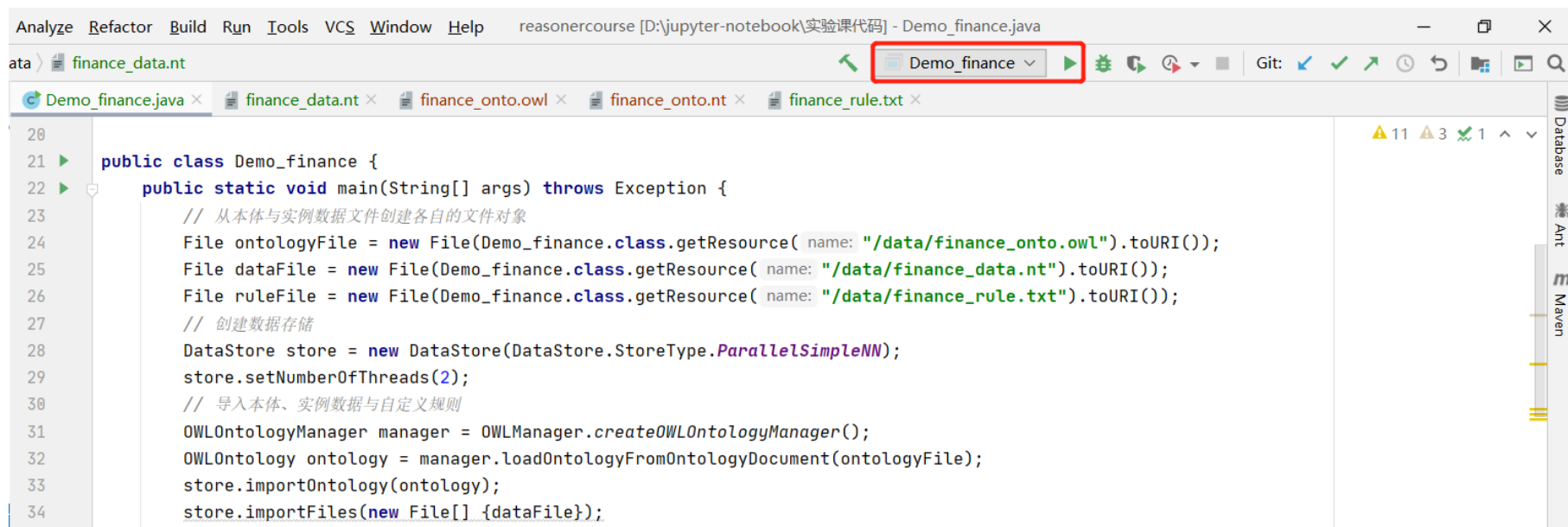
一般带有前缀，如<http://www.example.org/kse/finance#>。



```
Demo_finance.java × finance_data.nt × finance_onto.owl × finance_onto.nt × finance_rule.txt ×
1 <http://www.example.org/kse/finance#融创中国> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.example.org/kse/finance#PublicCompany> .
2 <http://www.example.org/kse/finance#孙宏斌> <http://www.example.org/kse/finance#control> <http://www.example.org/kse/finance#融创中国> .
3 <http://www.example.org/kse/finance#贾跃亭> <http://www.example.org/kse/finance#control> <http://www.example.org/kse/finance#乐视网> .
4 <http://www.example.org/kse/finance#王健林> <http://www.example.org/kse/finance#control> <http://www.example.org/kse/finance#万达集团> .
5 <http://www.example.org/kse/finance#孙宏斌> <http://www.example.org/kse/finance#hold_share> <http://www.example.org/kse/finance#乐视网> .
6 <http://www.example.org/kse/finance#万达集团> <http://www.example.org/kse/finance#main_income> <http://www.example.org/kse/finance#地产事业> .
7 <http://www.example.org/kse/finance#融创中国> <http://www.example.org/kse/finance#acquire> <http://www.example.org/kse/finance#乐视网> .
8 <http://www.example.org/kse/finance#融创中国> <http://www.example.org/kse/finance#acquire> <http://www.example.org/kse/finance#万达集团> .
```

RDFox实践-执行程序

选择执行文件为Demo_finance.java，
点击运行键执行代码。



The screenshot shows an IDE window titled "reasonercourse [D:\jupyter-notebook\实验课代码] - Demo_finance.java". The "Run" menu is open, and "Demo_finance" is selected. The code editor displays the following Java code:

```
20
21 public class Demo_finance {
22     public static void main(String[] args) throws Exception {
23         // 从本体与实例数据文件创建各自的文件对象
24         File ontologyFile = new File(Demo_finance.class.getResource( name: "/data/finance_onto.owl").toURI());
25         File dataFile = new File(Demo_finance.class.getResource( name: "/data/finance_data.nt").toURI());
26         File ruleFile = new File(Demo_finance.class.getResource( name: "/data/finance_rule.txt").toURI());
27         // 创建数据存储
28         DataStore store = new DataStore(DataStore.StoreType.ParallelSimpleNN);
29         store.setNumberOfThreads(2);
30         // 导入本体、实例数据与自定义规则
31         OWLOntologyManager manager = OWLManager.createOWLOntologyManager();
32         OWLOntology ontology = manager.loadOntologyFromOntologyDocument(ontologyFile);
33         store.importOntology(ontology);
34         store.importFiles(new File[] {dataFile});
```

The right sidebar shows the "Database" tab with 11 warnings and 3 errors, and the "Ant" and "Maven" tabs.

RDFox实践-程序输出

推理结果显示推理出了新的三元组，
本体和规则均被触发。

```
<http://www.example.org/kse/finance#融创中国>rdf:type<http://www.example.org/kse/Company>  
  
<http://www.example.org/kse/finance#融创中国><http://www.example.org/kse/finance#acquire><http://www.example.org/kse/finance#万达集团>  
  
<http://www.example.org/kse/finance#融创中国><http://www.example.org/kse/finance#acquire><http://www.example.org/kse/finance#乐视网>  
  
<http://www.example.org/kse/finance#万达集团><http://www.example.org/kse/finance#main_income><http://www.example.org/kse/finance#地产事业>  
  
<http://www.example.org/kse/finance#孙宏斌><http://www.example.org/kse/finance#hold_share><http://www.example.org/kse/finance#乐视网>  
  
<http://www.example.org/kse/finance#王健林><http://www.example.org/kse/finance#control><http://www.example.org/kse/finance#万达集团>  
  
<http://www.example.org/kse/finance#贾跃亭><http://www.example.org/kse/finance#control><http://www.example.org/kse/finance#乐视网>  
  
<http://www.example.org/kse/finance#孙宏斌><http://www.example.org/kse/finance#control><http://www.example.org/kse/finance#融创中国>  
  
<http://www.example.org/kse/finance#融创中国>rdf:type<http://www.example.org/kse/finance#PublicCompany>
```

查询时间: 5ms

原有的三元组总数量为8个

推理过后的三元组总数量为22个

推理出来的三元组总数量为14个

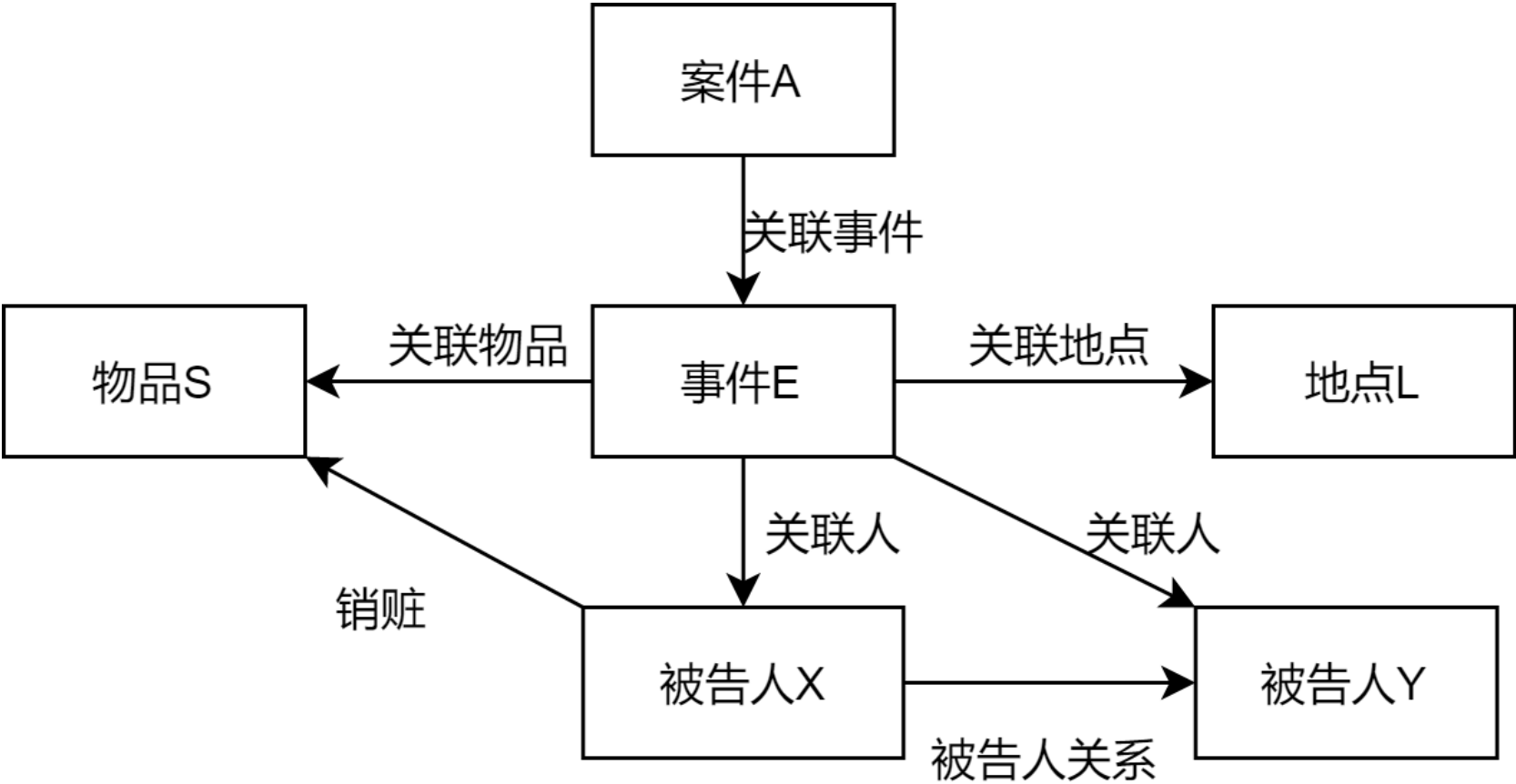
程序运行时间: 8ms

查询时间: 5ms

当前JVM占用的内存总数: 243.5M

已经使用内存: 38.014984130859375M

示例二：法律领域知识推理



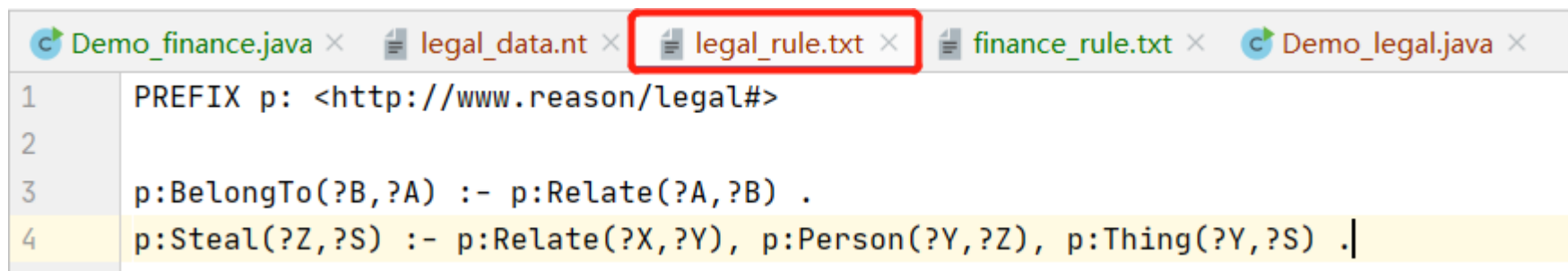
RDFox实践-程序输入

规则：描述具体情况，以三元组的形式给出

对应“实验课代码\src\main\resources\data\legal_rule.txt”

例如：

- 案件A关联一个事件B，那么事件B属于案件A，形式化规则为：
关联事件(A,B)->属于(B,A)
- 案件X关联事件Y，且事件Y关联人物Z，且事件Y关联盗窃物S，那么人物Z盗窃了物品S,形式化规则为：
关联事件(X, Y), 关联人物(Y, Z), 关联盗窃物(Y, S)->盗窃(Z, S)



The screenshot shows a code editor with five tabs: Demo_finance.java, legal_data.nt, legal_rule.txt (highlighted with a red box), finance_rule.txt, and Demo_legal.java. The content of legal_rule.txt is as follows:

```
1 PREFIX p: <http://www.reason/legal#>
2
3 p:BelongTo(?B,?A) :- p:Relate(?A,?B) .
4 p:Steal(?Z,?S) :- p:Relate(?X,?Y), p:Person(?Y,?Z), p:Thing(?Y,?S) .|
```


RDFox实践-程序输入

数据：记录事实信息，以三元组的形式给出；

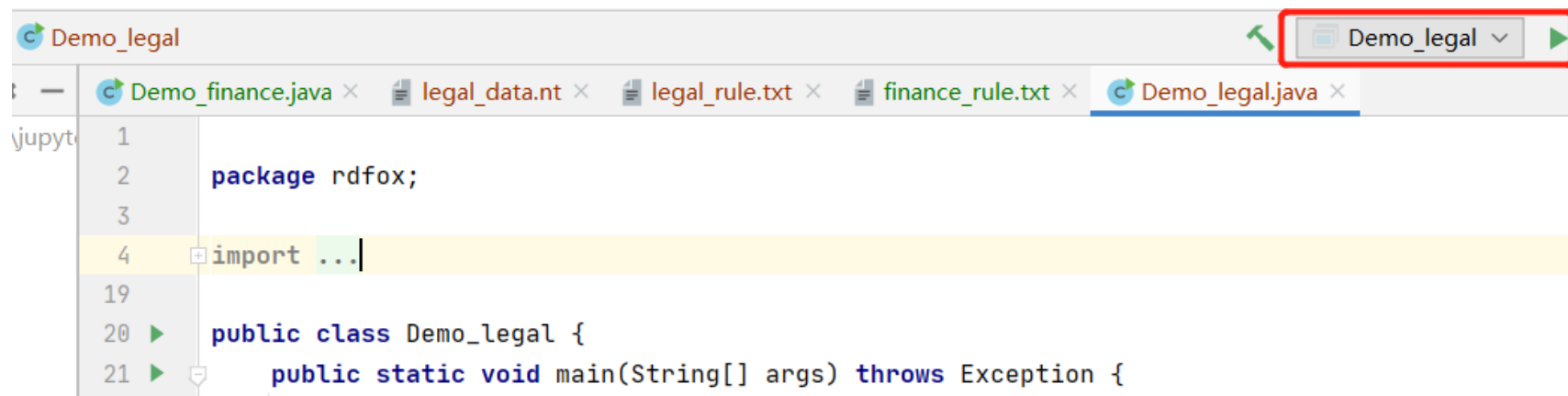
对应“实验课代码\src\main\resources\data\legal_data.nt”；

一般带有前缀，如<http://www.reason/legal#>。

```
Demo_finance.java × legal_data.nt × legal_rule.txt × finance_rule.txt × Demo_legal.java ×
1 <http://www.reason/legal#case1> <http://www.reason/legal#Relate> <http://www.reason/legal#event1> .
2 <http://www.reason/legal#event1> <http://www.reason/legal#Time> <http://www.reason/legal#2017年8月> .
3 <http://www.reason/legal#event1> <http://www.reason/legal#Locate> <http://www.reason/legal#潮州市潮安区> .
4 <http://www.reason/legal#event1> <http://www.reason/legal#Person> <http://www.reason/legal#邱伟> .
5 <http://www.reason/legal#event1> <http://www.reason/legal#Thing> <http://www.reason/legal#被害人梁金萍的人民币2100元> .
6 <http://www.reason/legal#case1> <http://www.reason/legal#Relate> <http://www.reason/legal#event2> .
7 <http://www.reason/legal#event2> <http://www.reason/legal#Time> <http://www.reason/legal#2017年7月31日> .
8 <http://www.reason/legal#event2> <http://www.reason/legal#Locate> <http://www.reason/legal#广东省潮州市潮安区> .
9 <http://www.reason/legal#event2> <http://www.reason/legal#Person> <http://www.reason/legal#邱伟> .
10 <http://www.reason/legal#event2> <http://www.reason/legal#Thing> <http://www.reason/legal#被害人潘玉香的支付宝账号> .
11 <http://www.reason/legal#event2> <http://www.reason/legal#Thing> <http://www.reason/legal#密码> .
12 <http://www.reason/legal#event2> <http://www.reason/legal#Thing> <http://www.reason/legal#支付密码> .
13 <http://www.reason/legal#event2> <http://www.reason/legal#Thing> <http://www.reason/legal#余额中的人民币6400元> .
14 <http://www.reason/legal#event2> <http://www.reason/legal#KeyWords> <http://www.reason/legal#销赃> .
15 <http://www.reason/legal#event2> <http://www.reason/legal#OtherPeople> <http://www.reason/legal#王小虎> .
16 <http://www.reason/legal#event2> <http://www.reason/legal#物品> <http://www.reason/legal#余额中的人民币6400元> .
```

RDFOX实践-执行程序

选择执行文件为Demo_legal.java,
点击运行键执行代码。



The screenshot shows an IDE window with the title bar "Demo_legal". The toolbar at the top right contains a green left-pointing arrow, a dropdown menu showing "Demo_legal", and a green right-pointing arrow (the Run button). The Run button is highlighted with a red rectangle. The editor displays the following Java code:

```
1  
2 package rdfox;  
3  
4 import ...  
19  
20 public class Demo_legal {  
21     public static void main(String[] args) throws Exception {
```

RDFOx实践-程序输出

推理结果显示推理出了新的三元组，
规则被触发。

```
Run: Demo_legal x
<http://www.reason/legal#event2><http://www.reason/legal#Time><http://www.reason/legal#2017年7月31日>
<http://www.reason/legal#case1><http://www.reason/legal#Relate><http://www.reason/legal#event2>
<http://www.reason/legal#event1><http://www.reason/legal#Thing><http://www.reason/legal#被害人梁金萍的人民币2100元>
<http://www.reason/legal#event1><http://www.reason/legal#Person><http://www.reason/legal#邱伟>
<http://www.reason/legal#event1><http://www.reason/legal#Locate><http://www.reason/legal#潮州市潮安区>
<http://www.reason/legal#event1><http://www.reason/legal#Time><http://www.reason/legal#2017年8月>
<http://www.reason/legal#case1><http://www.reason/legal#Relate><http://www.reason/legal#event1>

查询时间: 3ms
原有的三元组总数量为16个
推理过后的三元组总数量为23个
推理出来的三元组总数量为7个
程序运行时间: 4ms
查询时间: 3ms
当前JVM占用的内存总数: 243.5M
已经使用内存: 12.767059326171875M

Process finished with exit code 0
```

三、课堂总结

课堂总结

RDFox配置

RDFox推理过程

RDFox规则撰写

四、课堂作业

金融领域知识推理

1. 阅读程序Demo_finance.java源码，分别注释本体和规则的部分，观察推理结果的变化；
2. 撰写Datalog规则进行推理，观察新的推理结果（对应“实验课代码\src\main\resources\data\finance_rule.txt”）：
 - 1) 如果A是B的子类，B是C的子类，那么A是C的子类（对应finance_data.nt中的谓词“subClassOf”）
 - 2)* 如果A的类型是PublicCompany，那么PublicCompany的任意父类也是A的类型（对应finance_data.nt中的谓词“type”）。

*：2)的难度较高，首先原子中涉及常量的表示，其次需要声明新的前缀；其中常量的表示可以参考谓词的表示。

法律领域知识推理

1. 撰写Datalog规则进行推理，观察新的推理结果（对应“实验课代码\src\main\resources\data\legal_rule.txt”）：
 - 1) 如果案件A关联事件B,事件B的发生时间是案件A的关键节点（对应legal_data.nt中的谓词“Relate”和“Time”）

参考资料

- RDFox官方文档
<https://docs.oxfordsemantic.tech/index.html>