

实验二：活跃变量分析

2024.3.14



中山大學
SUN YAT-SEN UNIVERSITY

实验二：活跃变量分析



- 实验平台配置
- 实验内容
- API介绍
- 作业提交

实验平台配置——Tai-e



■ 按照 <https://tai-e.pascal-lab.net/intro/overview.html> 配置

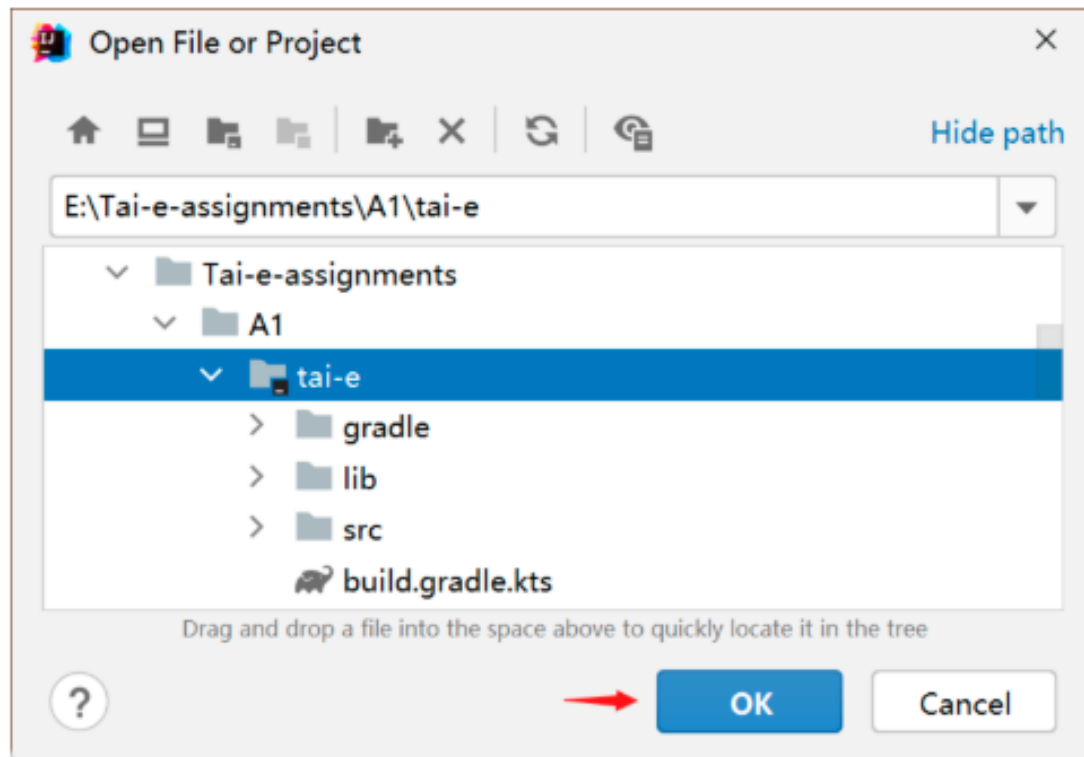
- Download assignments at <https://github.com/pascal-lab/Tai-e-assignments>

实验平台配置——Tai-e



■ 导入项目

选择 A1/tai-e/ 文件夹，点击“OK”。



实验平台配置——Tai-e



■ 导入项目

IntelliJ IDEA 可能会弹出下图窗口询问你是否信任该 Gradle 项目。点击 “Trust Project” 信任该项目（别担心，Tai-e 是可信的 😊）。



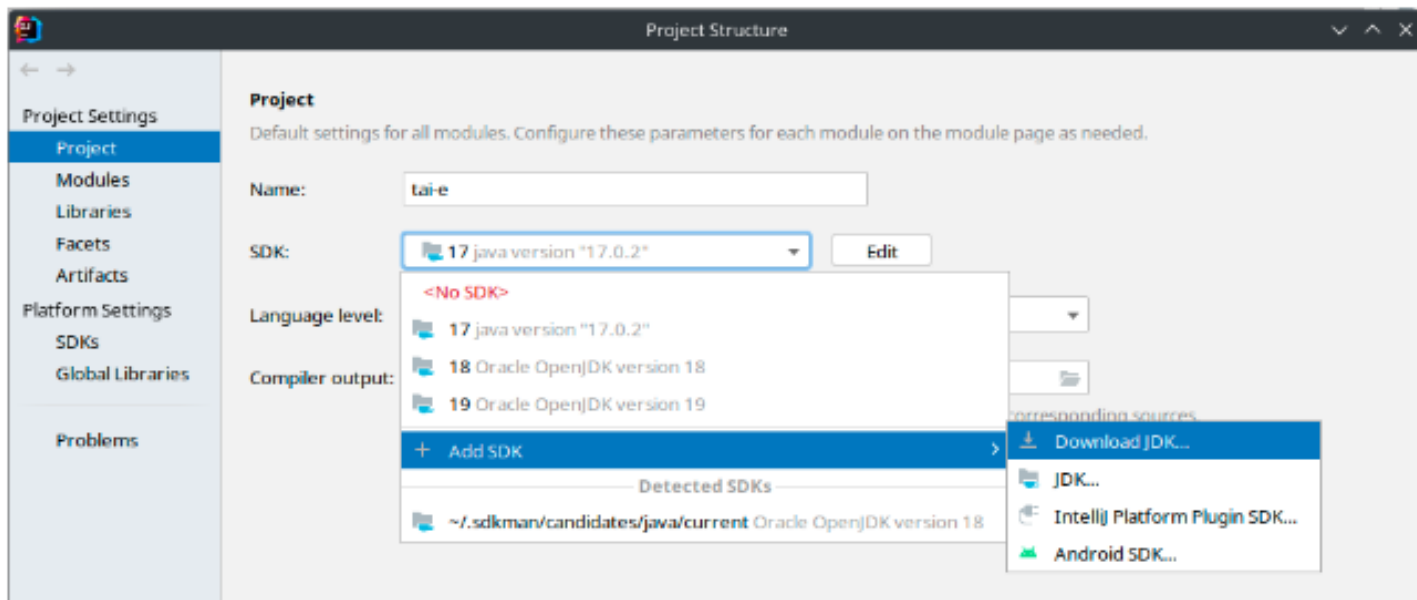
这样导入操作就完成了。你可能需要等待一段时间以导入 Tai-e。之后，tai-e/ 文件夹中会生成一些与 Gradle 相关的文件和文件夹，你可以忽略它们。

实验平台配置——Tai-e



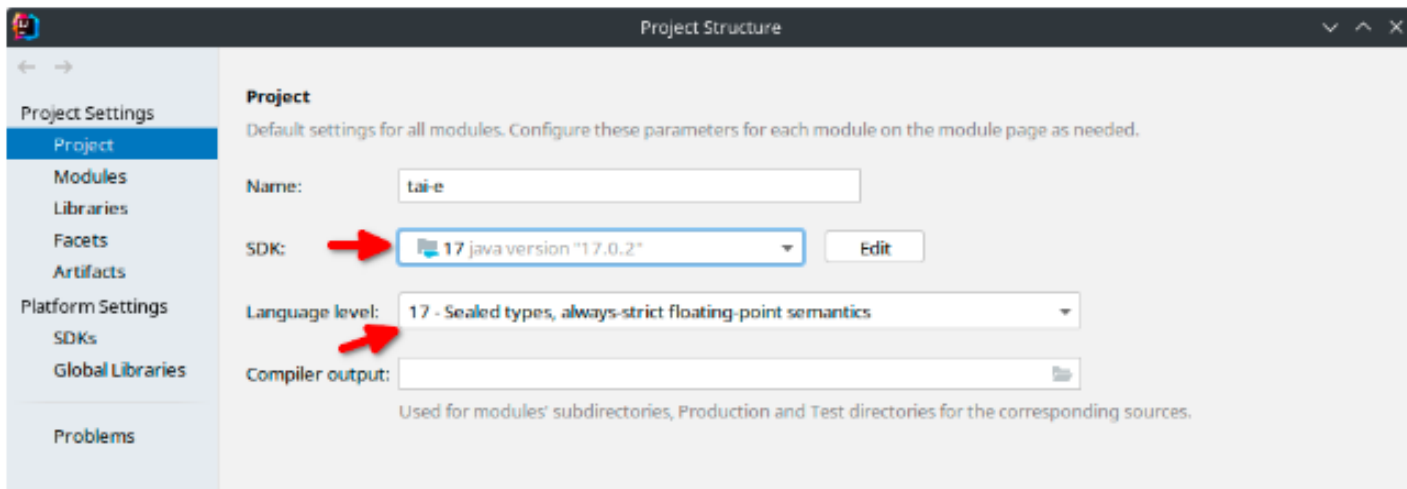
■ 配置project setting

打开 File > Project Structure..., 展开“SDK”下拉菜单, 选择 Add SDK > Download JDK..., 在弹出的窗口中选择 Version 为 17, Vendor 任意 (通常选择 Oracle OpenJDK), Location 选择安装位置, 一般保持默认即可, 点击 Download 开始后台下载。



■ 配置project setting

然后展开 “Language level”，选择 “SDK default”（如果默认值是后者）或 “17 - Sealed types, always-strict floating-point semantics”。

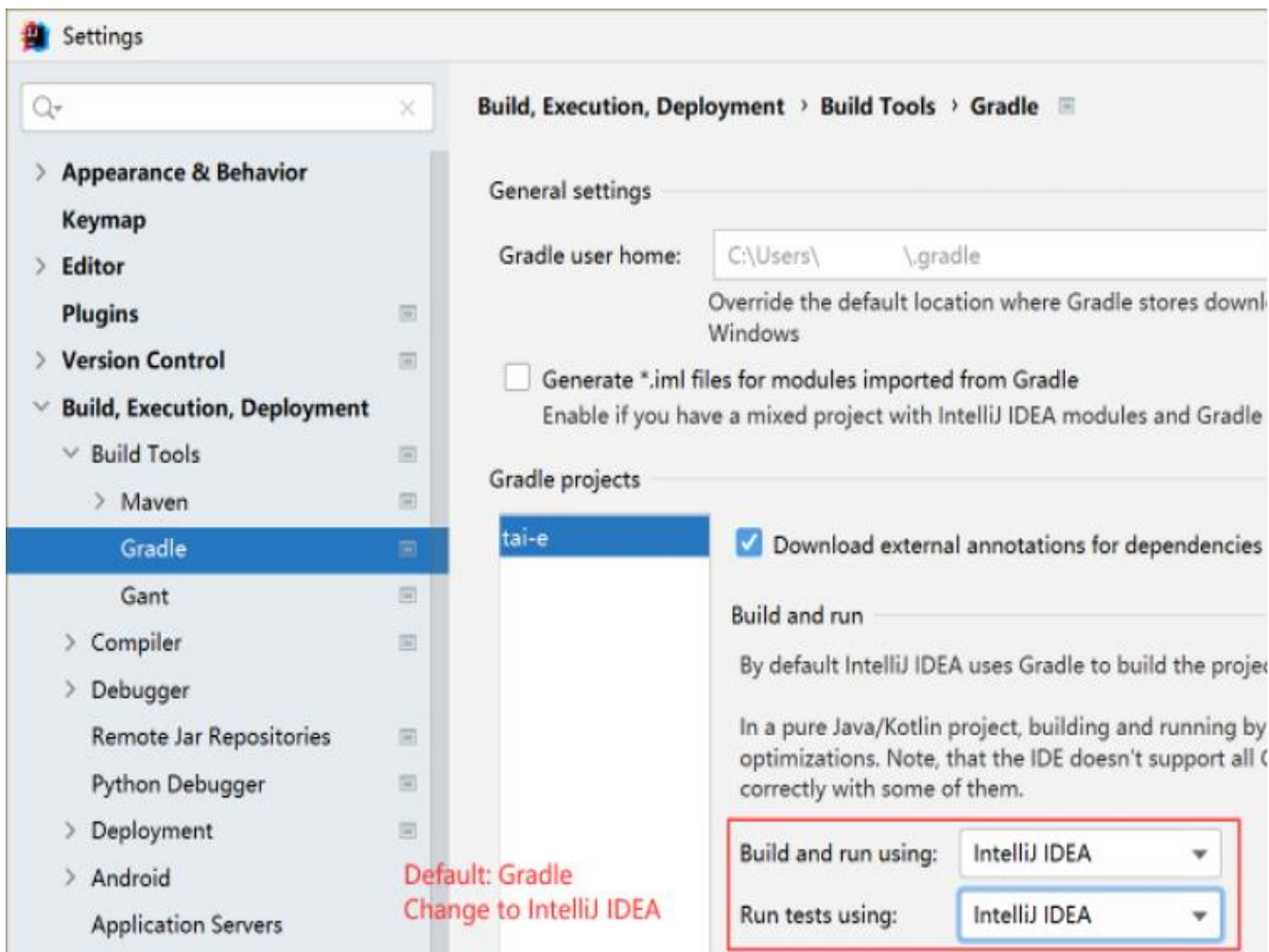


注：如果你已安装 JDK 17，在展开 SDK 下拉菜单后，直接选择你安装的 JDK 并选择 Language level 即可。如果你的 IntelliJ IDEA 已经选择 JDK 17 和 Java 17 作为默认 SDK 及 “Language level”，则配置后续作业时可跳过此步。

实验平台配置——Tai-e



■ 修改默认构建和运行工具



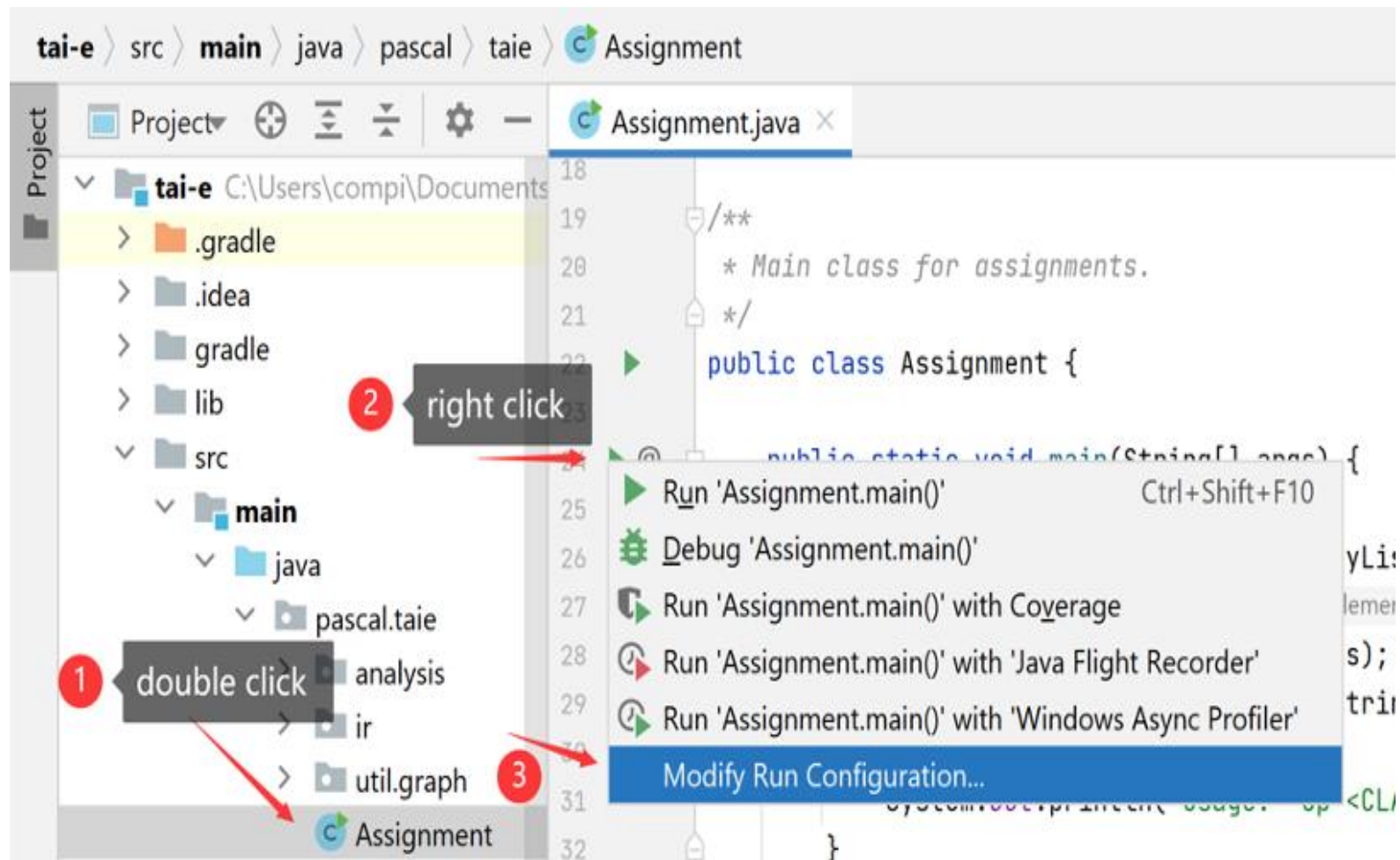
实验平台配置——Tai-e



■ 运行测试

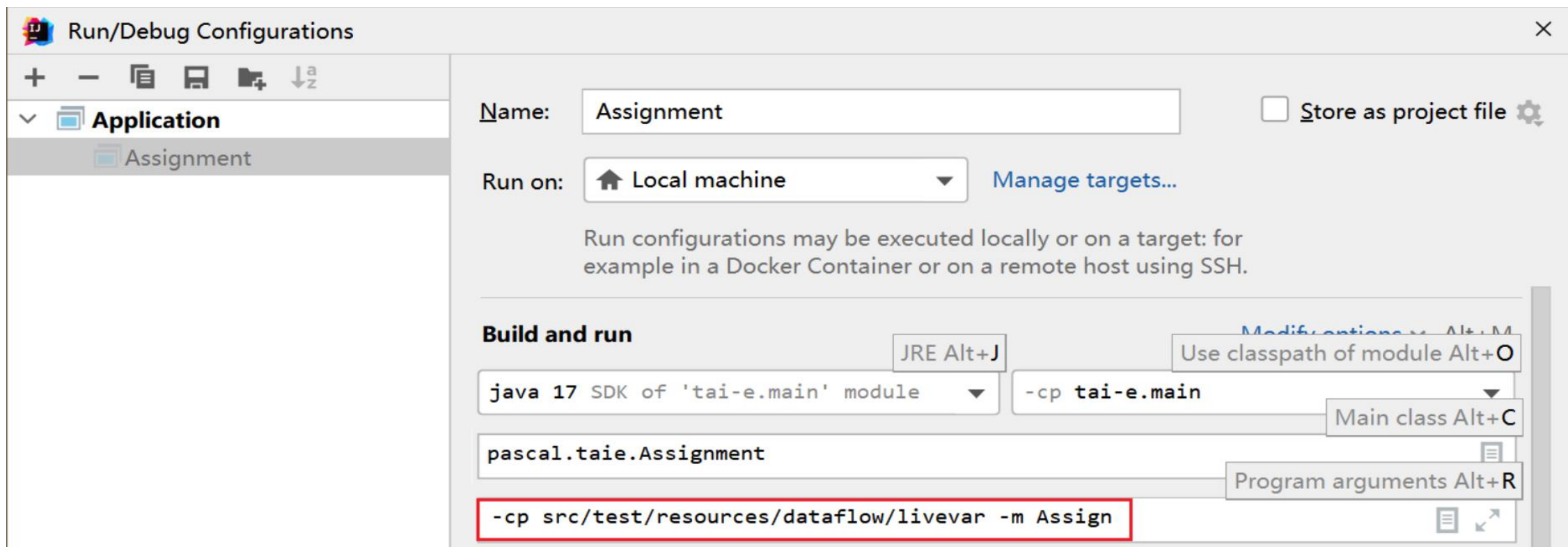
以分析

src/test/resources/dataflow/live
var 中的 Assign.java为例，首
先在 IntelliJ IDEA 中打开
Assignment 的 “Run
Configuration”：



■ 运行测试

配置 Program arguments: -cp <CLASS_PATH> -m <CLASS_NAME>



■ 运行测试

配置完成后，运行Assignment测试是否配置成功，若配置成功将得到如下输出

```
----- <Assign: int assign(int,int,int)> (livevar) -----  
[0@L4] d = a + b; null  
[1@L5] b = d; null  
[2@L6] c = a; null  
[3@L7] return b; null
```

完成代码后，运行Assignment测试代码是否有问题，正确的分析结果输出如下：

```
----- <Assign: int assign(int,int,int)> (livevar) -----  
[0@L4] d = a + b; [a, d]  
[1@L5] b = d; [a, b]  
[2@L6] c = a; [b]  
[3@L7] return b; []
```

■ 实现活跃变量分析算法

你的第一个任务是实现 LiveVariableAnalysis 中的如下 API:

- SetFact newBoundaryFact(CFG)
- SetFact newInitialFact()
- void meetInto(SetFact, SetFact)
- boolean transferNode Stmt, SetFact, SetFact)

你的第二个任务是实现上面提到的 Solver 类的两个方法:

- Solver.initializeBackward(CFG, DataflowResult)
- IterativeSolver.doSolveBackward(CFG, DataflowResult)

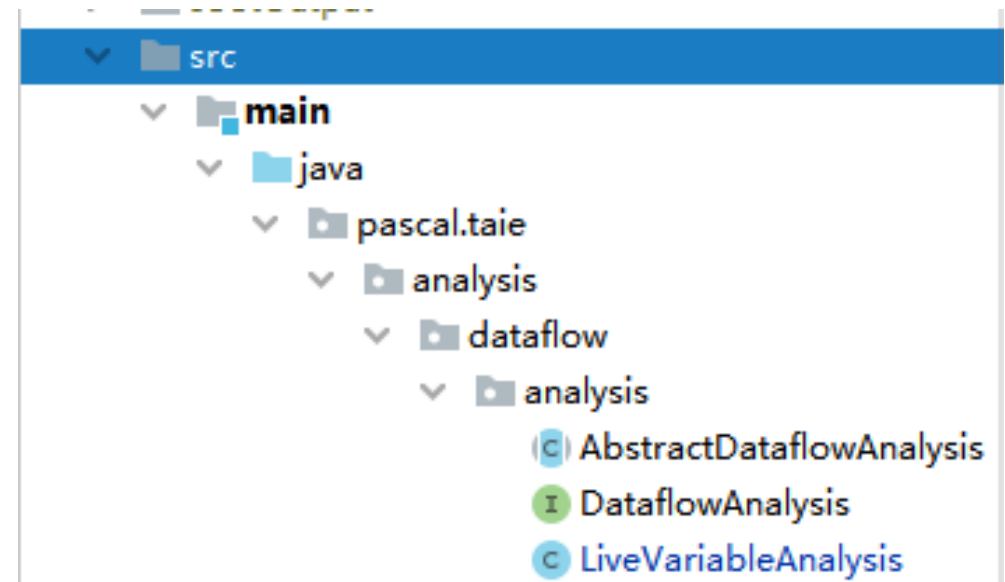
```
IN[exit] =  $\emptyset$ ; <- newBoundaryFact()
for (each basic block  $B \setminus \text{exit}$ )
    IN[B] =  $\emptyset$ ; <- newInitialFact()
```

```
while (changes to any IN occur)
    for (each basic block  $B \setminus \text{exit}$ ) {
        OUT[B] =  $\bigcup_{S \text{ a successor of } B} \text{IN}[S]$ ;
        meetInto() IN[B] =  $\text{use}_B \cup (\text{OUT}[B] - \text{def}_B)$ ;
    }
    transferNode()
```

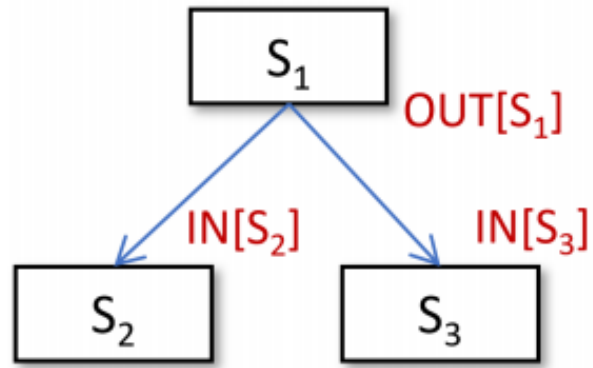
■ pascal.taie.analysis.dataflow.analysis.DataflowAnalysis

一个抽象类，用于具体的数据流分析方法（如本实验中的活跃变量分析等）实现其接口，本实验涉及五个关键API，对应着本实验 LiveVariableAnalysis 中具体要实现的部分：

- 分析方向
- 边界条件: `SetFact newBoundaryFact(CFG)`
- 初始条件: `SetFact newInitialFact()`
- Meet操作: `void meetInto(SetFact,SetFact)`
- Transfer函数: `boolean transferNode(Stmt,SetFact,SetFact)`



■ void meetInto(SetFact,SetFact) : 有点不同!



Equation: $OUT[S_1] = IN[S_2] \cup IN[S_3]$

meetInto():
 $OUT[S_1] \cup= IN[S_2] \text{ (meetInto(IN[S_2], OUT[S_1]))}$
 $OUT[S_1] \cup= IN[S_3] \text{ (meetInto(IN[S_3], OUT[S_1]))}$

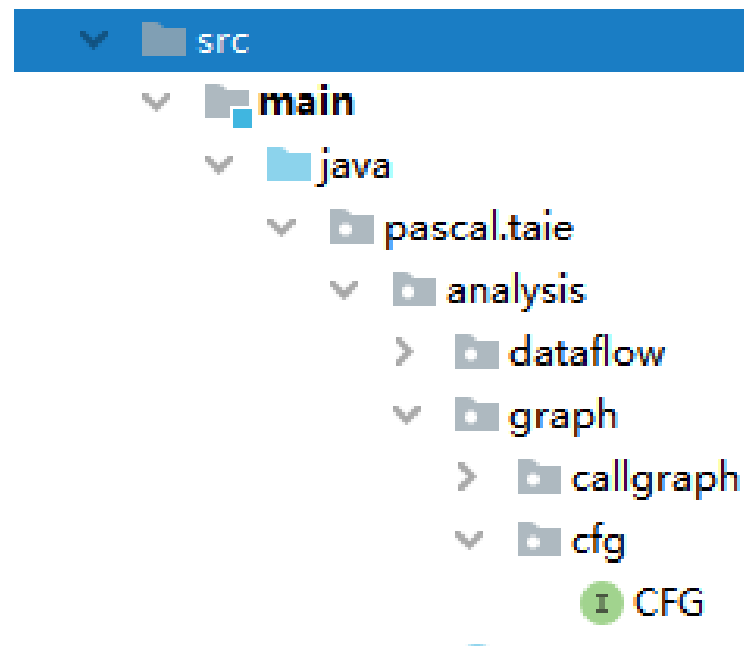
To support such meet strategy, you also need to give $OUT[S]$ an initial fact (the same initial value as in $IN[S]$) for each statement.

■ pascal.taie.analysis.graph.cfg.CFG

用来表示控制流图，其中getInEdgesOf(Node)和getOutEdgesOf(Node)分别返回当前节点的所有前驱和所有后继

```
/**
 * @return incoming edges of the given node.
 */
@Override
Set<Edge<N>> getInEdgesOf(N node);

/**
 * @return outgoing edges of the given node.
 */
@Override
Set<Edge<N>> getOutEdgesOf(N node);
```

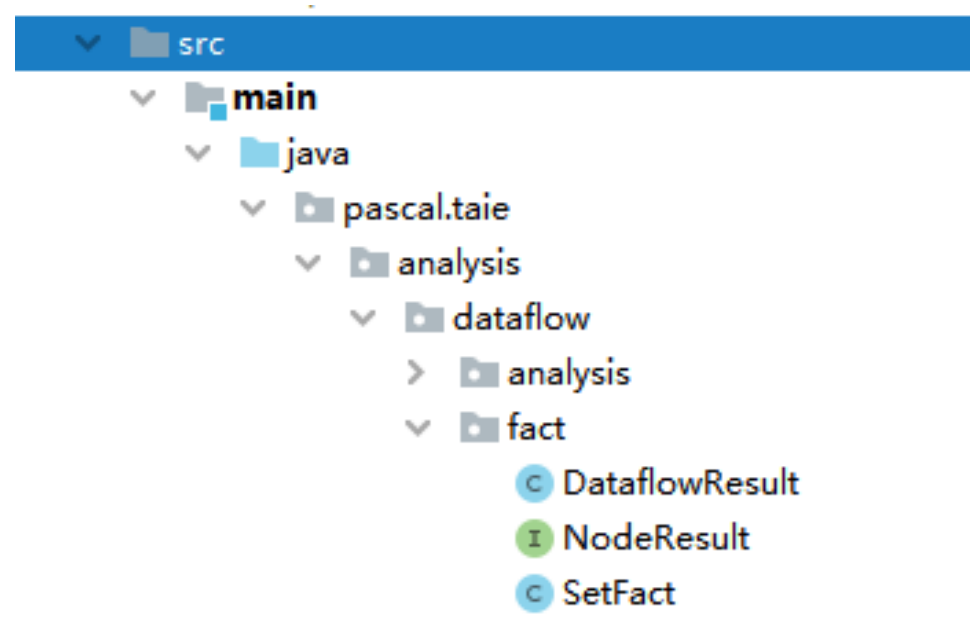


■ pascal.taie.analysis.dataflow.fact.DataflowResult

建立起每个节点与其IN集合和OUT集合之间的映射，分别提供对当前节点的IN/OUT集合的读写操作函数，方便在进行迭代过程中对节点IN/OUT集合的维护。

```
/**
 * @return the flowing-in fact of given node.
 */
@Override
public Fact getInFact(Node node) { return inFacts.get(node); }

/**
 * Associates a data-flow fact with a node as its flowing-in fact.
 */
public void setInFact(Node node, Fact fact) { inFacts.put(node, fact); }
```

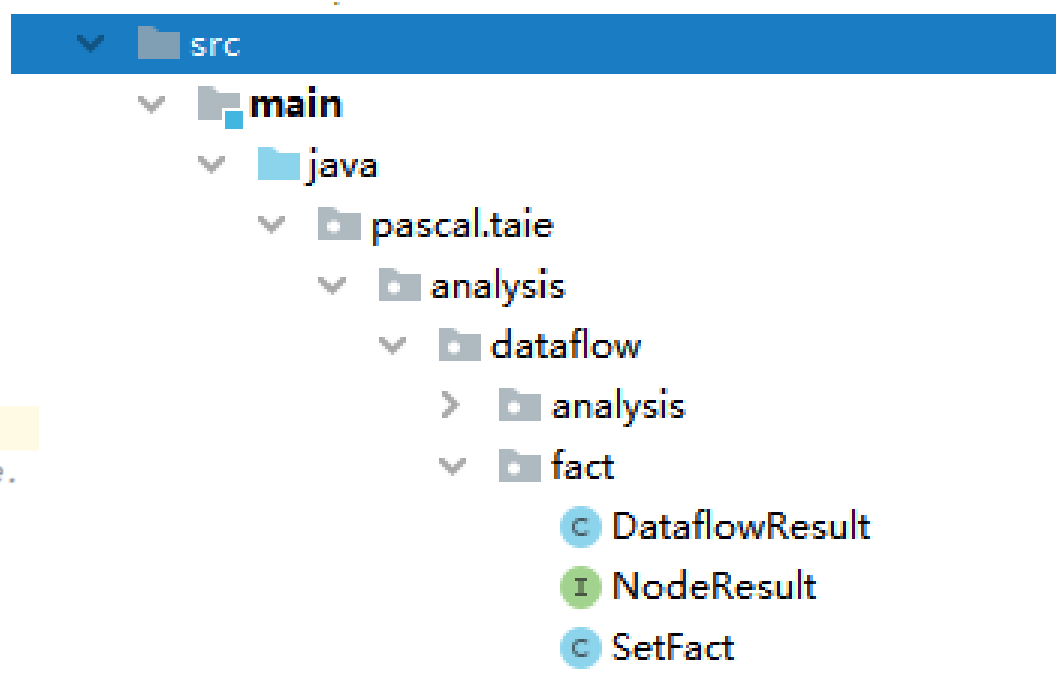


■ pascal.taie.analysis.dataflow.fact.SetFact

在维护节点IN/OUT集合过程中涉及到大量的集合操作，为方便，抽象出SetFact类，将对集合的各种操作集成到一起，包括增删元素、集合并操作等，具体看源码及注释

```
/**
 * Unions other fact into this fact.
 *
 * @return true if this fact changed as a result of the call, otherwise false.
 */
public boolean union(SetFact<E> other) { return set.addAll(other.set); }

/**
 * Removes an element from this fact.
 *
 * @return true if an element was removed as a result of the call, otherwise false.
 */
public boolean remove(E e) { return set.remove(e); }
```



■ pascal.taie.ir.exp

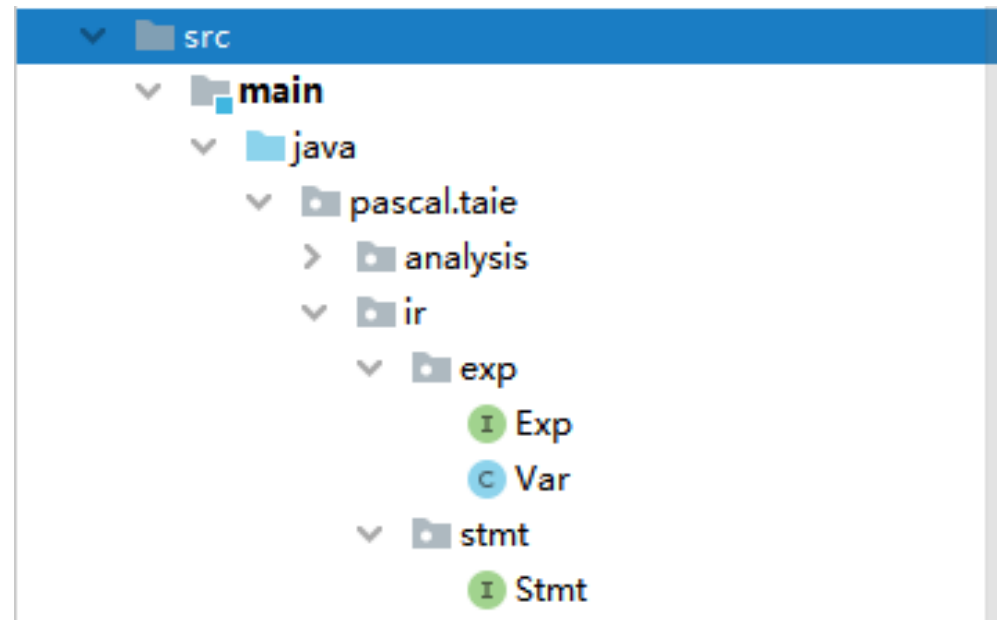
用来表示程序中所有的表达式，本实验只需关注Var类

■ pascal.taie.ir.stmt

用来表示程序中所有的语句，提供获取Def/Use变量的方法

```
/**
 * @return the (optional) left-value expression defined in this Stmt.
 * In Tai-e IR, each Stmt can define at most one expression.
 */
Optional<LValue> getDef();

/**
 * @return a list of right-value expressions used in this Stmt.
 */
List<RValue> getUses();
```



$a = b + d$

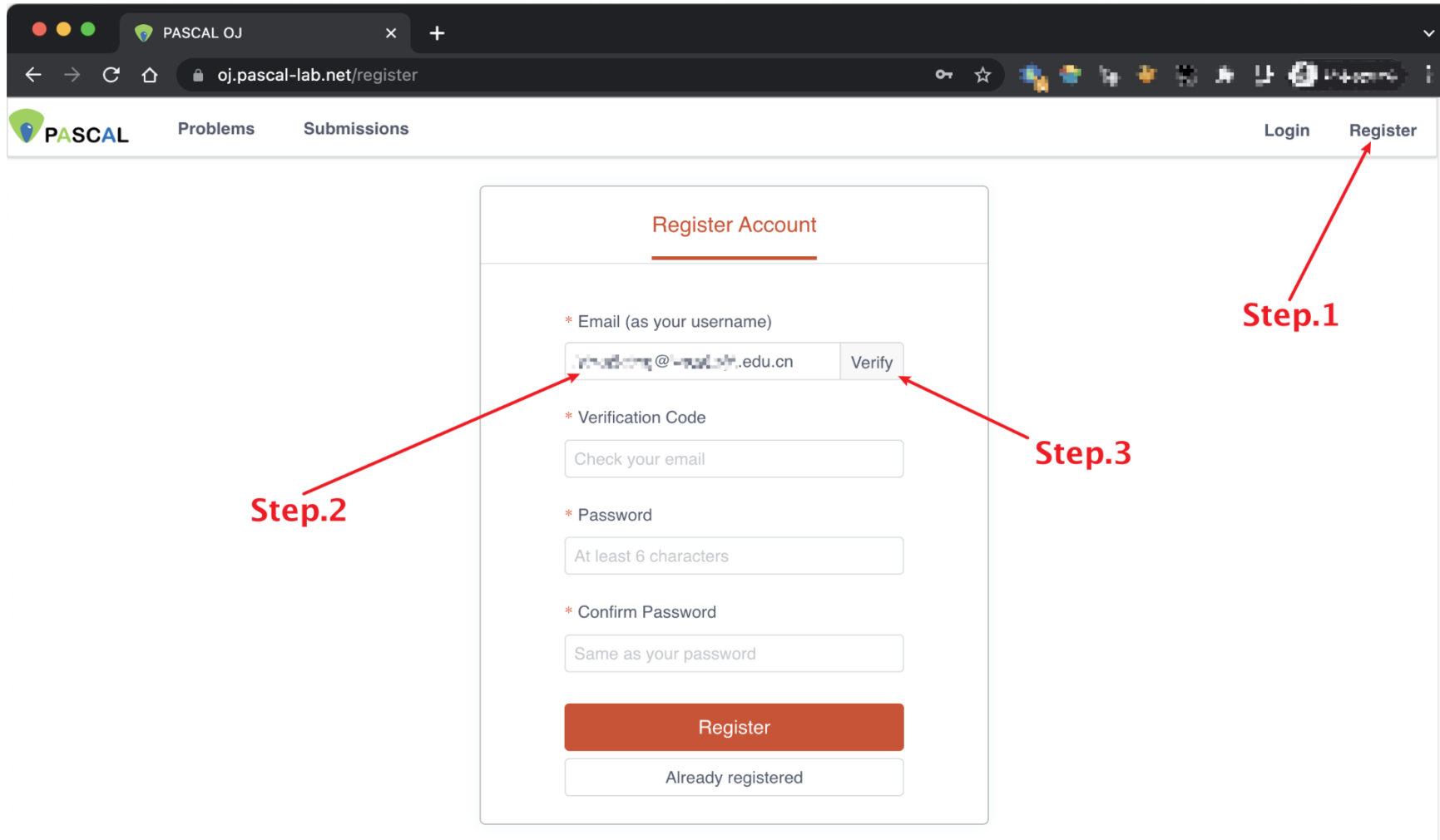
stmt.getDef(): a

stmt.getUses(): [b,d]

作业提交



■ 使用教育邮箱注册: <https://oj.pascal-lab.net/problem>



The screenshot shows the registration page of the PASCAL OJ system. The browser address bar displays `oj.pascal-lab.net/register`. The page header includes the PASCAL logo, navigation links for "Problems" and "Submissions", and buttons for "Login" and "Register". The "Register" button is highlighted with a red arrow labeled "Step.1". The main content area is titled "Register Account" and contains the following fields:

- * Email (as your username):** A text input field containing an email address ending in ".edu.cn". A red arrow labeled "Step.2" points to this field. To its right is a "Verify" button, which is pointed to by a red arrow labeled "Step.3".
- * Verification Code:** A text input field with the placeholder text "Check your email".
- * Password:** A text input field with the placeholder text "At least 6 characters".
- * Confirm Password:** A text input field with the placeholder text "Same as your password".

At the bottom of the form are two buttons: a prominent orange "Register" button and a white "Already registered" button.

作业提交



■ 在线测试平台: <https://oj.pascal-lab.net/problem>

提交一个zip文件, 包括实现好的以下类:

- LiveVariableAnalysis.java
- Solver.java
- IterativeSolver.java

PASCAL OJ

oj.pascal-lab.net/problem/tai-e-1

PASCAL Problems Submissions

Tai-e-1 : Assignment 1 - Live Variable Analysis and Iterative Solver

Description

评测大概需要 3 min, 期望的 zip 文件结构为:

```
A1.zip
├─ LiveVariableAnalysis.java
├─ Solver.java
└─ IterativeSolver.java
```

Assignment 1 (Java17)

Choose your Zip file

Submit

Details

Problem Code Tai-e-1

Recent Submissions

Result	Submit time
Show all submissions	

作业测试与提交



■ 提交测试

The screenshot shows a web browser window with the URL `oj.pascal-lab.net/problem/tai-e-1`. The page title is "Tai-e-1 : Assignment 1 - Live Variable Analysis and Iterative Solver". The interface includes a navigation bar with "PASCAL", "Problems", and "Submissions" tabs. The "Description" section contains the text: "评测大概需要 3 min, 期望的 zip 文件结构为:" followed by a directory tree:

```
A1.zip
├── LiveVariableAnalysis.java
├── Solver.java
└── IterativeSolver.java
```

On the right side, there are sections for "Details" (Problem Code: Tai-e-1) and "Recent Submissions" (Result, Submit time, Show all submissions). At the bottom, there is a dropdown menu showing "Assignment 1 (Java17)", a button labeled "Choose your Zip file" (indicated by a red arrow labeled "Step.1"), and a "Submit" button (indicated by a red arrow labeled "Step.2").



■ 测试通过

PASCAL OJ

oj.pascal-lab.net/submission/...

PASCAL

Problems

Submissions

✓ Accepted

Judge Log

Analyze 33 methods, pass 33 methods
There are 605 Stmtns in all test cases
Your submission correctly analyzes 605 Stmtns

Download Your Code

Submission

Assignment 1 - Live Variable Analysis and Iterat... >

Submission ID

...

Create Time

...

Judge Time

...

User

...

Judge Result

✓ Accepted

Judge Template

Assignment 1 (Java17)

■ 最终提交内容

- 测试截图
- 代码（zip文件）

将测试截图与代码.zip文件放在同一个文件夹下，再将其压缩为一个文件后上传

截止时间：2024-3-24 23:59

提交地址：

<https://send2me.cn/jLLpSxEg/TAa6q3cNmCQ8KQ>