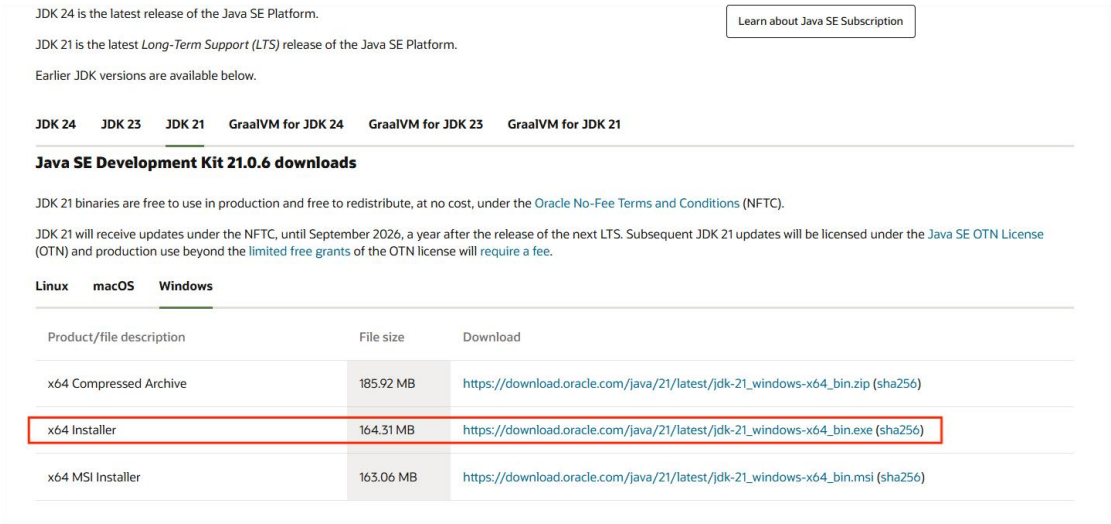
1. 配置环境

1）下载 JDK

进入 Oracle 官网：https://www.oracle.com/java/technologies/downloads/#jdk21-windows



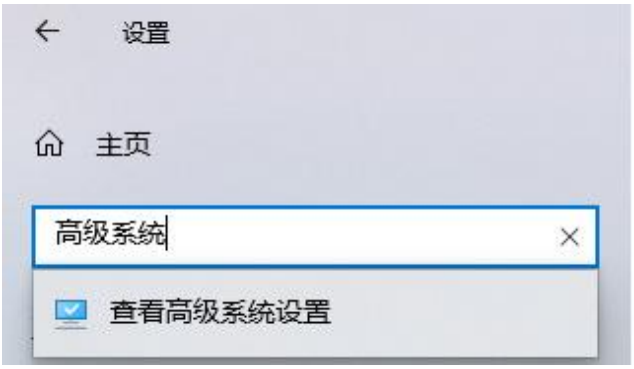
一路 next 点到底即可（**过程中出现安装 JRE 的弹窗请点同意**）

现在 JDK 安装包可以自动配置环境变量，如果不行就按下面手动配置一下（没有问题直接看下一步就行）

A.找到并记住 JDK 安装目录，如：C:\Program Files\Java\jdkxxx.xxx\

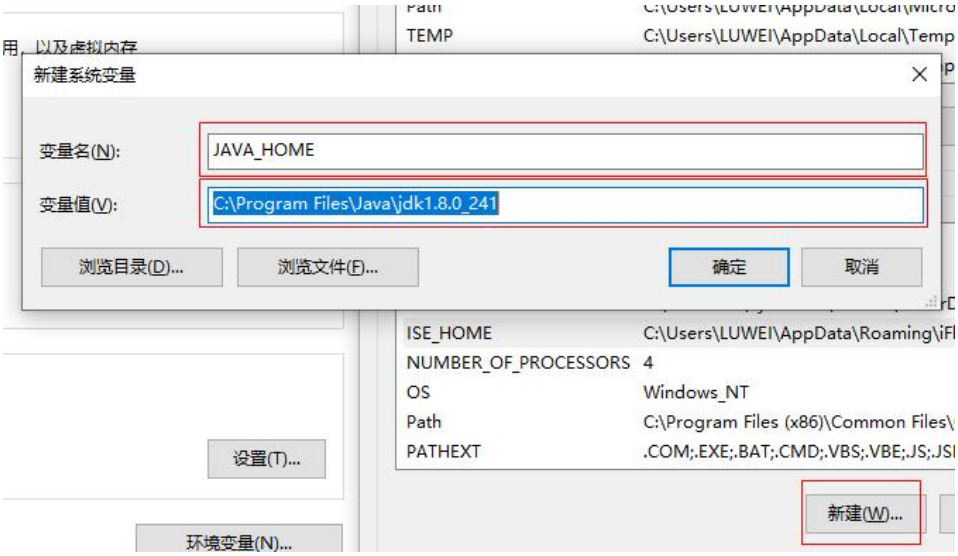
B.右键点我的电脑->属性->高级系统设置->高级->环境变量

或者如下图所示直接在设置里搜索

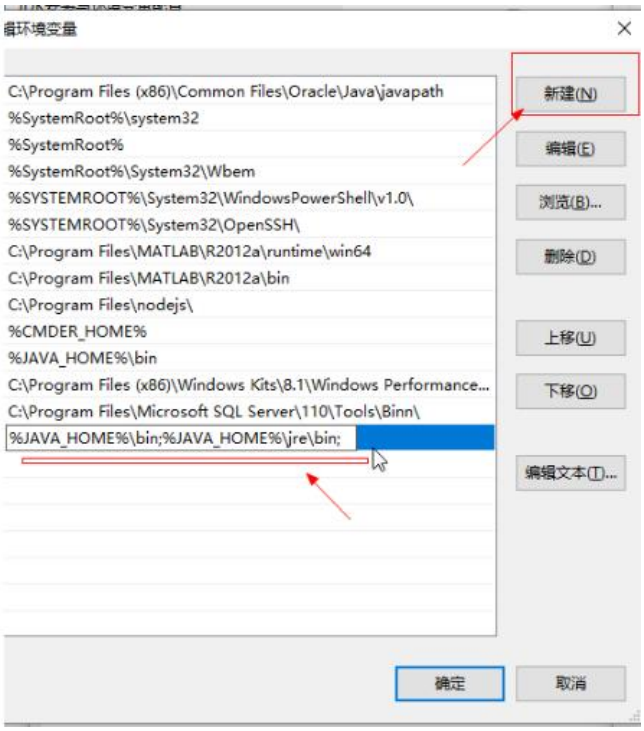


点击环境变量

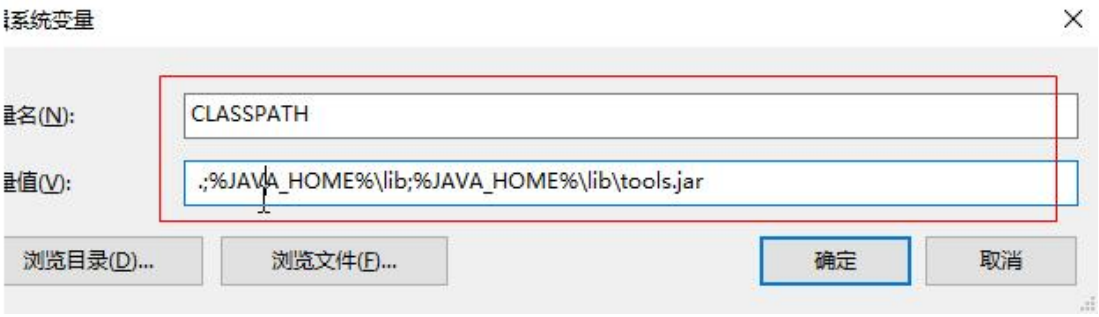
在系统变量中点新建，弹出的窗口中，变量名写 JAVA_HOME，变量值就是刚才的安装目录



C.在同一个地方找到 Path 变量，点编辑，然后新建，输入 %JAVA_HOME%\bin;%JAVA_HOME%\jre\bin;



D. 再在系统变量中新建一个 CLASSPATH 变量，变量值为 .;%JAVA_HOME%\lib;%JAVA_HOME%\lib\tools.jar



到这里环境变量应该就配置完了

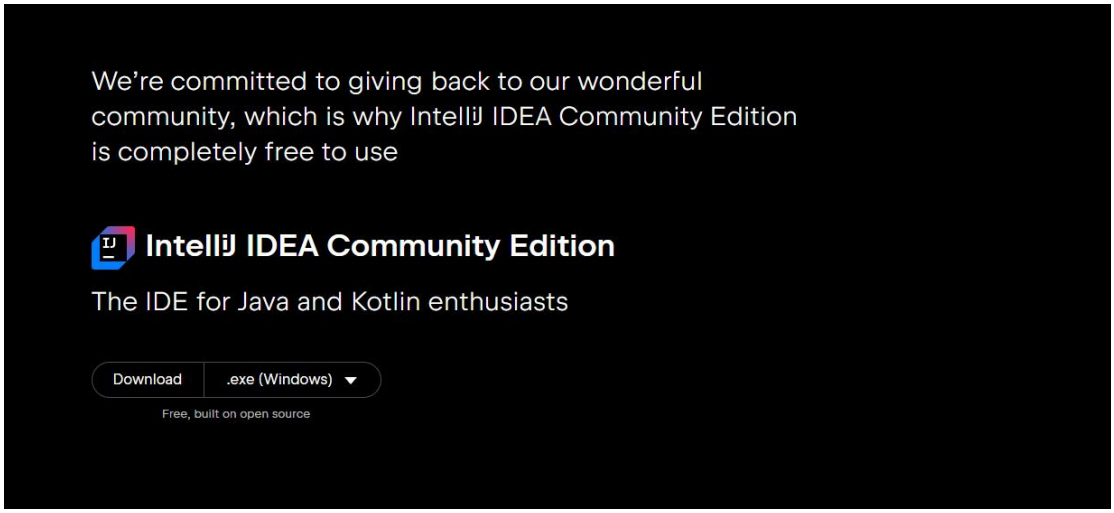
按 Win + R 打开 cmd，输入 java --version

```
C:\Users\yf wang>java --version
java 17.0.10 2024-01-16 LTS
Java(TM) SE Runtime Environment (build 17.0.10+11-LTS-240)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.10+11-LTS-240, mixed mode, sharing)
```

出现上述说明安装成功

2）下载 IDEA

进入官网：https://www.jetbrains.com/idea/download/?section=windows

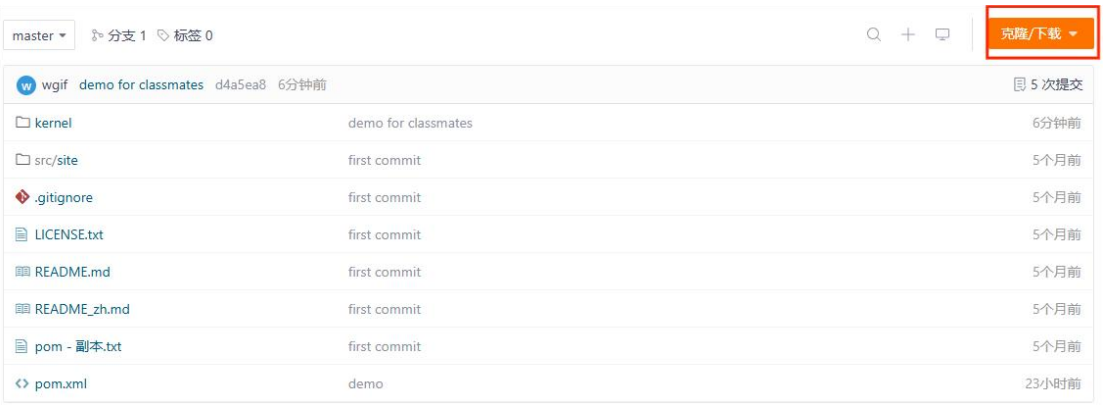


下拉找到 community 版本，下载并安装。

3）下载 CrowdOS

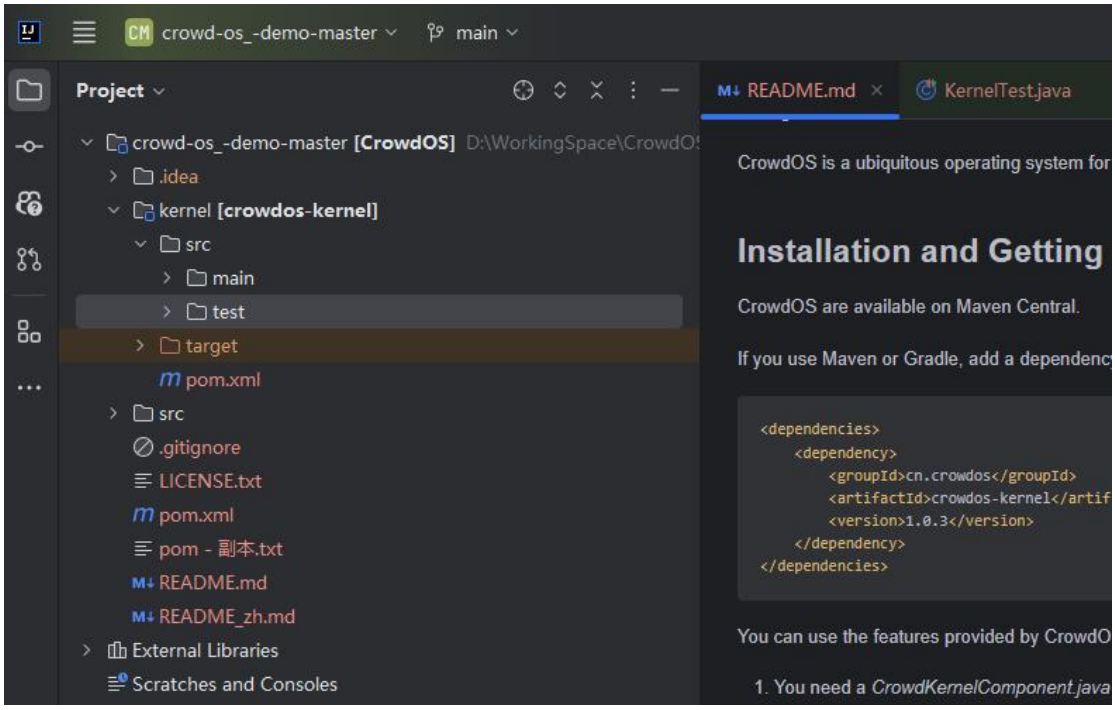
去 gitee 下载 CrowdOS 的 demo：https://gitee.com/wang-yifan10086/crowd-os_-demo

（如果熟悉 git 使用的话可以直接克隆）



点击克隆/下载，然后下载 zip 到本地

解压之后，用 IDEA 打开即可！

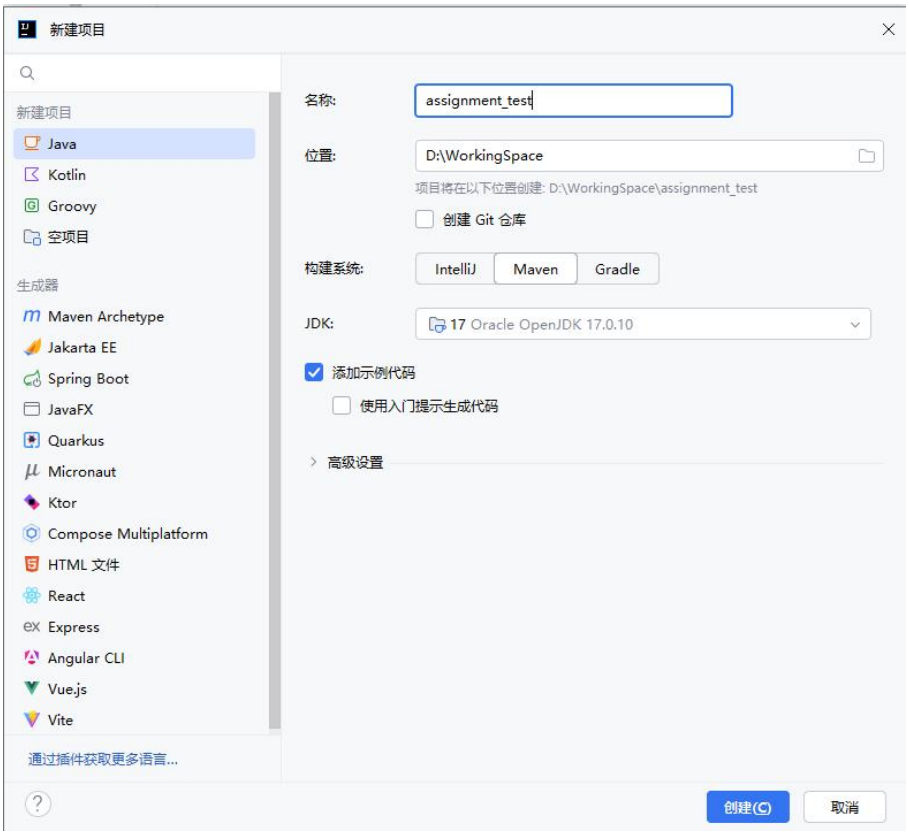


2. 简单任务——熟悉 java 与面向对象

介绍：让我们通过一个简单的任务来重新熟悉一下 java！我们先自己简单模拟一下群智感知中任务分配的基本过程！

要求：

(0) 在 IDEA 新建一个 java 项目



(1) 创建 Task 类，其中包含任务名称、参与者、任务进行时间段三个属性。

```
private String title;  2 个用法
private List<Participant> participants;  4 个用法
private List<Date> dateRange;  7 个用法
private SimpleDateFormat sdf;  4 个用法
```

(2) 创建 Participant 类，其中包含名字、空闲时间两个属性，以及 toString 方法（用来输出名字）。

```
public class Participant {  8 个用法
    private String name;  2 个用法
    private String date;  2 个用法
```

(3) 为两个类创建构造方法，初始化属性。

```
public Task(String title,String start,String end) throws ParseException {  1 个用法
    this.title = title;
    participants = new ArrayList<>();
    this.dateRange = new ArrayList<>();
    sdf = new SimpleDateFormat( pattern: "yyyy.MM.dd");
    this.dateRange.add(sdf.parse(start));
    this.dateRange.add(sdf.parse(end));
}
```

```
public Participant(String name, String date) {  2 个用法
    this.name = name;
    this.date = date;
}
```

(4) 为 Task 类设计 canAssinTo()方法用来判断参与者能否参与任务（空闲时间满足任务进行时间即可）；addParticipant()方法将满足条件参与者添加到任务的参与者列表中。
提示：用 date.compareTo 方法对日期进行比较

(5) 为 Task 类设计 getParticipants 方法，返回参与者列表（或者字符串，随便你喽）。

(6) 创建一个主函数，在其中创建新的参与者与任务，进行任务分配吧！

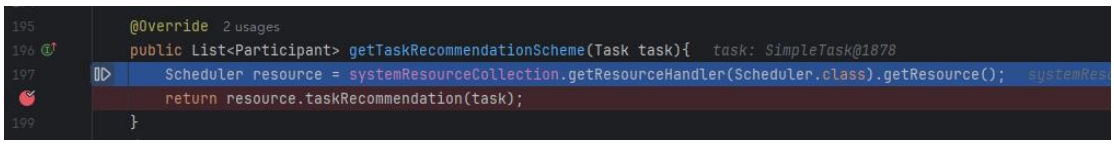
```
public static void main(String[] args) throws ParseException {
    List<Participant> participants = new ArrayList<Participant>();
    Task t1 = new Task( title: "SimpleTask", start: "2025.3.15", end: "2025.3.25");
    Participant p1 = new Participant( name: "虎哥", date: "2025.3.20");
    Participant p2 = new Participant( name: "刀哥", date: "2025.3.26");
    Participant p3 = new Participant( name: "团长", date: "2025.3.17");
    Participant p4 = new Participant( name: "小高", date: "2025.3.19");
    participants.add(p1);
    participants.add(p2);
    participants.add(p3);
    participants.add(p4);
```

3. 任务 2——熟悉 CrowdOS

介绍：刚刚我们自己模拟了简单的任务分配流程，接下来看看在 CrowdOS 系统中，任务分配是如何进行的吧！

要求：
(1) 用 IDEA 打开 CrowdOS_Demo
(2) 依次进入 kernel\src\test\ cn.crowdos.kernel\KernelTest
(3) 模板中已经创建好了 2 个 TimeParticipant 并注册到 kernel，并且耶创建并注册了一个简单的任务。请大家依葫芦画瓢的再创建 5 个 TimeParticipant、1 个 SimpleTask 并注册到 kernel，然后运行 getTaskRecommendationScheme 方法，看看任务分配情况。
(4) 大家自行打上断点，跟踪程序运行过程，了解 CrowdOS 分配任务的初步逻辑，为下一个任务做好准备。
提示：分别在下面几个地方打上断点

① Kernel 中的 getTaskRecommendationScheme 方法



② Scheduler 中的 taskRecommendation 方法

```
public List<Participant> taskRecommendation(Task task){  2 usages          task: SimpleTask@1878
    return taskRecommendationAlgo.getRecommendationScheme(task);   task: SimpleTask@1878     taskF
}
```

③ AlgoFactoryAdapter 中的 getRecommendationScheme 方法

```
@Override  2 usages
public TaskRecommendationAlgo getTaskRecommendationAlgo() {
    return new TaskRecommendationAlgo() {
        final ParticipantPool participantPool;  2 usages          participantPool:  size = 6
        {
            participantPool = resourceCollection.getResourceHandler(ParticipantPool.class).getResourceView();
        }
        @Override  2 usages
        public List<Participant> getRecommendationScheme(Task task) {   task: SimpleTask@1878
            List<Participant> candidate = new LinkedList<>();
            for (Participant participant : participantPool) {
                if (participant.available() && task.canAssignTo(participant)){
                    candidate.add(participant);
                }
            }
            return candidate;
        }
    };
}
```

④ AbstractTask 中的 canAssignTo 方法

```
public boolean canAssignTo(Participant participant) {   participant: "P(2024.06.01)"
    for (Constraint constraint : constraints) {   constraints:  size = 1
        Class<? extends Condition> conditionClass = constraint.getConditionClass();
        if(participant.hasAbility(conditionClass)){
            if (!constraint.satisfy(participant.getAbility(conditionClass)))
                return false;
        }
    }
    return true;
}
```

(5) 探究问题：

如图所示，为什么任务要求时间是”2024.6.1-2024.6.2”,推荐结果只有”P(2024.6.1)”1 个参与者？试着找出对应代码段！

```
SimpleTimeConstraint timeConst = new SimpleTimeConstraint("2022.6.1", "2022.6.2");
```

```
[P(2024.06.01)]
```

4. 任务 3——限制条件为 POI 的任务分配

介绍：群智感知中，有很多任务都是基于 POI（position of interest）进行任务分配的，请大家也试着创建一个简单的任务，规定一个任务范围，然后看看 CrowdOS 能不能将任务分配到范围内的参与者吧！

要求：

(1) 在 Kernel\src\main\java\cn.crowdos.kernel\目录下中找到 Condition 和 Constraint 接口的定义，并思考 Condition 与 Constraint 的关系

(2) 在 kernel\src\test\ cn.crowdos.kernel\common 中，参照 TimeParticipant 的形式创建新的参

与者类 PoiParticipant，思考并确定能力应该是什么。

```
public class TimeParticipant extends AbstractParticipant {
    @ability  3 个用法
    final DateCondition activeTime;
```

(3) 参照 TimeParticipant 的形式，在构造方法中，对能力属性和参与者状态进行初始化

```
public TimeParticipant(String time) {  8 个用法  wgif
    format = new SimpleDateFormat( pattern: "yyyy.MM.dd");
    try {
        this.activeTime = new DateCondition(format.parse(time).getTime());
    } catch (ParseException e) {
        throw new RuntimeException(e);
    }
    status = ParticipantStatus.AVAILABLE;
}
```

(4) 参照 TimeParticipant 的形式，覆写父类的三个方法

```
@Override  3 个用法  wgif
public boolean hasAbility(Class<? extends Condition> conditionClass) {
    return conditionClass == DateCondition.class;
}

@Override  wgif
public Condition getAbility(Class<? extends Condition> conditionClass) {
    if (!hasAbility(conditionClass))
        return null;
    else return activeTime;
}

@Override  wgif
public String toString() {
    return "P(" + format.format(activeTime) + ")";
}
```

(5) 参照 KernelTest 的形式，新建一个 ParticipantTest,同样也是创建参与者，创建任务，然后进行任务分配吧！

5. 附加任务 1——如果你成功完成上面的任务

如果你觉得以上都是小菜一碟，那不妨试着自己创建一个任务，自定义更复杂 constraint（比如同时限制时间和地点），并试着进行新的任务分配！

6. 附件任务 2

不会吧！您竟然还觉得简单！那不如告诉你，其实我们之前调用的 getTaskRecommendationScheme()方法并不是真正的任务分配，它只是判断参与者的 condition 是否满足任务的 constraint。

您可以试着在注册任务之后，加入 kernel.algoSelect("GGA_I");来设定任务分配算法，并调用 getTaskAssignmentScheme()方法，这个方法会用算法工厂中的任务分配算法进行真正的任务分配，您可以试着看看算法工厂的运行逻辑（我们还在 GGA_I 算法中留下了一个 bug，看看能不能找到！）