使用说明

- 建议采用给出的`RandomStoreGenerator`以及`RandomRoborGenerator`两个生成器，下面给出范例

```
rsg = RandomStoreGenerator(13,15,1)
# 初始化构造，以n=13，m=15生成仓库，初始点间隔skip=1个走道生成
store = rsg.generate_store()
# 生成仓库实例
rsg.generate_goods(store)
rsg.generate_supplys(store)
# 随机生成货物和补货方案

rbg = RandomRobotGenerator(store, 0, store.start_ms[0], 5, 4)
# 初始化构造器，在(0,store.start_ms[0])的位置生成智能体，规定随机生成5个订单，每个订单的sku不超过4
robot = rbg.generate_robot()
# 生成智能体实例
```

- 智能体有三种执行方案

```
store.start_tasks(0,1,False)
# 只出货，使用该方法，不要修改0，1的值，最后一个参数表示是否打印每一步骤
store.start_supplys(0,1,False)
# 只上货，使用该方法，不要修改0，1的值，最后一个参数表示是否打印每一步骤
store.round_robin(0.5,0.8,False)
# 交替进行，第一个参数为保有率低于0.5时进行上货，第二个参数为保有率高于0.8时进行出货
```

- `Task`和`Supply`的规范和含义

```
Task: ((n,m),{sku:1, sku:1, sku:1, ... , sku:1})
# 表示出发点位于(n,m)，订单内容包括的sku（这里的value值按照原意应该始终为1，保留规范以适应可能后续）

Supply: (((n,m),target),{sku:1, sku:1, sku:1, ... , sku:1})
# 表示出发点位于(n,m)，将货物运送到target列的货柜
```

- 关于读入 由于尚不清楚详细规范，后续想使用此功能，请仿照`RandomStoreGenerator`中的`generate_goods`和`generate_supplys`方法，实现读入功能。同理，仿照`RandomRobotGenerator`中的`generate_tasks`方法，实现读入功能。