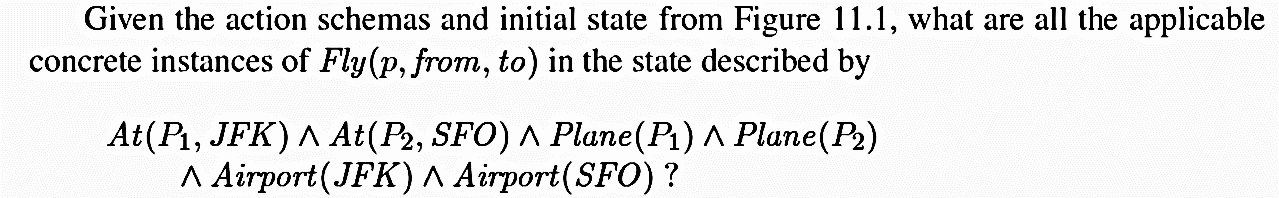
**Exercise 1**

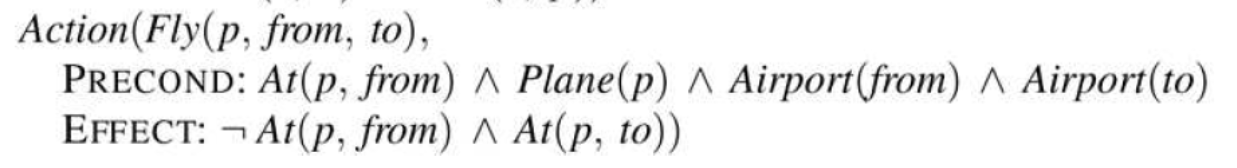
**问题解决与规划的区别和相似之处​​**

​​区别​​：问题解决聚焦于​​消除现有障碍或达成特定结果​​，而规划侧重于​​制定未来行动路径​​（如长期目标或资源分配）。问题解决强调​​即时性和灵活性​​（应对突发问题），规划则注重​​长期性和有序性​​（分阶段推进）。

​​相似性​​：两者均需明确目标，并通过​​分解任务​​实现（如问题拆解或步骤设计）；均依赖​​反馈机制​​，需根据执行情况修正方案或规划；均需根据现有资源，并分析可行性。

**Exercise 2**

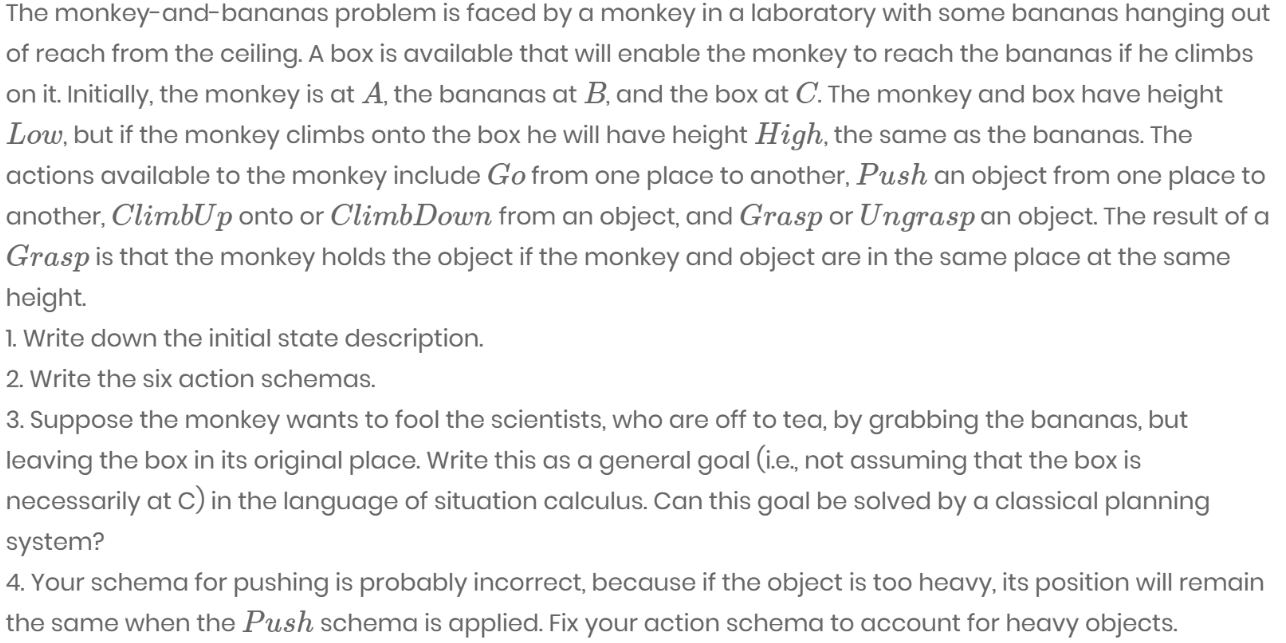
****

****

Fly(P1, JFK, SFO), Fly(P2, SFO, JFK)都是可行的动作

(若接受飞机不动，则可行的动作还有Fly(P1, JFK, JFK) , Fly(P2, SFO, SFO))

**Exercise 3**



1. 语义解释：At(obj, A): obj在A处

Height(obj, heit): obj的高度为heit

Holds(monkey, obj): monkey拿着obj

初始状态：

Init( At(monkey, A)∧At(bananas, B)∧At(box, C)∧Height(monkey, Low)∧Height(box, Low∧Height(bananas, High) ∧ ¬ Hold(monkey, bananas) ∧ ¬ Hold(monkey, box))

2.

Action(Go(from, to),

PRECOND: At(monkey, from)∧Height(monkey, Low)

EFFECT: ¬At(monkey, from)∧At(monkey, to)

)

Action(Push(obj, from, to),

PRECOND: At(monkey, from)∧At(obj, from)∧Height(monkey, Low)∧ Height(Object, Low)

EFFECT: At(monkey, to)∧At(obj, to)∧ ¬At(monkey, from) ∧ ¬At(obj, from)

)

Action(ClimbUp(obj),

PRECOND: At(monkey, place)∧At(obj, place)∧Height(monkey, Low)∧Height(obj, Low)

EFFECT: Height(monkey, High)

)

Action(ClimbDown(obj),

PRECOND: At(monkey, place)∧At(obj, place)∧Height(monkey, High)∧Height(obj, Low)

EFFECT: Height(monkey, Low)

)

Action(Grasp(obj),

PRECOND: At(monkey, place)∧At(obj, place)∧Height(monkey, h)∧Height(obj, h)∧¬Hold(monkey, obj)

EFFECT: Hold(monkey, obj)

)

Action(Ungrasp(obj),

PRECOND: Hold(monkey, obj)

EFFECT: ¬Hold(monkey, obj)

)

3. Goal(Hold(monkey, bananas)∧At(box, InitialPosition) ) 其中InitialPosition为盒子的初始位置。

此问题可以由经典规划解决，假设InitialPos为盒子的初始位置，可规划动作序列为：

Go(A, InitialPos)，Push(box, InitialPos, B)，ClimbUp(box)，Grasp(bananas)，ClimbDown(box)，Push(box, B, InitialPos)

4. 增加一个属性Heavy(obj): obj是沉重的

修改Push的动作模式

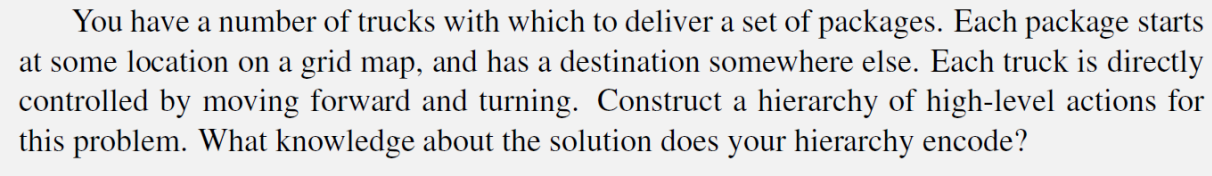
Action(Push(obj, from, to),

PRECOND: At(monkey, from)∧At(obj, from)∧Height(monkey, Low)∧ Height(Object, Low)∧¬Heavy(obj)

EFFECT: At(monkey, to)∧At(obj, to)∧ ¬At(monkey, from) ∧ ¬At(obj, from)

)

**Exercise 4**

（1）使用PDDL定义一组基元动作：

前进Forward (t)，左转TurnLeft(t)，右转TurnRight(t)

装货Load(p,t)，卸货Unload(p,t)

其中p是货物，t是卡车；动作模式具体定义略

（2）定义一些谓词：

Truck(t): t是卡车

Package(p): p是包裹

At(obj, [x,y])：obj在坐标[x, y]上

Destination（p，[x′, y′]): 包裹p的目的地为[x′, y′]

（3）定义高层动作HLA：

Nagivate(t，[x，y]）: 将卡车t开到坐标[x, y]

Deliver(t，p)：将包裹p使用卡车t的运送到p的目的地

Refinement(Deliver (t, p),

PRECOND:Truck(t) ∧ Package(p) ∧ At(p, [x, y]) ∧ Destination(p, [x ′ , y′ ])

STEPS: [Navigate(t, [x, y]), Load(p, t), Navigate(t, [x ′ , y′ ]), Unload(p, t)] )

Refinement(Navigate(t, [x, y]),

PRECOND:Truck(t) ∧ At(t, [x, y])

STEPS: [] )

Refinement(Navigate(t, [x, y]),

PRECOND:Truck(t)

STEPS: [Forward(t), Navigate(t, [x, y])] )

Refinement(Navigate(t, [x, y]),

PRECOND:Truck(t)

STEPS: [TurnLeft(t), Navigate(t, [x, y])] )

Refinement(Navigate(t, [x, y]),

PRECOND:Truck(t)

STEPS: [TurnRight(t), Navigate(t, [x, y])] )

（4）编码知识：

卡车一次只能携带一个包裹的知识；

卡车需要在目的地放下包裹；

两辆或以上卡车不能同时在一个坐标格中；

前方格中有其他卡车时，卡车不能前进。