PDDL实验

本节安排

- PDDL简介
- 规划器使用和语法
- 实验任务

PDDL简介

- PDDL = **P**lanning **D**omain **D**efinition **L**anguage
- 是对人工智能规划语言进行标准化的一种尝试。PDDL提供了标准化，好处是能使研究**更易于重用**和比较。
- 与针对特定论域的系统相比，这要付出一些表达能力的代价。

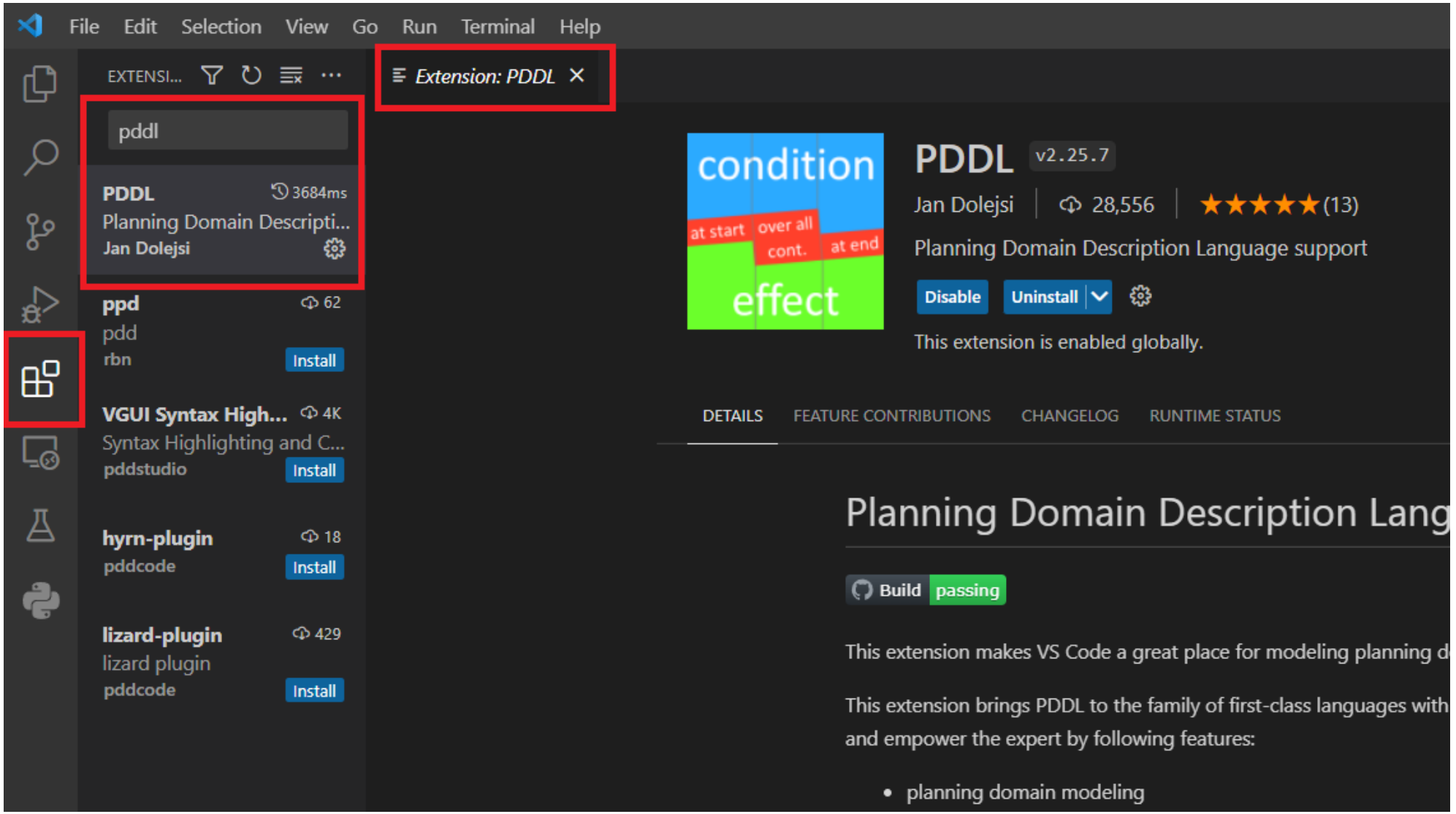
- 更多内容：
  - PDDL Domain - Planning.wiki 包含论域、问题定义、依赖项的例子
  - STRIPS — AI Planning in PDDL-Descripted World (tridu33.github.io) pddl基本介绍和例子
  - planning.domains 线上规划网站（不稳定）

实验环境

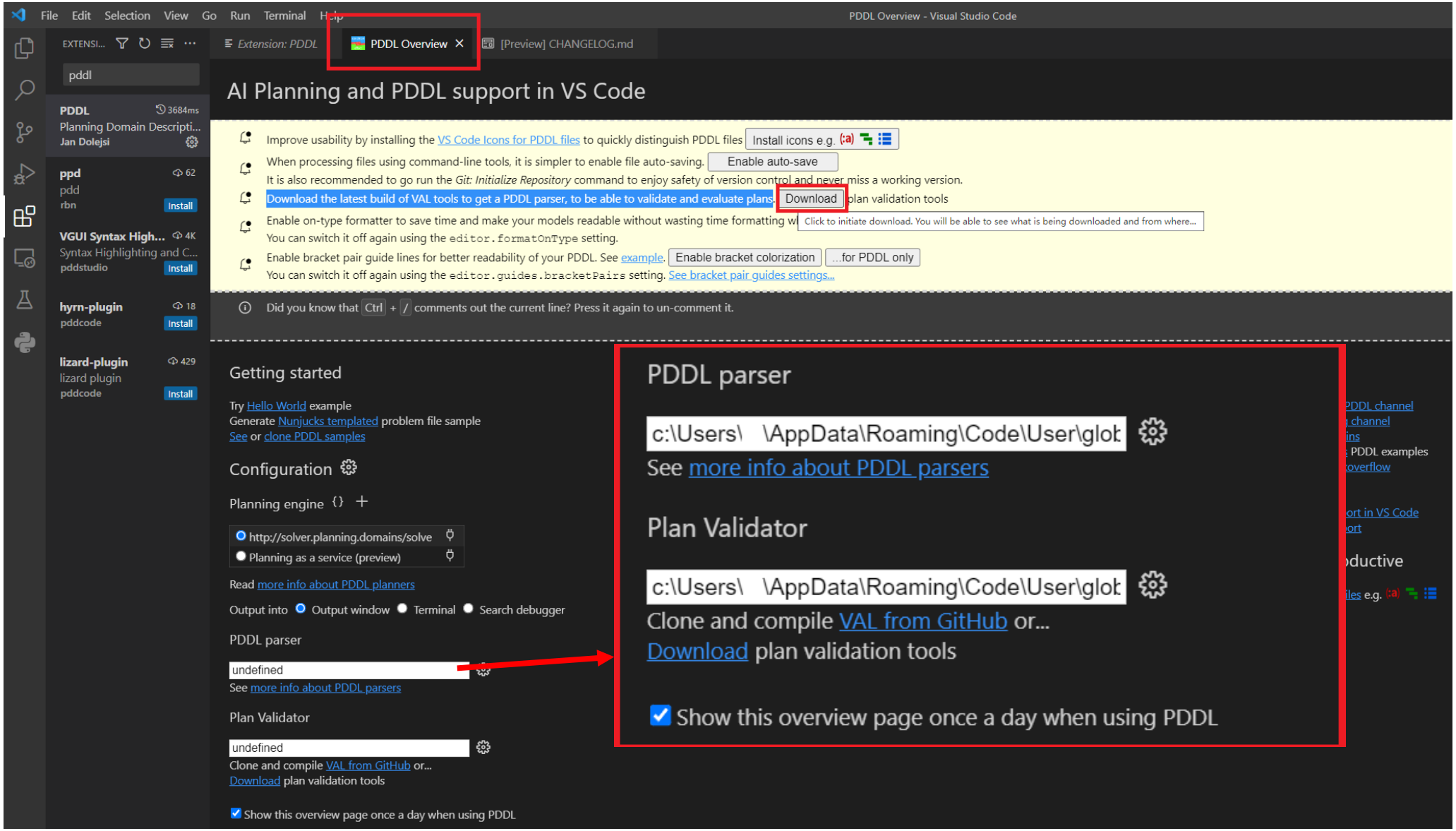
- FF （Fast Forward）规划器 链接
  - 采用经典的前向搜索方法
  - 结合启发式算法提高效率
  - 不保证最优性
- 线上使用 http://editor.planning.domains
- VSCode插件（详解）
- Linux可安装FFv2.3.tgz（详见压缩包文档）

PDDL 环境及语法

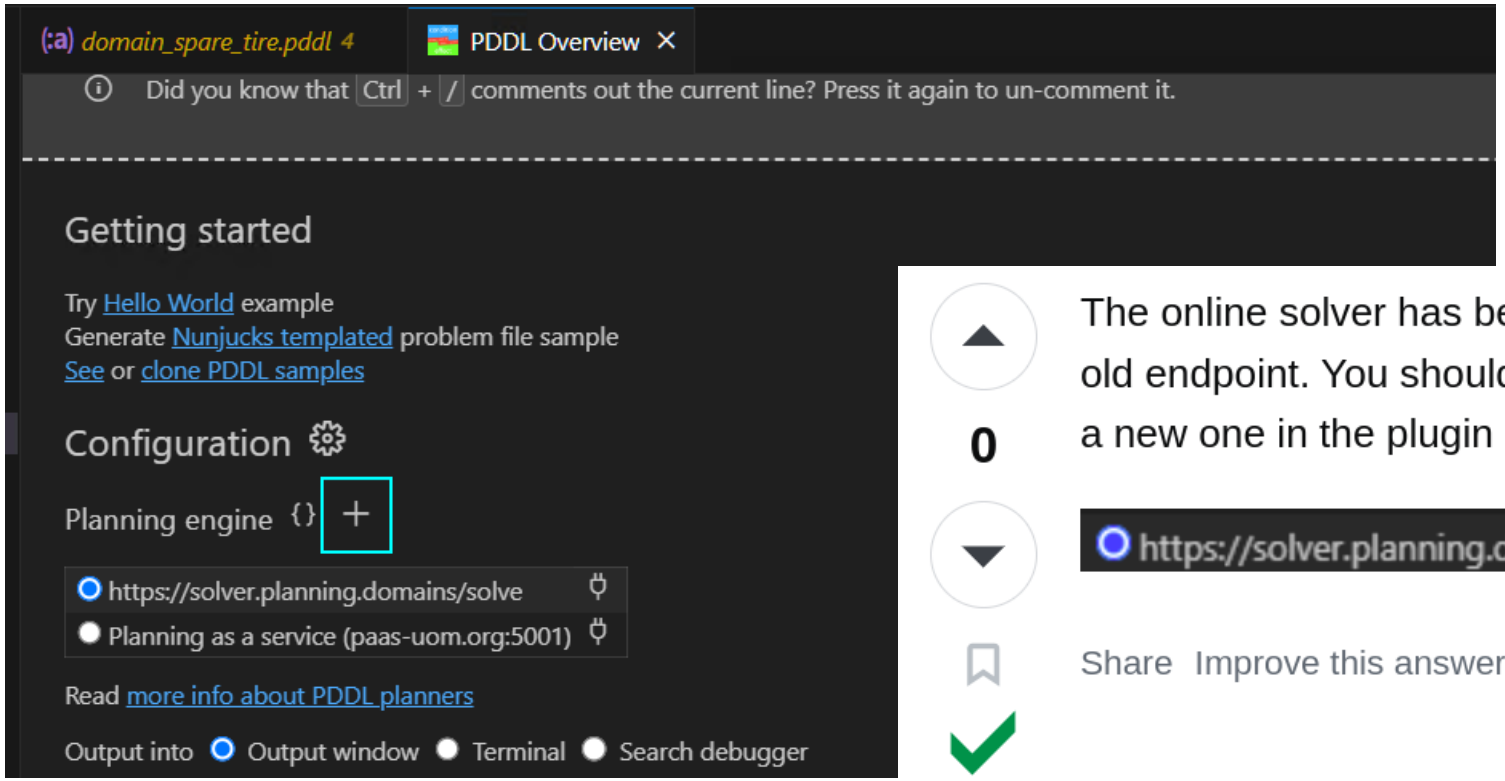
- Step 1 [安装]： 安装pddl插件

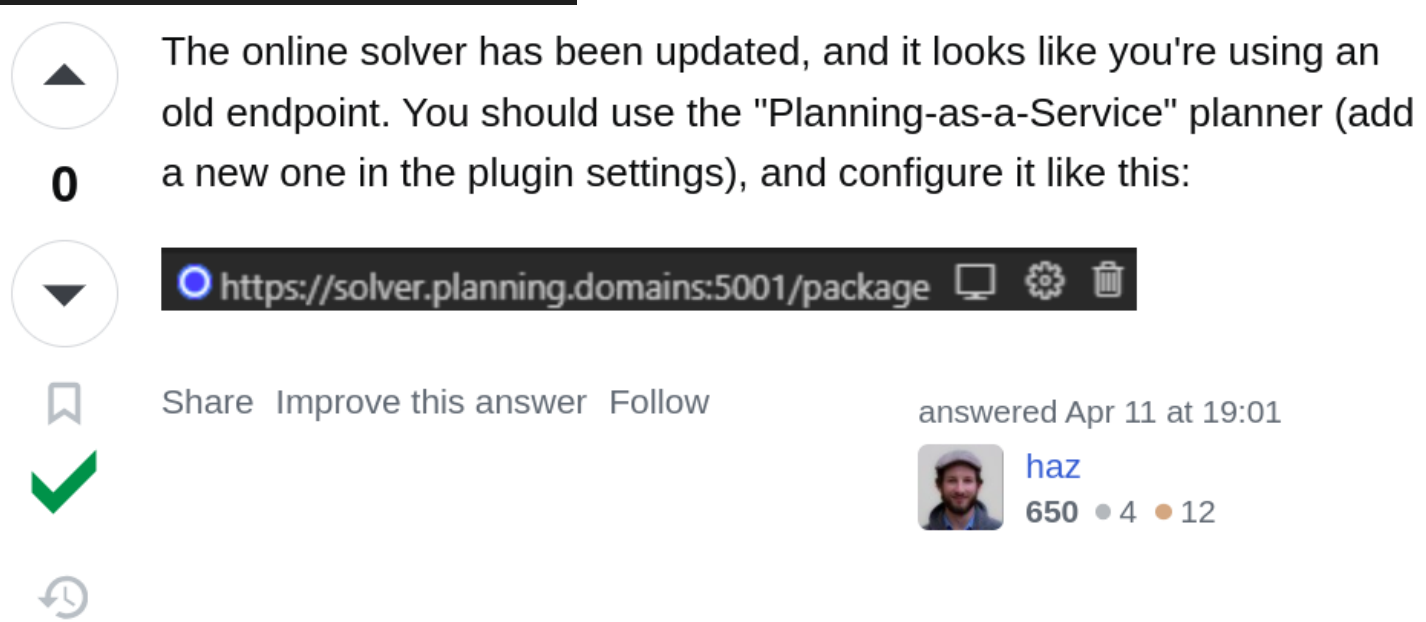


- Step 2 [配置]： Ctrl+shift+p 搜索 pddl: Show overview page
  - 下载VAL（一键自动下载），用于辅助解析和验证



- Step 2 [配置]：Ctrl+shift+p 搜索 pddl: Show overview page
  - Planning engine->Planning-as-a-service->填入下面的网址
  - https://solver.planning.domains:5001/package
  - 注：在pddl文件中不要使用中文注释，服务器端可能无法正确识别





https://solver.planning.domains:5001/package

- Step 3 [编写]：编写domain文件（xxx.pddl）
  - domain文件示例：
  - 使用括号进行匹配!

week 9 PDDL > domains > spare_tire > (;a) spare_tire.pddl > {} Remove

```
(define (domain spare_tire)
  (:requirements :strips:equality:typing:disjunctive-preconditions:negative-preconditions)
  Show hierarchy
  (:types physob location)
  (:predicates  (Tire ?x - physob) 1
                (At ?x - physob ?y - location) 3 2 2)

  (:constants
    Ground Axle - location
    Flat - physob
  )

  (:action Remove
           :parameters (?x - physob ?y - location)
           :precondition (At ?x ?y)
           :effect (and (not (At ?x ?y)) (At ?x Ground))
  )

  (:action PutOn
           :parameters (?x - physob)
           :precondition (and (Tire ?x) (At ?x Ground) (not (At Flat Axle)))
           :effect (and (not (At ?x Ground)) (At ?x Axle))
  )
  ; (:action LeaveOvernight
  ;           :effect (and (not (At Spare Ground))(not (At Spare Axle))(not (At Spare Trunk))
(not (At Flat Ground))(not (At Flat Axle))(not (At Flat Trunk)))
  ; )
 )

```

定义 **domain** 论域名

:requirements 依赖项

- strips（strips求解）
- equality（允许使用谓词=）
- typing 使用类型
- disjunctive-preconditions 使前提条件可以带析取（用析取但没加该依赖项会报错）
- negative-preconditions（报warning，可不加）
- 动作前提默认可使用合取和否定

定义**类型**，**谓词**tire和at，**动作**。

- 用?x表示变量x，- type_name表示类型
- (action a :parameters() :precondition() :effect())

注意！前提条件之间可能有合取、析取、否定等关系，使用括号和 and or not连接。

- (and (…) (…))
- (and (or (…) (not (…))) (not (…)))

常量constants

- 需要的时候可以用:constants在domain文件中定义一些对象，如图中的Gound, Axle 和 Flat

- Step 3 [编写]：编写problem文件（xxx.pddl）
  - problem文件示例：

week 9 PDDL > domains > spare_tire > (:a) problem.pddl > ...

```
(define (problem prob)
 (:domain spare_tire)
 Show hierarchy
 (:objects  Spare - physob  Trunk  - location)

 View
 (:init
  (Tire Flat)
  (Tire Spare)
  (At Flat Axle)
  (At Spare Trunk)

 )
 (:goal
  (At Spare Axle)
 )
)
```

定义 **problem** 问题名
指定domain论域

- :objects 定义实体
  - 用 item - name 表示变量
  - 用 a b - name1 c d e - name2 表示 ab类型为 name1 而 cde为name2
- :init 初始化，列出一组谓词
- :goal 目标，给出目标结果（多个合取时用and连接）

注意！与前提、结果、goal不一样，**init<u>不需要</u>使用**(and()()())的格式连接。

*在problem文件里不需要再次声明domain文件中的常量（constant）

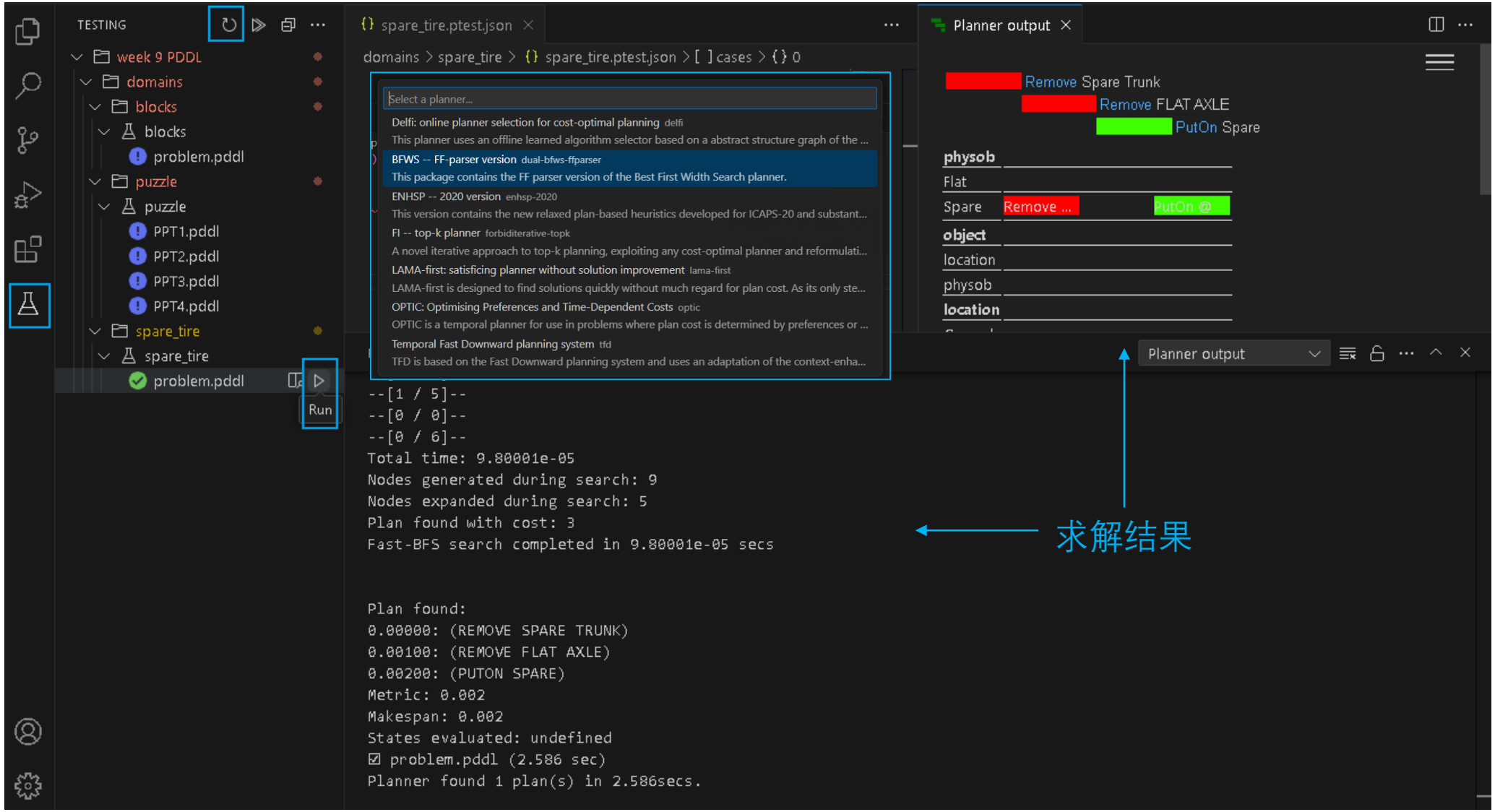
- Step 3 [编写]：进阶
  - effect中可能有条件效果，可以用(when (conditions) (effects)) 表示，意为当满足条件conditions的时候计算谓词2。
    - 需要依赖conditional-effects

> The semantics of (**when** P E) are as follows: If P is true before the action, then effect E occurs after. P is a *secondary precondition* [**?**]. The action is feasible even if P is false, but the effect E occurs only if P is true.

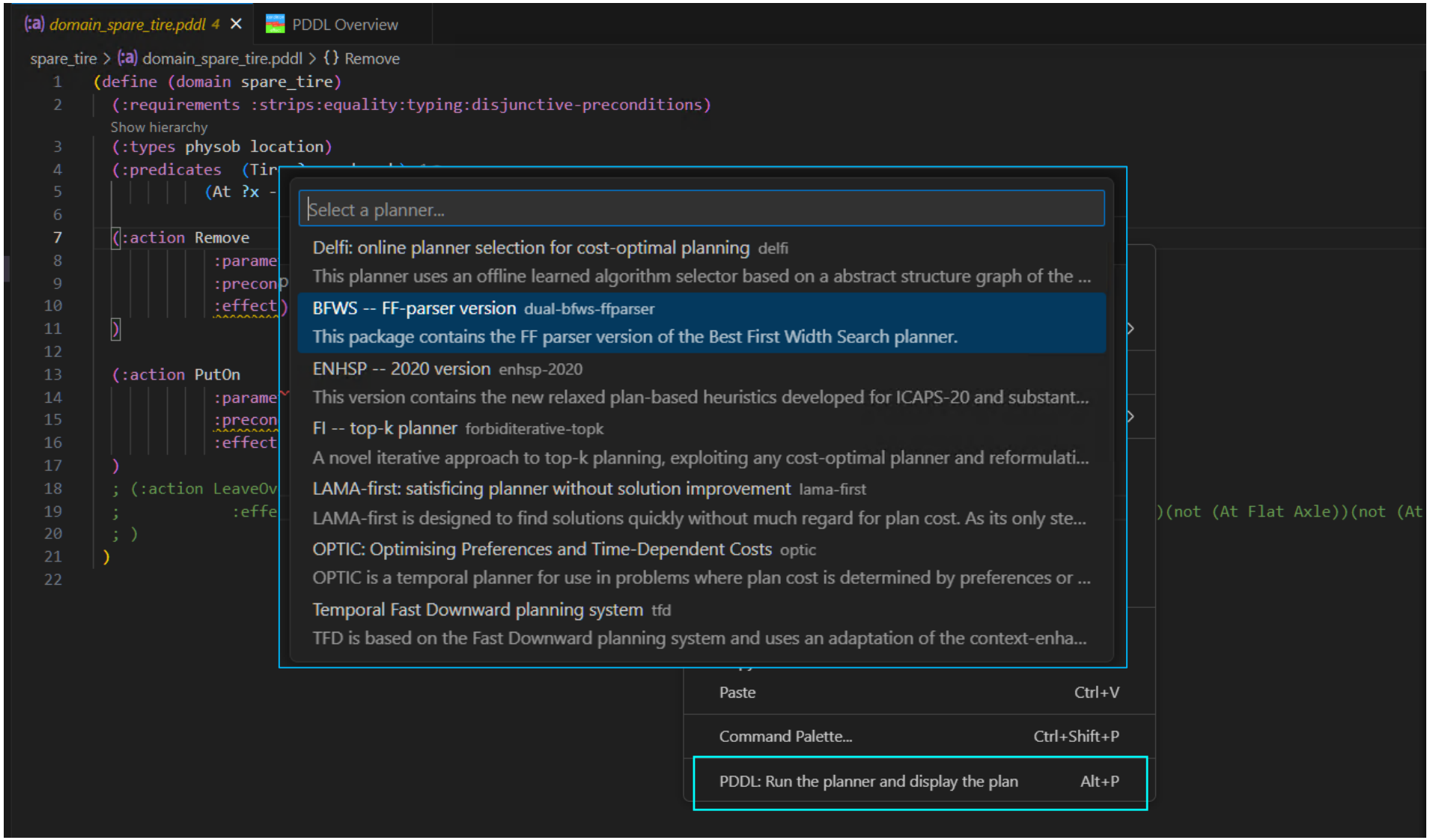
  - forall用于循环，如(forall (?z - block) (clear ?z)) 表示所有块都clear。
    - 可以嵌套使用，但要注意先对哪个变量进行循环
  - (forall (?z - block) (when (clear ?z) (…))) 对于所有block，如果clear则…

  - 可参考压缩包中的pddl.pdf p3/p10
  - 更多内容可参考本课件p3链接。

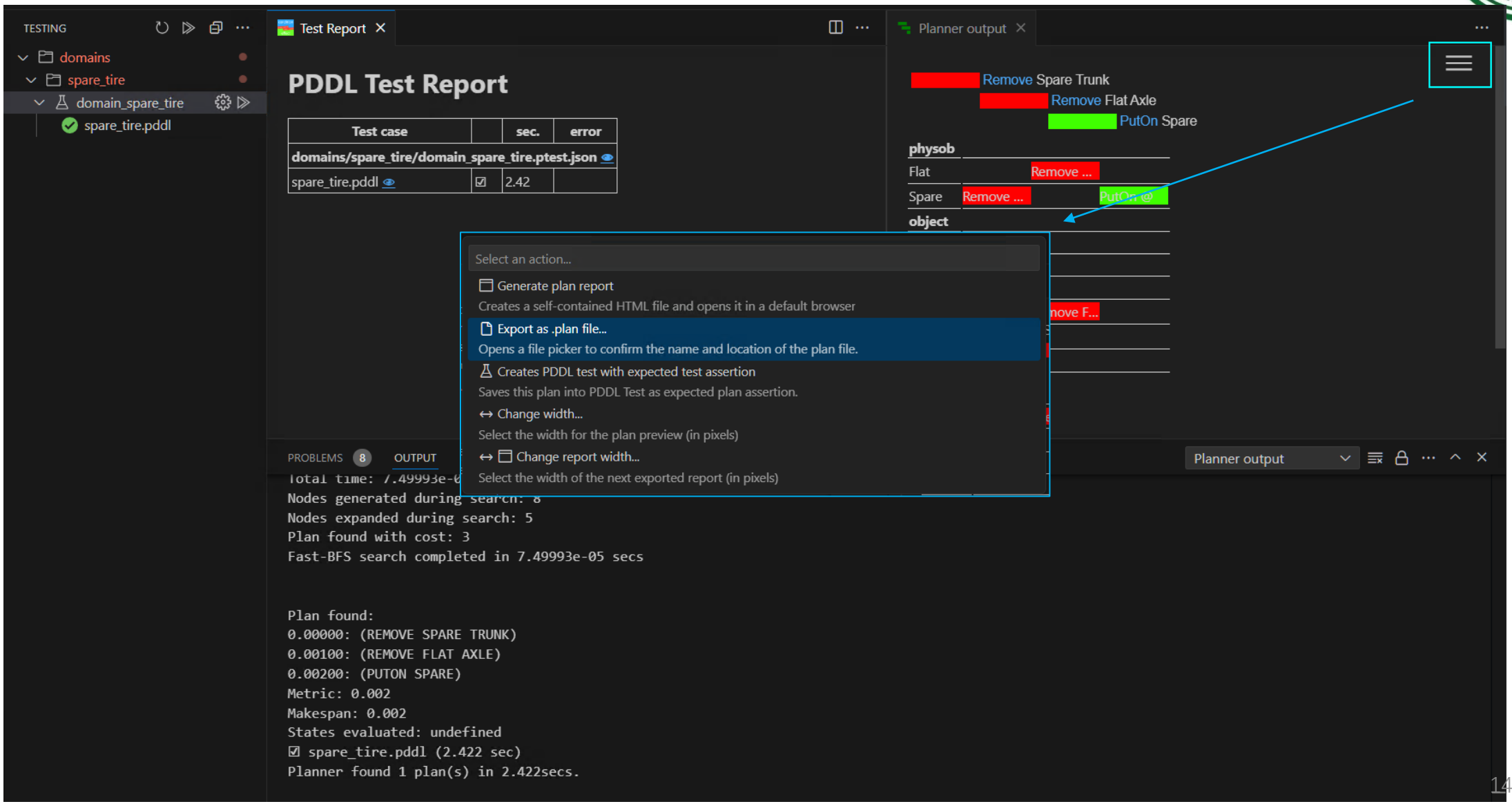
- Step 4 [求解]：方式1:通过Testing进行测试。
  - 点击侧边栏上的Testing->刷新->点击Run->选择planner
  - json配置文件已经都填好了，只需要补充完对应的文件后执行上述操作



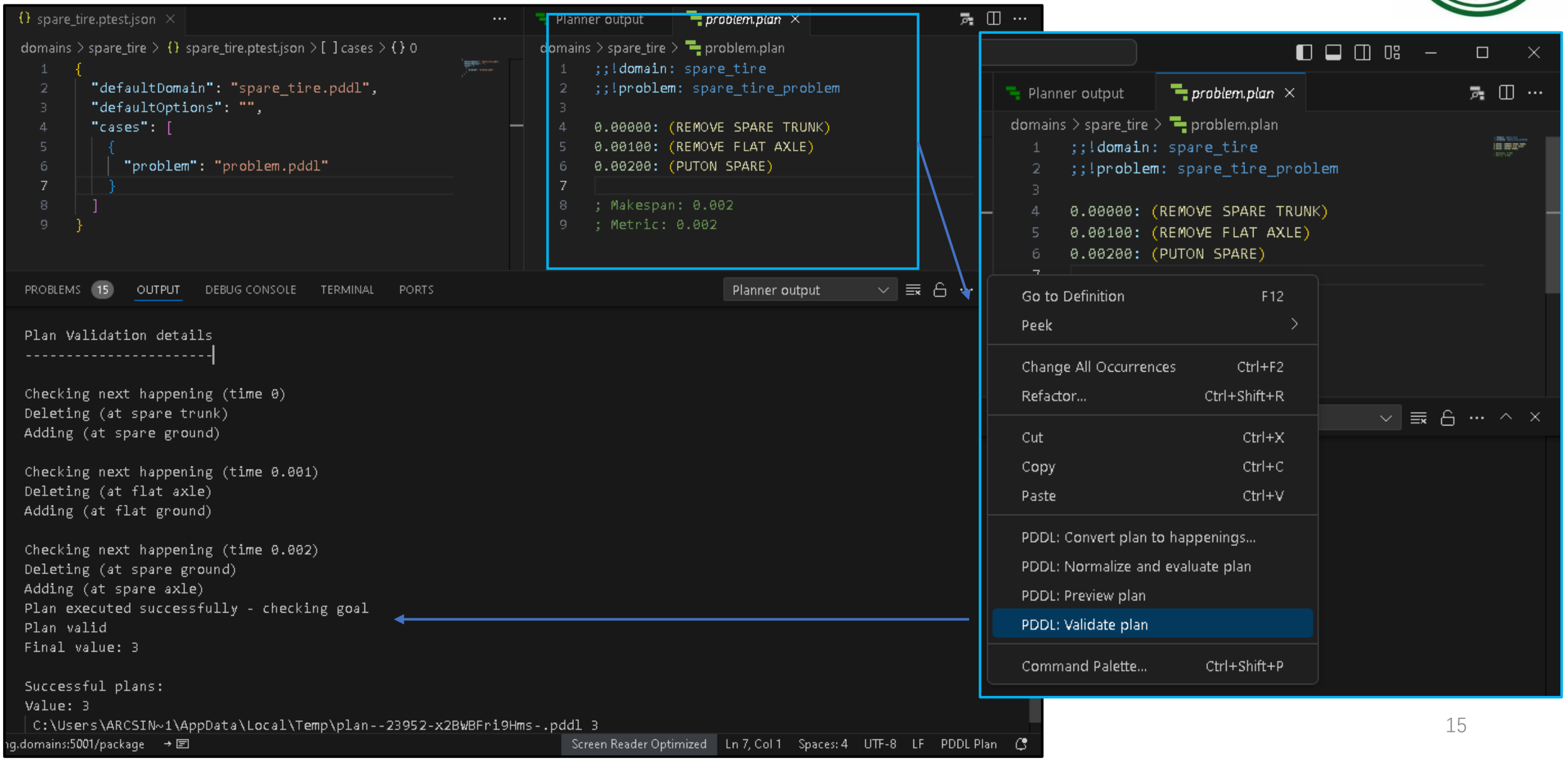
- Step 4 [求解]：方式2:右击运行
  - 右击domain文件或problem文件->选择PDDL: Run…->进一步选择planner
  - 若同个文件夹下还有多个pddl文件，会让你进一步进行选择



- 扩展 [验证]：Validate Plan
  - 右上方可以选择导出plan

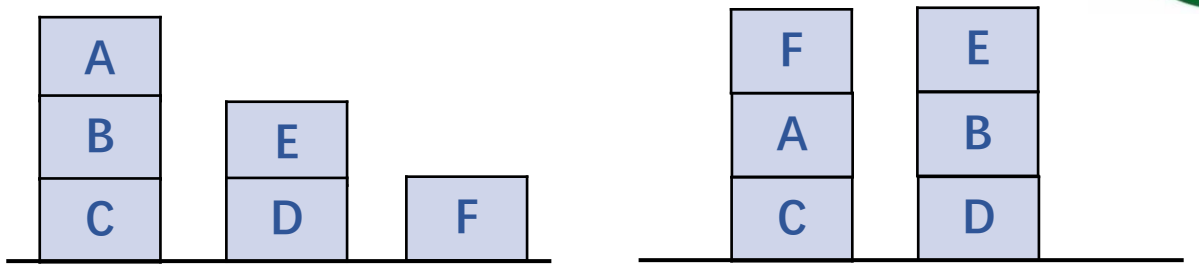


- 扩展 [验证]： Validate Plan
  - 右键生成的plan文件，选择PDDL: Validate plan即可



实验任务

1、Blocks world



blocks.pddl

```
(define (domain blocks)
  (:requirements :strips :typing:equality
                 :universal-preconditions
                 :conditional-effects)
  (:types physob)
  (:predicates
       (ontable ?x - physob)
           (clear ?x - physob)
     (on ?x ?y - physob))

  (:action move
            :parameters (?x ?y - physob)
            :precondition ()
            :effect ()
            )

  (:action moveToTable
            :parameters (?x - physob)
            :precondition ()
            :effect ( )
)
```

其他定义已给出，把动作**前提和效果补全**

problem.pddl

```
(define (problem prob)
 (:domain blocks)
 (:objects A B C D E F - physob)
 (:init (clear A)(on A B)(on B C)(ontable C) (ontable D)
  (ontable F)(on E D)(clear E)(clear F)
)
 (:goal  (and (clear F) (on F A) (on A C) (ontable C)(clear E) (on E B)
        (on B D) (ontable D)) )
 )
```

2、15puzzle

- 从A*和IDA*中PPT上的四个15数码问题中任选一个

补全问题文件
PPTx.pddl

```
(define (problem PPT1)
 (:domain puzzle)
 Show hierarchy
 (:objects )
 View
 (:init )
 (:goal ())
)
; 14 10 6 0
; 4 9 1 8
; 2 3 5 11
; 12 13 7 15
```

选中多行, Ctrl+“/” 取消注释后进行复制

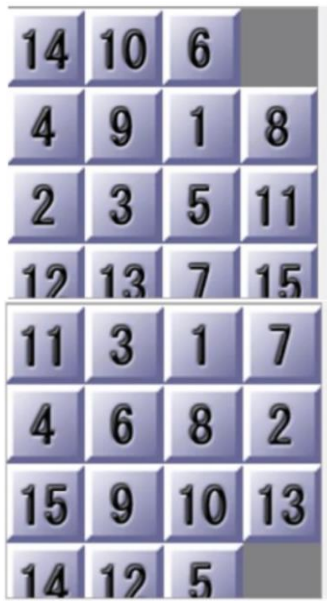
补全论域文件
puzzle.pddl

```
(define (domain puzzle)
  (:requirements :strips :equality:typing)
  Show hierarchy
  (:types num loc)
  (:predicates  ())
  (:constants  )


(:action slide
            :parameters ()
            :precondition ()
            :effect ()
 )
)
```

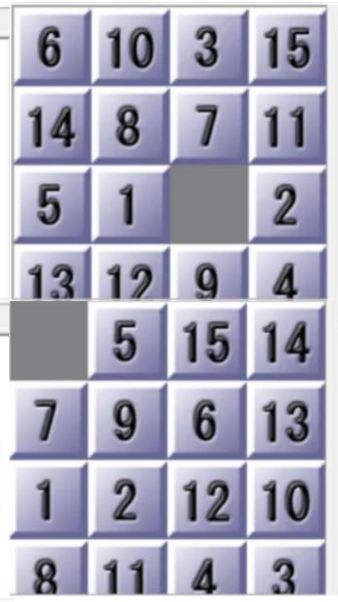
例1

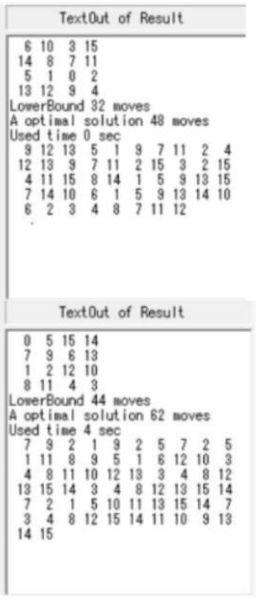
14 10 6 0
4 9 1 8
2 3 5 11
12 13 7 15



例2

6 10 3 15
14 8 7 11
5 1 0 2
13 12 9 4





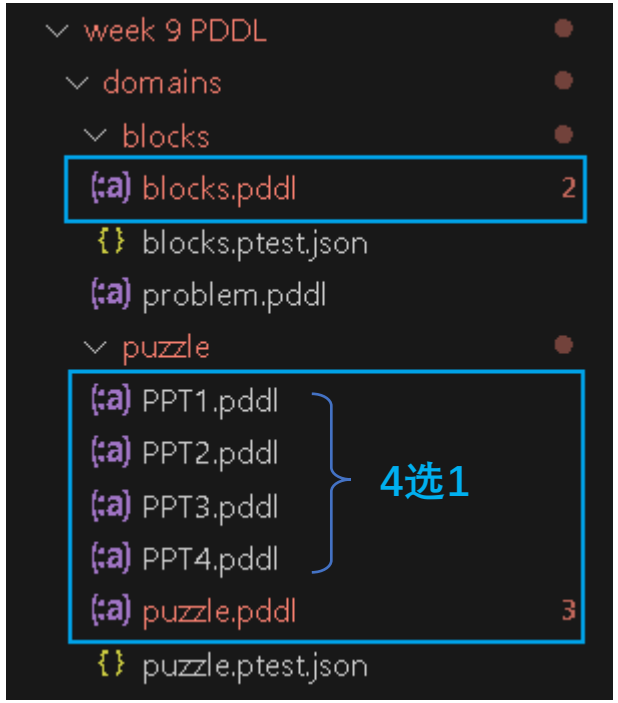
例3

11 3 1 7
4 6 8 2
15 9 10 13
14 12 5 0

例4

0 5 15 14
7 9 6 13
1 2 12 10
8 11 4 3

实验内容详解



- 建议复习理论课件Planning24（10页始）
- 积木世界：
  - 补全动作(blocks.pddl)
  - 你可能需要forall和when
- 15数码
  - Planning24，21页始
  - 不需要考虑解的最优性！！
  - 补充完整domain文件(puzzle.pddl)和一个PPTx.pddl文件，只允许有slide这一个动作，其他自行定义
  - 表示空格位置的两种方法：
    - 直接像PPT那样和block一起编码，但是否需要在action中引入常量？
    - 将“某位置是否为空格”单独作为谓词，但要怎么修改动作的前提结果和知识库？
  - 最好先判断是否在空格，再判断邻居（遍历的顺序）
  - 目标状态
    - 如何表示？是否一定要写全十六个位置的状态？15个可以吗？为什么？

（以上提问的回答不要求写到报告里，但想写可以写写）

实验要求

- 本次算法原理部分和伪代码/流程图部分可简写。主要是写清论域和问题的定义。可参考理论课件，和课件有出入的要描述清楚。
- 补全要求的pddl文件，放在code文件夹中一并提交。
- 在实验结果中展示使用规划器求解的结果（不要求对解进行Validate）

- 提交命名：E5_学号
- 截止时间：4月29号23:59（一周）

可能有用（15puzzle）↓

```
r=[]
for i in range(4):
    tmp=input().split(' ')
    r+=['(at block%s position%d)'%(x,j+i*4+1) for j,x in enumerate(tmp)]
```

- 附加题（无加分，有兴趣的同学可以试一下）

  - 有兴趣的同学可以考虑玩二阶魔方问题：
    - Online 2x2 Rubik's Cube Solver (rubiks-cube-solver.com)
    - 仅实现U（顶层往左转）,R（右列往上转）,F（正面顺时针转）即可求解，其他动作（URF连续做两次，或者反向拧）可选。
      - 根据这些动作，8个角里面有1个的位置是固定的，是哪个呢？
    - 定义：可以按色块位置（6*4）和颜色（6）来判断。也可以定义8个角？
    - 随机打乱魔方试试吧！

  - 其他的问题也可以尝试用pddl解决