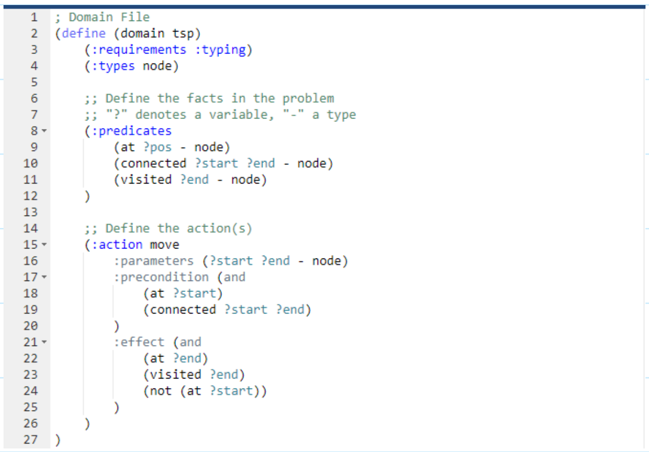
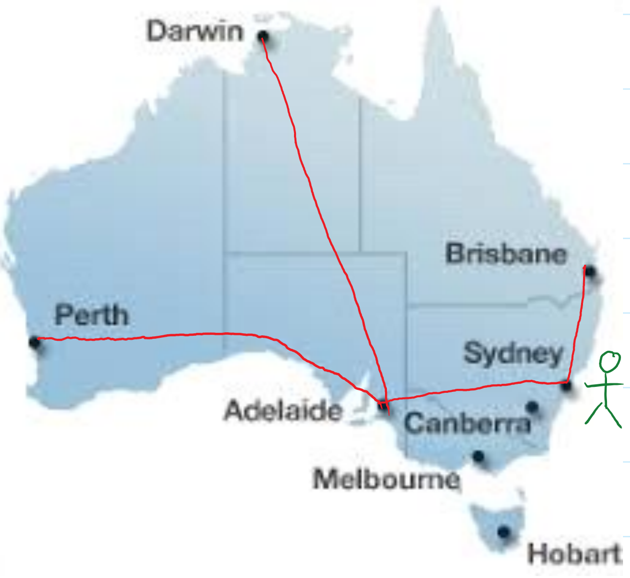
Week 5 T4

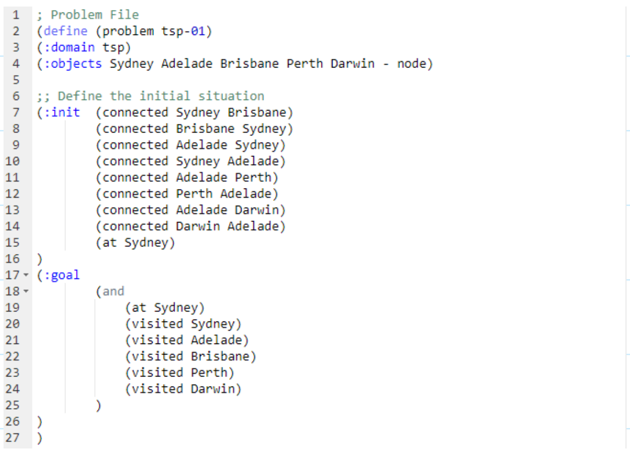
Friday, 19 August 2022 8:29 PM

PDDL

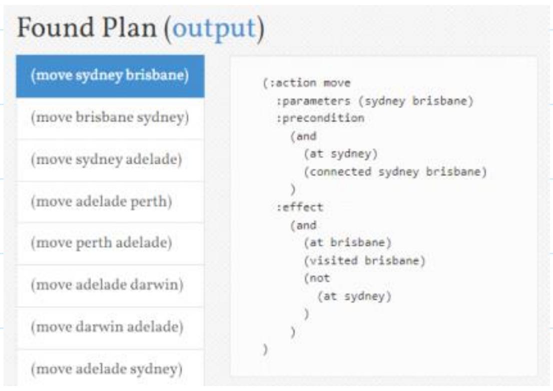


```
; Domain File
(define (domain tsp)
    (:requirements :typing)
    (:types node)

    ;; Define the facts in the problem
    ;; "?" denotes a variable, "-" a type
    (:predicates
        (at ?pos - node)
        (connected ?start ?end - node)
        (visited ?end - node)
    )

    ;; Define the action(s)
    (:action move
        :parameters (?start ?end - node)
        :precondition (and
            (at ?start)
            (connected ?start ?end)
        )
        :effect (and
            (at ?end)
            (visited ?end)
            (not (at ?start))
        )
    )
)
```





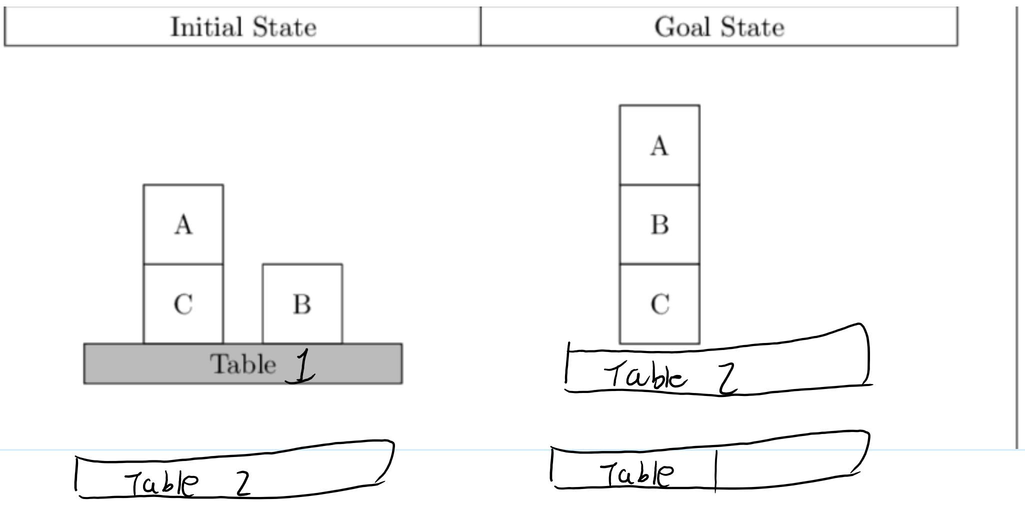
```
; Problem File
(define (problem tsp-01)
(:domain tsp)
(:objects Sydney Adelade Brisbane Perth Darwin - node)

;; Define the initial situation
(:init  (connected Sydney Brisbane)
        (connected Brisbane Sydney)
        (connected Adelade Sydney)
        (connected Sydney Adelade)
        (connected Adelade Perth)
        (connected Perth Adelade)
        (connected Adelade Darwin)
        (connected Darwin Adelade)
        (at Sydney)
)
(:goal
        (and
            (at Sydney)
            (visited Sydney)
            (visited Adelade)
            (visited Brisbane)
            (visited Perth)
            (visited Darwin)
        )
)
)
```



Found Plan (output)

```
(move sydney brisbane)
(move brisbane sydney)
(move sydney adelade)
(move adelade perth)
(move perth adelade)
(move adelade darwin)
(move darwin adelade)
(move adelade sydney)
```

```
(:action move
  :parameters (sydney brisbane)
  :precondition
    (and
      (at sydney)
      (connected sydney brisbane)
    )
  :effect
    (and
      (at brisbane)
      (visited brisbane)
      (not
        (at sydney)
      )
    )
)
```



```
;; Domain file: specifies Facts and Actions
;define (domain [name of the domain])
(define (domain blocksMultiTable)
;similar to importing modules in python
;(:requirements [list of requirements to import])
(:requirements :strips :typing)
; from typing requirement: specifies all types for the problem
(:types block table)

;; fact list
; variables denoted with leading ?
; type denoted with [- type]
(:predicates
    (on ?x ?y - block)
    (clear ?x - block)
    (onTable ?x -block ?t - table)
)

;define actions here
;actions similar to functions in python
(:action moveFromBlockToTable ;remove block x from on top of block y and place on
table t
    ;specify parameters for action as well as variable types
    ;no need to specify variable types elsewhere within action function
    :parameters (?x - block ?y - block ?t - table)
    :precondition (and ;leading 'and' for multiple preconditions
                (on ?x ?y)
                (clear ?x)
    )
    :effect (and ;inclues both add list and delete list
                (onTable ?x ?t)
                (clear ?y)
                (not (on ?x ?y)) ; elements of delete list enclosed in (not)
bracket i.e. (not (predicate))
    )
)
(:action moveFromTableToBlock
    :parameters (?x - block ?y - block ?t - table)
    :precondition (and
                (onTable ?x ?t)
                (clear ?x)
                (clear ?y)
    )
    :effect (and
                (on ?x ?y)
                (not (onTable ?x ?t))
                (not (clear ?y))
    )
)

)
```

```
;;Problem file: specifies Initial and Goal situations
(define (problem problem_name)
(:domain blocksMultiTable) ; links problem file to domain file with same name
(:objects ;; list of all the objects for our problem
    A B C - block Table1 Table2 - table
)
(:init ;; spaces/indents/newlines not part of syntax of pddl
       ;; can organise however you wish
    (clear A) (clear B)
    (onTable B Table1) (onTable C Table1)
    (on A C)
)
(:goal (and
    (onTable C Table2)
    (on A B)
    (on B C)
```

))
)