# Part1 Domain

(define (domain cookies-world)

  (:requirements :strips :typing :disjunctive-preconditions :negative-preconditions)

  (:types location key cookies - object)

  (:predicates

    (robot-at *?loc* - location)

    (key-at *?loc* - location)

    (cookies-at *?loc* - location)

    (lock-at *?loc* - location)

    (gate-open)

    (has-key)

    (has-cookies)

    (monster-at *?loc* - location)

    (connected *?from* *?to* - location)

    (gate-between *?loc1* *?loc2* - location)

  )

  ;The robot can move between connected locations.

  (:action move

    :parameters (*?from* *?to* - location)

    :precondition (and (robot-at *?from*)

                       (connected *?from* *?to*)

                       (not (monster-at *?to*))) ;The robot cannot move to B5, where the monster is located.

    :effect (and (not (robot-at *?from*))

                 (robot-at *?to*))

  )

;The robot can move from A6 to B4, or vice versa, if the gate is opened.

  (:action move-through-gate

    :parameters (*?from* *?to* - location)

    :precondition (and

                  (gate-open)

                  (robot-at *?from*)

                  (gate-between *?from* *?to*))

    :effect (and (not (robot-at *?from*)) (robot-at *?to*))

  )

  ;The robot can grab the key, or put it down in a location.

  (:action grab-key

    :parameters (*?loc* - location)

    :precondition (and (robot-at *?loc*) (key-at *?loc*) (not (has-key)))

    :effect (and (not (key-at *?loc*)) (has-key))

  )

  (:action drop-key

    :parameters (*?loc* - location)

    :precondition (and (robot-at *?loc*) (has-key))

    :effect (and (not (has-key)) (key-at *?loc*))

  )

  ;The robot can open or close the gate, using the key.

  (:action open-gate

    :parameters (*?loc* - location)

    :precondition (and (not (gate-open)) (has-key)(robot-at *?loc*)(lock-at *?loc*))

    :effect (gate-open)

  )

  (:action close-gate

    :parameters (*?loc* - location)

    :precondition (and (gate-open) (has-key) (robot-at *?loc*)(lock-at *?loc*))

    :effect (not (gate-open))

  )

  ;The robot can grab the cookies, or put it down in a location.

  (:action grab-cookies

    :parameters (*?loc* - location)

    :precondition (and (robot-at *?loc*) (cookies-at *?loc*)(not(has-cookies)))

    :effect (and (has-cookies) (not (cookies-at *?loc*)))

  )

  (:action drop-cookies

    :parameters (*?loc* - location)

    :precondition (and (robot-at *?loc*) (has-cookies))

    :effect (and (not (has-cookies)) (cookies-at *?loc*))

  )

)

# Part2 Problem

(define (problem cookies-world)

  (:domain cookies-world)

  (:objects

    A1 A2 A3 A4 A5 A6 A7 A8 A9 B1 B2 B3 B4 B5 B6 B7 B8 B9 - location

  )

  (:init

    (robot-at A1) ;The robot is in A1

    (lock-at A6) ;The gate is closed

    (key-at A8) ;The key is in A8,

    (cookies-at B6)  ;The cookies are in B6

    (monster-at B5) ;The cookie monster is in B5

    ; Define the connections between locations

    (connected A1 A2) (connected A2 A3) (connected A3 A2) (connected A2 A1)

    (connected A4 A5) (connected A5 A6) (connected A6 A5) (connected A5 A4)

    (connected A7 A8) (connected A8 A9) (connected A9 A8) (connected A8 A7)

    (connected A1 A4) (connected A4 A7) (connected A7 A4) (connected A4 A1)

    (connected A2 A5) (connected A5 A8) (connected A8 A5) (connected A5 A2)

    (connected A3 A6) (connected A6 A9) (connected A9 A6) (connected A6 A3)

    (connected B1 B2) (connected B2 B3) (connected B3 B2) (connected B2 B1)

    (connected B4 B5) (connected B5 B6) (connected B6 B5) (connected B5 B4)

    (connected B7 B8) (connected B8 B9) (connected B9 B8) (connected B8 B7)

    (connected B1 B4) (connected B4 B7) (connected B7 B4) (connected B4 B1)

    (connected B2 B5) (connected B5 B8) (connected B8 B5) (connected B5 B2)

    (connected B3 B6) (connected B6 B9) (connected B9 B6) (connected B6 B3)

    ; Gate connections

    (gate-between A6 B4) (gate-between B4 A6)

  )

  (:goal

    (and

      (robot-at A6) ;any of the A locations

      (has-cookies) ;he has the cookies

      (not (gate-open)) ; gate must be closed

    )

  )

  ; (:goal

  ;   (and

  ;     (or (robot-at A1)

  ;         (robot-at A2)

  ;         (robot-at A3)

  ;         (robot-at A4)

  ;         (robot-at A5)

  ;         (robot-at A6)

  ;         (robot-at A7)

  ;         (robot-at A8)

  ;         (robot-at A9))

  ;     (has-cookies)

  ;     (not (gate-open))

  ;   )

  ; )

)

# Part3 Plan

