Destinee Hill
Jaren Robbins
Kelsey Lohr
Can Ngo

<u>Question 1:</u>

State Definition:

- current_state = [total_left,
- cannibal_left,
- missonaries_left,
- total_right,
- cannibal_right,
- missionaries_right,
- people_on_boat,
- boat_side]

A vector where:

- total_left - total people on left side
- cannibal_left - total number of cannibals on the left
- missionaries_left - total number of cannibals on left
- total_right - total number of people on the right
- cannibal_right - total cannibals on right side
- missionaries_right - total missionaries on right side
- people_on_boat - the number of people on the boat currently
- boat_side - which side the boat is on

Initial State:
current_state = [0, 0, 0, 6, 3, 3, 0, right]

Actions:

- add_on_boat(isCannibal): adds a person onto the boat. 1 if the person is a cannibal. 0 otherwise.
- drop_off_left(isCannibal): drops off on the left.
- add_on_right(isCannibal): adds a person onto the boat from right.
- go_right(): takes the boat to the right side of the shore

Transition Model:

add_on_left(true): current_state = [-1, -1, 0, 0, 0, 0, +1, left]
add_on_left(false): current_state = [-1, 0, -1, 0, 0, 0, +1, left]
drop_off_left(true): current_state = [+1, +1, 0, 0, 0, 0, -1, left]
drop_off_left(false): current_state = [+1, 0, +1, 0, 0, 0, -1, left]
add_on_right(true): current_state = [0, 0, 0, -1, -1, 0, +1, right]

add_on_right(false): current_state = [0, 0, 0, -1, 0, -1, +1, right]
drop_off_right(true): current_state = [0, 0, 0, +1, +1, 0, -1, right]
drop_off_right(false): current_state = [0, 0, 0, +1, 0, +1, -1, right]
Create if statement is m > c for each side. Call each time someone is dropped off/added.

Path Cost:
number _of_trips

Goal Test/Condition:
current_state = [6,3,3,0,0,0]
Everyone on the opposite side of the river.

<u>**Question 2:**</u>

State Definition:
Current_position = (x,y)

Initial State:
Agent is at position (0,0) on the grid
Current_position = (0,0)

Actions:
Move(direction): moves to that coordinate if it is not blocked (returns false if blocked, returns true if agent can move)

direction = up, down, right, left, bottom_left, top_left, top_right, bottom_right

check_position(): returns the coordinates of the agent

Transition Model:

Move(up): returns true: current_state = (0,1)
check_position(): returns (0,1)
Move(up): returns true: current_state = (0,2)
check_position(): returns (0,2)
move(left): return false: current_state = (0,2)
Move(right): return true: current_state = (1,2)
Move(top_left): return true: current_state = (0,3)
Move(top_right): return true: current_state = (1,4)
Move(down): return false: current_state(1,3)
Move(up): return true: current_state(1,4)
Move(up): return true: current_state(1,5)
move(right): return true: current_state(2,5)

move(right): return true: current_state (3,5)
check_position(): returns (3,5)
move(right): return true: current_state(4,5)
move(right): return true: current_state (5,5)
check_position(): returns (5,5)

Path Cost:
Number of calls it takes to get to position (5,5)

Goal Test/Condition:
check_position() == (5,5)