Karan Sharma
107424405
HW: 0
Dr. Ioana Fleming
Section: 101

-

Diamond Room:

I will first separate the diamond room into a leftward and a rightward set of operations. The leftwards will be separated into left and right halves and likewise for the rightward set of operations. The picobot will progress up and down the columns, assessing the walls and always moving east or west (depending on if there is obstruction) upon filling the column at the top. I will create an initial state with accounts for leftmost, northmost, rightmost, N+E obstructed, and N+W obstructed positions as the picobot will start off by going north. The bot will finish either at the rightmost or at the leftmost point.

-

```
# check the walls on the right half / check starting
# position
0 x*** -> N 0 # start off, if able to go north go north
0 NExS -> W 1 # if north east and south blocked initially, go west
          # and change to state 1
0 NxWx -> S 4 # if north and west blocked initially, go south and
          #change to state 4
0 NExx -> W 1 # if north and east blocked initially, go west and
          #change to state 1
0 NEWx -> S 1 # if north east and west blocked initially, go south
          # and change to state 1
0 NxWS -> E 7 # if north west and south blocked initially, go east
          # and change to state 7

# this is the up down part
1 x*** -> N 1 # travel upward on right half going left
1 N*** -> S 4 # travel downward on right half going left and
          # change to state 4

# check the walls on the left half
4 ***x -> S 4 # keep going south to fill the column
4 xExS -> N 5 # if stuff to east and south, change to state 5 and
          # go north
4 xxWS -> N 5 # if stuff to west and south, change to state 5 and
          # go north
4 xEWS -> N 5 # if stuff to east, west, and south, change to state
          # 5 and go north
```

```
5 **x* -> W 6 # in current state go west and change to state 6

#check if at left endpoint
6 NxWS -> E 7 # if at leftmost point, go east and change to state
             # 7
6 x*** -> N 1 # because nothing to the north, go north and change
             # to state 1

#check walls on left half on way back
7 ***x -> S 7 # if able to go south go south
7 xExS -> N 8 # if east and south blocked, go north and change to
             # state 8
7 xxWS -> N 8 # if west and south blocked, go north and change to
             #state 8
7 xEWS -> N 10 # if east west and south blocked, go north and
              #change to state 10

8 x*** -> N 8 # keep going north to fill column
8 N*** -> E 9 # after reaching top of filled column, go east and
             # change state to 9

9 xxxx -> N 7 # if able to go all directions, go north one and
             # return to state 7

10 xxxx -> E 11 # when one above lowest point where nothing
               # resides, go east and change state to 11

#check walls on the right half on the way back
11 x*** -> N 11 # keep going north to fill column
11 NE** -> S 12 # if north and east blocked, go south and change
               # state to 12

12 xxxx -> E 13 # if able to go all directions, go east and change
               # to state 13

13 ***x -> S 13 # keep going south to fill column
13 xExS -> N 11 # if east and south blocked, go north and return
               # to state 11
```