IDE | Midpoint Karel

Reset

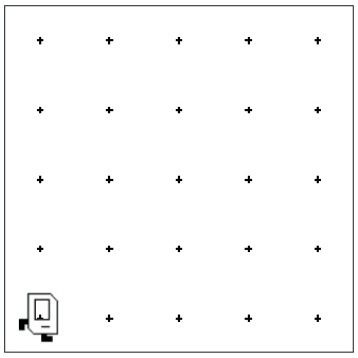
Share

Instructions

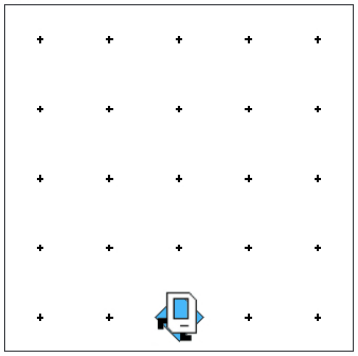
Restart

As an exercise in solving algorithmic problems, you will program to find the midpoint of 1st Street. Say Karel starts in the 5x5 world. Karel should end in the center of 1st row.

Before:



After:



Note that the final configuration of the world should have only a single beeper at the midpoint of the 1st row. Along the way, Karel is allowed to place additional beepers wherever it wants to, but must pick them all up again before it finishes. Similarly, if Karel paints/colors any of the corners in the world, they must all be uncolored before Karel finishes.

In solving this problem, you may count on the following facts about the world:

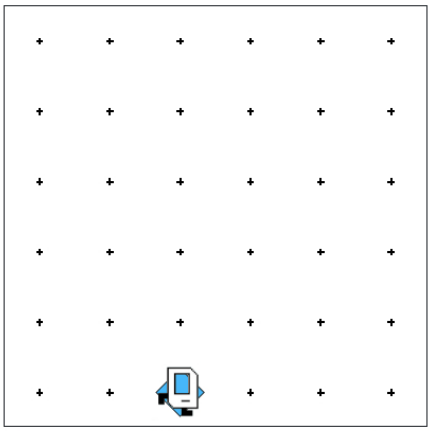
- Karel starts at bottom left corner of the world, facing east.
- The initial state of the world includes no interior walls or beepers.
- The world need not be square, but you may assume that it is at least as tall as it is wide.
- If the width of the world is odd, Karel must put the beeper in the center square. If the width is even, Karel must drop a beeper on the left most of the two squares.

There are many different algorithms you can use to solve this problem so feel free to be creative!
You should design your program to run successfully

main.py

```
1  from karel.stanfordkarel import *
2
3  def safe_move():
4      if(front_is_clear()):
5          move()
6
7  def turn_around():
8      for i in range (2):
9          turn_left()
10
11 def go_other_way():
12     while(front_is_clear()):
13         move()
14     turn_around()
15
16 def step_back():
17     turn_around()
18     safe_move()
19
20 def check():
21     safe_move()
22     while(no_beepers_present() and front_is_clear()):
23         move()
24     if(beepers_present()):
25         pick_beeper()
26         step_back()
27         put_beeper()
28
29 def main():
30     go_other_way()
31     put_beeper()
32     go_other_way()
```

World



World: 6x6

Terminal