Previously in CSCI 1170

Karel the Robot

A Gentle Introduction to the Art of Programming in C++

Previously in CSCI 1170

~~*Karel*~~ the Robot

A Gentle Introduction to the Art of Programming in ~~C++~~

Reeborg the Robot

A Gentle Introduction to the Art of Programming in Python

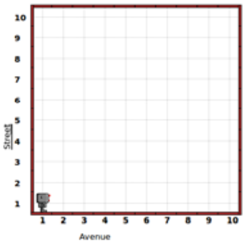
using RUR

Schools Using This Approach

- •Stanford
- •U. of Washington
- •Air Force Academy
- •Westpoint (USMA)
- •Purdue

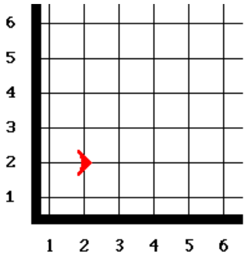
What is *Reeborg*?

- *Reeborg is* essentially a programmable image that can move across the flat world of a monitor screen.
- Shown on the screen is a grid work of vertical and horizontal lines, representing avenues and streets.



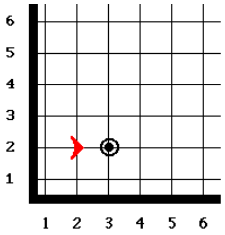
Reeborg's Capabilities

- *Reeborg* is restricted to moving from street corner to street corner, one such move at a time.
- *Reeborg* can pivot 90 degrees to the left when requested.
- *Reeborg* can only face north, south, east, or west.

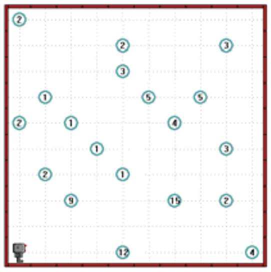


Beepers

- To add variety to *Reeborg's* environment, *Reeborg's* designer added beepers, tiny objects that *Reeborg* can detect (we say he "hears" them), pick up, carry, and put down.
- It is possible to program *Reeborg* to locate beepers, transport them, or place them in some graphic pattern.

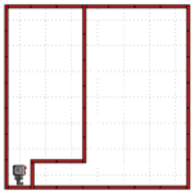


For example, it might look like the figure below on a computer screen.



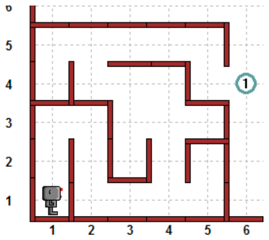
Walls

- Also in *Reeborgs's* environment are impenetrable wall sections that can be placed between streets.
- *Reeborg* can "see" wall sections that are immediately to his front or to his left and right sides.



Wall Sections

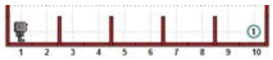
- Can be used to form a maze that *Reeborg* must navigate.



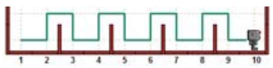
Wall Sections

…or to represent hurdles that Reeborg must "jump" in a hurdle race

Initial Situation



Final Situation



More about *Beepers*

- Beepers are so small that *Reeborg* can move right by them; only wall sections and boundary walls can block his movement.

Reeborg's Capabilities

- *Reeborg* is a mobile robot: he can move forward, in the direction he is facing, and he can turn in place.
- *Reeborg* possesses rudimentary senses:
  - *Sight*
  - *Sound*
  - *Direction*
  - *Touch*

- *Reeborg* is equipped with a mechanical arm that he can use to pick up and put down beepers.
- To carry these beepers, *Reeborg* has a soundproof pocket. (Actually putting them in and out of the pocket turns them off and on automatically.)
- *Reeborg* can determine if he is carrying any beepers in his pocket by probing it with his arm.

Tasks

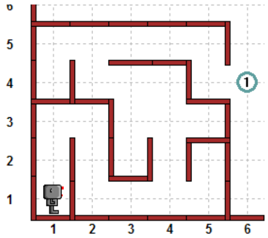
- A "task" is just something that we want *Reeborg* to do.
- Examples:
  - move to the corner of 3rd street & 5th avenue.
  - Run a hurdle race (with wall sections representing hurdles).
  - Run a maze.

A Detailed Set of Instructions

- Whenever we want Reeborg to accomplish a task in the world, we must apply a detailed set of instructions that explains how to perform the task. Reeborg is able to read and follow such a set of instructions, which is called a program.

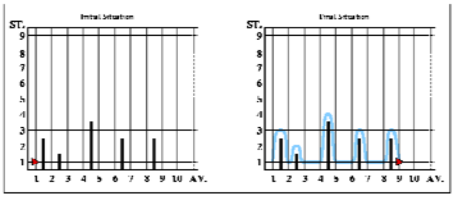
Situations

- A "situation", or "world", is an exact description of what *Reeborg's* world looks like.
- The basic structure of streets, avenues, and outer boundary walls is fixed.
- *What else do we need to specify?*



- *Reeborgs's position: location & direction*
- *Location & size of wall sections*
- *Location of beepers*
- *Initial number of beepers in his pocket*

Initial and Final Situations



- The initial situation for any task is defined to be the situation that Reeborg is placed in at the start of the task.
- The final situation is the situation that Reeborg is in when he turns himself off.

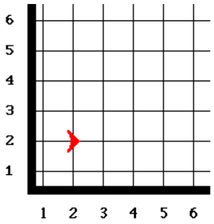
Reeborg initially understands only five imperative commands:

move()
turn_left()
pick_beeper()
put_beeper()
turn_off()

When these commands, or function calls, are executed in a Python program, the results are depicted on the screen.

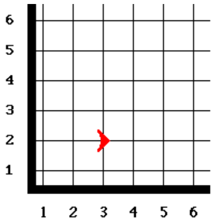
move()

- **A move() command will graphically show *Reeborg* moving from one street corner to the next.**

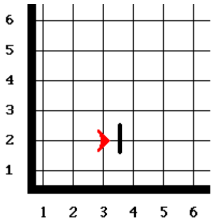


move()

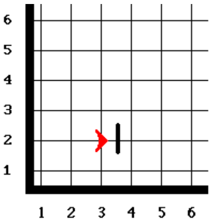
- **A move() command will graphically show *Reeborg* moving from one street corner to the next.**



- **Should a wall section be in the way, a move() command would cause an "Error Shutoff"**

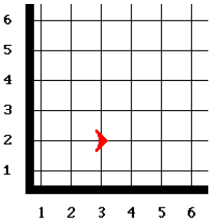


- Should a wall section be in the way, a move() command would cause an "Error Shutoff"



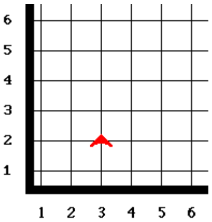
turn_left()

- A turn_left() instruction causes *Reeborg* to pivot 90 degrees to the left. *Reeborg* remains on the same street corner. For example, if *Reeborg* is facing east, a turn_left() causes him to face north.



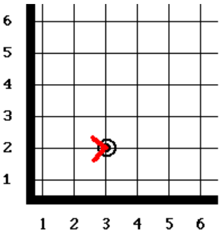
turn_left()

- A turn_left() instruction causes *Reeborg* to pivot 90 degrees to the left. *Reeborg* remains on the same street corner. For example, if *Reeborg* is facing east, a turn_left() causes him to face north.



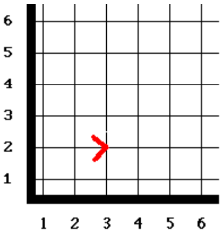
pick_beeper()

- When *Reeborg* executes a pick_beeper() instruction, he picks up a beeper from the corner on which he is standing and then deposits it in his pocket.
- On a corner with multiple beepers, *Reeborg* randomly picks up one – and only one – of the beepers.



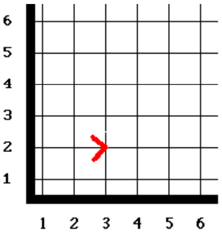
pick_beeper()

- When *Reeborg* executes a pick_beeper() instruction, he picks up a beeper from the corner on which he is standing and then deposits it in his pocket.
- On a corner with multiple beepers, *Reeborg* randomly picks up one – and only one – of the beepers.



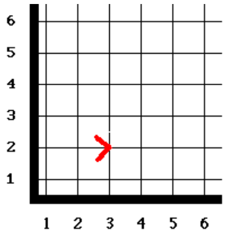
pick_beeper()

- If a pick_beeper() command is issued and there are no beepers on the corner, then he performs an "Error Shutoff".



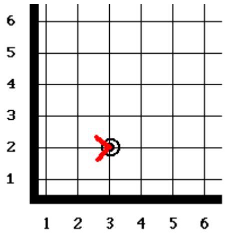
put_beeper()

- *Reeborg* executes a put_beeper() instruction by extracting a beeper from his pocket and placing it on his current street corner.



put_beeper()

- *Reeborg* executes a put_beeper() instruction by extracting a beeper from his pocket and placing it on his current street corner.



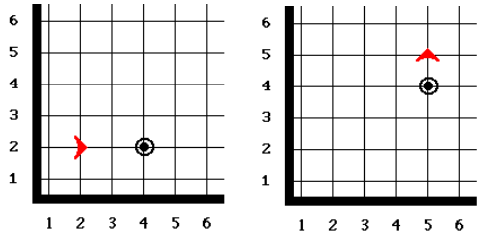
More about put_beeper()

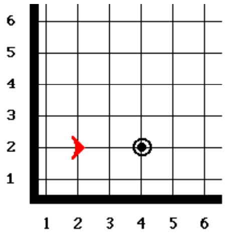
- If *Reeborg* is not carrying any beepers in his pocket and a put_beeper() command is issued, he performs an “Error Shutoff”.

turn_off()

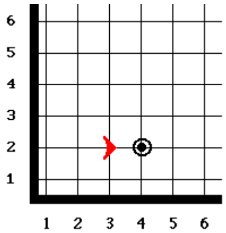
- When *Reeborg* executes a turn_off() instruction, he turns himself off and is incapable of executing any more instructions.
- The last instruction to be executed in every robot program <u>must</u> be a turn_off() instruction.

A complete robot program: Reeborg’s task is to transport the beeper from 2nd street, 4th avenue to 4th street and 5th avenue and then move one street further north before stopping.

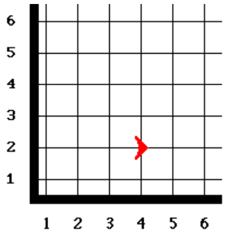




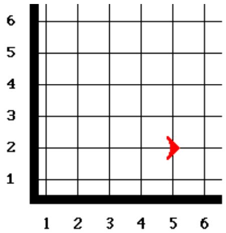
```
move()
```



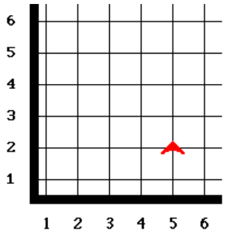
```
move()
move()
pick_beeper()
```



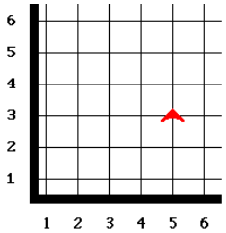
```
move()
move()
pick_beeper()
move()
```



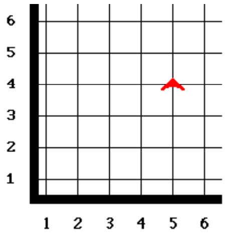
```
move()
move()
pick_beeper()
move()
turn_left()
```



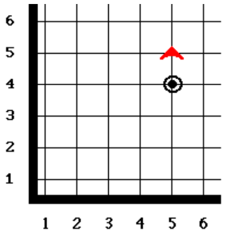
```
move()
move()
pick_beeper()
move()
turn_left()
move()
```



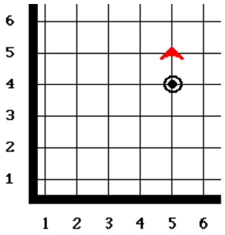
```
move()
move()
pick_beeper()
move()
turn_left()
move()
move()
```



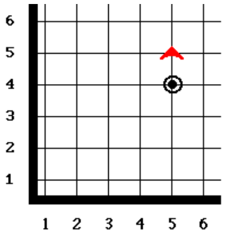
```
move()
move()
pick_beeper()
move()
turn_left()
move()
move()
put_beeper()
move()
```



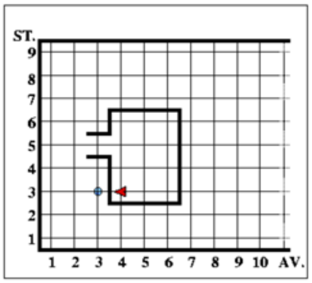
```
move()
move()
pick_beeper()
move()
turn_left()
move()
move()
put_beeper()
move()
```



```
move()
move()
pick_beeper()
move()
turn_left()
move()
move()
put_beeper()
move()
turn_off()
```



TASK: Every morning Reeborg is awakened in bed when his newspaper, represented by a beeper, is thrown onto the front porch of his house. Program Reeborg to retrieve his paper and bring it back to bed with him. The figure below illustrates the initial situation. (Newspaper is at street 3, avenue 3.) The final situation must have Reeborg back in bed (same corner, same direction) with the newspaper (that is, in his pocket).



Initial situation for the Newspaper Retrieval Task