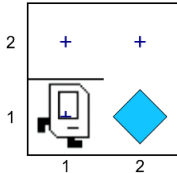


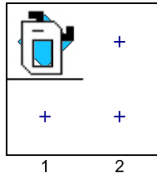
Karel the Robot Labs

Karel the Robot, an idea pioneered at Stanford, is a way to teach programming in a visual, entertaining way. Through the four basic commands of `move()`, `turn_left()`, `put_beeper()`, and `pick_beeper()`, you will navigate Karel through a series of mazes and learn about coding along the way! There is a Python version of Karel, and that's what we'll use.

Karel's commands

When Karel is shipped from the factory, it responds to a very small set of commands. Lets try out the commands. Use the buttons below to get the "world" to match the "goal":

World:

Goal:

Show Text Descriptions

move()

turn_left()

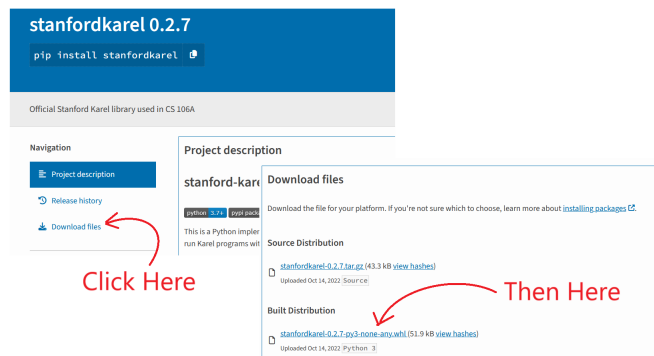
pick_beeper()

put_beeper()

You can try her yourself here: [Link](#).

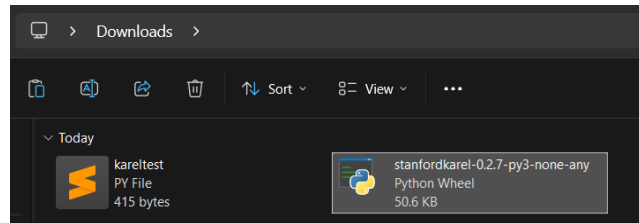
Installation

Navigate over to [this link](#) and click on the left sidebar “Download files.” From there, you want to download the “Built Distribution.”



Karel incoming.

Once she's downloaded, double click the file in FileExplorer to start the installation progress. A command prompt window will open up that will install Karel. **NOTE: This may take 30 seconds or longer. Do not close the installation window.**



My file explorer is in dark mode, so yours may look different.

Then, in a new command prompt window, type

```
python -m pip install stanfordkarel
```

Voilà! Now Karel will be ready to use.

```
C:\Users\Mia>pip install stanfordkarel
Collecting stanfordkarel
  Downloading stanfordkarel-0.2.7-py3-none-any.whl.metadata (5.9 kB)
  Downloading stanfordkarel-0.2.7-py3-none-any.whl (51 kB)
    51.9/51.9 kB 222.7 kB/s eta 0:00:00
Installing collected packages: stanfordkarel
Successfully installed stanfordkarel-0.2.7
```

Woohoo!

If you want to read a tutorial on Karel, you can find it [here](#). And [this link](#) has some more information if you scroll down, though it's a bit more techy. But I hope to explain what you need to do in this lab document.

Lab 1: Movement

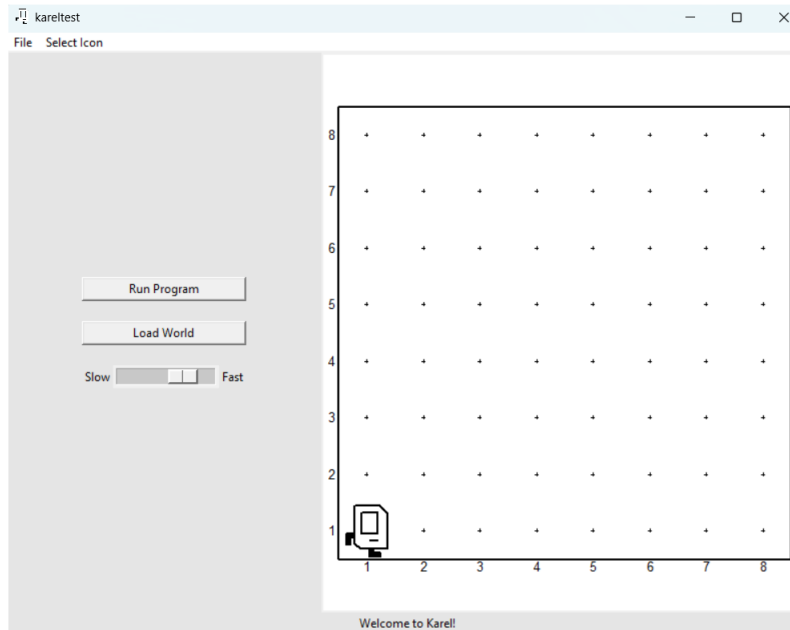
Prerequisites: Functions

In a new python file, copy the following code:

```
1     from stanfordkarel import *
2
3
4     def main():
5         """Karel code goes here!"""
6         turn_left()
7         move()
8         turn_left()
9
10
11     if __name__ == "__main__":
12         run_karel_program()
```

There's some code here that uses ideas we do not know yet. The statement `from stanfordkarel import *` just adds Karel to our file ready to be use. And the code on lines 11 and 12 is a bit beyond our level, but just know that it activates the Karel graphical window that you'll see when you run this code. All you need to worry about is writing Karel code inside `main()`!

When you run this code in command prompt, a window will open:

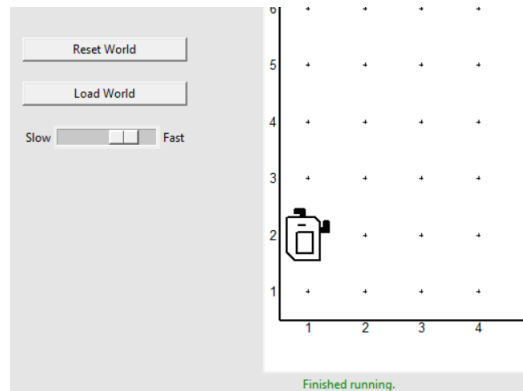


The Karel GUI.

In the caption above, I called this window a *GUI*. You’ll see that term come up all the time in programming. It stands for Graphical User Interface, and it just means that the Karel software isn’t command-line based but instead has a window you can click around in. Neat!

To open a world, click “Load World”. To run your code inside your `main()` function, click “Run Program.” Try it now and see what happens!

Notice that she goes up one? That’s because she turns left, which causes her to point up. Then she moves, and then she turns left, which has her pointing left. (Though in Karel the Robot, she’ll always be upside down when facing left. She can still move just fine though!)



Up One, Facing Left.

Notice also that instead of writing `move`, we write `move()`, and instead of writing `turn_left`, we write `turn_left()`. This is because remember how with functions we pass in inputs through the parentheses? For example, our function `bigger(3,5)` returned the bigger number of the two inputs. Well here, `move()` takes in no input, so the parentheses, while still necessary, are empty. You'll get used to writing empty parentheses like `()` a lot when programming.

Incorrect		Correct
<code>move</code> <code>turn_left</code> <code>pick_beeper</code> <code>put_beeper</code>	VS	<code>move()</code> <code>turn_left()</code> <code>pick_beeper()</code> <code>put_beeper()</code>

With the four main functions and knowledge on how to use Karel, you're ready to do some puzzles! Here's some problems I'd like you to do:



Pick what feels comfortable.

Problems

1. Write code that picks up the beeper on `collect_newspaper_karel.w`. You can directly launch a world without using the “Load World” button by changing the code to:

```
1         if __name__ == "__main__":
2             run_karel_program("collect_newspaper_karel")
```

Remember also there’s a slider to control the robot’s speed. For some reason it’s super low when you load a world directly as I showed above, so you may want to drag it up.

2. Write a function `turn_around()` that turns Karel around using only `turn_left()`.
3. Write a function `turn_right()` that turns Karel right using only `turn_left()`.
4. Write a function `turn_around_two()` that turns Karel around, but this time only uses `turn_right()` from the previous problem.
5. Modify your code for `collect_newspaper_karel.w` so that it places the beeper in the bottom left corner of the world.