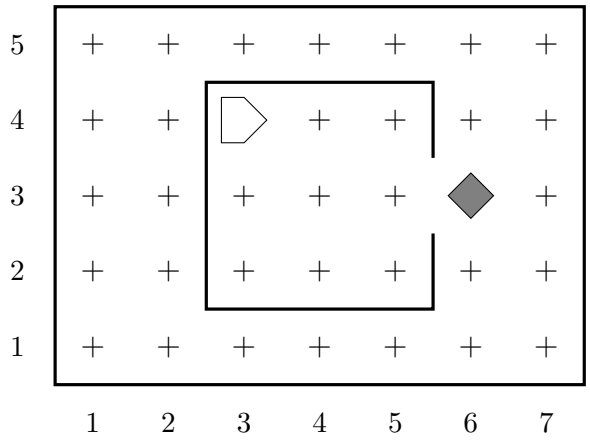
Both questions on this first problem set have template files in the repository. You'll need to grab the assignment from the link below and download the entire folder to your computer before proceeding. I would highly suggest then opening the *entire folder* in VSCode, as it will make it very easy to edit the different files and is necessary for VSCode to find certain libraries. When you are finished, upload your completed templates back to GitHub. Don't worry about changing any file names, you want to overwrite the original template files.

Get Assignment link: https://classroom.github.com/a/mS4t18BW

1. Your first computational task is to solve a simple story-problem in Karel's world. Suppose that Karel has settled into its house, which is the square area in the center of the following diagram:



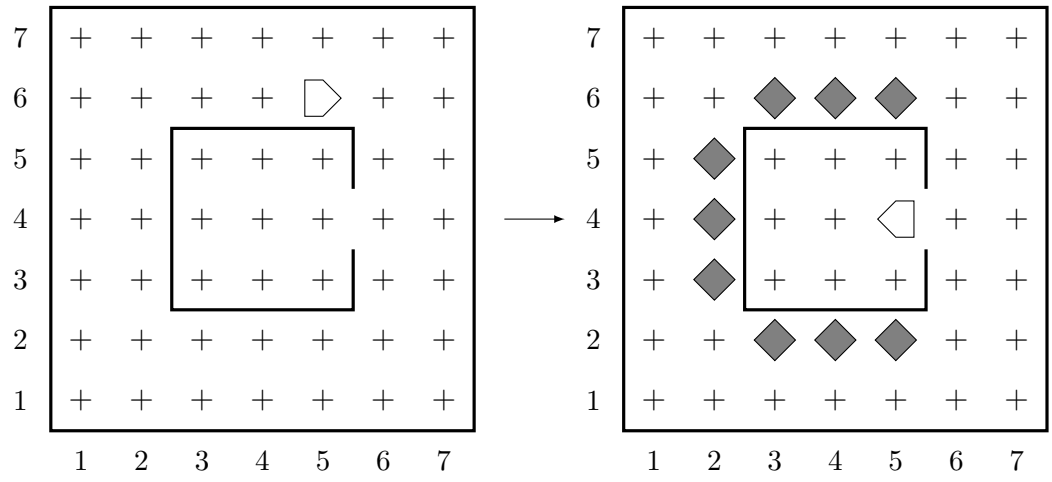
Karel starts off in the northwest corner of its house, facing east, as shown in the diagram. The problem is to program Karel to collect the newspaper–represented by the beeper–from outside the doorway and then to return to its initial position.

This exercise is no more complicated than our starting example and exists just to get you started and warmed up. You can assume that every part of the world looks just like it does in the above diagram. The house is exactly this size, the door is always positioned in the center of the east wall as shown, and the beeper is just outside the "door". Thus, all you have to do is write the sequence of commands necessary to have Karel

1. move to the newspaper,
2. pick up the newspaper,
3. and then return to its original starting point.

Even though the program is simple and only a few lines, it is still worth getting at least a little practice in decomposition. In your solution, include a helper function for each of the steps mentioned above and then use them together to accomplish your goal. You can write all code in the provided template entitled `Karel_Newspaper.py`.

2. In keeping with our story-problems involving Karel and its house, let us now suppose that Karel is wanting to "paint" the outside of its house on the north, west, and south sides. Karel will "paint" by placing beepers along those walls. Karel will always start on the east edge of the north wall, as depicted in the diagram to the below left.



Your goal in this problem is to write a program that paints *only* the north, west, and south walls, and then moves Karel to enter the house. The final state would look like the above right diagram.

This would feel very similar to the first problem, *except* that we want this program to be robust **no matter what the dimensions** of Karel's house are. In addition, while Karel will always start on the east edge of the north wall, **the direction Karel will initially be facing will not be consistent**, and so you will have to figure out your facing direction probably as a first step. You are safe to assume that:

- Karel has an infinite supply of beepers initially, and so could paint a house of any size without issue.
- The house will always have at least a 2 avenue/street gap between the sides of the house and the boundaries of the map.
- There will always be a door **somewhere** on the east side of the house. But its exact location could vary.

To help you test that your program will correctly solve the task for any general house, there are multiple worlds included in the repository that you can load and test your program against. `Karel_Painting.w` is the default, and matches the world sketched above. There are then 3 different worlds, which have differently sized houses, different directions Karel is initially facing, and different front door locations:

- `Painting1.w`
- `Painting2.w`

- `Painting3.w`

You can test your script against these worlds be running your program, and then clicking the far left button to open and load in the new world. Then you can press play to test your program against that world. Your same script should successfully work in *all* of the worlds, without any changes to the script. Proper use of loops and predicate functions will be key here! A basic template to get you started is in `Karel_Painting.py`.

Hints:

- Really think of how you want to decompose this problem. I would argue that you are essentially trying to accomplish the same task 3 different times, so that might guide or inspire how you want to break things down.
- Just because loops will be useful, you don't need to have *all* your code inside the loop! Frequently, you might need a line or two of code outside any loops to "reposition" Karel a bit.
- You have lots of potential predicate functions to choose from to determine where the wall is that you are trying to paint and move along. Many of them can work, but some streamline the code more than others.

Overall Checklist: Before you submit the assignment make sure that:

- ☐ You have practiced decomposition in both problems: breaking the overall problem up into smaller problems and using helper functions to group up the commands to solve that smaller problem.
- ☐ You have used comments to document your code. At the very least, you should have a comment explaining what each helper function is doing.
- ☐ You have filled out the meta-data at the top of the template, in particular indicating if you worked with anyone.