Eric Roberts
CS 151

Handout #3
September 1, 2020

Problem Set #2

Due: 5:00 P.M., Friday, September 16

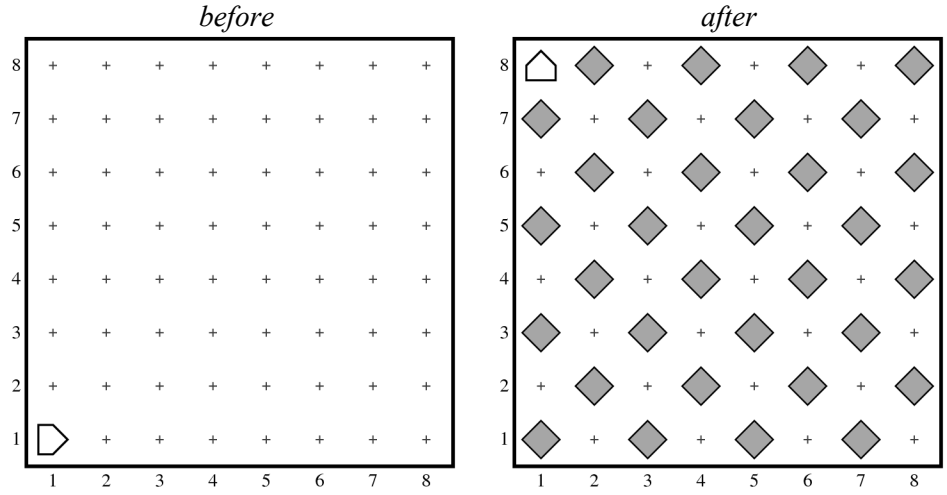
To make up for the fact that we lost a day to the Labor Day holiday, we've moved the final Karel problem to this week. The other two problems get you started with Python programming.

Before starting on this problem set, you should complete the following preliminaries:

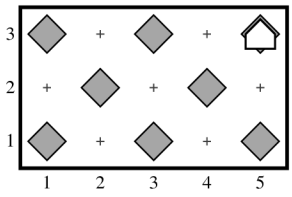
1. Download VSCode for your system from https://code.visualstudio.com/download and move the executable file into your applications folder.
2. Download the starter files for Problem Set #2 from the WISE site. You should then be able to open the Python starter file from inside VSCode.

Problem 1

In this exercise, your job is to get Karel to create a checkerboard pattern of beepers inside an empty rectangular world, as illustrated in the following before-and-after diagram:



This problem has a nice decomposition structure along with some interesting algorithmic issues. As you think about how you will solve the problem, you should make sure that your solution works with checkerboards that are different in size from the standard 8x8 checkerboard shown in the example. Odd-sized checkerboards are tricky, and you should make sure that your program generates the following pattern in a 5x3 world:



Another special case you need to consider is that of a world which is only one column wide or one row high. The starter folder contains several sample worlds that test these special cases, and you should make sure that your program works for each of them.

Problem 2 (Chapter 1, exercise 6, page 33)

It is a beautiful thing, the destruction of words.

—Syme in George Orwell's *1984*

In Orwell's novel, Syme and his colleagues at the Ministry of Truth are engaged in simplifying English into a more regular language called *Newspeak.* As Orwell describes in his appendix entitled "The Principles of Newspeak," words can take a variety of prefixes to eliminate the need for the massive number of words we have in English. For example, Orwell writes,

> Any word—this again applied in principle to every word in the language—could be negatived by adding the affix *un-*, or could be strengthened by the affix *plus-*, or, for still greater emphasis, *doubleplus-*. Thus, for example, *uncold* meant "warm," while *pluscold* and *doublepluscold* meant, respectively, "very cold" and "superlatively cold."

Define three functions—`negate`, `intensify`, and `reinforce`—that take a string and add the prefixes `"un"`, `"plus"`, and `"double"` to that string, respectively. Your function definitions should allow you to generate the following console session:

```
IDLE
>>> from newspeak import negate, intensify, reinforce
>>> negate("cold")
uncold
>>> intensify("cold")
pluscold
>>> reinforce(intensify("cold"))
doublepluscold
>>> reinforce(intensify(negate("good")))
doubleplusungood
>>>
```

Problem 3 (adapted from Chapter 2, exercise 3, page 68)

Why is everything either at sixes or at sevens?

—Gilbert and Sullivan, *H.M.S. Pinafore,* 1878

3a) Write a program that displays the integers in the range 1 to 100 (which in Python's interpretation of ranges does not include 100) that are divisible by either 6 or 7 but not both. In this version, your program should print each number on the console, one per line.

3b) Rewrite this problem so that it calls a function `list_divisible_by_six_or_seven`, which takes two parameters indicating the lower and upper limit of the range and *returns* a list containing the values you printed in part 3a. To create the list, you need to start with an empty list and then use concatenation to add each new value.