Eric Roberts
CS 208E

Handout #6
October 6, 2016

Assignment #2

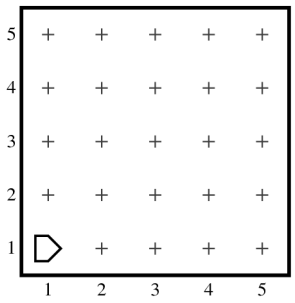
Due: Tuesday, January 19

Problem 1

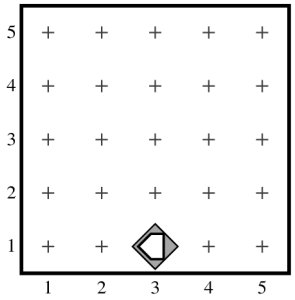
Solve the checkerboard problem in the puzzle box on page 24.

Problem 2

As an exercise in solving algorithmic problems in Karel's world, write a function called **`findMidpoint`** that causes Karel to place a single beeper at the center of 1st Street. For example, if Karel starts in the world



calling **`findMidpoint`** should leave Karel standing on a beeper in the following position:



Note that the final configuration of the world should have only a single beeper at the midpoint of 1st Street. Along the way, Karel is allowed to place additional beepers wherever it wants to, but must pick them all up again before it finishes.

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

- Karel starts at 1st Avenue and 1st Street, facing east, with an infinite number of beepers in its bag.
- 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.

Your program, moreover, can assume the following simplifications:

- If the width of the world is odd, Karel must put the beeper in the center square. If the width is even, Karel may drop the beeper on either of the two center squares.
- It does not matter which direction Karel is facing at the end of the run.

There are many different algorithms you can use to solve this problem. The interesting part of this assignment is to come up with a strategy that works.

Problem 3

Greek mathematicians took a special interest in numbers that are equal to the sum of their ***proper divisors,*** which is simply any divisor less than the number itself. They called such numbers ***perfect numbers.*** For example, 6 is a perfect number because it is the sum of 1, 2, and 3, which are the integers less than 6 that divide evenly into 6. Similarly, 28 is a perfect number because it is the sum of 1, 2, 4, 7, and 14.

Write a JavaScript function `isPerfect` that takes a number `n` and returns `true` if `n` is perfect, and `false` otherwise. Test your implementation by writing a test function that uses the `isPerfect` method to check for perfect numbers in the range 1 to 9999 by testing each number in turn. When a perfect number is found, your program should use the `Console.println` function to display it on the console. The first two lines of output should be 6 and 28. Your program should find two other perfect numbers in the range as well.