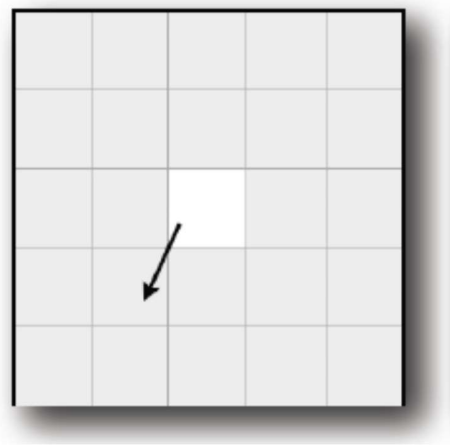
Purpose

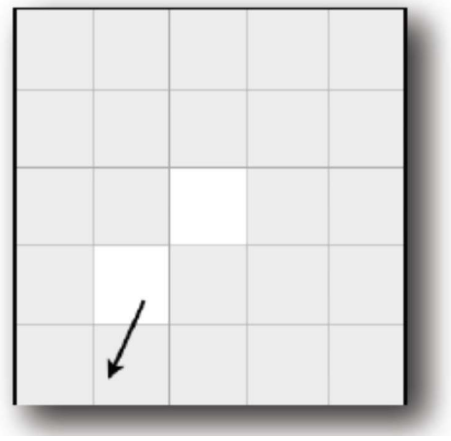
This exercise must be solved without using objects or classes. Please do not use the object and class resources in this exercise.

You have to simulate a robot floor cleaner. Let's see an example to explain what we are going to do (in the example the floor is 25 tiles, 5 x 5 units):

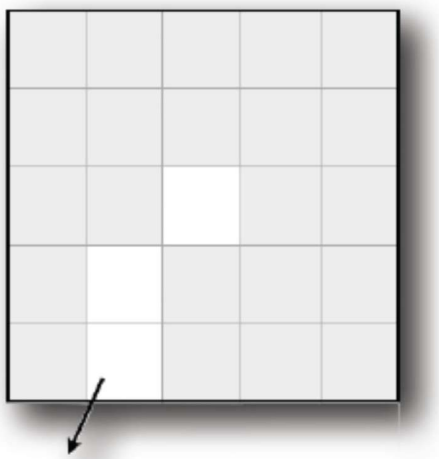
a. Time t = 0. The robot is at position (2.1, 2.2), looking at 205 degrees (clockwise, from "north"). The robot cleans the tile it is on.



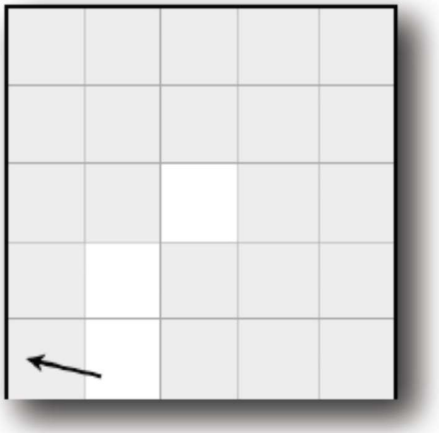
b. Time t = 1. The robot moves one unit, in the direction it was looking, to position (1.7, 1.3), and cleans another tile.



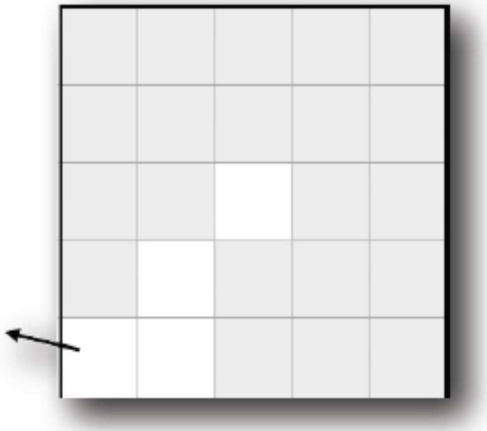
c. Time t = 2. The robot moves another unit in the same direction, to position (1.2, 0.4), and cleans another tile.



d. d. Time t = 3. The robot cannot move in the same direction because it collides with the wall. So it changes its direction randomly. In this example, 287 degrees.



e. hTime t = 4. The robot moves one unit in the new direction up to (0.3, 0.7), cleaning another tile.



Los detalles de la simulación son:

- La habitación es rectangular, de largo *h* y ancho *w*. El tamaño de la habitación es pasado por línea de comandos y es tal que h <= 100 y w <=70.
- El piso está dividido en baldosas de una unidad por una unidad. Las baldosas pueden estar limpias o sucias. Inicialmente todo el piso está sucio.
- Hay *n* robots limpiando en simultáneo. La cantidad de robots se pasa por línea de comando.
- Para simplificar las cosas asumiremos que los robots son puntos y se pueden cruzar sin interferirse.
- La posición de un robot se describe con dos números reales, (x, y), y su dirección con un ángulo real *d* ($0 \leq d < 360$). La baldosa en que se encuentra es (floor(x), floor(y)).
- Inicialmente los robots se encuentran dentro de la habitación, en posiciones y direcciones al azar.

Simulation Details:

• The room is rectangular, of length h and width w. The size of the room is passed through the console.

• The floor is divided into one-unit per one-unit tiles. The tiles can be clean or dirty. Initially the whole floor is dirty.

• There are n robots cleaning simultaneously. The number of robots is passed on the console at the beginning of the program.

• To keep things simple, we will assume that robots are points and can be crossed without interfering.

• The position of a robot is described by two real numbers, (x, y), and its direction by a real angle d ($0 \leq d < 360$). The tile it is on is (floor (x), floor (y)).

• Initially the robots are inside the room, in random positions and directions.

• The simulation progresses one time unit at a time (“tick”). In each unit of time a robot moves one unit of distance from where it is, in the direction it is.

• When a robot collides with a wall it retains its previous position, and selects a new direction at random.

• When a robot steps on a tile, it is considered to have cleaned that tile.

• The program ends when all the tiles are clean.

• It is requested to develop a program that allows the user to choose between two modes to be entered by console:

a. Mode 1: The program will show in real time how the floor is being cleaned by the robots; graphing the clean and dirty tiles, the robots and their directions.

b. Mode 2. The program will create a graph of the average delay time in cleaning the order as a function of the number of robots (n), for the given floor size. This way it will start with n = 1 and continue iterating until the mean time for n and n + 1 is less than 0.1. For each step in n (each increment in n), 1000 simulations should be run to estimate the average time it takes to clean the floor with that number of robots.

Notes:

• The program receives by console, the size of the floor (Width and Height), the number of robots (Robots) and the simulation mode (Mode).

• In Mode 2, the Robots parameter is ignored.

• The program in Mode 1 reports the number of ticks required to clean the floor according to the parameters received. In Mode 2 it displays the graph as described above.