ADA - Análisis y Diseño de Algoritmos, 2019-2
Tarea 4: Semanas 11 y 12

Para entregar el martes 22 de octubre de 2019

Problemas conceptuales a las 09:00 en clase

Problemas prácticos a las 23:59 en la arena de programación

Tanto los ejercicios como los problemas deben ser resueltos, pero únicamente las soluciones de los problemas deben ser entregadas. La intención de los ejercicios es entrenarlo para que domine el material del curso; a pesar de que no debe entregar soluciones a los ejercicios, usted es responsable del material cubierto en ellos.

Instrucciones para la entrega

Para esta tarea y todas las tareas futuras, la entrega de soluciones es *individual*. Por favor escriba claramente su nombre, código de estudiante y sección en cada hoja impresa entregada o en cada archivo de código (a modo de comentario). Adicionalmente, agregue la información de fecha y nombres de compañeros con los que colaboró; igualmente cite cualquier fuente de información que utilizó.

¿Cómo describir un algoritmo?

En algunos ejercicios y problemas se pide "dar un algoritmo" para resolver un problema. Una solución debe tomar la forma de un pequeño ensayo (es decir, un par de párrafos). En particular, una solución debe resumir en un párrafo el problema y cuáles son los resultados de la solución. Además, se deben incluir párrafos con la siguiente información:

- una descripción del algoritmo en castellano y, si es útil, pseudo-código;
- por lo menos un diagrama o ejemplo que muestre cómo funciona el algoritmo;
- una demostración de la corrección del algoritmo; y
- un análisis de la complejidad temporal del algoritmo.

Recuerde que su objetivo es comunicar claramente un algoritmo. Las soluciones algorítmicas correctas y descritas *claramente* recibirán alta calificación; soluciones complejas, obtusas o mal presentadas recibirán baja calificación.

Ejercicios

34.1-1, 34.1-2, 34.1-3, 34.1-4 (página 1060), 34.1-5, 34.1-6 (página 1061), 34.2-1, 34.2-2, 34.2-3 (página 1065), 34.2-5, 34.2-6, 34.2-8, 34.2-9 (página 1066), 34.3-2, 34.3-3, 34.3-5, 34.3-6 (página 1077), 34.3-7, 34.3-8 (página 1078), 34.4-3 (página 1085), 34.4-5, 34.4-6 (página 1086), 34.5-1, 34.5-2 (página 1100), 34.5-4, 34.5-6 (página 1101).

Problemas conceptuales

1. Ejercicio 12.3: *DNF* (Erickson, página 416).
2. Ejercicio 12.5 (a-e): *Magic Black Boxes* (Erickson, página 416).
3. Ejercicio 12.8: *Hamiltonian Cycles* (Erickson, página 418).

Problemas prácticos

Hay un problema práctico cuyo enunciado aparece a partir de la siguiente página.

A - Collecting Beepers

Source file name: `beep.py`
Time limit: 1 second

Karel is a robot who lives in a rectangular coordinate system where each place is designated by a set of integer coordinates (x and y). Your job is to design a program that will help Karel pick up a number of beepers that are placed in her world. To do so you must direct Karel to the position where each beeper is located. Your job is to write a computer program that finds the length of the shortest path that will get Karel from her starting position, to each of the beepers, and return back again to the starting position.

Karel can only move along the x and y axis, never diagonally. Moving from one position (i, j) to an adjacent position $(i, j + 1)$, $(i, j - 1)$, $(i - 1, j)$, $(i + 1, j)$ has a cost of one. You can assume that Karel's world is never larger than $20x20$ squares and that there will never be more than 10 beepers to pick up. Each coordinate will be given as a pair (x, y) where each value will be in the range 1 through the size of that particular direction of the coordinate system.

Input

First there will be a line containing the number of scenarios you are asked to help Karel in. For each scenario there will first be a line containing the size of the world. This will be given as two integers (x-size and y-size). Next there will be one line containing two numbers giving the starting position of Karel. On the next line there will be one number giving the number of beepers. For each beeper there will be a line containing two numbers giving the coordinates of each beeper.

The input must be read from standard input.

Output

The output will be one line per scenario, giving the minimum distance that Karel has to move from her starting position to get each of the beepers and back again to the starting position.

The output must be written to standard output.

Sample Input	**Sample Output**
1 10 10 1 1 4 2 3 5 5 9 4 6 5	The shortest path has length 24