IKI10100I: Data Structures & Algorithms • 2015-16
Faculty of Computer Science, Universitas Indonesia
Tutorial 1 (Week 1): ***Rendering a 2D gameworld***
Deadline: Monday, 15th February 2016, 17:00

In this worksheet you are asked to read a file that contains a definition for a 2D gameworld, which consists of a 30 × 30 grid, where each cell represents a location in the gameworld. Each location will have a code that defines the environment or static object at that location. There are 5 codes:

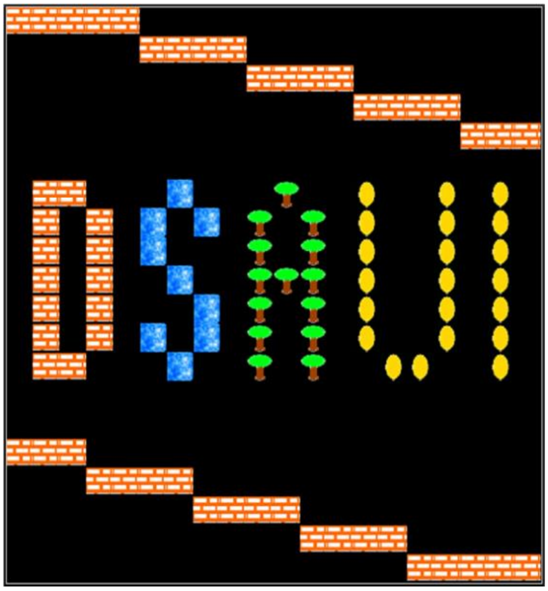
0 is free area: , 1 is a wall: , 2 is a tree: , 3 is water: , and 4 is a collectible coin: .

The gameworld can be represented as a two-dimensional array of integers, where the integer stored at cell position [*y*][*x*] represents the code of the location at that coordinate, i.e. at row *y* and column *x*.

Gameworld maps are stored in *map files*, which are simply text files with a `.map` extension. A map file contains 30 lines, where each line contains exactly 30 integers separated by a comma (‘`,`’) character.

Here is an example of a 20x20 map file called `test.map` and to the right is a screenshot of how the gameworld should appear once it is correctly rendered:

```
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0,1,1,0,0,0,2,0,0,0,3,0,0,4,0,0,4,0,4,0
0,1,0,1,0,2,0,2,0,3,0,3,0,4,0,0,4,0,4,0
0,1,0,1,0,2,0,0,0,3,0,3,0,4,0,0,4,0,4,0
0,1,0,1,0,0,2,0,0,3,3,3,0,4,0,0,4,0,4,0
0,1,0,1,0,0,0,2,0,3,0,3,0,4,0,0,4,0,4,0
0,1,0,1,0,2,0,2,0,3,0,3,0,4,0,0,4,0,4,0
0,1,1,0,0,0,2,0,0,3,0,3,0,0,4,4,0,0,4,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0,0
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1,1
```



You need to write a function with the following definition: `def loadMap(filename)`

When this function is called with a parameter containing the name of a map file, it will read the contents of that file and return a “2D list”, i.e. a list-of-lists, that contains the representation of the gameworld.

Once you have obtained such a gameworld representation, the second task is to *render* this gameworld on a Tkinter object known as a `Canvas`. On the canvas, each cell is represented by an icon/image with a size of 20 × 20 pixels. Since the gameworld is a 30 × 30 grid, the canvas size should be 600 × 600 pixels. You can use the provided image files (`0.gif`, `1.gif`, `2.gif`, `3.gif`, and `4.gif` in the directories `sprite` and `sprite/emoji`), or you can create your own.

The function that carries out this rendering should have the following definition: `def renderMap(map)`

This function should first create the Tk root object and the `Canvas` object. Once you have created your `Canvas` object, you may use its `create_image` method to render an image at a specific location on the canvas. See the documentation[1] for details, and make sure you understand the necessary options!

[1] http://infohost.nmt.edu/tcc/help/pubs/tkinter/web/create_image.html