README – Project 1, part B

Subject: Finding the exit in a labyrinth.

Labyrinth is given as a text file of 0 and 1, where 0 is considered path, 1 is considered wall, E is the entrance point in the labyrinth.

import java.util.Scanner, java.io.File: used to read and store the (labyrinth) file locally in the program

import java.util.Arrays: used for arrays

import java.io.FileNotFoundException: for exceptions handling

main (String [] args): the args of main will be used to pass the wanted labyrinth file

String[][] readAndStoreFile(String stringAddress): reading, checking, storing and return of the array

About the program:

In String [] args we store the name of the (main) class (1st argument) and the address of the labyrinth file (2nd argument) . Αποθήκευση του πίνακα συμβολοσειράς σε συμβολοσειρά - τοπική μεταβλητή.

readAndStoreFile: reads the File labyrinthFile, dimensions, coordinates of Entrance, stores the rest of the labyrinth in a 2-d String array which is then returned

pathStack: stack that stores the path you walk, used for backtracking when you find a deadend

goingRightIsOk, goingDownIsOk, goingLeftIsOk, goingUpIsOk: booleans to avoid going out of bounds if E is at the borders (and we do not want to search "outside" the labyrinth)

Ex, Ey the entrance coordinates,x, y the current coordinates of the player searching for the exit.

While (exitNotFound) loop: the player starts walking in the labyrinth, checking (in this order) right, down, left and upper cell if it can move. If it can move it proceeds to this direction and at every cell it looks again right, down, left, up if it can move. As long it can the pathStack is being pushed with the cells visited and the cells visited are marked as CFL, CFU, CFR, CFD showing the direction from where the player came to this cell (so it can backtrack to its previous one if it finds a dead end). If dead end is found, the player returns to its previous cell (current cell is popped out of the pathStack) and checks the rest of its options (e.g. if at the previous cell it went down, then the next choice is going left). Every time the player backtracks-returns to a previous cell because it finds a dead end, before leaving the current cell it blocks the cell by marking it as a 1-valued cell (Wall) so it may never be chosen again.

c81-visual example.txt and c81-stack.txt visualize the way the player moves inside the labyrinth file c81.txt. The c81-visual example.txt shows the path of the player beginning from first (0) line and the forth (3) row and the c81-stack.txt shows how the stack works while this is happening.

You can create your own 0 and 1 labyrinths and let the pc find a way out of them. Just keep in mind the “Readme about the labyrinth files” txt. Have fun !