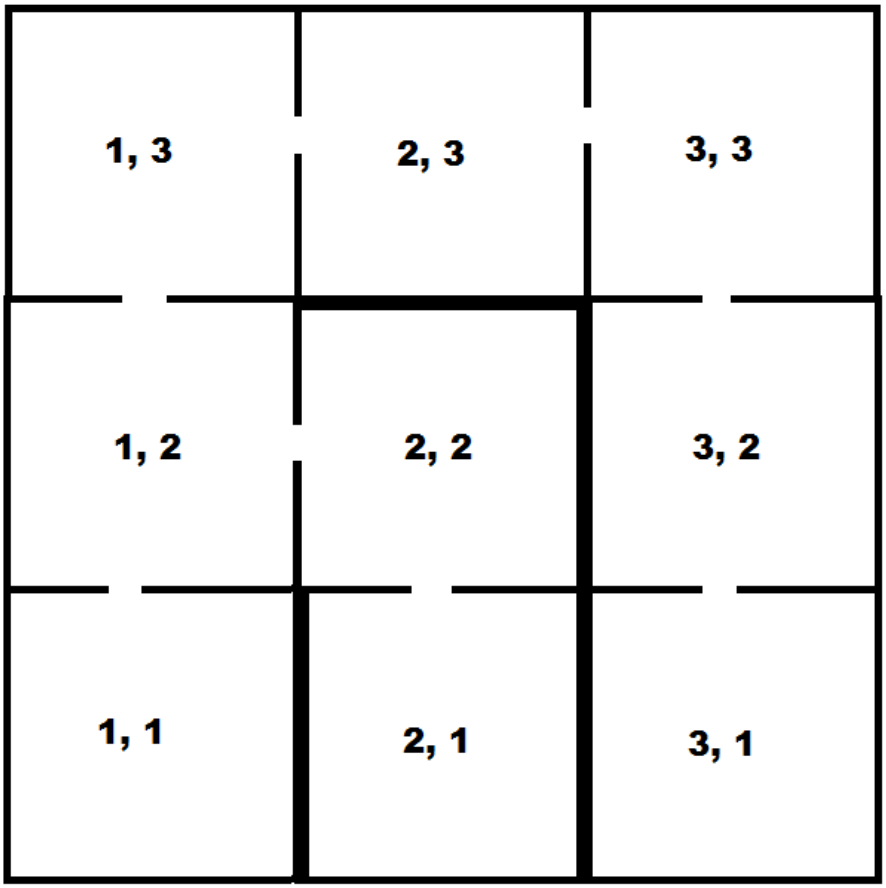
# Tile Traveller

This assignment contains two parts. You need to implement one program in each part and you have to use git during the development. Create one directory, named TileTraveller, on your local computer that will contain the two programs. In addition, you need to push your solution to github.

The assignment is to develop a computer game in which the player is located in a certain tile in a grid. At each iteration, the program displays the directions for which there are adjacent tiles that the player can travel to.

The program only displays text, so you don't actually draw the tile grid, but the program should behave as if the player is in a 3x3 grid with open and closed walls as seen in the following image:




The player starts in tile (1,1). At the beginning, and after each move selected by the player, the program should print the player's travel options. If there is an open wall in a direction, write that direction as a possible travel direction.

At each iteration, the player enters the first letter of the direction he/she wishes to travel, after which the player should "be" in another tile and the options for the new tile are then printed out.

The player enters:

- n/N for north (up)
- e/E for east (right)
- s/S for south (down)
- w/W for west (left)

If the player enters an invalid direction, the program prints “Not a valid direction!” and allows the player to enter the direction again.

For example, in tile (1,1) it’s only possible to move north. In tile (1,2), the possible moves are north, east and south, and in tile (3,3) the valid moves are south and west.

Tile (3,1) is the victory location. When the player enters this tile the program should notify him/her of their victory and quit running.

Example run:

```
You can travel: (N)orth.
Direction: s
Not a valid direction!
Direction: n
You can travel: (N)orth or (E)ast or (S)outh.
Direction: N
You can travel: (E)ast or (S)outh.
Direction: w
Not a valid direction!
Direction: E
You can travel: (E)ast or (W)est.
Direction: e
You can travel: (S)outh or (W)est.
Direction: s
You can travel: (N)orth or (S)outh.
Direction: S
Victory!
```

**Implementation 1:**

First, together as a group, design an algorithm as a description for solving the game. Then implement the game without using any functions. Put your algorithm at the top of the program file as a comment. When you have thoroughly tested your implementation in VS Code, submit your solution in Mimir.

**Remember: You need use git while developing your solution.**
Put the path to your github repo as a comment in your program file.

Next, discuss in your group what problems you see with the first implementation. Can some of the problems be solved using functions?

**Implementation 2:**

Make a copy of your first program. Refactor your program, now moving some of the functionality into functions. When you are done, put a comment at the top of the program file which answers the following questions:

1. Which implementation was easier and why?
2. Which implementation is more readable and why?
3. Which problems in the first implementations were you able to solve with the latter implementation?

Again, test your implementation thoroughly in VS Code before submitting your solution in Mimir.

**Remember: You need use git while developing your solution.**
Put the path to your github repo as a comment in your program file.

Your grade will be based on the functionality of your implementation, the answers you give and the repo that contains your solutions.