









# Practice 1: Informed search

We want to study ways to find paths to move quickly through a map with different heights.

The map will be represented by a matrix of  $X \times Y$  cells (10x10 in the example below). Each cell has an associated height value; some cells may have an insurmountable cliff. Each problem will be defined by a map configuration, an initial cell ( $x_i, y_i$ ), and a final cell ( $x_f, y_f$ ). A specific problem may look like this:

0	1	2	3	3		0	1	2	2
2	1	2	4	3	2	1	3	3	3
2	2		4	5	3	2		4	4
3	3		4	6	4	2	3	3	3
2	2	3	3	5	3	3	2	3	3
2	1	1	3	4		2	2	3	4
2	0	0	1	2	1	1	1	1	
	1	0	1	2	0	2	3	2	3
2	2	2	1	1	0	2	3	3	4
4	3	2	2		1	1	2	4	5

We can move horizontally and vertically, but not diagonally. The time it takes to move will be:

- $1 + (\text{height difference between destination cell and origin cell})$ , if the difference is positive or 0.
- $1/2$  unit, if the difference is negative.
- We cannot move to a cell with a cliff.

We can consider that the height distribution is geographically consistent (mountains, valleys, etc.), but cliffs are random.

The information available to calculate heuristics are:

- The coordinates ( $x, y$ ) of the current and final cells
- The height of the current and final cells
- The path followed until the current cell

# Tasks

We ask you to:

- Formalise the problem by **defining the states (with attributes) and the operators**.
- Design **3 well differentiated heuristics** (they do not need to be the 3 best ones, but they must be different) to find the fastest paths from the start to the end.
- For each heuristic, **justify whether they are admissible or not with regards to time**. It is not necessary that the 3 heuristics are admissible, but at least one of them should.
- Implement a Java program with the **best first** and **A\*** algorithms with the formalization and heuristics you have defined. The input map should be read from a file, and the output of the program should be shown in the console.
- Test **both algorithms with the 3 heuristics for different problems** (the map shown above, with start a 0,0 and end at 9,9, and, at least, another 10x10 map you design with start and end at the most distant corners), and report:
  - the solution (path) and its time,
  - the number of cells that has been “treated” by the algorithm (that is, the number of search iterations made),
  - whether the solution is optimal or not with respect to time,
  - analyse the influence of **each heuristic, algorithm and map into the number of treated cells and the solution found**
- Discuss whether the **hill climbing** algorithm would have been able to find any solution for each of the heuristics and maps. You do not need to implement the algorithm but justify your answer.

## Delivery

Practices should be done in groups of 2 students. Each group should do one delivery in the task in the virtual campus with the following:

- **Report** answering the questions above
- **Java code fragments** with the implementation of the **algorithms (Best-first and A\*)** and the **heuristics**. You only need to include the code of the specified parts. If they rely on any auxiliary function or class, they should also be included.

The delivery should be done in a zipped file with the name «**P1\_[NameSurname1-NameSurname].zip**»

There will be an interview with the teacher at the laboratory class following the delivery.

**Similar or identical deliveries will have a 0 mark.**

## Delivery dates

- Until March 16 at 20h (maximum grade 10, first call).
- Until May 27 at 12h (maximum grade 7, first call).
- Until June 12 at 12h (maximum grade 5, second call).