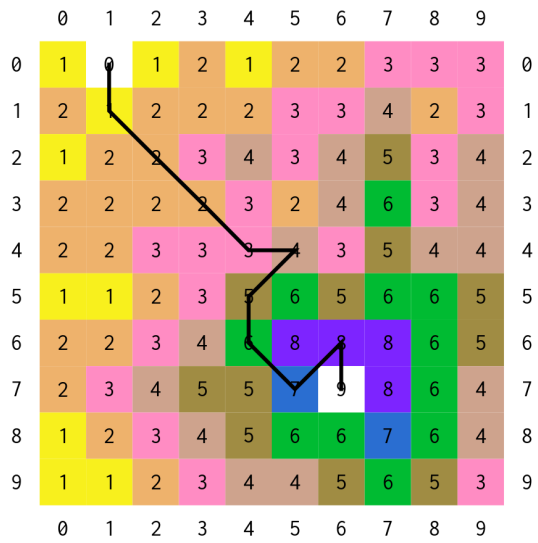
COS 485: Program #4 – Mountain Climber

Objectives: Designing a greedy algorithm.

The input is a height map of a mountain stored in a 2D array.

Write a program to find the best hiking path from the starting point to the peak. Where "best" is a combination of shortest and least steep. The hiker can move 1 cell at a time horizontally, vertically, or diagonally.



The cost of moving is the distance + a function of the change in height (Δh) of any elevation change.

The cost of a horizontal or vertical move is = $1 + |\Delta h|^3$

The cost of a diagonal move is = $\sqrt{2} + \frac{1}{2}|\Delta h|^3$

In the example it starts at (0,1) at height 0 and moves south to (1,1). The distance is 1 and $\Delta h = 1$ so by the cost function, the cost of that first move is 2.

The next move is diagonally from (1,1) to (2,2). The distance is $\sqrt{2}$ and $\Delta h = 1$ so by the diagonal cost function, the cost of the second move is $\sqrt{2} + \frac{1}{2}$

Steeper climbs are more difficult and have higher cost, and they are thus generally avoided if possible.

Setting up the project in Eclipse:
Create a new project similar to how you set up program 1:

- It will use MountainClimberTester.jar, the same Scaffold jar, and starting code MountainClimber.java
- In the run configuration set **Main class** to: tester/MountainClimberTester

What to turn in:
Written Report turned in through Brightspace

1. Your algorithm description
2. A screen shot of the report tab
3. A screen shot of your results for test4.txt

Electronic Submit
From a Unix machine in the lab run the program "submit" to submit your files.
Submit your source code (.java files) and compiled code (.class files) to the directory: **prog4**

Grading:

- 10 points – a brief description of your algorithm
- 40 points – finding paths with cost less than half that of the starting code's paths
- 50 points – finding optimal paths