Running the Example Code for the Final Exercise

To download and set up the code for searching for pillars work through the following steps.

1. Create a new directory for the final exercise in your home directory named e.g. **finalExercise.**
2. Download the file **finalExercise.tar.gz** from the summer school web repository: http://www.gs.unina.it/
3. Unzip and extract the files using the commands
   $ gunzip finalExercise.tar.gz
   $ tar xvf finalExercise.tar
4. Set the classpath using the script classpath.sh
   $ source ./classpath.sh
   Do not use previous versions of this script.
5. Compile the "glue" component code for accessing the web services by running the script wscompile.sh i.e.
   $ ./wscompile.sh
   If the script is not executable then use the command
   $ chmod 755 wscompile.sh
   And run the script again
6. Compile the code
   $ javac –classpath $CLASSPATH *.java

The code should now be set up to run.

Look at the code. Most of it has been provided previously for the "Day 2 Web Services" progressive exercise. The new code is contained in "Scanner.java". The description of how this works is given below. The idea is to use Scanner to identify an edge of a pillar and then use the program "Regular.java" to visualise the pillar, and locate the plaque and read the text. Regular generates a postscript file and the program will have to be used repeatedly for ever smaller bounding boxes to close in on the plaque and the text.
The Regular program can be run using the command line:

$ java –classpath $CLASSPATH Regular <lowerX> <lowerY> <upperX> <upperY> <count> <postscript file> <surface service URL>

Where the command is all on one line.

The inputs <lowerX>, <lowerY>, <upperX>, <upperY> are all doubles specifying the bounding box for sampling the surface (the lower corner and the upper corner). <count> is the number of grid points to use for sampling the surface (this is only an approximate number, the program does not necessarily produce that number only a number close to <count>). <postscript file> is the name for the output file and the <surface service URL> = http://server5.gs.unina.it:8080/PillarsOfWisdom/surface.

To visualise a postscript file with name "pillar.eps" use ghostview with the command:

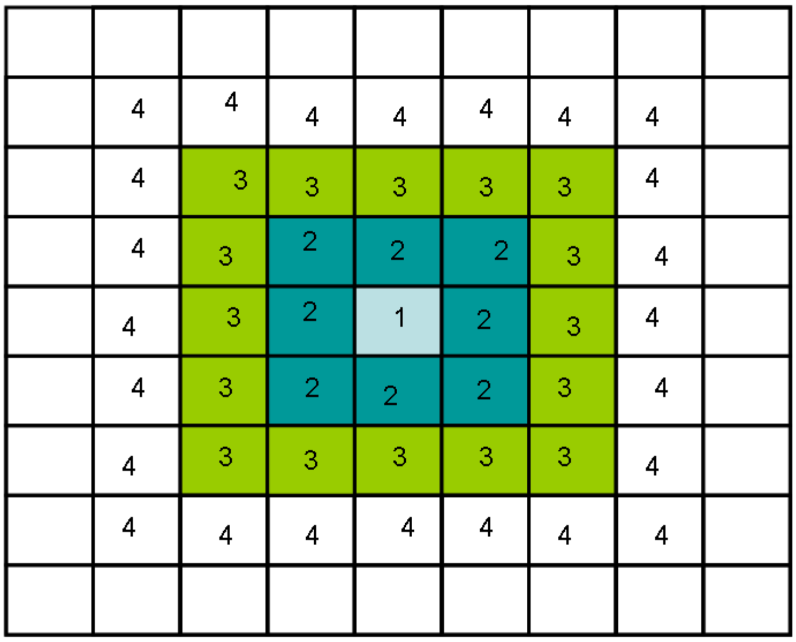
$ gv pillar.eps

Scanner

The feature scanner program is designed to look for flat features embedded in the surface that you have been working with. To run the scanner, use the following command line:

```
java –classpath $CLASSPATH Scanner <start X> <start Y> <radius>
<step>
```

The scanner repetitively samples a square section of the surface, retrieving at most one hundred sample points per section. It examines every point in the sample set, looking for a point on the surface with a gradient of zero. If it finds such a point, then it prints a message in the console detailing the coordinates of that point. The scanner moves the sample section around the surface in ever increasing concentric squares, scanning a larger area each time (but never over-lapping). This is shown in the diagram below, where the first square to be scanned is labelled ‘1’ and is centred on <start X> <start Y> .
The second set of sampling sections are labelled ‘2’ and form a square surrounding the first, and so on. The `<radius>` parameter determines the size of the largest concentric square, and thus the size of the area that is searched (in the diagram below the radius is half the length of the side of the square).
The `<step>` parameter is the size of the sample area (i.e. the dimensions of each square), and thus determines the density of the samples retrieved. All of these parameters are represented by double precision floating point variables.



Example Usage

A good general search configuration, for finding the initial location of features:

```
java –classpath $CLASSPATH Scanner <start X> <start Y> 10.0 1.0
```

To find a very small feature, the following parameters would be a good start:

```
java –classpath $CLASSPATH Scanner <start X> <start Y> 2.0 0.01
```