Geonomics Java Developer Programming Test #4.0.1:

Geo Clusters

Background

A rectangular grid is overlaid on a piece of a map such that the map is divided up into squares of land. These squares are called *Geos* and the encompassing rectangle is called a *GeoBlock*. Users can purchase these Geos. When they are purchased, they become "occupied".

Each Geo has an ID within the GeoBlock, starting from 0 in the bottom-left corner, moving left to right along the bottom row, then up to the next row, left to right along that and so on until the top right Geo is reached.

Below is an example GeoBlock. The occupied Geos are shown in red:

24	25	26	27
20	21	22	23
16	17	18	19
12	13	14	15
8	9	10	11
4	5	6	7
0	1	2	3

A *Geo Cluster* is defined as being a set of occupied Geos where each one is adjacent to (i.e. directly above, below, or to the left or right of) one or more other occupied Geos.

The clusters in the example above are:

- 4, 5 and 6
- 11 and 15
- 13, 17, 21 and 22

Note that another GeoBlock with the same number of Geos and occupied Geo IDs, but different dimensions (7x4 instead of 4x7) has different clusters:

21	22	23	24	25	26	27
14	15	16	17	18	19	20
7	8	9	10	11	12	13
0	1	2	3	4	5	6

The clusters in this GeoBlock are:

- 4, 5, 6, 11 and 13
- 15, 21 and 22
- 17

Occupied Geos are specified using an ASCII CSV file format. Each line contains the following fields:

Geo ID, Occupier's username, Date the Geo was occupied from in YYYY-MM-DD format

The CSV in this example looks like:

```
4, Tom, 2010-10-10
5, Katie, 2010-08-24
6, Nicole, 2011-01-09
11, Mel, 2011-01-01
13, Matt, 2010-10-14
15, Mel, 2011-01-01
17, Patrick, 2011-03-10
21, Catherine, 2011-02-25
22, Michael, 2011-02-25
```

Task

Write a command-line program in Java that takes three parameters:

- Width of GeoBlock (in number of Geos)
- Height of GeoBlock (in number of Geos)
- Name of the CSV file that defines the occupied Geos for that GeoBlock

It should output the details of the Geos in the cluster with the largest number of Geos in it. The details should include all the information about the Geos that was available in the CSV. Thus, using the example set of Geo Ids above, assuming the main() method of your program is in a class called GeoClusterAnalyser, the output should be:

```
$ java GeoClusterAnalyser 4 7 GeoBlockExample.csv
The Geos in the largest cluster of occupied Geos for this GeoBlock are:
13, Matt, 2010-10-14
17, Patrick, 2011-03-10
21, Catherine, 2011-02-25
22, Michael, 2011-02-25

$ java GeoClusterAnalyser 7 4 GeoBlockExample.csv
The Geos in the largest cluster of occupied Geos for this GeoBlock are:
4, Tom, 2010-10-10
5, Katie, 2010-08-24
6, Nicole, 2011-01-09
11, Mel, 2011-01-01
13, Matt, 2010-10-14

$
```

In the event that there is more than one cluster with the largest size, details of the cluster with the lowest sum of Geo IDs should be returned. So for example if there was a GeoBlock with two clusters (4, 8, 12) and (6, 7, 10), then details of the (6, 7, 10) cluster would be returned as (6+7+10) < (4+8+12).

Notes

- Your code should produce correct answers in under a second for a 10,000 x 10,000 Geo GeoBlock containing 10,000 occupied Geos.
- Your code needs to be 'production quality': it should be clear, easy to follow and maintainable. It should also have good error handling - including being able to tell the end user whether and *exactly where* in the CSV input there is an error (eg line too short or too long or invalid value).
- This is your chance to show that you can write idiomatic object-oriented Java code (as opposed to, say, Java-in-the-style-of-C).
- There is no time limit, but, as a guide, we wouldn't envisage the task taking longer than a couple of hours or so.
- Please don't publish the question or your solution online, but instead submit the answer in a zip file attached to an email.