CONGRESS

an, rev. XX, ad

U.S. Congress Apportion. The members of the U.S. Congress represent the states. There are more seats than states, in fact, there are 435 and 50 states, so many states receive more than one seat in Congress. This is how it should be: California has 33,930,798 residents while Wyoming has only 495,304, so California ought to receive many more seats. On the other hand, even though California has almost 70 times as many residents, it would be absurd if it had 70 times as many seats as Wyoming.

Huntington–Hill. Here's how it's done, using the Huntington–Hill method from 1911: First, each state receives a seat. (In fact, Article 1, Section 2 of the U.S. Constitution requires that.) Then, as long as there are seats left, we repeat the following: Give the next seat to the largest state, and decrease the state's population by a certain constant. The exact procedure, including the formula for the constant, can be found among the links below.

Requirements

Your algorithm must run in time proportional to $(m + n) \log m$ for m states and n seats. (Otherwise the large input files will indeed take too much time.)
You can write your own priority queue from scratch, or use Java's, or the book's, or grab one from the internet (make sure it's correct).

Tips

Yes, this is an exercise in priority queues, and to defeat the largest instance you need an implementation that runs in time proportional to $(n + m) \log m$. Depending on how much you trust yourself, you may want to first write a naïve algorithm with running time proportional to nm, to make sure you get the weird modification of populations right, before you switch to a fast priority queue. (As far as I know, the official U.S. Congress apportion is indeed computed in quadratic time, which is fast enough for the relatively tiny input produced by 435 seats.)

The files for this exercise are in the `congress` directory of the repository at `github.com/thorehusfeldt/bads-labs`.

Data files. Data files are in the `data/` directory. The files `1990-in.txt` and `2000-in.txt` contain the real-world data for the 1990 and 2000 U.S. censuses. The corresponding output files are the actual seats computed in the 106th and 108th congresses. The other files I made up. Files called `small-*-in.txt` are small instances that are useful for debugging, they are meant to make your life easier. The files called `huge-*-in.txt` are fictional population data for planets, the largest of these distributes the 1 million seats in the galactic senate among 200,000 planets. (I had fun with this. The names of the planets are created by an algorithm similar to what was used in the 1984 computer game Elite, picking random digrams from "..LEXEGEZACEBISOUSESARMAINDIREA.ERATENBERALAVETIEDORQUANTEISRION".)

Deliverables. Your java sources, including your own priority queue (if you wrote one – you don't have to.)

input file format:

```
2      ⟵ number of states
4      ⟵ number of seats
Gondor ⟵ name of 1st state
100    ⟵ population of 1st state
West Rohan ⟵ name of 2nd state
10     ⟵ population of 2nd state
```

output format:

(lines can be in any order)

state name ⟶ `West Rohan 1` ⟵ *number of seats*
`Gondor␣3`

separated by single space

References

1. "United States congressional apportionment." *Wikipedia: The Free Encyclopedia.*
2. "Huntington–Hill method." *Wikipedia: The Free Encyclopedia.*
3. Congressional apportionment, online resources maintained by the U.S. Census Bureau. www.census.gov/population/apportionment/.
4. "Oolite planet list." *Elite Wiki.* wiki.alioth.net/index.php/Oolite_planet_list.