B4 - Computer Numerical Analysis – Trade

B-CNA-410

Groundhog

Let's get an extended weather forecast today from a jittery, inconsistent, reddish brown rodent

3.0.5.1

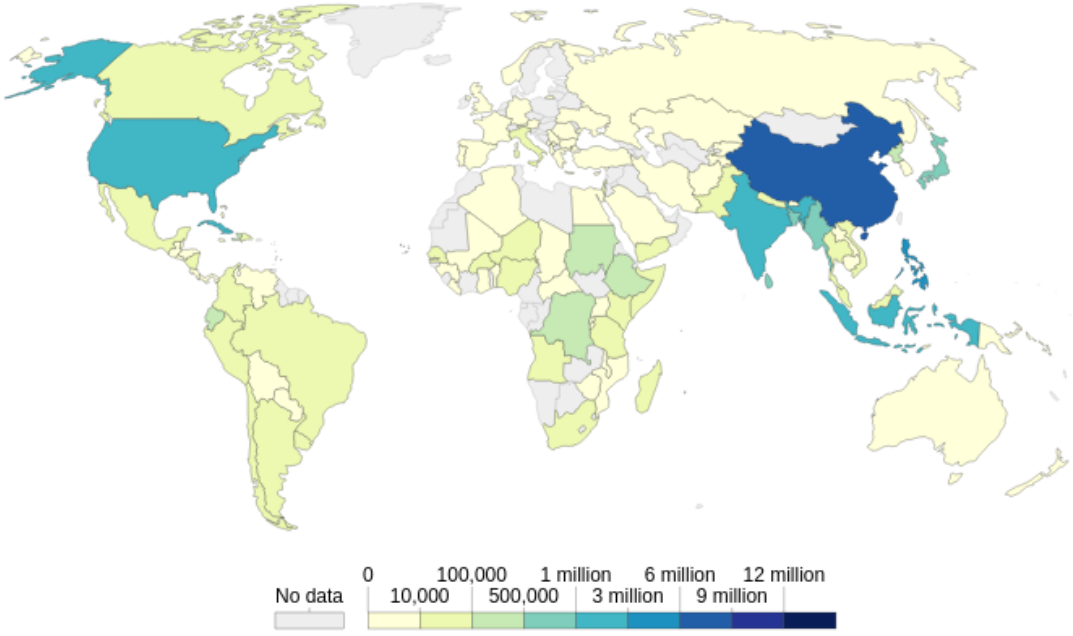
Groundhog

binary name: groundhog
language: everything working on "the dump"
compilation: via Makefile, including re, clean and fclean rules

- The totality of your source files, except all useless files (binary, temp files, obj files,...), must be included in your delivery.
- All the bonus files (including a potential specific Makefile) should be in a directory named *bonus*.
- Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).

You need to include a Makefile that builds a binary named groundhog - and remove the previous one if it exists

Internally displaced persons are defined as people who have been forced to flee or leave their homes a a results of natural or human-made disasters and who have NOT crossed an international border.



In 2016, weather disasters caused the displacement of 25 million people.
It also cost more than 100 billion dollars (*source*).

Despite popular opinion, weather forecasts are becoming more accurate than ever. Thanks to ingenious and talented people like you, who use smart tools to get better, faster and stronger predictions than the others.

Your job is to extract some relevant information from the data received in real-time on standard input (each float representing a temperature).

The goal is to detect weather **tendency** and **aberrations** (droughts, severe colds, hurricanes or any other extreme climatic condition whatsoever) as soon as possible.

> ***aberration**: deviation from what is normal, expected, or usual.*
>
> *collinsdictionnary.com*

To do so, your program must, given a number of days (called *period*) as argument:

1. **wait** for the next value to be written on the standard input
2. output, once enough data has been gathered, some indicators
    a. the **temperature increase average** `g` observed on the last period (decrease in temperature are not taken into account)
    b. the **relative temperature evolution** `r` between the last given temperature and the temperature observed n-days ago
    c. the **standard deviation** `s` of the temperatures observed during the last period
    d. when appropriate, an **alert** as soon as it detects a switch in global tendency
3. return to the first step, until the *STOP* keyword is read

A tendency is perceived when some values are moving in a particular direction over time.

Read again and dig each term, then do some research before any calculations.

Eventually, once the keyword is entered, it must output:

1. the total number of tendency switches observed on the whole time-series
2. the list of the five biggest aberrations observed on the whole time-series (sorted by decreasing weirdness)

Later, you may want to predict things on another timescale and use other *periods*.

It is **impossible** to detect a switch at the precise moment it happens - otherwise your algorithm would be way too sensitive to noise. However, you need to detect it quite soon, not at the very end of the algorithm.

Terminal

```
~/B-CNA-410> ./groundhog -h
SYNOPSIS
    ./groundhog period

DESCRIPTION
    period        the number of days defining a period
```

Rigor is one key to success. Be thorough and stay accurate.

EXAMPLE

You are expected to respect the following output formatting.
Results are separated by 2 tabulations.

Terminal

```
~/B-CNA-410> ./groundhog 7
27.7
g=nan       r=nan%       s=nan
31.0
g=nan       r=nan%       s=nan
32.7
g=nan       r=nan%       s=nan
34.7
g=nan       r=nan%       s=nan
35.9
g=nan       r=nan%       s=nan
37.4
g=nan       r=nan%       s=nan
38.2
g=nan       r=nan%       s=3.46
39.5
g=1.69       r=43%       s=2.82
40.3
g=1.33       r=30%       s=2.50
42.2
g=1.36       r=29%       s=2.40
41.3
g=1.07       r=19%       s=2.06
40.4
g=0.90       r=13%       s=1.56
39.8
g=0.69       r=6%       s=1.19
38.7
g=0.57       r=1%       s=1.07
36.5
g=0.39       r=-8%       s=1.72       a switch occurs
35.7
g=0.27       r=-11%       s=2.24
33.4
g=0.00       r=-21%       s=2.65
29.8
g=0.00       r=-28%       s=3.50
27.5
g=0.00       r=-32%       s=4.20
```

Terminal

```
25.2
g=0.00        r=-37%        s=4.64
24.7
g=0.00        r=-36%        s=4.51
23.1
g=0.00        r=-37%        s=4.36
22.8
g=0.00        r=-36%        s=3.58
22.7
g=0.00        r=-32%        s=2.48
23.6
g=0.13        r=-21%        s=1.60
24.3
g=0.23        r=-12%        s=0.91
24.5
g=0.26        r=-3%        s=0.77
26.7
g=0.57        r=8%        s=1.29        a switch occurs
27.0
g=0.61        r=17%        s=1.61
27.4
g=0.67        r=20%        s=1.71
29.8
g=1.01        r=31%        s=2.02
29.4
g=0.89        r=25%        s=1.98
31.5
g=1.09        r=30%        s=2.16
29.6
g=1.06        r=21%        s=1.64
29.8
g=0.77        r=12%        s=1.43
28.9
g=0.73        r=7%        s=1.13
28.7
g=0.67        r=5%        s=0.84
27.2
g=0.33        r=-9%        s=1.20        a switch occurs
25.7
g=0.33        r=-13%        s=1.74
26.0
g=0.07        r=-17%        s=1.56
25.2
g=0.07        r=-15%        s=1.67
21.6
g=0.04        r=-28%        s=2.30
20.3
g=0.04        r=-30%        s=2.77
```

Terminal

```
21.1
g=0.16        r=-26%        s=2.57
20.4
g=0.16        r=-25%        s=2.41
19.8
g=0.16        r=-23%        s=2.31
19.1
g=0.11        r=-27%        s=1.85
19.6
g=0.19        r=-22%        s=0.80
21.2
g=0.41        r=-2%        s=0.72
21.0
g=0.41        r=3%        s=0.77        a switch occurs
21.4
g=0.36        r=1%        s=0.82
24.0
g=0.73        r=18%        s=1.52
25.5
g=0.94        r=29%        s=2.13
25.5
g=0.94        r=34%        s=2.20
26.4
g=1.00        r=35%        s=2.16
29.4
g=1.20        r=39%        s=2.71
32.1
g=1.59        r=53%        s=3.25
31.4
g=1.53        r=47%        s=2.95
32.3
g=1.29        r=35%        s=2.87
35.2
g=1.49        r=38%        s=3.20
38.3
g=1.93        r=50%        s=3.55
36.6
g=1.80        r=39%        s=2.93
38.4
g=1.63        r=31%        s=2.77
39.9
g=1.46        r=24%        s=2.98
40.5
g=1.54        r=29%        s=2.65
39.4
g=1.41        r=22%        s=1.74
```

```
Terminal
39.0
g=1.00        r=11%        s=1.18
40.5
g=0.77        r=6%        s=1.27
42.1
g=1.00        r=15%        s=1.12
38.7
g=0.74        r=1%        s=1.07
37.5
g=0.53        r=-6%        s=1.39        a switch occurs
38.1
g=0.53        r=-6%        s=1.43
36.5
g=0.53        r=-7%        s=1.74
35.4
g=0.53        r=-9%        s=2.13
STOP
Global tendency switched 5 times
5 weirdest values are [26.7, 24.0, 21.6, 36.5, 42.1]
```

THERE IS NOT ONE UNIQUE METHOD to define a trend or to forecast. As a consequence, the switches and abberations detection may vary from one program to another.
Keep calm and carry on coding: your program will be proved with **reasonable tests**, and the switches detection will only be evaluated to help you improve for the main project. They won't be take into account for any grade.

You should find, in a more convenient format, the input/ouptut of the example above along with this subject.

Bonus

You could implement a lot of extra features that you may find relevant (and will be very useful for the trade algorithm anyway):

- a graphical interface with charts, indicators, tendencies, …
- a forecast for the next X days
- an evaluation of the difference between your forecast and the actual result
- different metrics for detecting switches and/or outliers
- a benchmark of various methods

Analysis relies on observation, plotting the data should provide you better insights. Diagrams should be an easy extra-feature that will be a (huge) advantage for your understanding!