# The Wall

## Story

*- "The White Walkers sleep beneath the ice for thousands of years. And when they wake up..."*
*- "And when they wake up... what?"*
*- "I hope the Wall is high enough."*
*The Wall is a colossal fortification which is being built to stretch for 100 leagues (300 miles) along the northern border of the Seven Kingdoms. Its purpose is to defend the realm from the wildlings who live beyond. The Wall is reported to be 30 foot tall and is made of solid ice. The Sworn Brothers of the Night's Watch have arranged that each section has its own construction crew.*

## Task

### Description

Write a program that keeps track of material quantities and cost for the construction of the 30-foot wall.

You will be given a series of numbers. These numbers represent the initial heights of different mile-long sections of the wall(Wall profile), in feet. Each of these sections has its own construction crew that can add 1 foot of height per day. All crews work simultaneously (see examples), meaning all sections that aren't completed (are less than **30 feet** high) grow by 1 foot every day. When a section of the wall is completed, its crew is relieved. Each foot added uses **195 cubic yards of ice**. To process one cubic yard, it costs the Night's Watch **1900 "Gold Dragon" coins** for salaries and food for the brothers who work on it.

Your program needs to expose a rest API using Django and Django Rest Framework that allows the managers of the process to keep track of how much ice is used daily until the completion of the entire wall profile.

At the end, your program needs to expose two API endpoints one for checking the amount of ice used during a particular day (given as a parameter) for a particular wall profile and second that gives the overview of how much does the construction cost until the given day per profile and overall.

## Input

The input is a config file containing the wall profiles with one line for each profile, containing numbers, separated by space.

## Output

Daily API: Print the amount of ice used on a given day for a profile.
Overview API: Shows the final cost of the wall. Or if given a parameter shows the cost per wall profile.

## Constraints

The wall profile may contain up to 2000 sections
Starting height for each section is within range [0...30]

## Multi-threaded version

Instead of automatically having a team for each section of the wall profile take as an input also the number of available teams. Each team can work only one wall section in one profile. The APIs should return the same results but each team that finishes a section instead of being relieved will move to the next section and the next wall profile if this is completed.

Implement the dynamic of the teams with a Pool of parallel processes / threads that process one section every day and feed their job queue with the next section if available. Each process/thread should write into a log file the section that it has completed on the concrete day or “relieved” if there are no jobs.

Scroll down to see detailed examples.

## Example

### Input

21 25 28
17
17 22 17 19 17

Explanation:
On the first day, all crews work simultaneously, each adding 1 foot to
their section:
First profile:  3 crews  x 195 = 585 cubic yards
Second profile: 1 crew x 195 = 195 cubic yards
Third profile: 5 crews x 195 = 975 cubic yards
In total: 1 755  cubic yards

On the second day, it's the same. However, the last section of the first wall profile reaches 30 feet and its crew is being relieved.

On the third day, only two crews work from the first wall profile, using up 390 cubic yards in total.

Etc.

Output:

```
GET /profiles/1/days/1/
RETURNS:  {
                day: "1";
                ice_amount: "585"
          }

GET /profiles/1/overview/1/
RETURNS:  {
                day: "1";
                cost: "1,111,500"
          }
```

```
GET /profiles/overview/1/
RETURNS:  {
                day: ”1”;
                cost: “3,334,500”
          }

GET /profiles/overview/
RETURNS:  {
                day: None;
                cost: “32,233,500”
          }
```