# C++ Programming: Judge Assignment 2 (JA2)

The following tasks should be submitted to the SoftUni Judge system, which will be open starting Saturday, 22 April 2017, 10:00 (in the morning) and will close on Sunday, 30 April 2017, 23:59. You will be provided with a link to the “contest” (where you will submit the assignment) later.

Solutions for each task will be submitted in the form of compressed archive (.zip) files, containing .h and .cpp files. Depending on the task, some .h and/or .cpp files will be available in the Judge system and your code will be compiled alongside them (so that either your files can use them, or the other files will use the files you submitted). The files available for a task we will call a “solution skeleton”. Detailed instructions on what solution skeletons you are provided with, as well as instructions for submitting your code, are given in each task.

Please be mindful of the strict input and output requirements for each task, as well as any additional requirements on running time, used memory, etc., as the tasks are evaluated automatically and not following the requirements strictly may result in your program’s output being evaluated as incorrect, even if the program’s logic is mostly correct.

You can use C++03 and C++11 features in your code.

Unless explicitly stated, any integer input fits into int and any floating-point input can be stored in double.

NOTE: the tasks here are NOT ordered by difficulty level.

**NOTE: memory and time restriction for these tasks may be altered slightly up to Friday 21 April 23:59 (we’re still testing out the feature for multiple file submission for C++ projects). Please check the task descriptions in the Judge system (when it opens on 22 April, 10:00) for the final memory and time restrictions.**

## Task 3 – Populations (JA2-Task-3-Populations)

The food company Buck Fitches Met Goney (*you probably read that wrong*) has developed a new GMO crop that has the effect of raising the fertility of the human population in a city where it is distributed by a factor within a certain range… Ok, I’ll explain slowly. I-F---P-E-O-P-L-E---E-A-T---T-H-E---G-M-O---T-H-E-Y---S-T-A-R-T---M-A-K-I-N-G---M-O-R-E---B-A-B-I-E-S (*you read that slowly, didn’t you?)*. Since you’re solving a homework problem for a C++ course, I’m going to assume you’ve never had… intercourse… before, and I’ll give you a formula:

If a city’s population is P0, then the following equation P0 \* L <= P1 <= P1 \* H is true, where:

* L and H are two integer numbers such that 2 <= L < H <= 180, representing the expected low and high factors of increase in the population
* P1 is the city’s population after approximately 1 year
* We will call the range [P0\*L, P0\*H] (within which P1 falls) the city’s growth range

To get the most profits out of their new GMO crop, the company wants to demonstrate it by increasing the populations of some cities in the world to match other larger cities. But the L and R values (the low and high factor range) could be very far apart, so the company wants to be sure that for any city they decide to increase, there are a **minimum number of larger cities** which **fall within the range** for the grown population after a year. That way BFMG can give their GMO to a city and then after a year at least one of the larger cities within the range will be close to the target city’s population – which will be good publicity for BFMG, because then they can say that their method works and is very predictable in scope. Ok, again with the formulas.

We will say that a city’s C’s population PC is viable for L and H, if N >= M is true, where:

* M is an integer number and M > PC
* N is the number of cities, which are within P’s growth range(before giving the GMO)
* i.e. PC \* L <= Pi <= PC \* H, where Ni is the ith city’s population

Simply said, a **city** is a **viable target** if it has a **minimum number** of **other cities** in its growth range. Meaning that if we give that city the GMO, it’s population is expected to reach the population of one of a minimum number of other cities.

For example, if we have the cities A: 10000 people, B: 20000 people, C: 30000 people, D: 40000 people, and **E: 60000** people, and if we have L = 2 and H = 3, and we want a minimum number of cities in growth range M = 2, then:

* The **growth range** for A is [10000 \* 2, 10000 \* 3], which is [20000, 30000], which includes the populations of **B** and **C**, a total of 2 >= M, so A is a **viable** target.
* For B the range is [40000, 60000], it includes D and E, so it is also a **viable** target
* For C the range is [60000, 90000] – it only includes E, i.e. 1 < M, so C is NOT a viable target
* D and E analogously are also NOT viable targets (they have 0 other cities in their range)

The total number of viable targets in this example is 2 (the cities B and A).

BFMG has provided you with a database of populations of cities they are considering as targets – see the file populations.txt.

Write a program which determines the **number** of viable targets, from the list of cities in populations.txt, when provided L, H**,** and M.

Hint: notice the low time limit for your program’s execution. You probably can’t afford to spend time reading from a file during execution.

Hint: you probably don’t need all the info in populations.txt, only some of it.

Hint: this is a non-standard task, so you will probably need to do some non-typical things.

### Input

A single line, containing the numbers L, H, and M, separated by single spaces.

### Output

A single line, containing the total number of viable cities of those stored in populations.txt, based on the L, H and M values read from the console.

### Restrictions

1 < L < H < 181; 0 < M < 501

There are exactly 6342 cities described in populations.txt. The highest population value in the file is 22315474, the lowest is 66175. There are NO duplicate population values in the file.

The total running time of your program should be no more than 0.15s

The total memory allowed for use by your program is 20MB

### Submission Instructions

Submit a .zip file containing any files (.h and .cpp**)** your program needs to be compiled. The Judge system will NOT overwrite or add any files when compiling your solution.

### Credits

Data in populations.txt has been obtained from GeoNames (<http://www.geonames.org/>) and is distributed by the Creative Commons license. The data was extracted and edited to meet the needs of the task. Link to the original file: <http://download.geonames.org/export/dump/cities15000.zip>

### Example I/O

|  |  |  |
| --- | --- | --- |
| Example Input | Expected Output | Explanation |
| 2 180 1 | 6335 | There are 6342 cities in populations.txt.  The highest city population is 22315474. It will obviously not have any city in its growth range.  There are 6 more city populations, which when multiplied by 2 give a value larger than the highest city population – so they will not have anything in their range either (the lowest of these 6 cities has a population of 11174257, multiplied by 2 that’s 22348514 > 22315474).  The H multiplier is high enough that no city in the list will be restricted by it.  That leaves us with the 1 + 6 = 7 largest cities, which are not viable targets. The total number of cities is 6342, so subtracting 7 from that we get 6335 viable targets. |