Assignment 4

You are given a list of numbers:

```
[28, 22, 7, 2, 8, 14, 24, 56]
```

Your task is to find the longest "conglomerate" of numbers from this list, such that the conglomerate satisfies the "divisibility requirement." You may assume the list contains integers, no negative values, and no duplicates.

The Divisibility Requirement
Select any two numbers from the conglomerate, `L` and `S`. For those two numbers one of them will be larger, and the other will be smaller. Let's say `L` is larger. The divisibility requirement states `L % S == 0`.

More concisely, the divisibility requirement holds for a conglomerate iff…
For ***all*** pairs `L` and `S` in the conglomerate, `L % S = 0` where `L > S`

For example, in the above list there is a conglomerate `{14, 2, 7}`. This conglomerate does not satisfy the divisibility requirement because `7 % 2 != 0`.

Another example from the list above is the conglomerate `{8, 2, 24}`. This conglomerate satisfies the divisibility requirement because *all* the pairings demonstrate perfect divisibility: `8 % 2 == 0`, `24 % 8 == 0`, and `24 % 2 == 0`.

However `{8, 2, 24}` is *not the biggest* conglomerate that can be found that satisfies the divisibility requirement. That would be: `{7, 14, 28, 56}`. Actually, there are multiple possible answers for this list. Another would be: `{2, 14, 28, 56}`.

Code Details
Write a program that finds the biggest conglomerate for a given list of numbers. Of course the conglomerate must satisfy the divisibility requirement. You have been given a file `main.cpp` and the `bdc.h` header file. You should write a `biggest-divisible-conglomerate.cpp` file which interfaces with both of them.

The `biggest-divisible-conglomerate.cpp` file should implement the methods given in the `bdc.h` header file and should be run via `main.cpp`.

Finally, you should write a `Makefile` to compile the program to a binary called `bdc`.

Note #1: You may write a `test()` function, or you might want to hard-code some more test list(s) into the main method. In either case, know that your program will be tested (graded) using a variety of input lists.

Note #2: In general your program only needs to output one solution for a given input, even when there are multiple solutions possible. The solution is the conglomerate itself (not the size, or the number repetitions to find the answer, etc.)

Note #3: Your program should work if the input list is empty (it should return empty list) and also if the input list contains only one number (it should return that same list with one number).

Note #4: If the input has multiple items, but among those items there are no pairs that satisfy the divisibility requirement, you can return the largest or the smallest item in the list as the answer. For example if the `input: [7, 11, 17, 3]` then `output: [17]` or `output: [3]` are both acceptable.

Pro-Tip: Using recursion to solve this problem is a good strategy. You don't have to implement dynamic programming until part 2 of this assignment!

Pro-Tip: Come ask for me for help during office hours for a hint!

Sample Output

```
user@machine:solution$ ./bdc
input: [28, 22, 7, 2, 8, 14, 24, 56]
output: [56, 14, 7, 28]

input: [10, 5, 3, 15, 20]
output: [20, 5, 10]
```

Proper Comments and Citations

Your code should have comments that explain "why" the code is written the way it is. You should also use comments to cite sources you used such as websites, tutors, and classmates.

Submission: Your program will be submitted using GitHub classroom.

Following the instructions on canvas you should have a repository cloned to your computer which contains these assignment instructions. That same repository will be used for you to submit your solutions.

1) Edit and write code in the repo on your computer. Use `git add`, `git commit`, and `git push` to put that code onto github. I have access to the repo. You can make as many commits as you like for this assignment with no penalty.

2) I will grade the final commit made before the due-date.

3) You don't have to make any submission on canvas.