Summer 2021, CS3130, Dr. He

Project 2B. Calculate the Greatest Common Divisor Between Two Large Integers

(Optional, Due 8/7/2021 Saturday)

Description:

In this project, we will calculate the *greatest common divisor* (GCD) between two large integers. We will use the *Euclid's Algorithm* we studied in this class. Since the implementation is pretty straightforward, we will do the computation on two large integers with `100` digits. Thus, we have to find a good way to handle the *overflow* problem.

Requirements:

1. (*Avoid Overflow*)

   Since the largest primitive integer data type, (e.g. in Java, the `long` data type), can suffer the *overflow* problem if we work on *sufficiently large* integers, we have to find a way to get around this problem. To overcome this *overflow* problem, we need to go to a much larger data type in this situation. In Java, we can use the `BigInteger` class to do the calculation. A Java example using the `BigInteger` class is provided to show you how to do basic mathematical operations on the `BigInteger` operands. While in Python, you do not need to worry about this problem, you just do your usual mathematical operations.

   I do not have the information on the way to solve this problem in C/C++. If you cannot find a good way in C/C++, try to use Java or Python in this project.

2. (*Generate Two Random Integers As Input*)

   Use a random integer generator to create two input integers with `100` digits. In the given Java example, you can see this part already.

3. (*Euclid's Algorithm*)

   The *Euclid's Algorithm* is the best way to find the GCD, and its implementation is very easy. But here, we just need to use the `BigInteger` operations instead of our usual mathematical operations.

4. (*Display Output*)

   When you display your GCD result, you also need to display the intermediate problem reduction steps, so that I can see your algorithm implementation details. You can get an idea on the problem size reduction from the example in our notes. After you display the GCD result, also display the response time in milliseconds (or nanoseconds if it is too small).

5. (*Work independently*)

   Everyone should work on the project independently. When I do grading, if I see similar behaviors, I will compare those programs line-by-line. Although you are allowed to discuss with your classmates about the ideas, the final code should be written by yourself.

Any updates about this project will be posted in Canvas as announcements.