Goals:

- Practice getting around the command line compiling and running Java programs.
- Practice getting around in and using GitHub.
- Explain some key concepts we covered in class.
- Get some lab points.

Part One. Description:

In cryptography we often employ the Euclidean algorithm to perform certain calculations. The algorithm is designed to find the greatest common divisor between two integers \( a, b > 0 \), i.e. \( \gcd(a, b) \). The algorithm assumes that \( a \geq b \), otherwise their values are switched. Then, it applies the division algorithm to find \( a = q_1 b + r_1 \). If the remainder \( r_1 = 0 \) then the algorithm stops, meaning that the greatest common divisor is \( b \). Otherwise, the algorithms continues dividing until the remainder is zero, as follows:

\[
\begin{align*}
  b &= q_2 r_1 + r_2 \\
  r_1 &= q_3 r_2 + r_3 \\
  r_2 &= q_4 r_3 + r_4 \\
  \ldots \\
  r_{n-2} &= q_n r_{n-1} + r_n \\
  r_{n-1} &= q_{n+1} r_n + 0
\end{align*}
\]

Once the algorithm terminates, the remainder \( r_n \) is the greatest common divider, i.e. \( r_n = \gcd(a, b) \). Your task is to implement this algorithm.

Input:

The input has \( L \) lines; each line has two integers, \( a, b > 0 \) separated by a single space.

Output:

The output is composed of \( L \) lines, each with an integer corresponding to \( \gcd(a, b) > 0 \).
Sample Input 1:

```
100  74
```

Sample Output 1:

```
4
```

Sample Input 2:

```
1071  462
148  75
```

Sample Output 2:

```
21
1
```

**Requirements:**

Create a method with the following signature `long euclidAlg(long a, long b)` that receives a pair of positive integers, and returns the corresponding greatest common divisor as an integer. Note that for multiple lines of input you can call this function repeatedly. Also, make sure you follow the style guidelines [http://www.reev.us/mscs630s17/style.html](http://www.reev.us/mscs630s17/style.html) that were given for this course.
**Part Two. Description:**

The Euclidean algorithm is widely used in cryptography to perform a number of different calculations. The original algorithm can be extended to be more efficient and to keep record of values. The extended algorithm still finds the greatest common divisor between two integers \( a, b > 0 \), i.e. \( \gcd(a, b) \), under the assumption that \( a \leq b \), otherwise their values are switched. Then, it computes the values for \( d, x, \) and \( y \) that satisfy the equation:

\[
d = ax + by
\]  

(1)

where \( d = \gcd(a, b) \). Your task is to implement this extended algorithm efficiently.

**Input:**

The input has \( L \) lines; each line has two integers, \( a, b > 0 \) separated by a single space.

**Output:**

The output is composed of \( L \) lines, each with three integers separated by a space corresponding to \( d = \gcd(a, b), x, \) and \( y \) that satisfy Eq (1).

Sample Input 1:

148 75

Sample Output 1:

1 37 -73

Sample Input 2:

1155 862
240 46

Sample Output 2:

1 203 -272
2 -9 47

**Requirements:**

Create a method with the following signature `long[] euclidAlgExt(long a, long b)` that receives a pair of positive integers, and returns an array long integers of length three, corresponding to \( d, x, \) and \( y \) in Eq (1). Note that for multiple lines of input you can call this function repeatedly. Also, make sure you follow the style guidelines [http://www.reev.us/mscs630s17/style.html](http://www.reev.us/mscs630s17/style.html) that were given for this course.
**Resources:**

- Your textbook (Stanoyevitch)!
- Project submission guidelines for this course: [www.reev.us/mscs630s17/project_submission.html](http://www.reev.us/mscs630s17/project_submission.html)
- Coding style guidelines for this course: [www.reev.us/mscs630s17/style.html](http://www.reev.us/mscs630s17/style.html)
- “How to” use the command line “shell”: [www.reev.us/mscs630s17/shell.html](http://www.reev.us/mscs630s17/shell.html)
- The official Java reference: [http://docs.oracle.com/javase/tutorial/collections/TOC.html](http://docs.oracle.com/javase/tutorial/collections/TOC.html)
- Stack Overflow Java Tag: [http://stackoverflow.com/questions/tagged/java](http://stackoverflow.com/questions/tagged/java)

---

**Submission:**

- Push your work to your GitHub repository before the due date (see the top of this document). Remember to include your name, the date, and the assignment in the (copious, meaningful, and accurate) commit messages. Then **double check your files are on GitHub** and that your professor has been added as a collaborator (you hopefully did this in Homework 0), his GitHub username is **pablorp80**.

- Make sure that your program (*.java) is in a folder called labs/1/ inside your repository folder. Your repository folder (say **mscs630lastname**) should look something like this:

```
mscs630lastname/
    labs/ ... This directory will have all your labs, organized as one directory per lab, where you will have any .java files as required.
    1/ ... Files for this specific lab would go in here: i.e., your .java file and test cases.
    2/ ...
    hw/
    prj/
```