CMSC B113 Computer Science 1 – Spring 2025
Lab Activity #3

Overview

In this lab activity, you will practice working with **`while`**-loops.

For this and other Lab Activities, you are encouraged to collaborate with your classmates, especially when it comes to addressing any issues that come up, error messages that you receive, etc.

However, it is expected that you will work *together* and not merely delegate the tasks among you, or simply copy/paste each other's solutions. This is an opportunity for you to get a firmer grasp on the material before you try it on your own in the upcoming Programming Assignment.

Part 1: Reverse an Integer

Write a program called **`Reverse.java`** that takes a positive integer **`n`** as a runtime argument and then prints its digits in reverse order, using the algorithm described below.

For instance

- **`java-introcs Reverse 1234`** should print `4321`
- **`java-introcs Reverse 19103`** should print `30191`

Once your program has read **`n`** as a command line argument, it should use the following algorithm to reverse the digits and store the value in the variable **`r`**:

```
initialize r to 0
while n != 0
     set d to the last digit of n
     multiply r by 10 and add d
     divide n by 10
```

When this algorithm terminates (i.e. when the **`while`**-loop is done), you can print the variable **`r`** to see the digits in reverse.

The trick here is figuring out how to set **`d`** to the last digit of **`n`**. Think about the different arithmetic operators we've seen in class this semester. Discuss with the person sitting next to you. Or, ask your instructor if you are not sure.

Part 2: Greatest Common Divisor

The Greatest Common Divisor, or GCD, of two numbers **a** and **b** is defined as the largest number that divides both of them without leaving any remainder. For example:

- GCD of 15 and 25 is 5
- GCD of 18 and 12 is 6
- GCD of 33 and 28 is 1
- GCD of 210 and 45 is 15

Euclid, who authored *Elements* (circa 300 BC), devised an algorithm to compute the GCD of two numbers. It is regarded as the oldest known algorithm!

Here is Euclid's algorithm for computing the GCD of two numbers **a** and **b**:

```
while (a ≠ b)
   if a > b
      set a to a – b
   else
      set b to b - a
```

When the loop terminates, both **a** and **b** will contain the GCD. Write a Java program called **GCD.java** that takes two numbers **a** and **b** as command line arguments and uses the above algorithm to compute and display their GCD.

For instance:

```
$ java-introcs GCD 15 25
The GCD of 15 and 25 is 5

$ java-introcs GCD 33 28
The GCD of 33 and 28 is 1
```

Once done, try out the other number pairs above to confirm your answers.

Then, once you are done implementing the algorithm, modify the program so that it displays "**`Invalid input`**" and exits if either **a** or **b** is 0 or negative. Note that the code should do this *before* executing Euclid's algorithm, since you cannot calculate the GCD in such cases.