Homework 1, CSCE 350, Spring 2016

Objectives

This is a simple exercise to get you into the process of thinking about data structures and algorithms and how to gauge what's going on.

You will recall the subtract-and-shift version of the GCD algorithm. You will recall that we did ONE instance of a worst case (all 1s and 1), but that it's not trivial to figure out all the worst case examples.

So let's punt on doing the math. Sometimes it's not necessary to be able to prove what is true, as long as we know what it is that is in fact true.

In other words, let's compute.

First Step

The first step is that you will have to write the code to compute GCDs by the subtract and shift algorithm.

```
Given nonnegative integers a and b
Remove the k_a powers of 2 in a to produce a_odd
Remove the k_b powers of 2 in b to produce b_odd

Now that we have two odd integers a_odd and b_odd,
we know that the gcd is odd

Flip a_odd and b_odd if necessary so that
a_odd = big
b_odd = small
and big >= small

while small != 0
    big = big - small
    shift big right one bit
    while big is even
        shift big right one bit
    exchange big and small if necessary so that big > small

the odd part of the gcd is big

Put back min(k_a, k_b) powers of 2
Return the gcd
```

Second Step

Now, the key issue is to find out what the worst case situations are, at least computationally if not mathematically.

Write the function to return the bit length of the largest integer permitted in the random integers.

Instrument your code to count frequencies of occurrence of the bit lengths.

Instrument your code to count frequencies of occurrence of the number of shifts (after the first free shift) divided by the max bit length of the operands.

That is, if the random numbers for which we compute the gcd are 255 (bitlength 8) and 37 (bitlength 5), and we count a total of 7 shifts beyond the guaranteed shift in doing the algorithm, then we have $7/5 = 1.40$ as the fraction of bit length for the extra shifts.

Rules

You will need to count the number of values of each bit length that are generated by the random number generator. Note that this is twice the number of tests, because you need two numbers for each test.

I have not seen any shift counts bigger than 1.5 times the bit length, so your count of fractional hundredths can go up to 2.0 and probably you'll be ok. If it breaks with this, then you have found a truly pathological example, and that's a good thing.

Random Numbers

I have provided random number genrators.