

Data Structures and Algorithms Homework 1

Due Wednesday Sept 4; Joseph Sepich (jps6444)

1 Problem 1

I understand the course policies.

2 Problem 2

2.1 Part a. Prove that if a and b are even, then $\gcd(a, b) = 2\gcd(a/2, b/2)$

1. By the definition of even we can state that $a = 2n$ and $b = 2m$, where n and m are both integers.
2. Using the definition in step 1 we can write $\gcd(a, b) = \gcd(2n, 2m)$.
3. Since 2 is a common divisor we can also write $\gcd(2n, 2m) = 2\gcd(n, m)$.
4. Using step 1, we also know that $n = \frac{a}{2}$ and $m = \frac{b}{2}$.
5. Plugging step 4 into the equalities in step 2 and 3 we get $\gcd(a, b) = \gcd(2n, 2m) = 2\gcd(n, m) = 2\gcd(\frac{a}{2}, \frac{b}{2})$.

Therefore if a and b are both even, $\gcd(a, b) = 2\gcd(\frac{a}{2}, \frac{b}{2})$.

2.2 Part b. Prove that if a is even and b is odd, then $\gcd(a, b) = \gcd(a/2, b)$

1. By the definition of even we can state that $a = 2n$, where n is an integer.
2. Using the definition in step 1 we can write $\gcd(a, b) = \gcd(2n, b)$.
3. Since b is odd, it cannot be divided by 2, so the 2 in the term $2n$ is unnecessary information (cannot contribute to the gcd). We can then write $\gcd(a, b) = \gcd(n, b)$.
4. Using step 1, we also know that $n = \frac{a}{2}$.
5. Plugging step 4 into the equalities in step 2 and 3 we get $\gcd(a, b) = \gcd(2n, b) = \gcd(n, b) = \gcd(\frac{a}{2}, b)$.

Therefore if a is even and b is odd, then $\gcd(a, b) = \gcd(\frac{a}{2}, b)$.

2.3 Part c. Prove that if a and b are both odd and $a \geq b$, then $\gcd(a, b) = \gcd((a-b)/2, b)$

1. By the definition of odd we can state that $a = 2n + 1$ and $b = 2m + 1$, where n and m are both integers.
2. $a - b = (2n + 1) - (2m + 1) = 2n - 2m + 1 - 1 = 2(n-m)$. This must be greater than 0, since $a \geq b$.
3. $a - b$ is therefore by definition an even number since $2(n - m)$ can be written as $2x = (a-b)$ where x is the integer $n - m$.
4. Definition of gcd means that $d \mid a$ and $d \mid b$, where d is an integer. This means $a = d * z$ and $b = d * y$ where z and y are integers.
5. Through step 4 $b - a = d(z - y)$, so d must also divide the integer $(z - y)$. This means $\gcd(a, b) = \gcd(a-b, b)$.
6. Since we determined in step 3 $a-b$ is even (and b is even) and in step 5 $\gcd(a, b) = \gcd(a-b, b)$, then through the proof in part b of the problem we can conclude that $\gcd(a, b) = \gcd(\frac{a-b}{2}, b)$.

2.4 Part d.

We know that by the definition of gcd

```
int gcd(int a, int b) {
    // input a >= b
    int d = 1;
    if (a == b) return a;
    bool a_is_even = isEven(a); // test parity of variables
    bool b_is_even = isEven(b); // unit time
    while (a > 0 && b > 0) {
        if (a_is_even && b_is_even) { // 2 is not a common divisor (part b)
            a = a / 2;
        } else if (a_is_even && !b_is_even) { // 2 is not a common divisor (part b)
            b = b / 2;
        } else if (!a_is_even && b_is_even) { // part c
            a = (a - b) / 2; // our input requires a > b
        } else { // both a and b are even, so both can be divided by 2 (as in part a)
            a = a / 2;
            b = b / 2;
            d += 1;
        }
    }
    // d is how many times divided by 2
    // a is non even part of gcd
    return a * 2^d;
}
```

Now let us assess running time. We are assuming testing parity and halving are in unit time, so let's focus on subtraction. As we can recall a positive integer a has at most $\log(a)$ bits, and here a is the larger number. Subtraction two n bit integer takes $O(n)$ time, so here it would take us $O(\log(a))$ time. Therefore the algorithm meets the requirements of the problem.

3 Problem 3

- a) $f(n) = \Omega(g(n))$, this is true because we know with polynomials, the higher degree will always grow faster.
- b) $f(n) = \Theta(g(n))$, this is true because $2^{n-1} = 2 * 2^{n-2}$ and we know that coefficients do not make a difference in larger values of n .
- c) $f(n) = \Omega(g(n))$, this is true because $f(n)$ has an exponent which grows, while $g(n)$ has a constant exponent, therefore $f(n)$ must grow faster.
- d) $f(n) = \Omega(g(n))$, this is true because in $g(n)$ the 2^n term is dominant. While 2^n and 3^n both have the same exponent term of n , the integer that has the exponent is greater in $f(n)$.
- e) $f(n) = \Theta(g(n))$, this is true because you can transform the exponent on each into a coefficient, due to properties of logarithms, then change the c you put in front of $g(n)$ to obtain an identical function.
- f) $f(n) = \Omega(g(n))$, this is true because $f(n)$ is growing a constant rate, while $g(n)$ is growing at less than a constant rate.
- g) $f(n) = O(g(n))$, this is true because for each additional n , $f(n)$ is multiplied by 2, but $g(n)$ is multiplied by n , so it must be growing faster.
- h) $f(n) = O(g(n))$, this is true because by the logarithm properties you are comparing $n \log(e)$ and $n \log(n)$, and since we don't look at coefficients for growth rates you are comparing n and $n \log(n)$, so while $f(n)$ grows at a constant rate, $g(n)$ is a function that increases at an increasing rate.
- i) $f(n) = \Theta(g(n))$, this is true because n in each equation is the dominating term. This makes both equations $\Theta(n)$, since $\log(n)$ grows more slowly than n .
- j) $f(n) = \Theta(g(n))$, this is true because of the same concept in the last part. n grows faster than both $\log(n)$ and $n^{0.5}$. If you chose c to be 5, then they would be identical equations.

4 Problem 4

4.1 Part a.