Lesson 1 Summary

Five problems

- gcd(n, m) log P
- sorting nlogn P
- subset sum exponential EXP
  - maybe someone will come up with a polynomial time algorithm. Then it will belong to class P.
  - It belongs to class NP. That is, given the solution, there is a polynomial time algorithm to verify it.
- n x n chess exponential EXP-complete
  - No polynomial time algorithm is possible.
  - It is not in class NP.
  - It will never belong to P.
- Halting problem No algorithmic solution.

gcd (greatest common divisor)

divisor

divisors of 12: 1, 2, 3, 4, 6, 12.

divisors of 30: 1, 2, 3, 5, 6, 10, 15, 30.

common divisors

1, 2, 3, 6.

greatest common divisor

6

gcd (greatest common divisor)

Algorithm gcdOne(int a, int b)

Step 1. Create a list L1 of all divisors of a.

Step 2. Create a list L2 of all divisors of b.

Step 3. Create a list L3 of common divisors of a and b.

Step 4. Pick the "greatest" from L3.

Imrovements ?

gcd (greatest common divisor)

30 % 12 = 6

12 % 6 = 0

gcd (greatest common divisor)

Algorithm gcd_Euclid(int a, int b)

Step 1. c <- a % b

Step 2. a <- b; b <- c

Step 3. Repeat steps 1 and 2 until b is 0.

Step 4. return a.

Optional - 1

1. (*GCD Problem*) Given two positive integers m, n, is there a positive integer d that is a factor of both m and n and that is bigger than or equal to every integer d' that is also a factor of m and n?

```
static int gcd(int a, int b)
{
        if(b == 0)
        {
                return a;
        }
        return gcd(b, a % b);
}
```

1. Given two integers, can you show (prove) the algorithm will halt (end) ?
2. If a = 3, which value of b less than 3 will result in maximum number of recursive calls?
3. If a =5, which value of b less than 5 will result in maximum number of recursive calls?
4. If a =8, which value of b less than 8 will result in maximum number of recursive calls?
5. What is 1, 1, 2, 3, 5, 8, …?

Optional - 2

1. Given two integers, can you show (prove) the algorithm will halt (end) ?
2. If a = 3, which value of b less than 3 will result in maximum number of recursive calls?
3. If a =5, which value of b less than 5 will result in maximum number of recursive calls?
4. If a =8, which value of b less than 8 will result in maximum number of recursive calls?
5. What is 1, 1, 2, 3, 5, 8, …?
6. Let $f(x) = x^2$. What is its inverse function?
7. Let g(x) = sqrt(x). Then f(g(x)) = g(f(x)) = x. Hence f and g are inverse of each other.
8. Let f(x) = exp(x). What is its inverse?
9. If Fibonacci has exponential growth, what can you say about the growth of its inverse?

Optional - 3

1. (*GCD Problem*) Given two positive integers m, n, is there a positive integer d that is a factor of both m and n and that is bigger than or equal to every integer d' that is also a factor of m and n?

```
static int gcd(int a, int b)
{
        if(b == 0)
        {
                return a;
        }
        return gcd(b, a % b);
}
```

gcd(21, 13) gcd(Fib(8), Fib(7))

gcd(13, 8) first recursive call

gcd(8, 5) second recursive call

gcd(5, 3) third recursive call

gcd(3, 2) fourth recursive call

gcd(2, 1) fifth recursive call

gcd(1, 0) sixth recursive call

There are 6 recursive calls. gcd(Fib(n), Fib(n-1)) will make n-2 recursive calls.

You must know - 1

3. The subset sum problem

S = {2, 5, 9}

{2}, {5}, {9},

{2, 5}, {2, 9}, {5, 9}

{2, 5, 9}

There are 7 nonempty subsets.

$7 = 8 - 1 = 2^3 - 1$.

Generalize this!

If S has n elements then there $2^n - 1$ nonempty subsets.

All known algorithms take exponential amount of steps. Hence subset sum problem belongs to EXP.

You must know

- What is a factor?
- What is gcd of two positive integers?
- What is P?
- What is NP?
- What is EXP?
- What is EXP-complete?
- What are the two conditions a problem must satisfy to belong to class EXP-complete
- Example of a problem in those classes.
- Learn to write an algorithm.
- Learn to write a nondeterministic algorithm.
- What is the Halting Problem?
- Why is it important?