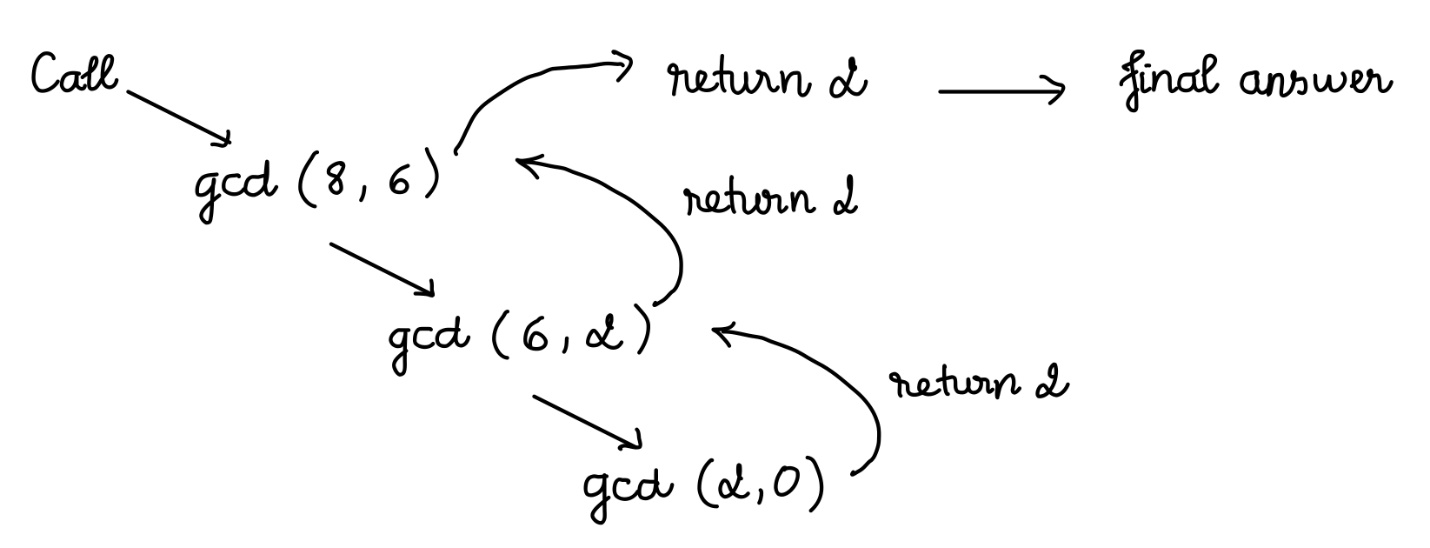
Thinh Le  
CS 272  
Lab 11

1. Recursive trace for gcd(8, 6)



1. Prove its correctness by induction over N (the number of recursive call):

* Base case: N = 0. In this case, it must be that b = 0. As a result, gcd(a, b) = gcd (a, 0) = a. Which is the correct answer.
* Inductive case: we assume that there are a, b such that gcd(a, b) makes N (N > 0) recursive calls. a = qb + r with q and r are 2 integers and 0 < r < b. Therefore, a%b = r.  
  gcd(a, b) = gcd(b, a%b) = gcd(b, r). So the algorithm returns the right value.

1. Analyze the time complexity of the Tower of Hanoi solution

The solution:

static void HanoiTower(int n, char pile1, char pile2, char pile3) {

if (n == 1) {

System.out.println("Move disk 1 from pile " + pile1 + " to pile " + pile2);

return;

}

HanoiTower(n - 1, pile1, pile3, pile2);

System.out.println("Move disk " + n + " from pile " + pile1 + " to pile " + pile2);

HanoiTower(n - 1, pile3, pile2, pile1);

}

Let the time required to move n disks be T(n).

There are 2 recursive calls for n – 1 disks and one constant time operation to move a disk from source pile to destination pile. It would be k1. So we have an equation of T(n):

T(n) = 2 T(n – 1) + k1

We can analyze:

T(1) = k1

T(2) = 2 T(1) + k1 = 2 k1 + k1

T(3) = 2 T(2) + k1 = 2(2 k1 + k1) + k1 = 4 k1 + 2 k1 + k1

T(4) = 2 T(3) + k1 = 8 k1 + 4 k1 + 2 k1 + k1

…

T(n) = ( 2n – 1 + 2n – 2 + … + 21 + 20 ) k1 = (2n – 1) k1

Therefore, the time complexity is O( 2n ).