Name: Ning Wei

Question1: Falling Glass

(a) Describe the optimal substructure/recurrence that would lead to a recursive solution

Optimal solution of problem is based on optimal solution of subproblems.

Drop 1 sheet from x floor:

if it breaks, go down to x-1 floor and sheets number -1.

If it does not break, go up, floors = total floor number - x, and sheets number does not change.

Base case:
1. When floors equal 1, only one trials.
2. If we only have 1 sheet, we have to traverse all the floors.

(b) Draw recurrence tree for given (floors = 4, sheets = 2)

(d) How many distinct subproblems do you end up with given 4 floors and 2 sheets?

Distinct subproblems: 4*2 = 8

(e) How many distinct subproblems for n floors and m sheets?

Distinct subproblems: m*n

(f) Describe how you would memoize GlassFallingRecur

1. Create an array to store a calculated item.
2. if memorized array's item already has been calculated, get the value from cache directly.
3. If item does not has been calculated, go to calculate it and store it in the array.

Falling Glass Output:
Question2: Rod cutting

RodCutting.java

RodCutting.java | GlassFalling.java

```java
/**
 * Rod cutting problem described in Chapter 15 of textbook
 */
public class RodCutting {

  // Do not change the parameters!
  public int rodCuttingRecur(int rodLength, int[] lengthPrices) {

    int[] check = new int[rodLength + 1];
    return rodCuttingRecur2(rodLength, lengthPrices, check);

  }
  public int rodCuttingRecur2(int rodLength, int[] lengthPrices, int check[]) {

    if (check[rodLength] != 0){
      return check[rodLength];
    }
    if(rodLength <= 0){
      return 0;
    }
    int maxRevenue = -1;
    for(int i = 0; i < rodLength; i++){
      maxRevenue = Math.max(maxRevenue, lengthPrices[i] + rodCuttingRecur2(rodLength - i -1, lengthPrice
    }
    check[rodLength] = maxRevenue;
    return maxRevenue;

  }

  // Do not change the parameters!
  public int rodCuttingBottomUp(int rodLength, int[] lengthPrices) {
    int[] max = new int[rodLength +1];
    max[0] = 0;
```

Task List

Connect Mylyn

Connect to your task

Outline

RodCutting

rodCuttingRecur

rodCuttingRecur

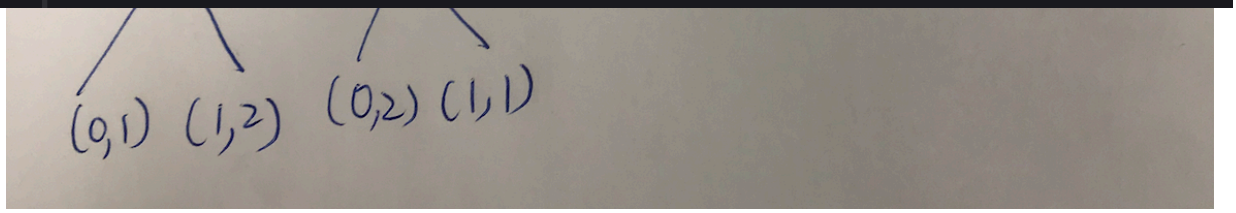
rodCuttingBotto

main(String[])

Problems | Javadoc | Declaration | Console

<terminated> RodCutting [Java Application] /Library/Java/JavaVirtualMachines/jdk1.8.0_144.jdk/Contents/Home/bin/java (Apr 2, 2019, 9:43:57 PM)

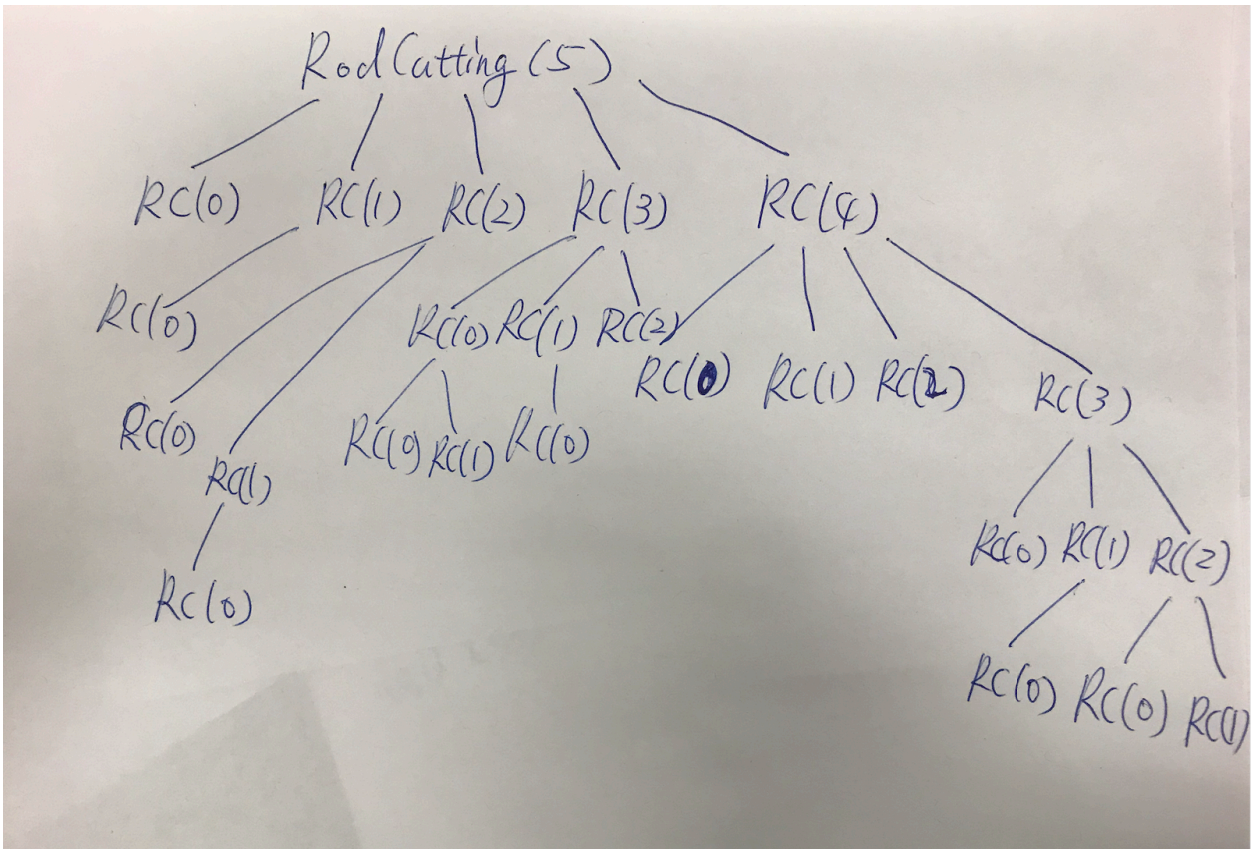
```
23 23
35 35
```



(a) Draw the recursion tree for a rod of length 5

Max value = 13

(b) On page 370: answer 15.1-2 by coming up with a counterexample, meaning come up with a situation / some input that shows we can only try all the options via dynamic programming instead



length : 1 2 3 4
price : 1 5 8 9
p/i : 1 2.5. 2.667 2.25

Maximum **density** p/i = 8/3=2.667 ,after first cut for length 3, so 1 left.
Total revenue is 9. But 5 + 5 =10 which is higher than 9 for length 2.