Question: https://leetcode.com/problems/coin-change/

if amount is 0 return 0

if amount<0 return -1

and if previously minimum number of coin for this amount is calculated return that.

make a choice at each step, to choose every coin from the coins array, and use recursion by making amount=amount-coins[i]

Now if that returns -1, then ignore it and select the minimum.

If no minimum is selected that means we have to return -1

We memoize the values to optimize it

Code:  
class Solution {

HashMap<Integer, Integer> mem = new HashMap<Integer, Integer>();

public int coinChange(int[] coins, int amount) {

if(amount == 0){

return 0;

}

if(amount<0){

return -1;

}

if(mem.containsKey(amount)){

return mem.get(amount);

}

int min =Integer.MAX\_VALUE;

for(int i=0; i<coins.length; i++){

int res=coinChange(coins, amount-coins[i]);

if(res!=-1){

min=Math.min(min,res);

}

}

if(min==Integer.MAX\_VALUE){

min = -2;

}

min++;

mem.put(amount, min);

return min;

}

}

Github Link :<https://lnkd.in/ecwtJeaz>