HW6

林宸昊 PB20000034

1. 重新定义r_n:

   $r_n = max\{p_n, r_1 + r_{n-1} - c, r_2 + r_{n-2} - c, ..., r_{n-1} + r_1 - c\}$

   然后将`MEMOIZED-CUT-ROD-AUX(p, n, r)`修改如下即可：

   ```
   6: for i = 1 to n - 1 do //最后没有切割成本故需要单独比较
   8: q = max{q - c, p[i]}
   9: r[n] = q
   ```

2. 设f(m, n)为m个苹果n个盘子的总放法，则由于只在乎组合不在乎放的次序，
   - n > m, 此时去掉n - m个必然空着的盘子对结果也不产生影响，即

     > f(m, n) = f(m, m)

   - n <= m, 此时有两种方案：

     > 所有盘子都放苹果，那么从所有盘子拿走一个是一样的：f(m, n) = f(m - n, n)
     >
     > 有一个空盘子，那么去掉这个空盘子是一样的：f(m, n) = f(m, n - 1)
     >
     > f(m, n) = f(m, n - 1), f(m - n, n)

   - n = 1 || m == 0，此时递归出口，返回1
3. 将给定点集从小到大排序，然后每一次考虑最小的点作为左边界，在单位长度内寻找下一个点，如果有，放入区间，如果没有，形成区间，以这个新的点重新作为左边界。因为每一步的选择都是必须包括且最小的点，也就是子问题的最优解，因此最终得到的也就是最优解。
4. 可以使用贪心方法。如果作业数小于机器数，那么直接分配即可；如果作业数大于机器数，那么可以先将作业所需时间从大到小排序，然后每一次从中选择所需时间最长的作业分配给当前离完成工作时间最短的机器，使得每台机器工作时间相近达到近似最优。
5. 每一次都选择可能的面额最大的硬币。下面针对本题有一个浅显的证明：
   - 只使用1美分，则最多表示4美分，否则可以使用1美分和5美分的组合；
   - 只使用1美分和5美分，则最多表示9美分，否则可以使用1美分，5美分和10美分的组合；
   - 只使用1美分，5美分和10美分，则最多表示24美分，否则可以使用1美分，5美分，10美分和25美分的组合；
   - 在这种条件下，任何面值的硬币都无法被面值比它小的硬币下位替代来达到更优，因此这种贪心得到的必然是最优解之一。
6.

0	0	0	0	0	0
0					
0					
0					
0					

构建这样的一张表格，横坐标是0-n，纵坐标是0-W，先进行如图所示的初始化，然后根据规则：

$$V[k,w] = \begin{cases} V[k-1,w] & \text{if } w_k > w \\ \max\{V[k-1,w], V[k-1.w-w_k]+b_k\} & \text{else} \end{cases}$$

更新表格。该规则的含义是对于总允许重量w的子问题S_k而言，

- 第k件商品重量大于总允许重量w，则不考虑该商品，直接继承；
- 不大于，则考虑加入与不加入第k件商品带来收益的较大值。

伪代码如下：

```
DP FOR KNAPSACK()
for w = 0 to W do
    V[0,w]=0
for i = 1 to n do
    V[i,0]=0
for i = 1 to n do
    for w = 0 to W do
        if w_i ≤ W then
            if b_i + V[i-1,w-w_i] > V[i-1,w] then
                V[i,w] = b_i + V[i-1,w-w_i]
        else
            V[i,w] = V[i-1,w]
```

由于需要遍历整个表格，时间复杂度为O(nW)