## 2.赛斯石(*Sise Problem*)

这个题目事实上是一个动态规划题目，只不过很容易想当然，有一些坑点。首先我们先了解一下方法。

方法很简单，每一艘船都得载满才不会造成浪费，毕竟是要最大盈利。然后求出每一艘船如果都载满后可得到的盈利。然后进行总体动态规划。

然而，坑点就在于，题目说过赛斯石是可以随便分割与合并的（只要保证为整数赛斯），那么在每艘船里面也可以进行再分割，你有没有考虑到呢？

那么此时我们就只需要讲一下如何用动态规划解这道题了。

按照动态规划的原则，将一个问题分成若干个子问题求解。我们从小处着眼，假设只有 1 赛斯赛斯石，那么情况就是一种，它本身。假设有 2 赛斯，那么可以不合并，也可以合并。我们假设 2 赛斯是一个整体，不分割就是它本身，我们之后就不讨论了，2 赛斯分割可以分割为 2 个 1 赛斯，这里尚且还不能体现动态规划的思想。那么我们假设有 5 赛斯，分割可以分割 1 到 4，我们看到当它分为 1 和 4 的时候，我们是不是要直接调用 1 和 4 的价格然后加起来呢？显然不是，因为我们不知道 4 往下分还有没有更大盈利，难道还要对 4 分下去吗？这也不是的，这样会造成重复考虑，你想吧，后面还要考虑 2 和 3 呢，4 往下分也会分到 3 来，那么 3 就重复考虑了。当总量特别大的时候，重复考虑就会更多，导致时间的浪费。

那么我们可以对考虑过的东西**进行储存**。

我们定义一个数组 solution[$n$]，每个 solution[$n$]都储存了 $n$ 赛斯赛斯石的时候的最大盈利，因此按照上述例子，5 分为 1 和 4 的时候，对 1 直接调用价格，对 4 调用 solution 数组，这样做是为了**防止重复**，那么我们就需要对其循环 5 次，每次递增 1，即继 5 分为 1 和 4 之后就是分为 2 和 3，然后是 3 和 2，直到 5 和 0，达到**不遗漏**的效果，找到其中最大的盈利。

那么，首先在确定每艘船满载时可得到的盈利进行一次小的动态规划，确定十个数据之后，在进行一次大的动态规划，得到总盈利，答案即可得。

转移方程为：**solution[$n$]=price[$k$]+solution[$n$-$k$]　($1\le k\le n$)**

值得一提的是，在大的动态规划的时候，最大合并长度不能超过 10 赛斯，否则就没有船可以租了。

比赛时有一个疑问提得好，就是商家可以一开始切得小，上岸后再合并。这是我的疏漏，后来更新为商家上岸不能再对赛斯石进行操作，此处对大家表示抱歉。

具体实现请见标程 SiseAnswer.cpp。

本题目改编自经典题：钢条切割问题。