# C3A 进厂切钢条

## 题目描述

互联网寒冬袭来，Zhoues 打算提前进厂，体验一下流水线的感觉是什么样的。

Zhoues 给一家钢条厂发送了一份自己的简历，希望可以寒假在这里实习。 厂里负责招聘的工作人员正好最近也遇到了一个难题，需要一个会动态规划算法的同学帮帮忙。他恰好看到 Zhoues 的简历说自己熟练掌握《算法导论》上的动态规划问题，于是打算用这个难题考一考 Zhoues， 问题如下：

- 给定一段长度为 $n$ 英寸的钢条和一个价格表，该价格表表示长度为 $i(i=1,2,\ldots,n)$ 英寸的钢条的价格为 $p_i$。求钢条切割方案，使得总销售价格最大，注意**钢条的长度必须为整英寸**。

## 题解思路

题目来源于《算法分析》课本第204页的钢条切割问题。课本中对本题有详细的分析，这里只简明扼要地总结一下。

本题的实现共有自顶向下的递归法、带备忘的自顶向下法和自底向上法。其中自顶向下的递归法复杂度为 $O(2^n)$，不可行。另外两种方法等价，复杂度均为 $O(n^2)$。

带备忘的自顶向下法采用递归的方法，每次递归首先检查所需值是否已知，如果是，则直接返回保存的值；否则，计算出所需的值并将其存入数组。自底向上法则是从1向上遍历，所以每一步所需的值都必定是已知的。

（本段用处不大）使用自底向上法，按照课本提供的伪代码，这两种方法都需要建立两个数组，其中一个数组 $p$ 存放不同长度钢条的价格，另一个数组 $r$ 存放不同长度钢条的最优收益值。但实际上，只用一个数组 $r$也可以解决问题。只需将不同长度钢条的价格直接存入数组 $r$ ，再用类似方法遍历即可。这是由于如果某长度钢条的价格低于此长度钢条的最优收益值，则在任何最优收益分割中，必定不会出现此长度的钢条，其必被分割成更小的钢条,所以 $q[i]$可直接被 $r[i]$代替。下面的代码展示的就是这种方法。

## 代码

```
#include <iostream>
using namespace std;
long long best[11451];
int main(){
    int n;
    scanf("%d", &n);
    for (int i = 1; i <= n; i++){
        scanf("%d", &best[i]);
    }
    best[0] = 0;
    for (int j = 1; j <= n; j++){
        long long max = -1;
        for (int i = 1; i <= j; i++){
            max = max > best[i] + best[j - i] ? max : best[i] + best[j - i];
        }
        best[j] = max;
    }
    printf("%lld", best[n]);
    return 0;
```

```
}
```

*Author : Shiny Sheff*