3-5：

> *给定 n 种物品和一背包。物品 i 的重量是 w_i，体积是 b_i，其价值为 v_i，背包的容量为 C，容积为 D。问：应该如何选择装入背包的物品，使得装入背包中物品的总价值最大？在选择装入背包的物品时，对每种物品 i 只有两种选择，即装入背包或不装入背包。不能将物品 i 装入背包多次，也不能只装入部分的物品 i。试设计一个解此问题的动态规划算法，并分析算法的计算复杂性。*

题目分析

本题目与 0-1 背包问题的区别在于多了一个限制条件“容积 D”，因此比较直观的想法就是多开一维数组进行限制。

最优子结构特性与标准的0-1背包问题完全相似，设$(x_1, x_2, \cdots, x_n)$是 n 元 0-1 向量，$(y_1, y_2, \cdots, y_n)$是所给问题的一个最优解，则$(y_2, \cdots, y_n)$是下面相应子问题的一个最优解。

$$max \sum_{i=2}^{n} v_i x_i$$

$$\begin{cases} \sum_{i=2}^{n} w_i x_i \leq C - w_1 y_1 \\ \sum_{i=2}^{n} b_i x_i \leq D - b_1 y_1 \\ x_i \in \{0, 1\}, 2 \leq i \leq n \end{cases}$$

固若不然，设$(z_2, \cdots, z_n)$是上述子问题的一个最优解，而$(y_2, \cdots, y_n)$不是它的最优解。由此可知，$\sum_{i=2}^{n} v_i z_i > \sum_{i=2}^{n} v_i y_i$且$w_1 y_1 + \sum_{i=2}^{n} w_i z_i \leq C \ \wedge \ b_1 y_1 + \sum_{i=2}^{n} b_i z_i \leq D$。因此有：

$$v_1 y_1 + \sum_{i=2}^{n} v_i z_i > \sum_{i=1}^{n} v_i y_i$$

$$w_1 y_1 + \sum_{i=2}^{n} w_i z_i \leq C$$

$$b_1 y_1 + \sum_{i=2}^{n} b_i z_i \leq D$$

这说明$(z_1, z_2, \cdots, z_n)$是所给问题的更优解，从而与假设矛盾。因此该问题同样具有最优子结构性质。

代码实现

详见 3-5 Code.cpp

输出示例

```
输入1：
25 30
7
4 6 8 3 6 7 4
9 3 7 9 8 4 7
5 2 8 8 3 3 9
输出1：
背包的最大价值为：28
装入的物品编号为：3 4 6 7
DP[0][4][9] = 5
DP[2][8][7] = 8

输入2：
3 100
5
1   4  5 6 7
101 2  6 4 3
3   20 9 1 8
输出2：
背包的最大价值为：0
没有办法装入物品。

输入3：
3 10
5
1 2  2 6 7
9 1  0 4 3
3 20 5 1 8
输出3：
背包的最大价值为：23
装入的物品编号为：1 2

输入4：
5 10
5
1 2  2 6 7
9 1  0 4 3
3 20 5 1 8
输出4：
背包的最大价值为：28
装入的物品编号为：1 2 3
```

算法分析

该算法本质与标准的0-1背包问题并无区别，由于我们要将 DP 这个三维数组给填充完整，因此时间复杂度为：$O(nCD)$。（其中，n 为物品数量，C 为背包容积，D 为背包体积）