

問題

ナップサック問題について、下記の説明文の空欄の番号に語群から記号を選んで記入しなさい。

説明文：図 1 に示すアルゴリズム 8.5 は 0-1 ナップサック問題の解を求める動的計画法を用いたアルゴリズムである。表 1 に示す荷物について、0-1 ナップサック問題として価格が最大になるようナップサックに荷物に詰めることを考える。1 番目から 4 番目までの荷物を順番に動的計画法の手順に従ってナップサックに詰めたときの重さに対する価値を求めて記入し、表 2 と末尾の解答欄を完成させなさい。ナップサックの容量は 8 とする。表内の数字の一部は記載を省略している。表の中には価値のみを解答する。

表 1 問題の表

番号 i	種類 x_i	重さ w_i [kg]	価格 v_i [円]
1	モカ	2	2000
2	キリマンジャロ	1	1000
3	コロンビア	4	3000
4	ブレンド	5	4000

表 2 動的計画法の表

番号 i \ 重さの合計	0	1	2	3	4	5	6	7	8
1		(1)						(2)	
2								(3)	
3				(4)					
4			(5)				(6)		

解答

- 価値の最大：(9)
- 荷物の組み合わせ：(10) (使用する荷物の番号を昇順で記載する。例：荷物 1、2、3 を使用する場合は「123」と記載する。)

アルゴリズム 8.5

0-1 ナップサック問題を解く動的計画法を用いたアルゴリズム

入力：荷物の重さを格納する配列 $W[1], W[2], \dots, W[n]$ ，荷物の価値を格納する配列 $V[1], V[2], \dots, V[n]$ ，およびナップサック容量を表す定数 C

2次元配列 T を準備し， T のすべての値を $(0, \phi)$ に初期化；

//アルゴリズムの実行のため， 表は0行目も準備する

```
for (i=1; i<=n; i=i+1) {
    for (j=1; j<=C; j=j+1) {
        if (j>=w[i]) {
            T[i-1][j-w[i]] に格納されている値を取り出し， その値を(v1,S1)とする；
            T[i-1][j] に格納されている値を取り出し， その値を(v2,S2)とする；
            if (v1+V[i]>v2) { T[i][j]=(v1+V[i],S1にiを追加) }
            else { T[i][j]=(v2,S2) }
        }
    }
}
```

図 1 問題のアルゴリズム