問３　ナップザック問題に関する次の記述を読んで，設問 1～3 に答えよ。

〔ナップザック問題〕

幾つかの種類の品物があり，それぞれの容積と価値が与えられているとき，選んだ品物の容積の合計が定められた値以下であるという条件（容量制限）を満たし，かつ，価値の合計（以下，価値合計という）が最大になるような品物の組合せを求める問題をナップザック問題という。

この問題では，一つの品物を選ぶ個数には制限がないものとする。

例えば，容積，価値を表 1 に示した 2 種類の品物 A，B があり，容量制限が 5 である問題を考える。この場合，品物 A を 1 個，品物 B を 2 個選ぶと，容積の合計は 5，価値合計は 14 となり，選んだ品物の価値合計が最大となる。

表 1　品物の容積と価値

	品物	
	A	B
容積	1	2
価値	2	6

〔動的計画法によるナップザック問題の解法〕

品物の容積や価値を正の整数に限定したナップザック問題に対して，動的計画法による解法が知られている。この問題に対する動的計画法は，元の容量制限以下の全ての値を容量制限としたときの，品物の種類を限定した問題（以下，小問題という）をあらかじめ解いておき，それらの解を用いることによって，元の問題の解を得る方法である。表 2 に示すような，選んだ品物の最大の価値合計を求める表に対して順に数値を埋めて考えると，この解法の手順は理解しやすい。

表 2　選んだ品物の最大の価値合計（作成途中）

	容量制限					
	0	1	2	3	4	5
A だけを選べる	0	2	4	6	①8	10
A，B を選べる	0	2	②6	8		

表 2 において，例えば，表 1 に示す品物 A，B を選べる場合の容量制限 3 までの小問題が解けているとする。この状態で，容量制限が 4 で品物 A，B を選べる場合の解は，次の考え方で求めることができる。

・容量制限が 4 で品物 A だけを選べる小問題の解は，表 2 から 8 であることが分かる（下線①部分）。

・品物 A，B を選べる場合，品物 B の容積が 2 であるので，4 から 2 を減じた容量制限 2 で品物 A，B を選べる小問題の価値合計 6（下線②部分）に，品物 B の価値 6 を加えた 12 の価値合計を得られることが分かる。

・8 と 12 を比較し，大きい方の 12 を，容量制限 4 の場合の価値合計として表 2 に記入する。これは，最後に品物 B を選んだことを意味する。

表 2 の空白の部分をこの手順に従って順に埋めていくと，容量制限が 5 のときの価値合計は 14 であることが分かる。

容量制限 4 の場合に最後に品物 B を選んだように，各容量制限の小問題を解いたときに最後に選んだ品物を表 3 に示す。

表 3　最後に選んだ品物

	容量制限					
	0	1	2	3	4	5
品物	なし	A	B	B	B	B

容量制限 5 では，最後に品物 B を選んだことが分かる。次に，品物 B の容積を引いた容量制限 3 では，最後に品物 B を選んでいる。これを続けていくと，価値合計 14 を実現する品物の組合せは，品物 A が 1 個，品物 B が 2 個であることが分かる。

表 1 の問題に対して，新たに，容積が 3 で価値が 9 である品物 C が追加されたときの問題を解くには，表 2 に品物 A，B，C を選べる場合の行を追加した表 4 を順に埋めていけばよい。

表4　品物A，B，Cを選べる場合の最大の価値合計

	容量制限					
	0	1	2	3	4	5
Aだけを選べる	0	2	4	6	8	10
A，Bを選べる	0	2	6	8	12	14
A，B，Cを選べる	0	2	ア	イ	ウ	エ

〔ナップザック問題に対する動的計画法によるプログラム〕

ナップザック問題に対する動的計画法によるプログラムを図1に示す。なお，Vはナップザックの容量制限，Nは品物の種類の数である。種類s（0～N－1）の品物の容積と価値は，それぞれ，size[s]，value[s]に格納されている。また，maxvalue[t]は，容量制限tの小問題の価値合計を保持する配列である。choice[t]は，容量制限tの小問題を解いたときに最後に選んだ品物を保持するための配列である。この配列は，選んだ品物の組合せを最後に出力するために使用する。なお，全ての配列の添字は0から始まる。

```
// 初期化
for(kを0からVまで1ずつ増やす)
    maxvalue[k] ← 0
    choice[k] ← −1    // −1は，品物が選ばれていないことを示す。
endfor

// 動的計画法メイン。Nは品物の種類の数，Vはナップザックの容量制限。
for(sを0からN−1まで1ずつ増やす)
    for(tをsize[s]からVまで1ずつ増やす)
        temp ← maxvalue[ オ ] + value[s]
        if(tempがmaxvalue[t]より大きい)
            カ ← temp
            choice[t] ← s
        endif
    endfor
endfor

// 結果の出力。選んだ品物と価値合計を出力する。
k ← V
while(choice[k]が0以上)
    choice[k]を出力
    k ← k − size[ キ ]
endwhile
ク を出力  // 価値合計を出力する。
```

図1　ナップザック問題に対する動的計画法によるプログラム

〔計算量に関する検討〕

動的計画法を用いてナップザック問題を解く場合の計算量のオーダは，品物の種類の数をN，ナップザックの容量制限をVとすると，$O($［　ケ　］)である。

設問1　品物 A，B，C を選べる問題について，〔動的計画法によるナップザック問題の解法〕に従って，(1)，(2)に答えよ。

(1)　表4中の［　ア　］～［　エ　］に入れる適切な数値を答えよ。

(2)　容量制限が5の場合に最大の価値合計を実現する品物 A，B，C それぞれの個数を答えよ。

設問2　図1中の［　オ　］～［　ク　］に入れる適切な字句を答えよ。

設問3　本文中の［　ケ　］に入れる適切な字句を答えよ。