Problem 1

The matrix for execution:

0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24	-26	-28	-30	-32	-34	-36	-38	-40	-42	-44
-2	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24	-26	-28	-30	-32	-34	-36	-38	-40
-4	0	4	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24	-26	-28	-30	-32	-34	-36
-6	-2	2	6	4	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24	-26	-28	-30	-32
-8	-4	0	4	8	6	4	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24	-26	-28
-10	-6	-2	2	6	5	8	6	4	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24
-12	-8	-4	0	4	8	6	5	3	6	4	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20
-14	-10	-6	-2	2	6	10	8	7	5	8	6	4	2	0	-2	-4	-6	-8	-10	-12	-14	-16
-16	-12	-8	-4	0	4	8	12	10	8	6	10	8	6	4	2	0	-2	-4	-6	-8	-10	-12
-18	-14	-10	-6	-2	2	6	10	9	7	5	8	12	10	8	6	4	2	0	-2	-4	-6	-8
-20	-16	-12	-8	-4	0	4	8	7	11	9	7	10	14	12	10	8	6	4	2	0	-2	-4
-22	-18	-14	-10	-6	-2	2	6	5	9	8	11	9	12	11	9	7	5	3	1	-1	-3	-5
-24	-20	-16	-12	-8	-4	0	4	3	7	6	9	8	11	9	13	11	9	7	5	3	1	-1
-26	-22	-18	-14	-10	-6	-2	2	1	5	4	7	11	9	8	11	15	13	11	9	7	5	3
-28	-24	-20	-16	-12	-8	-4	0	4	3	7	5	9	8	11	9	13	12	10	8	11	9	7
-30	-26	-22	-18	-14	-10	-6	-2	2	6	5	4	7	11	9	13	11	15	13	12	10	13	11
-32	-28	-24	-20	-16	-12	-8	-4	0	4	8	6	5	9	13	11	10	13	12	10	14	12	15
-34	-30	-26	-22	-18	-14	-10	-6	-2	2	6	10	8	7	11	10	8	11	10	9	12	11	13
-36	-32	-28	-24	-20	-16	-12	-8	-4	0	4	8	7	10	9	13	11	10	8	12	10	14	12
-38	-34	-30	-26	-22	-18	-14	-10	-6	-2	2	6	5	8	12	11	10	8	7	10	14	12	16
-40	-36	-32	-28	-24	-20	-16	-12	-8	-4	0	4	3	7	10	14	12	12	10	9	12	16	14
-42	-38	-34	-30	-26	-22	-18	-14	-10	-6	-2	2	1	5	8	12	11	14	12	12	10	14	13
-44	-40	-36	-32	-28	-24	-20	-16	-12	-8	-4	0	4	3	6	10	14	12	16	14	12	12	11
-46	-42	-38	-34	-30	-26	-22	-18	-14	-10	-6	-2	2	1	5	8	12	11	14	13	16	14	14

Directions

(d means diagonal , < means left, ^ means up)

0	<	<	<	<	<	<	<	<	<	<	<	<	<	<	<	<	<	<	<	<	<	<
^	d	<	<	<	<d	<	<	<	<d	<	<	<	<d	<	<d	<	<d	<	<d	<	<d	<
^	^	d	<d	<	<	<	<	<	<	<	<	<d	<	<	<	<d	<	<d	<	<	<	<
^	^	^d	d	<	<	<	<	<	<	<	<	<d	<	<	<	<d	<	<d	<	<	<	<
^	^	^	^	d	<	<d	<	<d	<	<d	<	<	<	<d	<	<	<	<	<	<d	<	<d
^	^	^	^	^d	d	d	<	<d	<	<d	<	<	<	<d	<	<	<	<	<	<d	<	<d
^	^d	^	^	^	d	^<	d	<d	d	<	<	<	<d	<	<d	<	<d	<	<d	<	<d	<
^	^	^	^	^d	^	d	<	d	<	d	<	<	<	<d	<	<	<	<	<	<d	<	<d
^	^	^	^	^	^	^	d	<	<	^<	d	<	<	<	<	<	<	<	<	<	<	<
^	^	^d	^d	^	^	^	^	d	<d	<d	^	d	<	<	<	<d	<	<d	<	<	<	<
^	^d	^	^	^	^d	^	^	^d	d	<	<	^	d	<	<d	<	<d	<	<d	<	<d	<
^	^	^	^	^	^	^	^d	^d	^	d	d	<	^	d	<d	<d	<d	<d	<d	<d	<d	<d
^	^d	^	^	^	^d	^	^	^d	^d	^d	^	d	d	^<d	d	<	<d	<	<d	<	<d	<
^	^	^d	^d	^	^	^	^	^d	^	^d	^	d	^<	d	^	d	<	<d	<	<	<	<
^	^	^	^	^d	^	^d	^	d	^	d	^<	^	d	d	^<	^	d	<d	<d	d	<	<d
^	^d	^	^	^	^d	^	^	^	d	^	d	^	d	^<	d	^<	d	<	d	<	d	<
^	^	^	^	^d	^	^d	^	^d	^	d	<	^	^	d	^<	d	^	d	^<d	d	<	d
^	^	^	^	^	^	^	^d	^	^	^	d	<	^	^	d	^<d	^	^d	d	^	d	^
^	^d	^	^	^	^d	^	^	^	^d	^	^	^	d	d	^	d	<	d	^<d	d	^<	d	<
^	^	^	^	^d	^	^d	^	^d	^	^d	^	^	^d	^	d	^	d	^<d	d	^	d	^<	d
^	^d	^	^	^	^d	^	^	^	^d	^	^	^	^d	d	^	d	<	d	<	d	^	d	^<
^	^d	^	^	^	^d	^	^	^	^d	^	^	^	^d	^d	^	^d	d	d	<	d	^<	^d	d
^	^	^d	^d	^	^	^	^	^	^	^	^	^	d	^	^	^	d	^<	d	<	<	^	^d
^	^	^	^	^d	^	^d	^	^d	^	^d	^	^	^d	d	^	^	d	^	d	d	<	d

An optimal alignment (red in the picture above)

U	A	A			G		G	U	G	C	A	U	C	U	A	G	U	G	C	U	G	U	U	A	G
U	A	A	G	U	G	C	G	U	G	C	A	U	G	U	A		U		A	U	G		U		G

Problem 2

---Table 0---

0	inf	inf	-1	-2
inf	0	inf	3	5
inf	4	0	inf	5
inf	3	4	0	inf
inf	4	4	inf	0

---Table 1---

0	inf	inf	-1	-2
inf	0	inf	3	5
inf	4	0	inf	5
inf	3	4	0	inf
inf	4	4	inf	0

---Table 2---

0	inf	inf	-1	-2
inf	0	inf	3	5
inf	4	0	7	5
inf	3	4	0	8
inf	4	4	7	0

---Table 3---

0	inf	inf	-1	-2
inf	0	inf	3	5
inf	4	0	7	5
inf	3	4	0	8
inf	4	4	7	0

---Table 4---

0	2	3	-1	-2
inf	0	7	3	5
inf	4	0	7	5
inf	3	4	0	8
inf	4	4	7	0

---Table 5---

0	2	2	-1	-2
inf	0	7	3	5
inf	4	0	7	5
inf	3	4	0	8
inf	4	4	7	0

Problem 3

Proof

Using Induction

Inductive hypothesis H(n):

For n⩾1, n is the total number of blocks in a stack, a stack can have at most two blocks with the same shape.

Base case:

n=1, stack has only one blocks, satisfy the hypothesis above.

Maintenance:

Assume H(n) is true for n greater than 1.

Let's prove H(n+1)here can be at most two blocks with the same shape. The (n+1) th block have three possiblilties:

1) If the n+1 th block is in a different shape with all other n blocks, so there is only one block in (n+1)th shape. According to assumption, H(n+1) is true.

2) If the n+1 th block is in same shape with one of the n blocks, assuming i th block

2.1) if there is only one block of the i-th shape in stack, then this time there is two. According to assumption,H(n+1) is also true.

2.2) if there are already two blocks of the i-th shape in stack. Let's prove it not exist. Using contradiction.

Let assume i-th shape has a width w, height h, depth d. (w>d>h).

(if two of them equals, the three base area is not strictly decresing)

A block can have at most three different base area. (w*h, w*d, d*h).

we know $w*h > d*h$

2.2.1)if we have used w*h, w*d as base, there do not exist, because w is longest edge, we can not place a edge with w considering two edge has a length w.

2.2.2)if we have used w*h, d*h as base, so when placing a third one , we need to use base w*d, however we have a longest length w now, can not put on any of the present two

2.2.3) if we have used w*d, d*h as base. same with 2.2.2). we cannot put the w*h on any of the present two.

Thus, we cannot put a third one of same shape directly on any of these two, let alone we put additional blocks on this two and make the base even smaller.

Thus 2.2) do not exist.

In conclusion, we prove H(n+1) is true.

(b)

General idea:

using Dynamic Programming

H(j) represents the tallest stack of boxes with box j on top. We need to put j box on top of stack [i](i-th box is the current top of the stack). Thus H(j) = Max (i<j and wi> wj and di>dj {H(i)} + hj.)

1. SubProblem: DP(i)

   choose the i th block (from i to n) in order to have the max height
2. Label of Subproble: i to n.
3. DP(i) = MAX{DP(i-1)+ height[k] for k in i to n which block(k) can be put on block[i-1]}.
4. G = (V, E) which V is set of subproblems.E is the dependency of subproblems Because block is ordered, so subproblem's dependency is a DAG.
5. The sink node of subproblem is DP(0) which can be computed directly from the height.
6. Main problem = DP[n]

algorithm

Input: A list of legal blocks which only has most two blocks are of the same shape.

Output: The Max height we can have if stack them according to the rule that a block can be put on the other if its bottom is strictly smaller than the other.

```
Sort the blocks in decreasing base area order.
initial
for item in order:
    H[item] = 0
H[1] = h1

for j=2 to n in order
    for i in 1 to j-1:
        if wi > wj and di>dj:           # can be put on the other
            if H(j) < H(i) + hj:        # find the max one
                H(j) = H(i) + hj
8□□□□□
```

Complexity Analysis:

time:

Sort = O(nlogn)

double for circulation = O(n*n)

Total T(n) = O(n*n)

Space:

O(n)

we just need to maintain a array to keep track of each height of each of the at most 3*n different base area .

Problem 4

Longest at position 1 is of length 1

There is no previous position

Longest at position 2 is of length 2

The previous position is 1

Longest at position 3 is of length 2

The previous position is 1

Longest at position 4 is of length 1

There is no previous position

Longest at position 5 is of length 3

The previous position is 3

Longest at position 6 is of length 4

The previous position is 5

Longest at position 7 is of length 2

The previous position is 4

Longest at position 8 is of length 1

There is no previous position

Longest at position 9 is of length 4

The previous position is 5

Longest at position 10 is of length 3

The previous position is 7

Longest at position 11 is of length 5

The previous position is 9

Longest at position 12 is of length 2

The previous position is 8

Longest at position 13 is of length 2

The previous position is 8

Longest at position 14 is of length 4

The previous position is 10

Longest at position 15 is of length 3

The previous position is 12

Longest at position 16 is of length 4

The previous position is 15

Longest at position 17 is of length 5

The previous position is 14

Longest at position 18 is of length 6

The previous position is 11

Longest at position 19 is of length 5

The previous position is 14

Longest at position 20 is of length 6

The previous position is 19

The longest increasing subsequence has length 6

one of the subsequences with the longest length is [8, 9, 14, 15, 17, 19]

The positions of the entries in this sequence are [1, 3, 5, 9, 11, 18]