Describe a Dynamic programming solution to the following problems:

Q1 An $m \times n$ rectangle of wood is to be cut into smaller rectangles. An $a \times b$ rectangle is worth $m_{a,b}$. The machine that cuts rectangles can only cut full length or full width. I.e. if after cutting there is an $x \times y$ rectangle then the machine can cut it into two rectangles $z \times y$ and $(x-z) \times y$ for some $z$ or into two rectangles $x \times z$ and $x \times (y-z)$. All rectangles cut must have integral side lengths.

**Solution** Let $f(p,q)$ be the maximum value obtainable from a $p \times q$ rectangle. We can write

$$f(p,q) = \max \left\{ \max_{1 \leq x \leq q} \{ m_{p,x} + f(p,q-x) \}, \max_{1 \leq y \leq p} \{ m_{y,q} + f(p-y,q) \} \right\}.$$ 

Q2 Consider a 2-D map with a horizontal river passing through its center. There are $n$ cities on the southern bank with $x$-coordinates $a(1)\ldots a(n)$ and $n$ cities on the northern bank with $x$-coordinates $b(1)\ldots b(n)$. You want to connect as many north-south pairs of cities as possible with bridges such that no two bridges cross. When connecting cities, you can only connect city $i$ on the northern bank to city $i$ on the southern bank. Construct a Dynamic Programming solution to this problem. (You can assume that $a(1) < a(2) < \cdots < a(n)$, but you *cannot* assume that $b(1) < b(2) < \cdots < b(n)$. If both sequences are increasing, then the problem is trivial).

**Solution:** Let $f(\ell)$ be the maximum number of bridges we can connect, considering only $a(1),\ldots,a(\ell)$. Then

$$f(\ell) = \max\{f(\ell-1), 1+\max\{f(k) : k < \ell \text{ and bridge } k \text{ avoids bridge } \ell \}\}.$$ 

Q3 Solve the infinite horizon problem for the given matrix of costs. Assume that $\alpha = 1/2$.

$$\begin{bmatrix} 5 & 4 & 1 & 8 \\ 2 & 1 & 5 & 6 \\ 3 & 1 & 5 & 4 \\ 4 & 3 & 6 & 1 \end{bmatrix}$$
Begin with the policy

\[ \pi(1) = 4, \pi(2) = 4, \pi(3) = 3, \pi(4) = 4. \]

**Solution:** We begin by evaluating the solution:

\[ y_1 = 8 + \frac{y_4}{2} = 9. \]
\[ y_2 = 6 + \frac{y_4}{2} = 7. \]
\[ y_3 = 5 + \frac{y_4}{2} = 10. \]
\[ y_4 = 1 + \frac{y_4}{2} = 2. \]

We must now check for optimality:

\[
\begin{array}{cccc}
5 + 9/2 & 2 + 9/2 & 3 + 9/2 & 4 + 9/2 \\
4 + 7/2 & 1 + 7/2* & 1 + 7/2* & 3 + 7/2 \\
1 + 10/2* & 5 + 10/2 & 5 + 10/2 & 6 + 10/2 \\
8 + 2/2 & 6 + 2/2 & 4 + 2/2 & 1 + 2/2*
\end{array}
\]

New policy is

\[ \pi(1) = 3, \pi(2) = 2, \pi(3) = 2, \pi(4) = 4. \]