COMP3121 Assignment 3

Question 2

2.1 For this question, we will solve this problem using dynamical programming by updating $T[i,j]$ for each (i,j) representing a panel in the grid and $T[i,j] = 1$ iff we can reach panel (i,j), and 0 otherwise. First, we notice that for all panels on the first row or first column, we can possibly have only one path to reach it. Once there is one block in the way we cannot reach it. So, we may update it iteratively: $T[1,i] = T[1,i-1] * (1 - B[1,i]), T[j,1] = T[j-1,1] * (1 - B[j,1])$. With $T[1,1] = 1$. So, we require that both the previous panel is reachable and there is no block on this very panel to reach it.
Then we update T using the recursive relation:
$T[i,j] = 1$ iff $B[i,j] = 0$ and at least one of $T[i-1,j]$ and $T[i,j-1]$ is 1. And we update T in increasing order of i and j. When we finish, we have $T[m,n]$, which is what we want. Note we have went through $O(mn)$ iterations, each costs constant time to check and update, hence total time cost is $O(mn)$.

2.2 We can use the same method of question 2.1 but fix it a little. We let $A[i,j]$ containing the min number of boxes to be removed if we need to get to this panel, so we can directly reach (i,j) iff $A[i,j] = 0$. And the previous initialization now becomes: $T[1,1] = 0$, $T[1,i] = T[1,i-1] + B[1,i]$, $T[j,1] = T[j-1,1] + B[j,1]$, since we still have only one path and need to remove all boxes along the way. And the new iterative relation is: $T[i,j] = B[i,j] + min(T[i-1,j], T[i,j-1])$. Note that this is because we can always reach a panel from up or from left, and we are free to pick the more efficient way. And if there is a box at this panel, we must remove it. And finally, we get $T[m,n]$, which is the min number of boxes to be removed to get to the exit. Similarly, total time complexity is $O(mn)$ to update the whole table T.

2.3 We need to make at least one step rightwards and one step downwards, so either we have a "down" followed by a "right", or inversely "right" followed by "down". So we may always replace this pair of movements by a "diagonal" movement.

2.4 Since we can use the diagonal move exactly once, we need to keep track of this information (whether we have used this move) alongside. Hence we Now have $T[i,j]$ and $A[i,j]$, where T records the min cost to reach (i,j) without using this diagonal move, just like we computed before but add up all costs instead of number of boxes removed (cost of passing a box is considered infinity cost since it is impossible); while A records the min cost along the way using exactly one diagonal move. Since we know we can build T in $O(mn)$ time, we just need to build A on top of it. The recursive relationship is just

$$A[i,j] = H[i,j] + min(A[i-1,j], A[i,j-1], T[i-1,j-1])$$

Since we can either use this diagonal move (then we need to go to $(i-1, j-1)$ without a diagonal move) or go horizontally/vertically from a previous panel. And $A[1,j]$ and $A[i,1]$ are initialized to be infinity, since we cannot reach the first line or column using a diagonal move.

Now that building A costs $O(mn)$ time further, our algorithm still costs overall $O(mn)$ time to find the shortest path to reach (m,n), given by $A[m,n]$.