1 Dance Contest (exam question from E16) Tomorrow is the big dance contest at DTU. You have a list of n songs that will be played during the contest, in chronological order. You know all the songs, all the judges, and your own dancing ability extremely well. For each integer k, you know that if you dance to the kth song on the schedule, you will be awarded exactly $\texttt{Score}[k]$ points, but some of the songs are very physically demanding. Thus if you want to dance to song k you will have to take a break for the $\texttt{Break}[k]$ songs before. E.g., if $\texttt{Break}[10] = 3$ and you choose to dance to song 10, then you cannot dance to song 7,8, and 9. You can assume that $\texttt{Break}[k] < k$.

Let $P(1,i)$ denote the maximal score you can have after the first i songs if you dance to song i. Similarly, let $P(0,i)$ denote the maximal score you can have after the first i songs if you do not dance to song i.

1.1 Fill out the table below when $\texttt{Score} = [3,4,8,1,2,1]$ and $\texttt{Break} = [0,1,2,3,1,2]$.

$P(d,i)$	1	2	3	4	5	6
0						
1						

1.2 Which of the following recurrences correctly computes $P(d,i)$: (give an argument for your answer)

A
$$P(d,i) = \begin{cases} 0 & \text{if } i = 0 \\ \max\{P(1,i-1), P(0,i-1)\} & \text{if } d = 0 \text{ and } i \geq 1 \\ P(1, i - \texttt{Break}[i]) + \texttt{Score}[i] & \text{if } d = 1 \text{ and } i \geq 1 \end{cases}$$

B
$$P(d,i) = \begin{cases} 0 & \text{if } i = 0 \\ \max\{P(1,i-1), P(0,i-1)\} & \text{if } d = 0 \text{ and } i \geq 1 \\ P(0, i - \texttt{Break}[i]) + \texttt{Score}[i] & \text{if } d = 1 \text{ and } i \geq 1 \end{cases}$$

C
$$P(d,i) = \begin{cases} 0 & \text{if } i = 0 \\ \max\{P(1,i-1), P(0,i-1)\} & \text{if } d = 0 \text{ and } i \geq 1 \\ \max\{P(0, i - \texttt{Break}[i]), P(1, i - \texttt{Break}[i])\} + \texttt{Score}[i] & \text{if } d = 1 \text{ and } i \geq 1 \end{cases}$$

1.3 Write pseudocode for an algorithm *based on dynamic programming and the recurrence from Question 1.2* that computes the maximum total score you can achieve. The input to your algorithm is the pair of arrays $\texttt{Score}[1 \ldots n]$ and $\texttt{Break}[1 \ldots n]$. Analyze the space usage and running time of your algorithm in terms of n.