Matthew Soicher
z5160737

Question 3

Let's assume that our enjoyment levels have been populated into a 2D array $Enj[N][3]$

For example:

$$Enj = \begin{bmatrix} [4,7,9] \\ [8,12,2] \\ [10,1,8] \end{bmatrix} \text{ where } N = 3$$

And where $Enj[i-1, j-1]$ is the enjoyment level received from activity j on day i

E.g., by the above matrix, the enjoyment level from day 3, activity 2 is $Enj[2,1] = 1$

Our approach is to use Dynamic Programming to keep track of optimal computations and find the maximum possible enjoyment without having the same activity 2 days in a row.

We can populate these results and continuously update the total enjoyment into a solution matrix $OptEnj[N][3]$.

Each row of the solution matrix refers to the optimal result (maximum enjoyment) received so far. At the end of our algorithm, our results are in the final row, and our solution is the maximum value in that row.

Algorithm

$OptEnj[N][3] = 0$

$if\ (N == 0)\ return\ 0;$

$for\ (int\ i = 1;\ i < N;\ i++)$

$OptEnj\ [i][0] \mathrel{+}= max(Enj[i-1][1], costs[i-1][2]);$
$OptEnj\ [i][1] \mathrel{+}= max(Enj[i-1][0], costs[i-1][2]);$
$OptEnj\ [i][2] \mathrel{+}= max(Enj[i-1][0], costs[i-1][1]);$

$return\ \max(OptEnj\ [N-1][0], OptEnj\ [N-1][1], OptEnj\ [N-1][2])\ ;$

Time Complexity is $\boldsymbol{O(n)}$