983. Minimum cost for tickets

DP problem. create a dp array with indexes till highest value of day. If the day is given, mark it as 1 in dp or else let it remain 0.

Now, work your way up the dp array. If the value in array is 0, just let the cost be dp[i-1].
If it is 1, choose min (dp[i-1] + cost[0],
dp[i-7] + cost[1],
dp[i-30] + cost[2]).

CODE

```
int mincostTickets ( vector <int> & days, vector <int> costs) {
    vector <int> dp ( days [days.size()-1]+1, 0);
    for ( int i = 0 ; i < days.size() ; i++) {
        dp [ days [i] ] = 1;
    }
    for ( int i = 1 ; i < dp.size() ; i++) {
        if ( dp[i] == 0 ) dp[i] = dp[i-1];
        else {
            dp[i] = min ({dp[i-1] + cost[0] ,
                dp[max (0, i-7)] + cost[1],
                dp[max (0, i-30)] + cost[2]});
```

```
return dp[dp.size() - 1];
}
```

Time complexity $O(w)$ → w is max no° of days in plan
$w = 365$

Space complexity $O(w)$.