CPSC 5031 Algorithms
HW #6 (10 pts)

II. Binomial coefficient (7 points)

a. Solve the recurrence relation for computing the binomial coefficients (seen below).

$$\begin{aligned} C(n,k) &= C(n-1,k-1) + C(n-1,k) \quad \text{for } n > k > 0, \\ C(n,0) &= C(n,n) = 1. \end{aligned}$$

Algorithm *Binomial*(n, k)

```
//Computes C(n, k) by the dynamic programming algorithm
//Input: A pair of nonnegative integers n ≥ k ≥ 0
//Output: The value of C(n, k)
for i ← 0 to n do
    for j ← 0 to min(i, k) do
        if  j = 0 or j = i
            C[i, j] ← 1
        else C[i, j] ← C[i − 1, j − 1] + C[i − 1, j]
return C[n, k]
```

Hint: Think about the "shape" of a table that records values for binomial coefficient, that has n+1 rows and k+1 columns.

b. The time and space efficiency for part a) is $\Theta(nk)$. Rewrite the `Binomial(n,k)` function with $\Theta(n)$ space efficiency.

III. Exercises 8.2 #1a,b (3 points)

a. Apply the bottom-up dynamic programming algorithm to the following instance of the knapsack problem:

item	weight	value
1	3	\$25
2	2	\$20
3	1	\$15
4	4	\$40
5	5	\$50

, capacity $W = 6$.

b. What is the maximal value? Which items make up the optimal subset?

Note(s):
- Use C++ or Java for those problems that require algorithm design.
- All problems may be found in the Levitin textbook.

Submission:
- Deadline: Monday, 5/22/2023, 11:59pm
- Submit your solutions on Canvas under HW #6