DYNAMIC PROGRAMMING

- Dynamic programming is an algorithm design technique was invented by a prominent U.S. mathematician, Richard Bellman, in 1950 is a general method for optimizing multistage decision processes.
- Dynamic programming is a technique for solving problems with overlapping subproblems.
- These subproblems arise from a recurrence relating a solution to a given problem with solutions to its smaller subproblems of the same type.
- Rather than solving overlapping subproblems again and again, each of the smaller subproblem is solved only once and the results are recorded in a table from which we can then obtain a solution to the original problem.
- One example of this category is generating the Fibonacci numbers.
  - ✓ The Fibonacci numbers are the elements of the sequence

    0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ... ,
  - ✓ Fibonacci numbers can be defined by the simple recurrence

    $F(n) = F(n-1) + F(n-2)$ for $n > 2$

    and two initial conditions are

    F(0) = 0, F(1) = 1.

Computing a Binomial Coefficient

- A *binomial coefficient,* denoted C(n, *k)*, is the number of combinations (subsets) of *k* elements from an n-element set ($0 \le k \le n$).
- The name "binomial coefficients" comes from the participation of these numbers in the binomial formula

$$(a + b)^n = C(n, 0)a^n + \cdots + C(n, k)a^{n-k}b^k + \cdots + C(n, n)b^n$$

- The important properties of binomial coefficients are

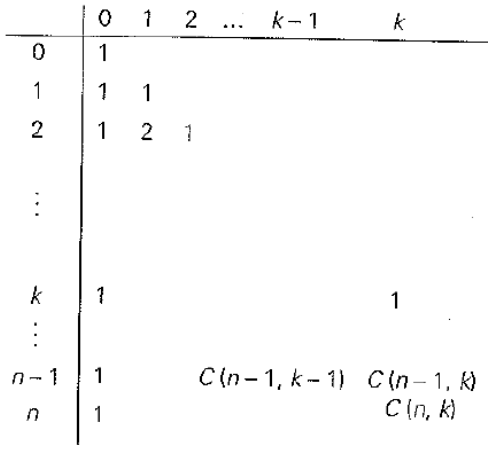
$$C(n, k) = C(n - 1, k - 1) + C(n - 1, k) \quad \text{for } n > k > 0$$

and

$$C(n, 0) = C(n, n) = 1$$

- *C(n, k)* can be computed in terms of the smaller and overlapping problems of computing *C (n* - 1, *k* - 1) and *C (n* - 1, *k),* i.e. it tends itself to be solved by the dynamic programming technique.

- The computed vales of the binomial coefficients can be recorded in a table of $n + 1$ rows and $k + 1$ columns, numbered from 0 to n and from 0 to k.



- To compute $C(n, k)$, the above table has to be filled row by row, starting with row 0 and ending with row n.
- Each row i ($0 \le i \le n$) is filled left to right, starting with 1 because $C(n, 0) = 1$.
- Rows 0 through k also end with 1 on the table's main diagonal: $C(i, i) = 1$ for $0 \le i \le k$.
- Other entries can be computed using the above formula i.e. adding the contents of the cells in the preceding row and the previous column and in the preceding row and the same column.
- The pseudocode for the algorithm is

```
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]
```

- Example

$^{7}C_{3} = ?$

	0	1	2	3
0	1			
1	1	1		
2	1	2	1	
3	1	3	3	1
4	1	4	6	4
5	1	5	10	10
6	1	6	15	20
7	1	7	21	**35**

- Time efficiency
  - ✓ The algorithm's basic operation is addition.
  - ✓ Here, the first $k + 1$ rows of the table form a triangle while the remaining $n - k$ rows form a rectangle, we have to split the sum expressing $A(n, k)$ into two parts:
  - ✓ The total number of additions is

$$A(n,k) = \sum_{i=1}^{k}\sum_{j=1}^{i-1} 1 + \sum_{i=k+1}^{n}\sum_{j=1}^{k} 1$$

$$A(n,k) = \sum_{i=1}^{k}(i - 1 - 1 + 1) + \sum_{i=k+1}^{n} k - 1 + 1$$

$$A(n,k) = \sum_{i=1}^{k} i - 1 + \sum_{i=k+1}^{n} k$$

$$A(n,k) = \sum_{i=1}^{k} i - \sum_{i=1}^{k} 1 + k\sum_{i=k+1}^{n} 1$$

$$A(n,k) = \frac{k(k+1)}{2} - (k - 1 + 1) + k(n - (k + 1) + 1)$$

$$A(n,k) = \frac{k(k+1)}{2} - k + k(n - k - 1 + 1)$$

$$A(n,k) = \frac{k(k+1)}{2} - k + k(n - k)$$

$$A(n,k) = \frac{k^2 + k - 2k}{2} + k(n - k)$$

$$A(n,k) = \frac{k^2 - k}{2} + k(n - k)$$

$$A(n,k) = \frac{(k-1)k}{2} + k(n - k) \in \theta(nk)$$