Programación Dinámica

Idea general

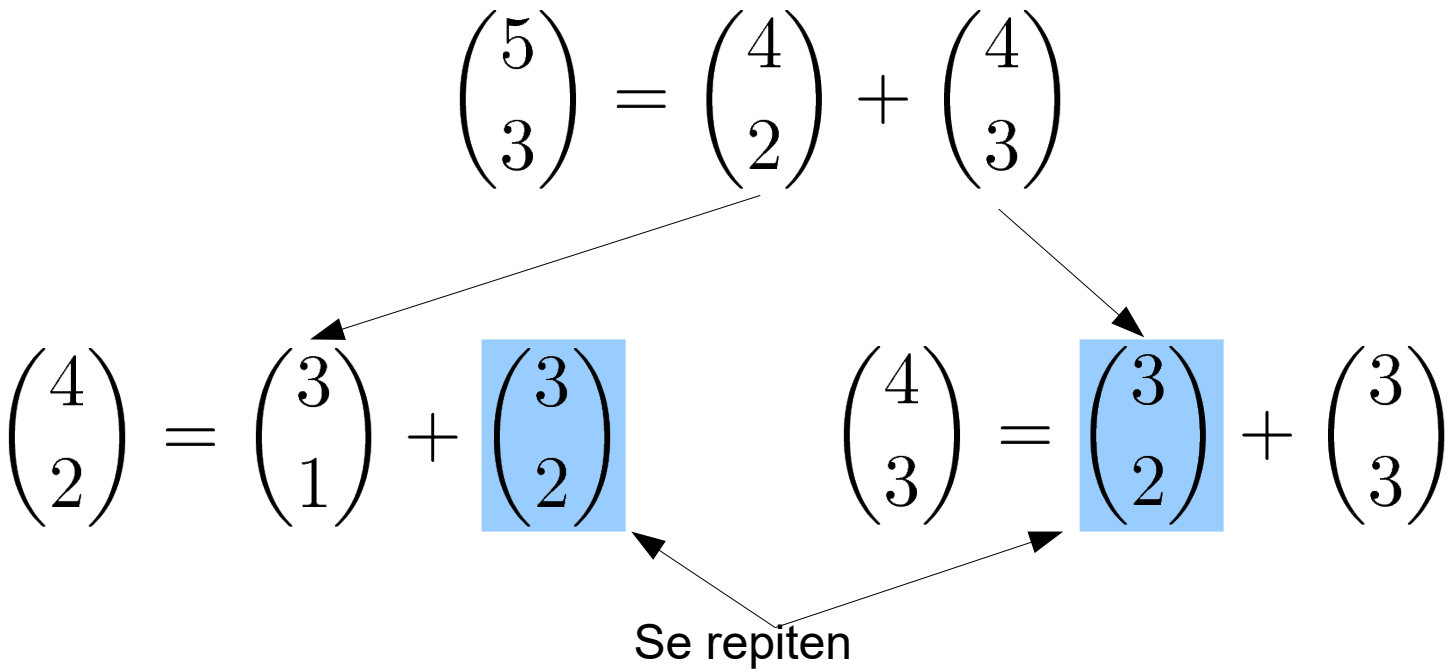
- En divide y vencerás se descompone un problema en muchos subproblemas.
- Cada uno de estos subproblemas se resuelve independientemente.
- Puede ocurrir que muchos subproblemas se repitan: Sería muy ineficiente resolver varias veces el mismo subproblema.
- La programación dinámica invierte el enfoque:
  - Se resuelven primero los subcasos,
  - los resultados se almacenan en memoria
  - hasta llegar al problema original.

Ejemplo

- Cálculo de los coeficientes binomiales divide y vencerás

```
función combinatorio(n,k)
  si k=0 o k=n devolver 1
  sino devolver combinatorio(n-1,k-1)+combinatorio(n-1,k)
```

Ejemplo de ejecución: combinatorio(5,3)



Solución por programación dinámica

- Se define una matriz A tal que a_{ij} es el número combinatorio C(i,j).
- Se calcula la matriz de abajo hacia arriba.
- Al llegar a a_{nk}, se obtiene la respuesta deseada C(n,k).
- Este algoritmo es conocido como el “triángulo de pascal”.

Triángulo de Pascal

```
función combinatorioDinámico(n,k)
    A ← matriz[0..n, 0..k]
    para i←0 hasta n
        para j←0 hasta min(i+1,k)
            Si j>i
                A[i,j] ← 0
            Sino, si j=0 o j=i
                A[i,j] ← 1
            sino
                A[i,j] = A[i-1,j-1] + A[i-1,j]
    devolver A[n,k]
```

Ejercicios:

- Demostrar que este algoritmo requiere espacio $\Theta(n*k)$.
- Proponer una mejora que solo requiera espacio $\Theta(n)$.
- Cuál es el tiempo de peor caso de este algoritmo?

Y una curiosidad

- Existe esta otra forma de calcular el número combinatorio:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

- Hacer el pseudo-código para hacer este cálculo y estimar su eficiencia asintótica.
- Como se compara con la versión dinámica?