

## Ranking für Zahl-Partitionen in umgedrehter lexikographischer Ordnung.

```
def rank_integer_partitions(pi):  
    """Bestimme den Rang der Zahlpartition pi (gegeben als absteigend  
    geordnetes numpy-array von ganzen Zahlen größer Null) in  
    umgedrehter lexikographischer Ordnung"""  
    # Anzahl der Teile:  
    nof_parts=len(pi)  
    # Bestimme die Partialsummen der Teile, mit Null beginnend  
    local_pi_partial_sums = np.zeros(nof_parts+1, dtype=int)  
    local_pi_partial_sums[1:] = np.cumsum(pi)  
    n = local_pi_partial_sums[-1]  
    # Berechne rekursiv die Funktionswerte  $p(n,k)$  in einer Tabelle  
    pnk = p_nk_by_recursion(n)  
    # Bestimme den Rang von pi  
    rank = pnk[n][n] - 1  
    # Obwohl pi und local_pi_partial_sums nicht gleich lang  
    # sind, können wir trotzdem "zip" anwenden:  
    for sum_pi_j, pi_j in zip(local_pi_partial_sums, pi):  
        rank-=pnk[n-sum_pi_j][pi_j-1]  
    return rank
```

Unter Verwendung der Funktion  $p(n,k)$  ist das Ranking einer Partition  $\pi$  ganz leicht: Diese Funktion ist uns aber *nur rekursiv* gegeben; hier wird sie in Form einer Tabelle berechnet.