

## Ranking für Funktionen von beschränktem Wachstum in lexikographischer Ordnung.

```
def rank_function_of_restricted_growth(the_func):
    """Berechne den Rang einer Funktion von beschränktem
    Wachstum in lexikographischer Ordnung."""
    # Vorbereitung: r-Bell-Zahlen und Funktion, die sie "ausliest".
    nnn = len(the_func)
    the_table = compute_r_Bell_numbers(nnn)
    def local_d(i,m):
        return the_table[i-1][m-1]
    # Wir bilden die "abschnittsweisen Maxima" von the_func, also
    #      [ max(the_func[:i]) for i in range(1, nnn + 1) ];
    # mit numpy-Funktionalität können wir das auch so erledigen:
    the_maxima = np.roll(np.maximum.accumulate(the_func),1)
    # (numpy.roll(a, k) "vertauscht zyklisch" nach rechts um k Stellen;
    # daß nun auf the_maxima[0] ein "falscher Wert" steht, macht hier
    # nichts, weil der in der folgenden Summe ohnehin nicht vorkommt.)

    # Berechne den Rang aus der Funktion als Summe:
    rank = 0
    for i, m_i, f_i in zip(range(2,nnn+1), the_maxima[1:], the_func[1:]):
        rank+= (f_i-1)*local_d(i,m_i)

    return rank
```

Rangfunktion:  $\text{function} f \mapsto \text{rank}$ . Wir brauchen hier als Vorbereitung eine Matrix von Zahlen, aus der wir dann Werte ablesen, die wir in der Berechnung verwenden.