

SOURCE CODE 1. Visualisierung mit `matplotlib`.

```
def visualize_ferrers_diagram_heights(n):
    """Visualisiere: Anzahl der Ferrers Diagramme der
    Zahl--Partitionen von n, die die einzelnen 1 x 1-Boxen
    im n x n-Quadrat enthalten"""
    # Wir erzeugen zuerst die Daten durch simples Zählen:
    ferrers_box = np.zeros(n**2, dtype=float).reshape((n,n))
    for pi in generate_integer_partitions(n):
        for row, pi_j in enumerate(pi):
            # Zeilen numerieren wir hier "von unten", damit das
            # mit der graphischen Ausgabe zusammenpaßt:
            ferrers_box[n-1-row][:pi_j] += 1.0
    # Wir dividieren alle Zahlen in dieser Matrix durch die Anzahl
    # aller Zahlpartitionen von n (die steht im Eintrag (n-1,0)):
    p_n = ferrers_box[n-1][0]
    ferrers_box /= p_n
    # Dann verwenden wir eine der (sehr zahlreichen!) Darstellungs-
    # möglichkeiten von matplotlib: Wir wollen eine einfache Abbildung
    # (figure) mit einem einzigen Achsenkreuz (axes):
    figure, axes = plt.subplots()
    # Die "Höhen" visualisieren wir durch Farben, die hier automatisch
    # gewählt werden:
    axes.pcolormesh(ferrers_box, shading='auto')
    # Um ein quadratisches Bild zu erhalten, das ja unserer Situation
    # entspricht, setzen wir die "aspect ratio" der Graphik auf 'equal':
    axes.set_aspect('equal')

    # Nun zeigen wir die Graphik:
    plt.title(f'Häufigkeit Kästchen in \n{int(p_n)} Ferrers Diagrammen:')
    plt.show()
```

Wenn man alle Ferrers Diagramme der Partitionen von n "übereinander schichtet", dann ergibt sich für jede "Box" im $n \times n$ -Quadrat eine "Höhe" (entsprechend der Anzahl der Ferrers Diagramme, in denen diese Box enthalten ist): Mit den Visualisierungs-Tools in `matplotlib` kann man das sehr hübsch sichtbar machen.