Serie 11

Aufgabe 1:

Schätzen Sie den Aufwand des *BubbleSort* Algorithmus (aus dem Skriptum) analytisch ab. Überprüfen Sie Ihre Abschätzungen indem Sie den Algorithmus implementieren und durch Zeitmessung den Aufwand mithilfe von `matplotlib` plotten.

Aufgabe 2:

Wir wollen den Aufwand von MergeSort analysieren. Der Einfachheit halber nehmen wir an, dass $n = 2^k$ (also immer durch 2 teilbar) ist und dass der Merge Schritt für zwei Listen der Länge $n/2$ den Aufwand bn ($b > 0$) hat (Warum macht diese Annahme Sinn?). Zeigen Sie dazu zuerst, dass der Aufwand $A(n)$ von MergeSort für eine Liste der Länge $n = 2^k$ die Rekursionsungleichung

$$A(n) \leq 2A(n/2) + bn + c$$

mit einer Konstante $c > 0$ erfüllt. Zeigen Sie weiters, dass eine Konstante $D > 0$ existiert, so dass eine Lösung der Rekursionsungleichung für alle $n \in \mathbb{N}$ durch

$$A(n) = Dn\log(n)$$

gegeben ist. (Diese Argumentation ist ein Spezialfall des Master-Theorems, welches in der Vorlesung nicht behandelt wurde. Zur Vollständigkeit müsste man noch zeigen, dass es keine größere Lösung gibt, was wir hier aber nicht tun.)

In [ ]:
```python
def merge(left, right):
    # Aufwand =  O(len(left)+len(right))

def mergesort(arr):
    if len(arr)<=1:
        return arr
    mid=len(arr)//2
    left=mergesort(arr[:mid])
    right=mergesort(arr[mid:])
    return merge(left,right)
```

Aufgabe 3:

Geben Sie Beispiele an wo der QuickSort Algorithmus für ein gegebenes $n \in \mathbb{N}$ aus der Vorlesung einen Aufwand von $\mathcal{O}(n^2)$ hat. Veranschaulichen Sie die Beispiele mithilfe `matplotlib`.

Aufgabe 4:

Installieren Sie da Modul `numpy`. Schreiben Sie eine Funktion, die eine Liste als Übergabeparameter hat und diese als `numpy`-array zurückgibt. Außerdem sollen die Dimensionen, sowie deren Anzahl am Bildschirm ausgegeben werden. Testen Sie Ihre Funktion mit einer 1-dimensionalen Liste, einer 2-dimensionalen Liste und einer 3-dimensionalen Liste. Demonstrieren Sie, wie man einen `view` und eine echte Kopie eines `numpy`-arrays erstellt. Erklären Sie den Unterschied zwischen beiden Konzepten.

Aufgabe 5:

Tagentialebene für Funktionen von 2 Variablen: Betrachten Sie die Funktion $f(x,y) = x^2 + y^2$ in $[-5,5]$. Plotten Sie die Funktion und die Tangentialebene im Punkt $(-4,4)$. Die Tangentialebene einer Funktion in einem Punkt (x',y') ist gegeben durch:

$$E(x,y) = f(x',y') + \nabla f(x',y') \cdot \begin{pmatrix} x - x' \\ y - y' \end{pmatrix},$$

also hier durch

$$E(x,y) = f(x',y') + \begin{pmatrix} 2x' \\ 2y' \end{pmatrix} \cdot \begin{pmatrix} x - x' \\ y - y' \end{pmatrix}.$$

Formatieren Sie den Plot indem Sie aussagekräftige Labels und Titel vergeben.

Aufgabe 6:

Laden Sie die Datei `xy1d.npy`. Die Datei enthält eine Liste der Form $[x_0, y_0, x_1, y_1, \ldots]$. Nutzen sie die reshape Funktion von numpy arrays um die Daten in eine nützlichere Form zu bringen und plotten Sie die x Daten gegen die y Daten mithilfe von `matplotlib`. Scalieren Sie die Axen mithilfe von `plt.axis('equal')`.