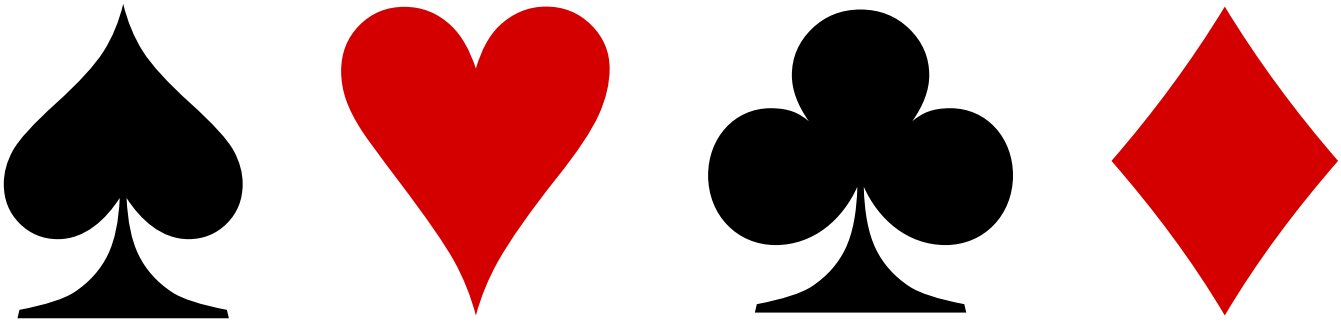
# Algorithmen und Datenstrukturen in Go

## Aufgabe: Sortieralgorithmen verstehen und anwenden



Für einen der unten aufgelisteten Sortieralgorithmen sollen die Eigenschaften stabil/instabil, iterativ/rekursiv, in-place/out-of-place geklärt und die Zeitkomplexität (O-Notation) im Best-, Average-, Worst-Case ermitteln werden.

Außerdem soll jeder Algorithmus in der Programmiersprache Go implementiert werden. Die implementierten Funktionen sollen ein Slice von Ganzzahlen beliebiger Menge sortieren können. Die Sortierfunktion soll die nachfolgende Signatur haben:

```go
func Sort(a []int) []int {
    // TODO: implement
    return a
}
```

Abschließend sollen die Implementierungen vorgestellt und der Ablauf der Algorithmen anhand von Spielkarten demonstriert werden.

**Die folgenden Sortieralgorithmen stehen zur Auswahl:**

- Heapsort
- Insertionsort
- Mergesort
- Quicksort
- Radixsort
- Selectionsort
- Shakersort
- Shellsort

# Links

- Sortieralgorithmen - Bleeptrack (https://deprecated.bleeptrack.de/tutorials/sortieralgorithmen/)
- Algorithmentänze - YouTube (https://www.youtube.com/user/AlgoRythmics)
- Heapsort - Wikipedia (https://de.wikipedia.org/wiki/Heapsort)
- Insertionsort - Wikipedia (https://de.wikipedia.org/wiki/Insertionsort)
- Mergesort - Wikipedia (https://de.wikipedia.org/wiki/Mergesort)
- Quicksort - Wikipedia (https://de.wikipedia.org/wiki/Quicksort)
- Radixsort - Wikipedia (https://de.wikipedia.org/wiki/Radixsort)
- Selectionsort - Wikipedia (https://de.wikipedia.org/wiki/Selectionsort)
- Shakersort - Wikipedia (https://de.wikipedia.org/wiki/Shakersort)
- Shellsort - Wikipedia (https://de.wikipedia.org/wiki/Shellsort)