Technische Universität Berlin **SoSe 2023**
Fakultät II, Institut für Mathematik
Sekretariat MA 6–2, Antje Schulz
Prof. Dr. Michael Joswig
Dr. Frank Lutz, Martin Knaack, Marcel Wack

3. Programmieraufgabe Computerorientierte Mathematik II

Abgabe: 19.5.2021 über den Comajudge bis 17:00 Uhr

Aufgabenstellung

In dieser Aufgabe sollen Sie einen Typ `Node`, sowie eine Funktion `top_sort(G)` schreiben, die mittels Tiefensuche eine topologische Sortierung eines einfachen gerichteten Graphen bestimmt, falls eine existiert. Der Graph soll dabei als Vektor von Objekten vom Typ `Node` gegeben sein. Hier hat jede `Node` als Attribute

- einen Vektor `successors::Vector{Node}` aller von diesem Knoten aus *direkt* erreichbaren Knoten,
- einen String `name` und
- ein Symbol `color::Symbol`, das bei Initialisierung auf `:white` gesetzt wird.

Es darf davon ausgegangen werden, dass alle Knoten unterschiedliche Namen haben.

Konkret gefragt sind Funktionen:

- `Node(name::String)::Node` Ein Konstruktor der ein `mutable struct` von Typ `Node` erstellt.
- `top_sort(G::Vector{Node})::Vector{Node}` Eine Funktion die eine topologische Sortierung des Graphen `G` zurück gibt.
  Eingabe Es wird ein Vektor G von `Node`-Objekten übergeben, der einen gerichteten Graphen G repräsentiert.

  Ausgabe Besitzt G eine topologische Sortierung, so werden die Knoten topologisch sortiert als Liste $[v_1, \ldots, v_n]$ zurückgegeben. Andernfalls wird der Fehler `error("Der Graph enthaelt einen Kreis!")` geworfen.

```
#Beispiel

S = Node("Senke");
Q = Node("Quelle");
Q.successors = [S];
Q
> Node("Quelle",:white,[Node("Senke",:white,[])])
top_sort([S,Q])
>2-element Vector{Node}:
> Node("Quelle",:black,[Node("Senke",:black,[])])
> Node("Senke",:black,[])
```