



Labor 3

Diskussion: 29./30./31. März, **Abgabe:** 4. April, **Wert:** 6/50

Eine **Dependency-Structure-Matrix** ist eine Darstellung einer Ansammlung von Elementen und deren Abhängigkeiten (siehe [Wikipedia](https://de.wikipedia.org/wiki/Dependency-Structure-Matrix)). Eine entsprechende Datenstruktur muss also die Namen der Elemente und deren Verbindungen speichern. Verbindungen haben ein Gewicht (ganzzahlig).

Implementieren Sie eine Klasse `DSM`, unter der Verwendung der Klassen `String`, `Vector` und eventuell anderen aus der Standard-Bibliothek.

A. Implementieren Sie zwei Konstruktoren

```
DSM(int elementCount)
```

```
DSM(vector<string> element_names)
```

Und fügen sie auch den entsprechenden Destruktor und Kopierkonstruktor hinzu.

B. Implementieren Sie die Methoden

```
a. public int size()
```

gibt die Anzahl der Größe der Matrix.

```
b. public void set_element_name(int index, string name)
```

```
c. public string get_name(int index)
```

Der Zugriff auf einen Index außerhalb des Gültigkeitsbereichs sollte eine Ausnahme auslösen.

```
d. public void add_link(string from_element, string  
to_element, int weight)
```

Wiederholtes Hinzufügen einer Verbindung zwischen Elementen A und B sollte die vorherige überschreiben. Das Hinzufügen einer Verbindung mit unbekanntem Elementnamen soll dieses zur Matrix hinzufügen.

```
e. public void delete_link(string from_element, string  
to_element).
```

C. Implementieren Sie die folgenden Analyse-Methoden

```
a. public bool have_link(string from_element, string to_link)
```

```
b. public int link_weight(string from_element, string  
to_link)
```

```
c. public int count_to_links(string element_name)
```

```
d. public int count_from_links(string element_name)
```



Labor 3

Diskussion: 29./30./31. März, **Abgabe:** 4. April, **Wert:** 6/50

D. Implementieren Sie die folgenden Methoden zur Bestimmung von Eigenschaften:

- a. `public int count_all_links()`
- b. `public double calculate_matrix_density()`

Die Dichte ist ein Wert zwischen 0 und 1 und ergibt sich aus der Proportion der vorhandenen Verbindungen zu den maximal möglichen Verbindungen.

Bonus A: Die Gewichtseigenschaft wird als Template-Parameter implementiert.

ODER

Bonus B: Die Matrix kann in eine Datei geschrieben und aus einer Datei gelesen werden, die aus 0..N Zeilen des folgenden Formats besteht:

```
from_element_name , weight , to_element_name
```

Die Lösung besteht aus 3 Dateien:

- `L3_Nachname_Vorname_DSM.h` enthält die Definition der Klasse (Methodensignaturen müssen exakt mit den oben angegebenen übereinstimmen.)
- `L3_Nachname_Vorname_dsm.cpp` enthält die Implementierung
- `L3_Nachname_Vorname_dsm_test.cpp` enthält `main()` mit den Aufrufen aller Konstruktoren und Methoden, sowie der Überprüfung der Ergebnisse.

In jeder Datei, für jede Klasse und jede Funktion ist es **erforderlich** aussagekräftige Kommentare zu schreiben!

Das Programm muss mit dem GnuCompiler compilierbar sein:

```
g++ -std=c++17
```