## Design Entscheidungen:

### GUI:

Die GUI hat einige kleine Erweiterungen bekommen:

* Im Menü unter den Algorithmen finden sich nun die beiden neu implementierten Algorithmen Kruskal und Prim.

### Algorithmen:

Ähnlich wie bei den *ShortestPath* Algorithmen entschieden wir uns auch bei den *MinimalSpanningTree* Algorithmen dazu gegen ein Interface zu programmieren und somit die Algorithmen in der Verwendung leicht austauschbar zu machen.

Des Weiteren wurden auch die innerhalb der *MinimalSpanningTree* Algorithmen genutzten *EdgeFinder* und *VertexFinder* extrahiert, abstrahiert und hinter ein Interface gestellt.

Dies ermöglichte es uns auch innerhalb der Algorithmen sehr leicht zwischen der Verwendung dieser „Tools“ zu wechseln.

Somit gibt es auch keine Implementationsunterschiede innerhalb des Algorithmus zwischen den Versionen mit und ohne *FibonacciHeap*.

Sämtliche Unterschiede befinden sich hier in den extrahierten jeweiligen *EdgeFinder* Implementationen.

Unsere Implementation des *EdgeFinders* mit *FibonacciHeap* stützt sich auf die *FibonacciHeap* Implementation aus der JGraphT Bibliothek.

Die *VertexFinder* werden lediglich dazu verwendet um zu Beginn der Algorithmen einen Knoten auszuwählen, welcher dem Spannbaum hinzugefügt wird.

### Tools:

Um uns die Implementierung der Algorithmen einfacher zu gestalten haben wir erneut neue Tools erzeugt und alte Tools erweitert bzw. gekürzt, wenn ersichtlich wurde das einige Funktionen nicht benötigt waren.

Der *GraphenGenerator* wurde erweitert und nutzt nun eine neue Familie von Tools, nämlich die *HeuristikGeneratoren*, welche die Berechnung von Heuristiken zu übernehmen.

Außerdem gibt es nun ein Tool für Gewichtete Graphen welches einige häufig verwendete Funktionalitäten anbietet.

Entfernt wurden die Funktionalitäten des *GraphGenerators* Kreise und Räder zu erzeugen, da diese bereits in der JGraphT Bibliothek vorhanden sind.

## 2. Textantworten:

### 2.1.(a)

Eine gegebene Kantenfolge ist genau dann ein Eulerkreis, wenn sie alle Kanten des Ausgangsgraphen beinhält, und der Startknoten identisch zum Endknoten ist.

### 2.1.(b)

### 5. Laufzeit & Zugriffsverhalten von *MinimalSpanningTree* Implementationen:

Lorem ipsum dolor sit amet, consectetur adipiscing elit.

## 3. Theorieteil:

### Aufgabe X:

1. (a,b b,c)
2. (a,b a,c a,d)

### Aufgabe XI:

### Aufgabe XII: