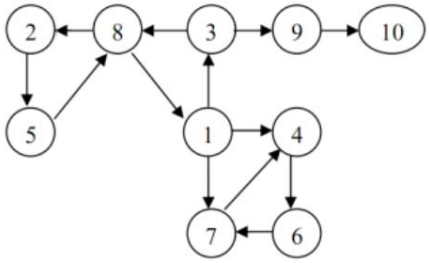
Algorithmen und Datenstrukturen

Backtracking

Aufgabe 1: Manuelle Graphen-Analyse [0 Punkte]

Gegeben ist folgender Graph G:



Führen Sie für den Graphen G die Tiefensuche durch (Vorsicht: leicht veränderter Graph), beginnend im Knoten 1. Geben Sie die Reihenfolge der Knotenbesuche an. "Besuchen" Sie dabei für einen Knoten seine Nachbarn immer in aufsteigender Reihenfolge (also von Konten 1 aus zunächst 3, dann 4, dann 7).

Aufgabe 2: Erstellen des Graphen [2 Punkte]

Lesen Sie das Labyrinth aus der Datei Labyrinth.txt ein, erstellen Sie einen Graphen. Verwenden Sie wieder den `DikstraNode`, da dieser die benötigten Felder enthält.

Hinweise:

- `Graph` und `AdjListGraph` aus dem letzten Praktikum verwenden.
- Nehmen Sie einfach z.B. "0-6" als Knotennamen; so können Sie zum Zeichnen die Koordinaten leicht wieder "berechnen" (z.B. x = 0; y = 6).

Abgabe

Praktikum: ADS8.2

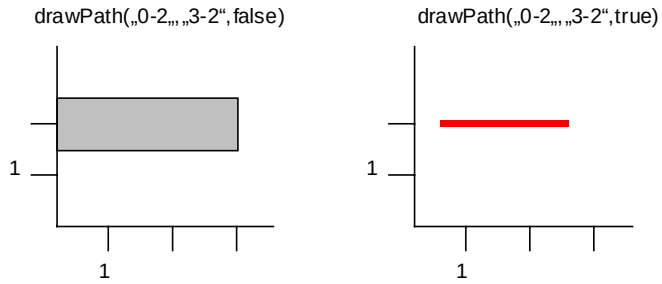
Filename: LabyrinthServer.java

Aufgabe 3: Zeichnen des Labyrinths [4 Punkte]

Zeichnen Sie das Labyrinth.

Hinweise:

- Zum Zeichen des Labyrinths kann die drawPath Methode von ServerGraphics verwendet werden. Mit dieser Methode kann sowohl *ein Weg* des Labyrinths (ein Pfad im Graphen: `line = false`) als *auch ein Teil der (Maus-)Spur* (line = true) gezeichnet werden.



- Zum Zeichnen des Labyrinths kann zuerst ein (dunkles) Rechteck gezeichnet werden und anschliessend mittels der Methode drawPath *in weiss* die Wege (`line = false`).

Abgabe

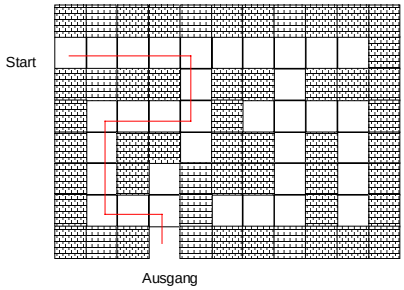
Praktikum: ADS8.3

Filename: LabyrinthServer.java

Aufgabe 4: Finden des Wegs aus dem Labyrinth [4 Punkte]

Backtracking bedeutet, dass die Ganze oder Teile einer gefundenen Lösung wieder verworfen werden. Eine elegante Art der Programmierung ist die Verwendung von Rekursion. Dabei wird eine rekursive Methode aufgerufen, die als Resultat zurückgibt, ob der eingeschlagene Weg zielführend war. Falls der Weg zum Ziel geführt hat wird *true* zurück gegeben, andernfalls *false*. Dank dem rekursiven Aufruf muss das Zurückgehen (Backtracking) nicht explizit ausprogrammiert werden, da der jeweilige Zustand (Position der Maus) auf dem Stack gespeichert wird.

Es soll nun eine Klasse `LabyrinthServer` erstellt werden, die den `CommandExcecutor` implementiert und den Ausgang aus dem gegebenen Labyrinth findet und den Weg als rote Markierung darstellt.



Implementieren Sie die rekursive Suche aus dem Script und geben Sie den gefundenen Weg als rote Spur aus (`line = true`).

Hinweise

- Setzen Sie das `prev` Feld beim rekursiven Aufruf und gehen Sie zum Zeichnen des Weges vom Ziel aus.

Abgabe

Praktikum: ADS8.4

Filename: LabyrinthServer.java