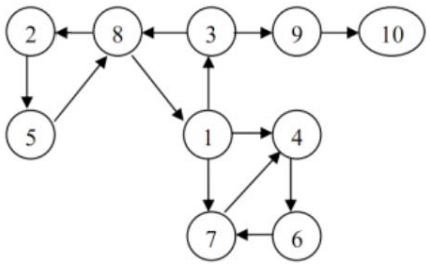
Algorithmen und Datenstrukturen

Backtracking

Aufgabe 1: Manuelle Graphen-Analyse [2 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).

1 - 3 - 8 - 2 - 5 - 9 - 10 - 4 - 6 - 7

Aufgabe 2: Neue Graphiken [2 Punkte]

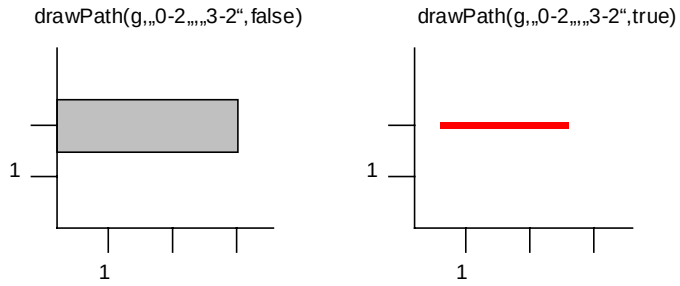
Es wird auch ein ServerGraphics bereitgestellt, das das Zeichnen auf der Serverseite erlaubt. Machen Sie sich mit den Klassen vertraut und zeichnen sie zur Übung ein möglichst grosses schwarzes gleichseitiges Dreieck und ein rotes Viereck.

Aufgabe 3: Zeichnen des Labyrinths [3 Punkte]

Lesen Sie das Labyrinth aus der Datei Labyrinth.txt ein, erstellen Sie einen Graphen und zeichnen Sie diesen. Definieren Sie einen `LabyrinthNode`, der 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).
- Zum Zeichen des Labyrinths kann untenstehende Methode zum LabyrinthServer hinzugefügt werden. Mit dieser Methode kann sowohl *ein Weg* des Labyrinths (**eine** Kante im Graphen: `mouse = false`) als *auch ein Teil der (Maus-)Spur* (mouse = true) gezeichnet werden.



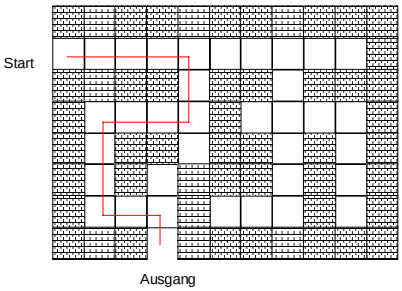

```
final double SCALE = 10;
  private void drawPath(ServerGraphics g, String from, String
to, boolean mouse) {
    double scale = 10;
    double xh0 = from.charAt(0) - '0';
    double yh0 = from.charAt(2) - '0';
    double xh1 = to.charAt(0) - '0';
    double yh1 = to.charAt(2) - '0';
    double x0 = Math.min(xh0,xh1)/SCALE;
    double y0 = Math.min(yh0,yh1)/SCALE;
    double x1 = Math.max(xh0,xh1)/SCALE;
    double y1 = Math.max(yh0,yh1)/SCALE;
    double w = 1/SCALE;
    if (mouse) g.drawLine(x0+w/2,y0+w/2,x1+w/2,y1+w/2);
    else {
      if (y0 == y1)
        g.fillRect(x0,y0,x1-x0+w,w);
      else
        g.fillRect(x0,y0,w,y1-y0+w);
    }
  }
```

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

Aufgabe 4: Finden des Wegs aus dem Labyrinth [3 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 (Faden) aus (`mouse = true`).

Hinweise

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