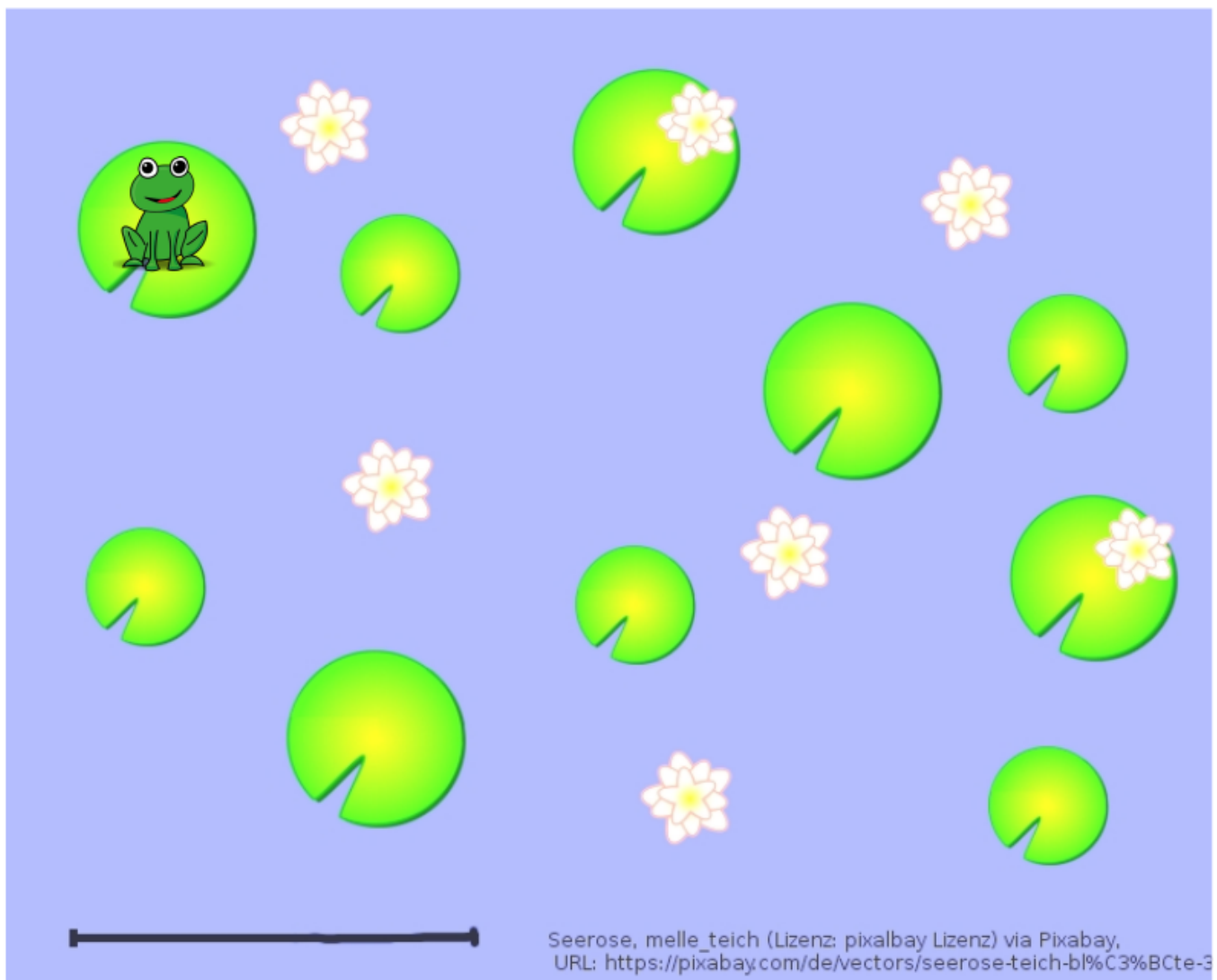


Kürzester Pfad nach Kantenzahl

Quacki der Frosch

Quacki sitzt in seinem wunderschönen Seerosenteich gemütlich auf einem Seerosenblatt und wartet auf Fliegen - seine Lieblingsspeise. Gelegentlich hüpft er zu einem anderen Blatt, wenn dort eine Fliege umherfliegt. Um möglichst schnell bei der Fliege zu sein, nimmt er denjenigen Weg, bei dem er am wenigsten Sprünge machen muss. Wie viele Sprünge muss Quacki maximal machen, um auf diese Weise zu einem beliebigen anderen Blatt zu kommen.



Hilf ihm bei der Antwort! Dabei muss die maximale Sprungweite (schwarzer Balken im Bild) berücksichtigt werden. Die Entfernung wird immer von Mitte zur Mitte der Blätter gemessen.



Wie ergibt sich die Anzahl der notwendigen Sprünge zu einem Blatt, wenn man weiß, dass man zu einem benachbarten Blatt fünf Sprünge (n Sprünge) braucht? Formuliere diese Aussage auch mit Knoten und Kanten.



(A3) Implementation

Implementiere den Algorithmus der erweiterten Breitensuche (Algorithmus von Mooore) im Graphentester. [Hilfe Grafentester](#)

Lösungsvorschlag

```
public void fuehreAlgorithmusAus() {
    gr = getGraph();
    for(Knoten k : gr.getAlleKnoten()) {
        k.setWert(Double.POSITIVE_INFINITY);
    }
    info("Setze alle Entfernungen auf unendlich");

    Queue<Knoten> q = new LinkedList<>();
    q.add(getStartKnoten());
    getStartKnoten().setMarkiert(true);
    getStartKnoten().setWert(0);
    info(" - Markiere und reihe ein: " + getStartKnoten().getInfotext());

    step();

    while(!q.isEmpty()) {
        Knoten v = q.remove();
        v.setFarbe(3);
        v.setBesucht(true);
        info("Bearbeite " + v.getInfotext());
        step();
        for(Knoten w: gr.getNachbarknoten(v)) {
            if(!w.isMarkiert()) {
                info(" - Markiere und reihe ein: " + w.getInfotext());
                w.setMarkiert(true);
                w.setWert(v.getIntWert()+1);
                q.add(w);

                step();

                info(" - Markiere Kante von " + v.getInfotext() + " nach "
+ w.getInfotext());
                gr.getKante(v,w).setMarkiert(true);
            }
        }
    }
}
```

}

}

}



(A4)

Für viele Anwendungen verwendet man **gerichtete** Graphen, d.h. die Kanten haben eine Richtung (z.B. bei einem Stadtplan mit Einbahnstraßen). Die Kanten dürfen dann nur in der vorgegebenen Richtung durchlaufen werden.

- Bestimme die Entfernungen in nebenstehendem Graphen vom markierten Knoten aus.
- Entscheide, ob der Algorithmus verändert werden muss, um auf gerichtete Graphen angewendet werden zu können.

Du kannst deine Ergebnisse im Graphentester nachvollziehen, wenn du den Graph `03_routenplanung/01_einbahnstrassen.csv` lädst.

1)

Du kannst [hier](#) spicken, wenn du es vergessen hast

From:
<https://www.info-bw.de/> -

Permanent link:
https://www.info-bw.de/faecher:informatik:oberstufe:graphen:zpg:kuerzeste_pfade:kpfad_kantenzahl:start?rev=1720074238

Last update: **04.07.2024 06:23**

