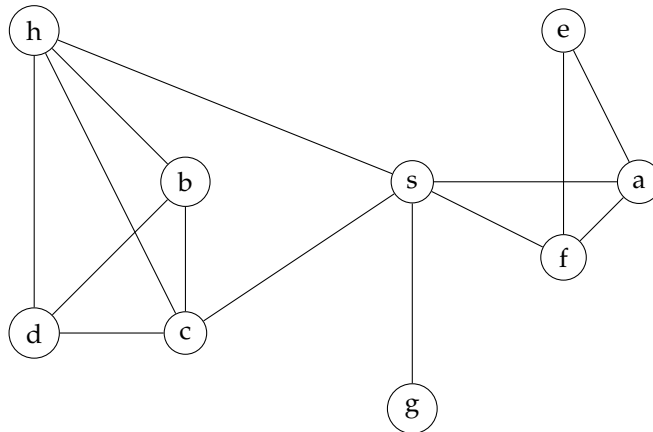


Aufgabe 8



- (a) Führen Sie auf dem folgenden ungerichteten Graphen G eine Tiefensuche ab dem Knoten s aus (graphische Umsetzung). Unbesuchte Nachbarn eines Knotens sollen dabei in *alphabetischer Reihenfolge* abgearbeitet werden. Die Tiefensuche soll auf Basis eines *Stacks* umgesetzt werden. Geben Sie die Reihenfolge der besuchten Knoten, also die *dfs-number* der Knoten, und den Inhalt des Stacks in jedem Schritt an.

```

1 public static void main(String[] args) {
2     TiefenSucheStapel ts = new TiefenSucheStapel(20);
3
4     ↪ ts.fügeKnotenUndKantenEin("a-e a-f a-s b-c b-d b-h c-d c-h c-s d-h e-f f-s g-s h-s");
5     ts.gibMatrixAus();
6     ts.starteTiefenSucheStapel("s");
7 }

```

In der Musterlösung auf Seite 3 lautet das Ergebnis $s, a, e, f, c, b, d, h, g$. Ich glaube jedoch diese Lösung ist richtig:

fett: Knoten, der entnommen wird.

kursiv: Knoten, die zum Stapel hinzugefügt werden.

Reihenfolge	Stapel	besucht
1	s	s
2	a, c, f, g, h	h
3	a, c, f, g, b, d	d
4	a, c, f, g, b	b
5	a, c, f, g	g
6	a, c, f	f
7	a, c, e	e
8	a, c	c
9	a	a

- (b) Führen Sie nun eine Breitensuche auf dem gegebenen Graphen aus, diese soll mit einer Queue umgesetzt werden. Als Startknoten wird wieder s

verwendet. Geben Sie auch hier die Reihenfolge der besuchten Knoten und den Inhalt der Queue bei jedem Schritt an.

```

1 public static void main(String[] args) {
2     BreitenSucheWarteschlange bs = new BreitenSucheWarteschlange(20);
3
4     ↪ bs.fügeKnotenUndKantenEin("a-e a-f a-s b-c b-d b-h c-d c-h c-s d-h e-f f-s g-s h-s");
5     bs.gibMatrixAus();
6     bs.starteBreitenSuche("s");
7 }

```

fett: Knoten, der entnommen wird.

kursiv: Knoten, die zur Warteschlange hinzugefügt werden.

Reihenfolge	Warteschlange	besucht
1	s	s
2	a, c, f, g, h	a
3	c, f, g, h, e	c
4	f, g, h, e, b, d	f
5	g, h, e, b, d	g
6	h, e, b, d	h
7	e, b, d	e
8	b, d	b
9	d	d

- (c) Geben Sie in Pseudocode den Ablauf von Tiefen- und Breitensuche an, wenn diese wie beschrieben mit einem Stack bzw. einer Queue implementiert werden.