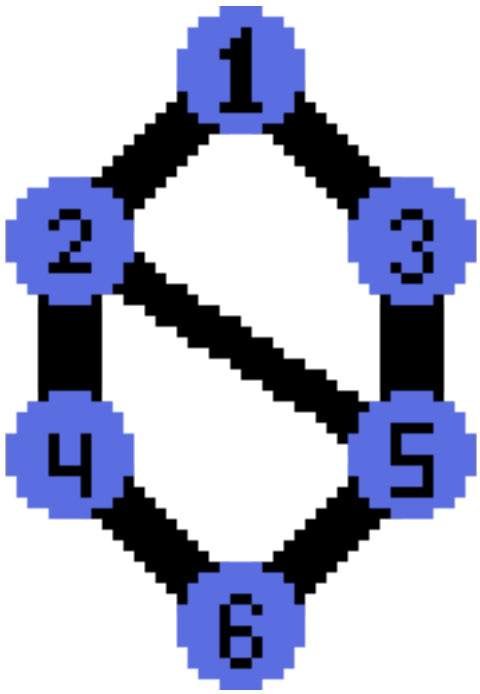
Vikublað 10 - tölvunarfræði 2

ttb3@hi.is

24. mars 2022

4.1.14

BFS notar *queue* til að geyma upplýsingar um hnúta sem búið er að skoða. Þetta gerir BFS kleift að finna stysta veg á milli tveggja hnúta. Ef við myndum nota *stack* myndi það hafa áhrif á leitina á þann hátt að öll önnur hliðin væri farin áður en byrjað væri að skoða hina. Frekar illa orðað en sjá teikningar,



Ef við byrjum með þetta tré og förum snöggt í gegnum það hvernig leitað væri í því með BFS þar sem er notað *queue*. Q táknar queue, P táknar print. Hver lína felur í sér að prenta næsta hnút í röðinni og setja tengda hnúta í röðina. Við byrjum á 1.

P	Q
	1
1	2,3
1,2	3,4,5
1,2,3	4,5
1,2,3,4	5,6
1,2,3,4,5	6
1,2,3,4,5,6	

Sjáið núna muninn ef við notum *stack* en ekki *queue*:

P	Q
	1
1	2,3
1,3	2,5
1,3,5	2,6
1,3,5,6	2,4
1,3,5,6,4	2
1,3,5,6,4,2	

Sama niðurstaða og við fengjum með DFS, þannig tldr. ef við notum *stack* í stað *queue* í BFS er ekki hægt að fá stystu leið því við erum í raun bara að nota DFS.

Bacon Tala

Forritið sem ég gerði notast við DegreesOfSeperation forritið sem gefið er í bókinni. Það skilar veginum auk Bacon tölunar, hér eru Bacon tölur allra leikaranna í Glengarry Glen Ross:

Leikari	Bacon Tala
Al Pacino	2
Jack Lemmon	1
Alec Baldwin	1
Ed Harris	1
Kevin Spacey	2
Alan Arkin	2
Jonathan Pryce	2
Jude Ciccolella	2
Lori Tan Chinn	2
Bruce Altman	2
Leigh French	2
George Cheung	2
Neal Jones	2
Murphy Dunne	2
Dana Lee	2
Gregory Snegoff	2
Julie Payne	2
Diane Dreyer	2
Paul Butler	1

BaconHistogram

```
    import edu.princeton.cs.algs4.*;

public class BaconHistogram {
    public static void main(String[] args) {
        String filename = args[0];
        String delimiter = args[1];
        String source  = args[2];

        SymbolGraph sg = new SymbolGraph(filename, delimiter);
        Graph G = sg.graph();
        if (!sg.contains(source)) {
            StdOut.println(source + " not in database.");
            return;
        }

        // run breadth-first search from s
        int s = sg.indexOf(source);
        BreadthFirstPaths bfs = new BreadthFirstPaths(G, s);


        // compute histogram of Kevin Bacon numbers - 100 for infinity
        int MAX_BACON = 100;
        int[] hist = new int[MAX_BACON + 1];
        for (int v = 0; v < G.V(); v++) {
            int bacon = Math.min(MAX_BACON, bfs.distTo(v));
            hist[bacon]++;

            // to print actors and movies with large bacon numbers
            if (bacon/2 >= 7 && bacon < MAX_BACON)
                StdOut.printf("%d %s\n", bacon/2, sg.nameOf(v));
        }

        // print out histogram - even indices are actors
        for (int i = 0; i < MAX_BACON; i += 2) {
            if (hist[i] == 0) break;
            StdOut.printf("%3d %8d\n", i/2, hist[i]);
        }
        StdOut.printf("Inf %8d\n", hist[MAX_BACON]);
    }
}
```

B-Tala	Fjöldi
0	1
1	1324
2	70717
3	40862
4	1591
5	125
ÓE	655