

Tölvunarfræði II

Skiladæmi 12

Sesar Hersisson - seh32

Dæmi 1:

0	$\xrightarrow{\text{.next}}$	2	$\xrightarrow{\text{.next}}$	6					
1	$\xrightarrow{\text{.next}}$	4	$\xrightarrow{\text{.next}}$	8	$\xrightarrow{\text{.next}}$	11			
2	$\xrightarrow{\text{.next}}$	5	$\xrightarrow{\text{.next}}$	6	$\xrightarrow{\text{.next}}$	0	$\xrightarrow{\text{.next}}$	3	
3	$\xrightarrow{\text{.next}}$	10	$\xrightarrow{\text{.next}}$	10	$\xrightarrow{\text{.next}}$	6	$\xrightarrow{\text{.next}}$	2	
4	$\xrightarrow{\text{.next}}$	1	$\xrightarrow{\text{.next}}$	8					
5	$\xrightarrow{\text{.next}}$	10	$\xrightarrow{\text{.next}}$	2					
6	$\xrightarrow{\text{.next}}$	2	$\xrightarrow{\text{.next}}$	3	$\xrightarrow{\text{.next}}$	0			
7	$\xrightarrow{\text{.next}}$	8	$\xrightarrow{\text{.next}}$	11					
8	$\xrightarrow{\text{.next}}$	1	$\xrightarrow{\text{.next}}$	11	$\xrightarrow{\text{.next}}$	7	$\xrightarrow{\text{.next}}$	4	
9									
10	$\xrightarrow{\text{.next}}$	3	$\xrightarrow{\text{.next}}$	5	$\xrightarrow{\text{.next}}$	3			
11	$\xrightarrow{\text{.next}}$	8	$\xrightarrow{\text{.next}}$	7	$\xrightarrow{\text{.next}}$	1			

Dæmi 2:

```
public boolean hasEdge(int v, int w) {  
    for(int vert : adj[v])  
    {  
        if(vert == w) return true;  
    }  
    return false;  
}
```

Dæmi 3-4

```
public class GraphProperties
{
    private Graph G;

    public GraphProperties(Graph G)
    {
        this.G = new Graph(G);
    }

    public int eccentricity(int V)
    {
        BreadthFirstPaths BFP = new BreadthFirstPaths(G, V);
        int max = -1;
        for(int i = 0; i < G.V(); i++)
        {
            if(BFP.hasPathTo(i))
            {
                if(BFP.distTo(i) > max)
                {
                    max = BFP.distTo(i);
                }
            }
        }
        return max;
    }

    public int diameter()
    {
        int max = 0;
        for(int i = 0; i < G.V(); i++)
        {
            if(eccentricity(i) > max)
            {
                max = eccentricity(i);
            }
        }
        return max;
    }

    public int radius()
    {
        int min = G.V() + 1;
        for(int i = 0; i < G.V(); i++)
        {
            if(eccentricity(i) == -1) {}
            else if(eccentricity(i) < min)
            {
                min = eccentricity(i);
            }
        }
        return min;
    }

    public int center()
    {
        int min = G.V() + 1;
        int center = 0;
```

```

    for(int i = 0; i < G.V(); i++)
    {
        if(eccentricity(i) == -1) {}
        else if(eccentricity(i) < min)
        {
            min = eccentricity(i);
            center = i;
        }
    }
    return center;
}

public int girth()
{
    BreadthFirstPaths[] BFP = new BreadthFirstPaths[G.V()];
    for(int i = 0; i < G.V(); i++)
    {
        BFP[i] = new BreadthFirstPaths(G, i);
    }
    int min = G.V() + 1;
    for(int i = 0; i < G.V(); i++)
    {
        for(int j : G.adj(i))
        {
            for(int k : G.adj(i))
            {
                if(k != j)
                {
                    if(BFP[j].distTo(k) + 2 < min) min = BFP[j].distTo(k) + 2;
                }
            }
        }
    }
    return min;
}

public static void main(String[] args){
    In I = new In();
    Graph G = new Graph(I);
    GraphProperties GP = new GraphProperties(G);
    int s = GP.girth();
    System.out.println(s);
}
}

```

Dæmi 5

```
import java.util.Scanner;

public class D4124
{
    public static void main(String[] args)
    {
        String filename = args[0];
        String delimiter = args[1];
        Scanner sc = new Scanner(System.in);
        SymbolGraph sg = new SymbolGraph(filename, delimiter);
        Graph graph = sg.graph();
        CC cc = new CC(graph);
        int m = cc.count();
        System.out.println("Number of connected components: " + m);
        Queue<Integer>[] components = (Queue<Integer>[]) new Queue[m];
        for (int i = 0; i < m; i++)
        {
            components[i] = new Queue<Integer>();
        }
        for (int v = 0; v < graph.V(); v++)
        {
            components[cc.id(v)].enqueue(v);
        }
        int count = 0;
        int max = 0;
        int maxIndex = 0;
        for (int i = 0; i < m; i++)
        {
            if(components[i].size() < 10) count++;
            if(components[i].size() > max)
            {
                max = components[i].size();
                maxIndex = i;
            }
        }
        System.out.println("Size of biggest component: " + max);
        System.out.println("Number of components with size less than 10: "
            + count);
    }
}
```

```
> run D4124 movie.txt /
Number of connected components: 23
Size of biggest component: 45851
Number of components with size less than 10: 0
> |
```

Dæmi 6

```
import java.util.Scanner;

public class D4122
{
    public static void main(String[] args)
    {
        String filename = args[0];
        String delimiter = args[1];
        Scanner sc = new Scanner(System.in);
        SymbolGraph sg = new SymbolGraph(filename, delimiter);
        Graph graph = sg.graph();
        String Bacon = "Bacon, Kevin";
        int B = sg.index(Bacon);
        String nafn = sc.nextLine();
        if (sg.contains(nafn))
        {
            int s = sg.index(nafn);
            BreadthFirstPaths Bac = new BreadthFirstPaths(graph, B);
            System.out.println(Bac.distTo(s)/2);
        }
        else
        {
            System.out.println("Nafn ekki í skra");
        }
        String mynd = sc.nextLine();
        int s = sg.index(mynd);
        for(int i : graph.adj(s))
        {
            BreadthFirstPaths Bac = new BreadthFirstPaths(graph, B);
            System.out.println(sg.nameOf(i) + " " + Bac.distTo(i)/2);
        }
    }
}
```