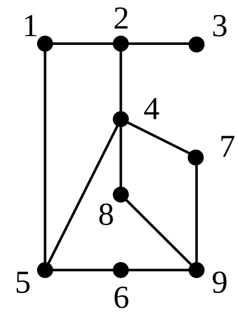
Exercises 3

29 January

All questions are unassessed.

1.
(a) State the order of traversal of the nodes in the graph starting from node 1 for (1) depth-first search (2) breadth-first search. Also draw the associated spanning trees. Assume that where there is a choice the numerically least node is chosen.
(b) The same question but starting from node 4.

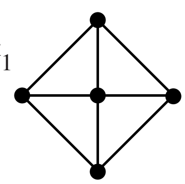


2. Suppose that G is a graph with no odd length cycles. We know from Sheet 2 Q3(b) that G is 2-colourable.
Write an algorithm based on depth-first search to perform the 2-colouring. Give a brief explanation. Use $0,1$ for the two colours. Use the adjacency list representation of G. You can use the pseudocode of the lectures or Java. Note that we do not assume that G is connected.

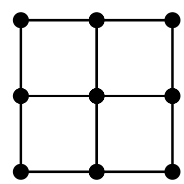
3. [From the 2008 exam] Let G be an acyclic graph with n nodes and k connected components. Give a formula for the number of arcs in G in terms of n and k. Prove that your formula is correct.

4. For each graph, state (a) whether it has a Hamiltonian path (HP) and (b) whether it has a Hamiltonian circuit (HC).

G_1



G_2



5. [From the 2002 exam] A mail delivery circuit (MDC, for short) is a path through a graph which uses every arc exactly twice and returns to the start node. Under what conditions does a graph have a MDC? Explain your answer.

6. [From the 2001 exam] Show by induction that a connected graph with n nodes has at least $n-1$ arcs.

7. [From the 2002 exam] Let G be a connected graph. An articulation point is a node x of G such that if x is removed from G (together with all arcs incident on x) then the resulting graph is no longer connected. In the following, you may assume that a connected graph with n nodes has at least $n-1$ arcs.
(a) Draw a connected graph with four nodes and exactly one articulation point. Mark the articulation point clearly.
(b) Show that for $n \geq 3$, a connected graph with n nodes has at least one node with degree ≥ 2.
(c) A graph G is 2-connected if it is connected and it has no articulation point. Show that a 2-connected graph with n nodes has at least n arcs (all $n \geq 3$).

8. [From the 2002 exam] A graph is acyclic if it has no cycles. A graph is a (nonrooted) tree if it is connected and acyclic.

(a) Show that a tree with at least two nodes must have at least one node of degree one.
(b) Show by induction that for all $n \geq 1$, if a tree has n nodes then it has exactly $n-1$ arcs.

9. [From the 2003 exam] (a) What is the greatest number of arcs possible for a simple graph with n nodes (any $n \geq 1$)? Justify your answer. Give an example for $n = 5$ to show that this greatest number can be achieved.
(b) What is the least number of arcs possible for a simple graph with n nodes, where each node has degree ≥ 3? Justify your answer.
Give an example for $n = 5$ to show that this least number can be achieved.

10. [From the 2014 exam] Let G be a connected graph, and let T be a spanning tree of G obtained by DFS starting at node x. Explain why if a is any arc of G (not necessarily in T), with endpoints y and z, then *either* y is an ancestor of z in T *or* z is an ancestor of y in T.
Here 'y is an ancestor of z in T' means that y lies on the (unique) path from x to z in T.

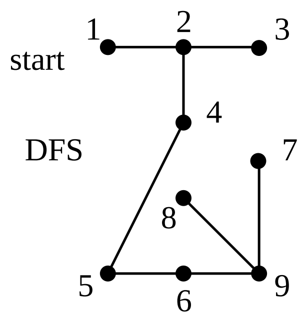
11. Here is a (non-meaningful) graph algorithm which iterates through the nodes of a graph and their adjacency lists.

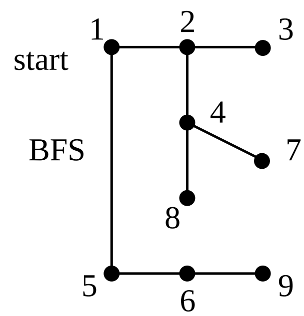
```
Algorithm Iterate:
count = 0
for x in Nodes:
   count += 1    # i.e. count++
   for y in adj[x]:
      count += 1
return count
```

If this algorithm is applied to a simple graph with n nodes and m arcs, what is the value (of count) returned? Give both an exact answer and one using big-O notation. You may wish to try some example graphs.
Notice that the algorithm counts its accesses to the graph, and so count is a measure of the running time of the algorithm.

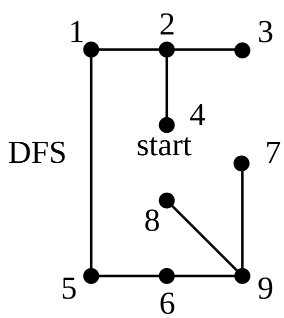
Answers to Exercises 3

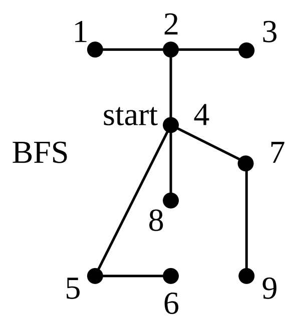
1. (a) DFS: 1,2,3,4,5,6,9,7,8 BFS: 1,2,5,3,4,6,7,8,9





(b) DFS: 4,2,1,5,6,9,7,8,3 BFS: 4,2,5,7,8,1,3,6,9





2. The idea is that each node is assigned the opposite colour from its parent.

Algorithm 2-Colour using DFS:

```
procedure twocol(x,col):
   visited[x] = true
   colour[x] = col
   for y in adj[x]:
      if not visited[y]:
         twocol(y,1−col)
for x in Nodes:
   if not visited[x]:
      # found a new connected component; start a new DFS at x with x coloured 0
      twocol(x,0)
```

3. $n-k$.
Let the ith component have n_i nodes. Then $n = \sum_{i=1}^{k} n_i$. Each component is a tree (connected and acyclic) and therefore has $n_i - 1$ arcs by Lectures slide 58. Total number of arcs is

$$\sum_{i=1}^{k}(n_i - 1) = (\sum_{i=1}^{k} n_i) - k = n - k$$

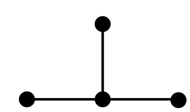
4. (a) both have HP

(b) G_1 has HC, G_2 does not.
To see that G_2 does not, either use *ad hoc* reasoning, or note that it is 2-colourable, and hence has no odd-length cycle, and therefore no HC since has an odd number of nodes (thanks to Petr Cermak).

5. Condition: graph is connected (apart from nodes of degree 0).
Transform a graph G satisfying the condition by (1) deleting nodes of degree 0 and (2) adding a new parallel arc for every existing arc to get graph G'. G' has *EC* iff G has MDC. But G' has EC since it is connected and every node has even degree.

6. Base case $n = 1$. Clearly any graph has at least 0 arcs.
Assume true for n. Take a connected graph G with $n+1$ nodes. If there is a node x with only one arc, then remove x and its arc to get G' with n nodes. G' must still be connected (since x is a needless diversion on any path between other nodes of G). G' has at least $n-1$ arcs by Ind Hyp. So G has at least n arcs.
So suppose that all nodes of G have degree at least 2. Then the sum of the degrees of $G \geq 2n$ and so G has $\geq n$ arcs (Ind Hyp not used in this case).

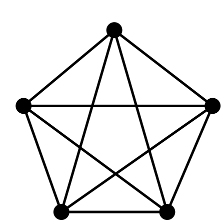
7. (a) For instance the accompanying graph (bottom centre node is the articulation point).
(b) Suppose for a contradiction that all nodes have degree ≤ 1. Then the sum of the degrees is $\leq n$. So since every arc is counted twice, #arcs$\leq n/2$. But G is connected, and so has $\geq n-1$ arcs. If $n \geq 3$ then $n/2 < n-1$. Contradiction.



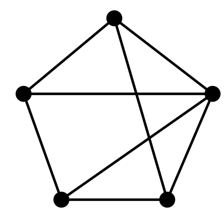
(c) Let G be 2-connected with n nodes ($n \geq 3$). Remove a node of degree ≥ 2 (exists by (b)). We must have removed ≥ 2 arcs. The new graph has $n-1$ nodes and is connected. So it has $\geq n-2$ arcs. Hence G has $\geq n-2+2 = n$ arcs.

8. (a) [Thanks to Sophia Drossopoulou for pointing out a mistake in a previous version] Since there are at least two nodes and the graph is connected, there can be no isolated nodes and all nodes have degree at least one. If there is no node of degree 1 then all nodes have degree ≥ 2. We can create a cycle as follows: start at any node and continue until a node is repeated. This works because we can continue past any new node since degree ≥ 2.
(b) Base case $n = 1$. If acyclic cannot have any arcs.
Assume true for n. Take tree G with $n+1$ nodes. Remove a node x of degree 1 (exists by (a)) and the single arc incident on it to get graph G' with n nodes. G' is still connected since the removed arc could only be part of a path to or from x. Also G' is acyclic since G is, and G' is a subgraph of G. So G' is a tree. By inductive hypothesis, G' has $n-1$ arcs and therefore G has $(n-1)+1 = n$ arcs as required. [NB A non-inductive proof is given in the lectures.]

9. (a) Let S = sum of degrees. Then $S = 2A$. Since graph is simple, degree of each node $\leq n-1$. So $S \leq n(n-1)$. Hence $A \leq n(n-1)/2$. Example: K_5 (see diagram).



(b) Let S = sum of degrees. Then $S = 2A$. Also $S \geq 3n$. Hence $A \geq \lceil 3n/2 \rceil$. NB In fact this is true whether or not graph is simple. Example: K_5 minus 2 arcs (see diagram).



10. Suppose that when performing depth-first search we reach y before z. Then while executing the procedure call dfs(y) we will process z as it belongs to adj$[y]$. At this point we either add z to the tree as a child of y, or else z has already been processed during dfs(y) and is a descendant of y.
The case where we reach z before y is similar.

11. Exact: $n + 2m$, which is $O(n+m)$. Each arc is accessed twice - once from each endpoint. This algorithm has the same running time as DFS and BFS.