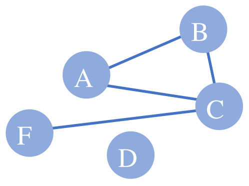
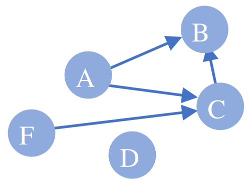
Graph Terminology



A *undirected graph* is a pair (V, E) where

1. V is a finite set of objects called *vertexes*, *vertices*, or *nodes*.
2. E is a set of pairs {u, v}, called *edges*, where u and v are nodes in V.



A *directed graph* is a pair (V, E) where

1. V is a finite set of objects called *vertexes*, *vertices*, or *nodes*.
2. E is a set of *ordered pairs* (u, v), called *edges*, where u and v are nodes in V. An edge (u, v) is depicted by an arrow leaving u and ending in v.

A directed graph is sometimes called a *digraph*.

We do not discuss graphs with loops

An edge from a node to itself is called a *loop*. To keep things simple, we assume through that graphs do not have loops.

Adjacency and degree

The nodes of an undirected edge {u, v} and a directed edge (u, v) are called the *endpoints* of the edge. For a directed edge (u, v), u is the *source* and v the *sink*.

Two nodes are *adjacent* if they are connected by an edge.

The *outdegree* of a vertex u in a directed graph is the number of edges for which u is the source.
The *indegree* of a vertex u in a directed graph is the number of edges for which u is the sink.
The *degree* of a node u in an undirected graph is the number of edges for which u is an endpoint.

Paths and cycles

A path is a sequence (v_0, v_1, …, v_p) of nodes such that for k, $0 \le k < p$,

If the graph is undirected, {v_k, v_{k+1}} is an edge,
If the graph is directed (v_k, v_{k+1}) is an edge.

The *length* of the path is the number of edges in it. This is 1 less than the number of nodes in the sequence!

Here are paths in the undirected and directed graphs shown above to the right:

(C) is a path of length 0 in both graphs.
(F, C, B) is a path of length 2 in both graphs.
(F, C, A) is a path of length 2 in the undirected graph. It is not a path in the directed graph.

A path is *simple* if all nodes in it are different. All three paths shown in the previous paragraph are simple. The path (F, C, A, B, C) in the undirected graph is not simple.

A path is a *cycle* if its length is ≥ 1 and its first and last nodes are the same. Paths (C, A, B, C) and (C, A, C, A, C) in the undirected graph are cycles. Path (F, C, A, B, C) in the undirected graph is not a cycle.

A cycle is *simple* if its only repeated nodes are its first and last nodes. Cycle (C, A, B, C) in the undirected graph is simple; cycle (C, A, C, A, C) in the undirected graph are is not simple.

Connected graph

A graph is *connected* if there is a path between any pair of nodes; otherwise, it is *disconnected*. A one-node graph is connected. You can verify that an n-node connected undirected graph has at least n-1 edges.

Acyclic graphs and dags

A graph is *acyclic* if does not contain a cycle. The undirected graph shown above is not acyclic; the directed graph is acyclic.

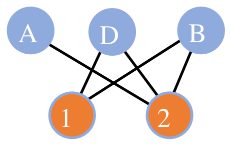
A *d*irected *a*cyclic *g*raph is called a *DAG*. The directed graph shown above is a DAG.

Here is an example of a DAG. Let the nodes be the courses that can be taken to satisfy a computer science (or other) major. Draw a directed edge from course c1 to course c2 if c1 is a prerequisite for c2. This graph better not have a cycle! Also, one can analyze the graph to find the longest prerequisite chain —it should not be more than 8, so students can graduate in eight semesters.

DAGs have many uses. In another pdf file on DAGs, we develop an algorithm called *topological sort*, which can determine whether a directed graph is a DAG.

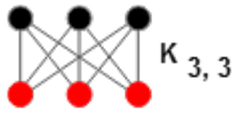
Bipartite graphs

A directed or undirected graph is *bipartite* if its nodes can be partitioned into two sets such that no edge connects two nodes in the same set. An example of a bipartite graph appears to the right, with the two sets being {A, D, B} and {1, 2}. The second one to the right is known as $K_{3,3}$. Its significance will become apparent when we discuss planar graphs.



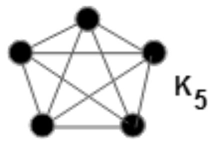
The following three properties are equivalent. You can verify the properties yourself.

1. Graph G is bipartite
2. Graph G is 2-colorable (see *Graph coloring*, below)
3. Graph G has no cycles of odd length.



Complete graph

A graph is *complete* if it has as many edges as possible. To the right is K_5, the complete (undirected) graph of 5 nodes. A complete directed graph of n nodes has n(n–1) edges, since from each node there is a directed edge to each of the others. You can change this complete directed graph into a complete undirected graph by replacing the two directed edges between two nodes by a single undirected edge. Thus, a complete undirected graph of n nodes has n(n–1)/2 edges.

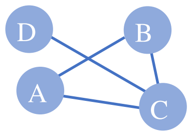


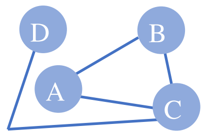
Graph $K_{3,3}$ is a complete bipartite graph, since it has as many edges as possible.

Planarity

A graph is *planar* if it can be drawn in the plane without any edges crossing.

Be careful with this definition! The first graph on the right doesn't look planar because edges (D, C) and (A, B) cross. But edge (D, C) can be redrawn so that the edges don't cross, so this graph *is* planar.





Is the 5-node bipartite graph shown at the top of the page planar? Determine that yourself. The second one, $K_{3,3}$, is not planar.

For a deeper discussion of planarity, look at JavaHyperText entry "planar graph".

Graph coloring

A coloring of an undirected graph is an assignment of a color to each node such that no two adjacent nodes have the same color.

Here's one application of graph coloring. Consider the graph in which the nodes are tasks to be performed. There is an edge between two nodes/tasks if they require the same shared resource, so they cannot be executed simultaneously. Produce a graph coloring. Then, two nodes/tasks have the same color if they can be carried out simultaneously. Think of the colors as *time* slots in which to schedule the tasks. Hence, the minimum number of colors needed to color the graph is the minimum number of time slots needed to carry out all tasks.

Sparse and dense graphs

Consider a connected graph of n nodes. The graph is *sparse* if the number of edges is close to the minimum, which is n-1. The graph is *dense* if the number edges is close to the maximum, which is n(n-1)/2.

These terms are not well defined in the literature. So we use these definitions: A graph of n nodes is dense if the number of nodes is proportional to or close to n*n; it is sparse if the number of edges is O(n).