1. Apply the Breadth-First Search algorithm to find a spanning tree of the Petersen graph. Explain why, no matter what choices you make, the isomorphism class of the resulting tree is always the same.

**Solution:** The Petersen graph is as follows (there is no standard choice of vertex set, but the letters a to j are as good as any).

![Petersen Graph Diagram]

To apply the Breadth-First Search Algorithm to this graph, we first have to choose a vertex to be $v_1$: suppose that we choose $a$. Then the next vertices to be added to the tree would be $e, f, b$, along with the edges joining these to $a$; we have to label these vertices as $v_2, v_3, v_4$ in some order, and it may as well be $v_2 = e, v_3 = f, v_4 = b$. Then the next vertices to be added to the tree would be $d, j$, along with the edges joining these to $e$; we can label these as $v_5 = d, v_6 = j$. In the next phase, the vertices to be added are $v_7 = i$ and $v_8 = h$, along with the edges joining these to $f$; and finally $v_9 = g$ and $v_{10} = c$, along with the edges joining these to $b$. The resulting ordering of the vertices and spanning tree are as follows:

![Spanning Tree Diagram]

The reason that the isomorphism class of the tree doesn’t depend on the choices is as follows. Since every vertex of the Petersen graph has degree 3, there are guaranteed to be three vertices $v_2, v_3, v_4$ adjacent to $v_1$ in the tree, no matter what $v_1$ is chosen. Each of $v_2, v_3, v_4$ is adjacent to two other vertices in the Petersen graph, and these six vertices are exactly the six remaining vertices of the graph, because the Petersen graph contains no 3-cycles or 4-cycles. So in the spanning tree, $v_5$ and $v_6$ are always adjacent to $v_2$, $v_7$ and $v_8$ to $v_3$, and $v_9$ and $v_{10}$ to $v_4$. Thus the spanning tree is always isomorphic to the one depicted above.

2. What sort of spanning tree do you obtain by applying the Breadth-First Search algorithm to the complete graph $K_n$? What about if you apply the Depth-First Search algorithm instead?
Solution: In $K_n$, very vertex is adjacent to every other. So if you apply the Breadth-First Search algorithm, you choose one vertex to be $v_1$, and then the other vertices are immediately numbered $v_2, \ldots, v_n$ by virtue of being adjacent to $v_1$. The spanning tree has the edges $\{v_1, v_i\}$ for every $i$ in $\{2, 3, \ldots, n\}$; so it is isomorphic to the graph $K_{1,n-1}$. If you apply the Depth-First Search algorithm instead, then the spanning tree is a path, because in the complete graph you never have to retrace your steps.

3. Show that if $T$ is a tree and $v, w$ are non-adjacent vertices of $T$, then $T + \{v, w\}$ contains exactly one cycle.

Solution: If $T + \{v, w\}$ contained a cycle which did not include the edge $\{v, w\}$, then that cycle would be contained in $T$, which is impossible since $T$ is a tree. So the only cycles in $T + \{v, w\}$ are those which contain the edge $\{v, w\}$. For any such cycle $C$, $C - \{v, w\}$ is a path in $T$ with end-vertices $v$ and $w$. By a result proved in lectures, there is a unique such path, and hence there is a unique cycle obtained by adding the edge $\{v, w\}$ to this path.

4. Use Prim’s Algorithm to find minimal spanning trees in these weighted graphs, starting with the vertex marked with a star. (Here the labels on the vertices have been omitted so as not to clutter the pictures.)

(a) ![Graph](image)

Solution: In Prim’s Algorithm, we build up a spanning tree from the chosen initial vertex by successively adding in an edge of smallest weight which connects a vertex already in the tree with one not yet in the tree. If there is more than one such edge, we choose one at random. For the given graph, the first edge chosen must be the one with weight 3 ending at the vertex *. The next edge could be either of the two edges with weight 4. Depending on the choice made in this step, one ends up with either of the following two spanning trees of weight 10:

(b) ![Graph](image)

Solution: Applying Prim’s Algorithm to this graph, the only choice is in the addition of the final edge of weight 6. The resulting spanning trees of
weight 13 are:

(c) 

Solution: The spanning tree of weight 25 resulting from Prim’s Algorithm is as follows:

5. Find the number of spanning trees of each of the following graphs.

(a) 

Solution: Since there are five vertices, any spanning tree has to use four of the five edges of the graph. Clearly the edge \{4, 5\} must be included in the spanning tree, but any one of the other four could be dropped. So there are 4 spanning trees of this graph.

(b) 

Solution: We could determine the number of spanning trees using reasoning similar to that in the previous part, but it is perhaps easier to apply the Matrix–Tree Theorem. The Laplacian matrix of this graph is

\[
M = \begin{pmatrix}
3 & -1 & 0 & -1 & -1 \\
-1 & 2 & -1 & 0 & 0 \\
0 & -1 & 3 & -1 & -1 \\
-1 & 0 & -1 & 3 & -1 \\
-1 & 0 & -1 & -1 & 3
\end{pmatrix},
\]
and the number of spanning trees equals any cofactor of $M$. The $(1, 1)$-cofactor is:

$$\det\begin{pmatrix} 2 & -1 & 0 & 0 \\ -1 & 3 & -1 & -1 \\ 0 & -1 & 3 & -1 \\ 0 & -1 & -1 & 3 \end{pmatrix} = 2 \det\begin{pmatrix} 3 & -1 & -1 \\ -1 & 3 & -1 \\ -1 & -1 & 3 \end{pmatrix} + \det\begin{pmatrix} -1 & -1 & -1 \\ 0 & 3 & -1 \\ 0 & -1 & 3 \end{pmatrix}$$

$$= 2(3 \times 8 - (-1) \times (-4) + (-1) \times 4) + ((-1) \times 8) = 24.$$  

*(c)*

**Solution:** This graph is the complete bipartite graph $K_{3,3}$. The approach using the Matrix–Tree Theorem is explained in the solution to Q10 (of which this is a special case). An alternative method is to classify the spanning trees according to how many vertices of degree 3 they contain. Note that there cannot be any vertices of degree greater than 3, since there are no such vertices in the graph. Also note that since the tree must have 5 edges, the sum of its degree sequence is 10.

If the tree has no vertices of degree 3, then it must be a path with 5 edges, and one of the end-vertices must be on the top (1, 2, or 3), with the other one on the bottom (4, 5, or 6). There are three choices for the end-vertex on the top, then three choices for the next vertex in the path (bottom), then two choices for the next vertex (top), then two choices for the next vertex (bottom), and no remaining choice for the last two vertices. So there are $3 \times 3 \times 2 \times 2 = 36$ such spanning paths.

If the tree has a single vertex of degree 3, then its degree sequence must be $(1, 1, 1, 2, 2, 3)$, and the two vertices of degree 2 must be adjacent to the vertex of degree 3. There are six choices for the vertex of degree 3, then $\binom{3}{2}$ choices for which of the adjacent vertices have degree 2, then two choices for which way round the remaining vertices are joined to the vertices of degree 2. So there are $6 \times \binom{3}{2} \times 2 = 36$ spanning trees of this type.

If the tree has two vertices of degree 3, then its degree sequence must be $(1, 1, 1, 1, 3, 3)$, and the two vertices of degree 3 must be adjacent, with one on the top and one on the bottom. There are $3 \times 3 = 9$ ways to choose these vertices of degree 3, and then the rest of the spanning tree is determined. So there are 9 spanning trees of this type.

We conclude that there are $36 + 36 + 9 = 81$ spanning trees in total.

**6.** How many spanning trees does the graph $K_n - \{1, n\}$ have? (*Hint:* since Cayley’s Formula gives the number of spanning trees of $K_n$, you just need to work out how many of these contain the edge $\{1, n\}$.)

**Solution:** Since there is complete symmetry between the $\binom{n}{2}$ edges of $K_n$, every edge must belong to the same number of spanning trees, say $d$. Then $\binom{n}{2}d$
overcounts the number of spanning trees by a factor of exactly $n - 1$, since every spanning tree has $n - 1$ edges. Hence \( \binom{n}{2} d = (n-1)n^{n-2} \), which shows that \( d = 2n^{n-3} \). In particular, there are \( 2n^{n-3} \) spanning trees which contain the edge \( \{1, n\} \), so there are \( n^{n-2} - 2n^{n-3} = (n-2)n^{n-3} \) spanning trees which do not contain the edge \( \{1, n\} \), and hence \( (n-2)n^{n-3} \) spanning trees of \( K_n - \{1, n\} \).

\*7. The adjacency matrix of a graph \( G \) with vertex set \( \{1, 2, \ldots, n\} \) is the \( n \times n \) matrix \( A \) whose entries are defined by

\[
a_{ij} = \begin{cases} 
1 & \text{if } \{i, j\} \text{ is an edge of } G, \\
0 & \text{otherwise.}
\end{cases}
\]

(a) Show that the \((i, j)\) entry of the \( \ell \)th power \( A^\ell \) equals the number of walks from \( i \) to \( j \) in \( G \) of length \( \ell \).

\textbf{Solution:} By definition of matrix multiplication, the \((i, j)\)-entry of the \( \ell \)th power \( A^\ell \) equals

\[
\sum_{1 \leq i_1, i_2, \ldots, i_{\ell-1} \leq n} a_{ii_1}a_{i_1i_2}\cdots a_{i_{\ell-1}j}.
\]

Each term in this sum is a product of 1s and 0s, so it is either 1 or 0 itself, and it is 1 if and only if \( \{i, i_1\}, \{i_1, i_2\}, \ldots, \{i_{\ell-1}, j\} \) are all edges of \( G \). This is equivalent to saying that \( i, i_1, i_2, \ldots, i_{\ell-1}, j \) is a walk from \( i \) to \( j \) in \( G \). So the sum has one nonzero term for every walk from \( i \) to \( j \) in \( G \) of length \( \ell \), and this term is always 1; the result is that the sum is the number of such walks, as claimed.

(b) Show that \( \frac{1}{2} \text{tr}(A^2) \) equals the number of edges in \( G \).

\textbf{Solution:} By definition of trace, \( \text{tr}(A^2) \) is the sum of the \((i, i)\)-entries of \( A^2 \) for all \( i \). By the previous part, this equals the number of walks in \( G \) of the form \( i, j, i \) (i.e. traversing an edge and then coming straight back). Clearly every edge \( \{i, j\} \) gives rise to two such walks, namely \( i, j, i \) and \( j, i, j \). So \( \frac{1}{2} \text{tr}(A^2) \) equals the number of edges in \( G \).

(c) Show that \( \frac{1}{6} \text{tr}(A^3) \) equals the number of 3-cycles in \( G \).

\textbf{Solution:} By similar reasoning to the previous part, \( \text{tr}(A^3) \) equals the number of walks in \( G \) of the form \( i, j, k, i \) (i.e. walks which traverse a 3-cycle (returning to the starting point). Every 3-cycle gives rise to 6 such walks, because there are 3 choices for the starting vertex and 2 choices for which way around to walk. So \( \frac{1}{6} \text{tr}(A^3) \) equals the number of 3-cycles in \( G \).

8. Let \( G \) be a connected graph. A strongly connected orientation of \( G \) is a way of putting a direction on each edge (i.e. rather than just being between \( v \) and \( w \), it is either from \( v \) to \( w \) or from \( w \) to \( v \)) so that there is a walk from any vertex to any other which obeys these directions.

(a) Find a strongly connected orientation of the following graph (indicate the directions with arrows on the edges).
Solution: The following is one strongly connected orientation (there are several):

(b) Prove that if $G$ has a strongly connected orientation, it contains no bridges.

Solution: Suppose that $G$ has a strongly connected orientation. Let $\{v, w\}$ be any edge of $G$, which is directed (say) from $v$ to $w$. By assumption, there is some walk from $w$ to $v$ which obeys the directions on the edges. We can assume that this walk does not use the edge $\{v, w\}$, since to obey the direction on this edge you need to be at the vertex $v$ already. But this means that $v$ and $w$ are linked in $G - \{v, w\}$, i.e. $\{v, w\}$ is not a bridge.

**(c)** Prove the converse to the previous part, as follows. Let $G$ be a connected graph with no bridges, and let $T$ be a spanning tree of $G$ produced using the Depth-First Search Algorithm, with the vertices labelled $v_1, \ldots, v_n$ in the order in which they were added to the tree. For any edge $\{v_i, v_j\}$ of $G$ where $i < j$, direct it from $v_i$ to $v_j$ if it belongs to the tree $T$, and from $v_j$ to $v_i$ if it does not. Show that this is a strongly connected orientation of $G$.

Solution: Let $G$ be a connected graph with no bridges, and define a spanning tree $T$ and an orientation as in the question. We need to show that for any $i, j$, there is a walk from $v_i$ to $v_j$ which obeys the directions on the edges. By the construction of $T$, for any $j$ there is a walk $v_i, v_{j_1}, v_{j_2}, \ldots, v_j$ in the tree $T$ such that $1 < j_1 < j_2 < \cdots < j$; this walk obeys the directions on these edges. So it suffices to show that for any $i \neq 1$, there is a walk from $v_i$ to $v_1$ which obeys the directions on the edges. In fact, it is good enough if we can find such a walk from $v_i$ to some $v_{i'}$ with $i' < i$, because then we can follow this with a walk from $v_{i'}$ to $v_{i''}$ for some $i'' < i$, and continuing in this way we must eventually reach $v_1$.

The fundamental fact about the DFS algorithm, proved in lectures, is that every edge of $G$ joins a vertex to one of its ‘ancestors’ in the tree $T$. According to our orientation, if the edge joins a ‘parent’ and a ‘child’, i.e. it actually belongs to the tree $T$, then it is directed from parent to child; otherwise, it is directed from descendant to ancestor. For our chosen $i \neq 1$, there is a unique edge in the tree $T$ which joins $v_i$ to its parent $v_k$; note that $k < i$.

By assumption this edge $\{v_i, v_k\}$ is not a bridge in $G$, so there is some walk in $G$ from $v_i$ to $v_k$ which does not use $\{v_i, v_k\}$. On the other hand, $\{v_i, v_k\}$ is definitely a bridge in $T$, so this walk must use some edges which do not belong to $T$. Suppose that the walk has the form

$$v_i, v_{i_1}, v_{i_2}, \ldots, v_{i_s}, v_{k_0}, \ldots, v_k$$

where $\{v_{i_s}, v_{k_0}\}$ is the first edge in the walk which does not belong to $T$. We can assume that the portion of the walk that stays in $T$ obeys the directions of the edges, i.e. $i < i_1 < i_2 < \cdots < i_s$, because otherwise our walk would include some back-tracking along the edge just used, which we could eliminate. If $v_{i_s}$ is an ancestor of $v_{k_0}$ in $T$, then we can replace the edge $\{v_{i_s}, v_{k_0}\}$ with a walk from $v_{i_s}$ to $v_{k_0}$ along the edges of $T$, again obeying
the directions. So we can assume that $v_{k_0}$ is an ancestor of $v_i$ in $T$. If $i \leq k_0 < i_s$, then $v_{k_0}$ is a descendant of $v_i$ in $T$, so we can replace the whole section $v_i, v_{i_1}, v_{i_2}, \ldots, v_{i_s}, v_{k_0}$ with a direct walk from $v_i$ to $v_{k_0}$ in $T$, obeying the directions. So we can assume that $k_0 < i$, and now $v_i, v_{i_1}, v_{i_2}, \ldots, v_{i_s}, v_{k_0}$ is our desired walk from $v_i$ to some $v_{k_0}$ with $k_0 < i$, obeying the directions of the edges.

**9.** Find the number of spanning trees of the complete bipartite graph $K_{p,q}$.

**Solution:** If $p = 1$ or $q = 1$, then $K_{p,q}$ is itself a tree, so the answer is 1. Assuming that $p, q \geq 2$, we use the Matrix–Tree Theorem. The Laplacian matrix of $K_{p,q}$ is

$$M = \begin{pmatrix} q & 0 & 0 & \cdots & 0 & -1 & -1 & \cdots & -1 \\ 0 & q & 0 & \cdots & 0 & -1 & -1 & \cdots & -1 \\ 0 & 0 & q & \cdots & 0 & -1 & -1 & \cdots & -1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & q & -1 & -1 & \cdots & -1 \\ -1 & -1 & -1 & \cdots & -1 & p & 0 & \cdots & 0 \\ -1 & -1 & -1 & \cdots & -1 & 0 & p & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & -1 & \cdots & -1 & 0 & 0 & \cdots & p \end{pmatrix},$$

where the diagonal entries are $q$ ($p$ times) and $p$ ($q$ times). Using row operations,
we simplify the \((1,1)\)-cofactor of this matrix:
\[
\begin{vmatrix}
q & 0 & \cdots & 0 & -1 & -1 & \cdots & -1 \\
0 & q & \cdots & 0 & -1 & -1 & \cdots & -1 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & q & -1 & -1 \cdots & -1 \\
-1 & -1 & \cdots & -1 & p & 0 & \cdots & 0 \\
-1 & -1 & \cdots & -1 & 0 & p & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
-1 & -1 & \cdots & -1 & 0 & 0 & \cdots & p
\end{vmatrix}
\]
\[= \det \begin{vmatrix}
0 & 0 & \cdots & 0 & 1 & 1 \cdots & 1 \\
0 & q & \cdots & 0 & -1 & -1 \cdots & -1 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & q & -1 & -1 \cdots & -1 \\
-1 & -1 & \cdots & -1 & p & 0 & \cdots & 0 \\
-1 & -1 & \cdots & -1 & 0 & p & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
-1 & -1 & \cdots & -1 & 0 & 0 & \cdots & p
\end{vmatrix}
\]
\[\text{(R}_1 \leftarrow \text{R}_1 + \cdots + \text{R}_{p+q-1})
\]
\[= \det \begin{vmatrix}
0 & 0 & \cdots & 0 & 1 & 1 \cdots & 1 \\
0 & q & \cdots & 0 & 0 & 0 \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & q & 0 & 0 \cdots & 0 \\
-1 & -1 & \cdots & -1 & p & 0 & \cdots & 0 \\
-1 & -1 & \cdots & -1 & 0 & p & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
-1 & -1 & \cdots & -1 & 0 & 0 & \cdots & p
\end{vmatrix}
\]
\[\text{(R}_2 \leftarrow \text{R}_1 + \text{R}_2, \text{ etc.})
\]
To continue the simplification, we use some column operations:
\[
\begin{vmatrix}
0 & 0 & \cdots & 0 & 1 & 1 \cdots & 1 \\
0 & q & \cdots & 0 & 0 & 0 \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & q & 0 & 0 \cdots & 0 \\
-1 & 0 & \cdots & 0 & p & 0 \cdots & 0 \\
-1 & 0 & \cdots & 0 & 0 & p \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
-1 & 0 & \cdots & 0 & 0 & 0 \cdots & p
\end{vmatrix}
\]
\[\text{(C}_2 \leftarrow -\text{C}_1 + \text{C}_2, \text{ etc.})
\]
\[
\begin{vmatrix}
2 & 0 & \cdots & 0 & 1 & 1 \cdots & 1 \\
0 & q & \cdots & 0 & 0 & 0 \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & q & 0 & 0 \cdots & 0 \\
0 & 0 & \cdots & 0 & p & 0 \cdots & 0 \\
0 & 0 & \cdots & 0 & 0 & p \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 0 & 0 & 0 \cdots & p
\end{vmatrix}
\]
\[\text{(C}_1 \leftarrow \text{C}_1 + \frac{1}{p}(\text{C}_p + \cdots + \text{C}_{p+q-1}))
\]
8
The matrix is now upper-triangular, so the determinant can be read off as the product of the diagonal entries:

\[
\frac{q}{p} - p^{q-2}p^q = p^{q-1}q^{p-1}.
\]

This formula does give the right answer in the cases \( p = 1 \) and \( q = 1 \), so the number of spanning trees of \( K_{p,q} \) is always \( p^{q-1}q^{p-1} \).