Using **adjacency matrices** to represent directed graphs (digraphs): A directed graph with n vertices can be represented by an \( n \times n \) matrix \( A \) with entry \( a_{i,j} \) equal to 1 if and only if there is an arc from vertex \( i \) to vertex \( j \); otherwise the entry is 0.

For example, the 4-vertex directed graph (vertices labeled 1 thru 4) with arcs \{(1,2), (2,3), (3,1), (3,4), (4,2)\} can be represented by

\[
\begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
1 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 \\
\end{pmatrix}
\]

Undirected graphs where there is an edge between vertices \( i \) and \( j \) are represented by symmetric matrices where both \( a_{i,j} \) and \( a_{j,i} \) are 1; otherwise both entries are 0. Entries on the main diagonal (e.g. \( a_{i,i} \)) are loops; edges/arcs that leave and terminate on the same vertex.

\[
\begin{pmatrix}
0 & 1 & 0 & 0 \\
1 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 \\
0 & 1 & 1 & 0 \\
\end{pmatrix}
\]

Alternately an upper right matrix can be used where entry \( a_{i,j} = 1 \) indicates an edge between vertices \( i \) and \( j \) with no direction attached.

\[
\begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 \\
\end{pmatrix}
\]

Finally if edges/arcs have weights or costs, then each entry \( a_{i,j} \) can be set to the value of the edge/arc.
Part 1: Using Mathematica, create a 9 x 9 adjacency matrix for the graph given by figure 8.10 on page 306. Since this is an undirected graph with edges (not arcs) there are two symmetric entries for each edge. Identify row/columns 1 – 9 with vertices a. – i.

If A is a square n by n matrix, the command \( \text{Sum}[A[[k]], \{k, 1, n\}] \) will sum the columns of A returning a list with the column sums (\( \text{Sum}[\text{Transpose}[A][[k]], \{k, 1, n\}] \) will return the row sums.)

Using Mathematica find the row sums and column sums for the matrix in Part 1

What information does the row sum (or column sum) give you? Hint: Konigsburg Bridge Problem.

Is the graph from Part 1 Eulerian (see page 300)?

Part 2 - Transitive Closures: If you take the matrix product of an adjacency matrix A with itself (i.e. in Mathematica, \( A \cdot A \)) the matrix obtained has for each entry the number of paths of length two from vertex i to vertex j. Recall from matrix multiplication that \( a_{i,j} = \sum_{k=1}^{n} a_{i,k} \times a_{k,j} \) and if for some k both \( a_{i,k} \) and \( a_{k,j} \) equal 1 then there is an arc from vertex i to vertex k and an arc from vertex k to vertex j hence a path of length 2 from vertex i to vertex j. Since you are summing over all k, you are counting all the paths of length 2.

Taking the matrix product of \( A \cdot A \) again with A (in Mathematica \( \text{MatrixPower}[A, 3] \)) for each entry you obtain the number of paths of length 3 from vertex i to j. Do this n times (assuming n vertices) and (component wise) sum all the matrices you obtain the transitive closure of the graph where each non-zero entry \( a_{i,j} \) indicates there is a path from vertex i to vertex j.

However, an adjacency matrix only uses 0’s and 1’s so instead using matrix multiplication, we use Boolean matrix multiplication where 1 times 1 is still 1 but 1 plus 1 is 1 (not 2) since we want our matrices to have only 0’s or 1’s for entries to indicate no-path/arc or path/arc between two vertices (and not the number).

Boolean multiplication of 0’s and 1’s is the same as algebraic multiplication (e.g. Mathematica’s \( \text{Times}[] \) command) but it can be implemented using Mathematica’s \( \text{Min}[] \) function which returns the smaller of its two arguments. Boolean addition being different from algebraic addition can be implemented using Mathematica’s \( \text{Max}[] \) function which return the larger of its two arguments (hence 1 plus 1 is 1, 1 plus 0 is 1, 0 plus 1 it 1 and 0 plus 0 is 0).

Thus we use the Boolean matrix expression \( \text{Inner}[\text{Min}, A, B, \text{Min}] \) where \( \text{Inner}[] \) is the generalized inner product function (you might want to check this out first - \( \text{Inner}[f, A, B, g] \) is the generalized inner product function where \( f \) takes the role of multiplication and \( g \) the role of addition.). In Mathematica the expression
returns a matrix $M$ whose entries $m_{i,j}$ equal $\max_{k=1}^{n} \min \{a_{i,k}, b_{k,j}\}$ (instead of $\sum_{k=1}^{n} a_{i,k} \times b_{k,j}$) the Boolean product. Essentially $m_{i,j}$ equals 1 if for some $k$ both $a_{i,k}$ and $a_{k,j}$ are 1 indicating a path from vertex $i$ to $k$ and a path from vertex $k$ to $j$.

Now if $A^k$ denotes the Boolean matrix product of $A$ with itself $k$ times (i.e. all path of length $k$), the transitive closure of $A$ is given by

$$\text{transitive closure of } A = A + A^2 + A^3 + \ldots + A^n$$

Of course the $+$ operation is Boolean matrix addition where each component of the sum $A+B$ is $\max \{a_{i,j}, b_{i,j}\}$. The effect is that the entry is 1 if either or both of its components are 1.

The module BooleanSum[] defined below defines a Boolean sum of two 0-1 matrices.

```mathematica
BooleanSum[A0_, B0_, n_]:= Module[{C0, i, j},
  (C0 = A0;
   For[i = 1, i <= n, i++,
    For[j = 1, j <= n, j++,
     C0[[i, j]] = Max[A0[[i, j]], B0[[i, j]]]]);
  C0]
```

So do the following. Using Mathematica compute the transitive closure the following 6 vertex directed graph

![Graph Diagram](image-url)
Part III – Vertex Cover: Formulate an integer program to find the minimum vertex cover for the graph in Figure 8.36 on page 337 then use Mathematica’s Minimize[] function to solve it.

Recall that a vertex cover of a graph G is a subset S of vertices V(G) such that every edge in G is incident with some element of S.

There are three parts to the integer programming model for the vertex cover problem

1. **Decision Variables:** \( x_i = \begin{cases} 1 & \text{if vertex } i \in S \\ 0 & \text{otherwise} \end{cases} \) for every vertex in V(G)

2. **Constraints:** \( x_i + x_j \geq 1 \) for every ij in E(G) – the edges of G
   
   \( x_i = 0 \) or \( x_i = 1 \) for every i in V(G)

3. **Objective Function:** Minimize \( z = \sum_{i \in V(G)} x_i \)

An integer programming model can be solved using Mathematica’s Minimize[] function

The syntax of Minimize[]: Minimize[\{f, cons\}, \{x, y, \ldots\}]: minimize function f subject to constraints cons.

**Example:** Find a minimal vertex cover for the following graph

Mathematica set up:

Minimize[{x1+x2+x3+x4, (x1+x2≥1) && (x2+x4≥1) && (x4+x3≥1) && (x1+x3≥1) && (x2+x3≥1) && (x1==0 || x1==1) && (x2==0 || x2==1) && (x3==0 || x3==1) && (x4==0 || x4==1)}, \{x1, x2, x3, x4\}]

Do the following

A. On a separate sheet of paper formulate an integer program to find a minimum vertex cover for Fig 8.36 on page 337 Identify the decision variables, constraints and objective function

B. Using Mathematica’s Minimize[] function implement and solve your integer program.
Do the Following: Formulate an integer program to find the minimum vertex cover for a 3 by 3 grid then use Mathematica’s `Minimize[]` function to solve it. Label the vertices $x_0, x_1, x_2, \ldots, x_8$.

\[
 \begin{array}{c}
 x_0--x_1--x_2 \\
 | \quad | \\
 x_3--x_4--x_5 \\
 | \quad | \\
 x_6--x_7--x_8 \\
\end{array}
\]

A. Identify the decision variables, constraints and objective function

Decision Variables:

Constraints

Objective Function

B. Using Mathematica’s `Minimize[]` function implement and solve your integer program.
Part IV Max-Min Flow Formulate an integer program to find the maximum flow for the graph in Figure 8.35 on page 337 then use Mathematica’s Maximize[] function to solve it.

1. Decision Variables: \( x_{i,j} \geq 0 \) is the flow through \( arc_{i,j} \)

2. Constraints: \( 0 \leq x_{i,j} \leq u_{i,j} \) where \( u_{i,j} \) is the capacity of \( arc_{i,j} \)

\[
\sum_{i} x_{i,j} = \sum_{k} x_{j,k} \quad \forall j \in V(G) - \{s,t\} \quad \text{- flow conservation constraint}
\]

3. Objective Function: Maximize \( \sum_{j} x_{s,j} \)

**Example 2:** Find the Maximum flow for the following directed graph

\[x_{1,2} + x_{3,2} = x_{2,4} \quad \text{(vertex 2 conservation constraint)}\]

\[x_{1,3} = x_{3,2} + x_{3,4} \quad \text{(vertex 3 conservation constraint)}\]

**Mathematica set up:**

```mathematica
Maximize[{x12 + x13, (0 \leq x12 \leq 2) && (0 \leq x13 \leq 4) && (0 \leq x32 \leq 3) && (0 \leq x34 \leq 3) && (0 \leq x24 \leq 3) && (x12 + x32 == x24) && (x13 == x32 + x34)}, {x12, x13, x32, x24, x34}, Integers]
```

Note the **Integers** directive.
Do the Following: Find the maximum flow for the graph in Figure 8.35 on page 337 (reproduced below).

A. Identify the decision variables (9), constraints (9 plus 4 conservation constraints) and objective function,

Variables

Constraints

Objective Function

B. Using Mathematica’s Minimize[] function implement and solve your integer program.