COSC 421/521 - Assignment 1: Flight Network Analysis

COSC 421/521 - Assignment 1: Flight Network Analysis

Student Name: Mahatav Arora
Student ID: 96593926
Partner Name: Yerdana Maulenbay **Student ID:** 24831786
Data Collection Date: [September 14th, 2025]

Question 0: Data Collection Date

On what date did you collect data?

Data was collected on: [**September 14th, 2025**] from FlightAware Api.

Question 1: Nodes and Edges

How many nodes and edges are in this graph?

```r
# Install & load igraph
if (!require(igraph)) {
  install.packages("igraph")
  library(igraph)
} else {
  library(igraph)
}
```

```
Loading required package: igraph
```

```
Attaching package: 'igraph'


The following objects are masked from 'package:stats':

    decompose, spectrum


The following object is masked from 'package:base':

    union
```

```r
# Load data
nodes <- read.csv("canadian_airports.csv", header=TRUE)
edges <- read.csv("flightnetwork.csv", header=TRUE)

# Fix column names
colnames(edges) <- c("from", "to")

# Ensure nodes has a 'name' column
if (!"name" %in% colnames(nodes)) {
  if ("code" %in% colnames(nodes)) {
    nodes$name <- nodes$code
  } else {
    nodes$name <- nodes[,1]
  }
}

# Create complete node list
all_airport_codes <- unique(c(edges$from, edges$to))
complete_nodes <- data.frame(name = all_airport_codes)

# Merge with original nodes
if ("city" %in% colnames(nodes)) {
  complete_nodes <- merge(complete_nodes, nodes, by = "name", all.x = TRUE)
}

# Build graph
airports <- graph_from_data_frame(d = edges, vertices = complete_nodes, directed = FALSE)

# Calculate nodes and edges
num_nodes <- vcount(airports)
num_edges <- ecount(airports)
```

```r
cat("Number of nodes:", num_nodes, "\n")
```

```
Number of nodes: 119
```

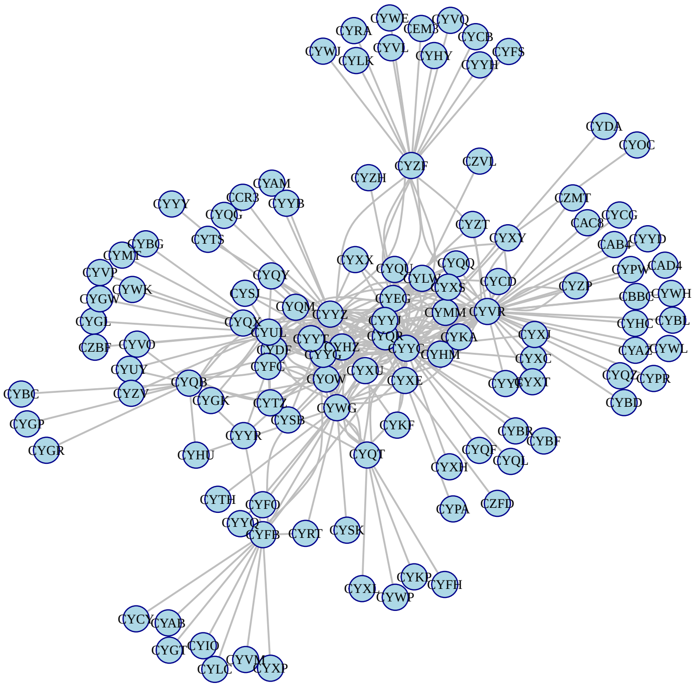
```r
cat("Number of edges:", num_edges, "\n")
```

```
Number of edges: 328
```

Q2. Plot of the undirected network graph

```r
set.seed(123) # For reproducible layout

plot(airports,
     vertex.label = V(airports)$name,
     vertex.size = 8,
     vertex.color = "lightblue",
     vertex.frame.color = "darkblue",
     vertex.label.color = "black",
     vertex.label.cex = 0.7,
     edge.color = "gray",
     edge.width = 1.5,
     layout = layout_with_fr,
     main = "Canadian Airport Network")
```

Canadian Airport Network



Q3. Degree of airports and mean degree

```
# Calculate degree statistics
deg <- degree(airports)
mean_degree <- mean(deg)

cat("Mean degree:", round(mean_degree, 2), "\n\n")
```

```
Mean degree: 5.51
```

```
# Most connected airports
sorted_deg <- sort(deg, decreasing = TRUE)
most_connected <- head(sorted_deg, 2)
```

```
# Least connected airports
non_zero_deg <- deg[deg > 0]
if (length(non_zero_deg) > 0) {
  least_connected <- head(sort(non_zero_deg), 2)
} else {
  least_connected <- head(sort(deg), 2)
}

cat("Two most connected airports:\n")
```

```
Two most connected airports:
```

```
for (i in 1:length(most_connected)) {
  cat("  ", names(most_connected)[i], ": ", most_connected[i], "connections\n")
}
```

```
   CYYC :  57 connections
   CYVR :  55 connections
```

```
cat("\nTwo least connected airports:\n")
```

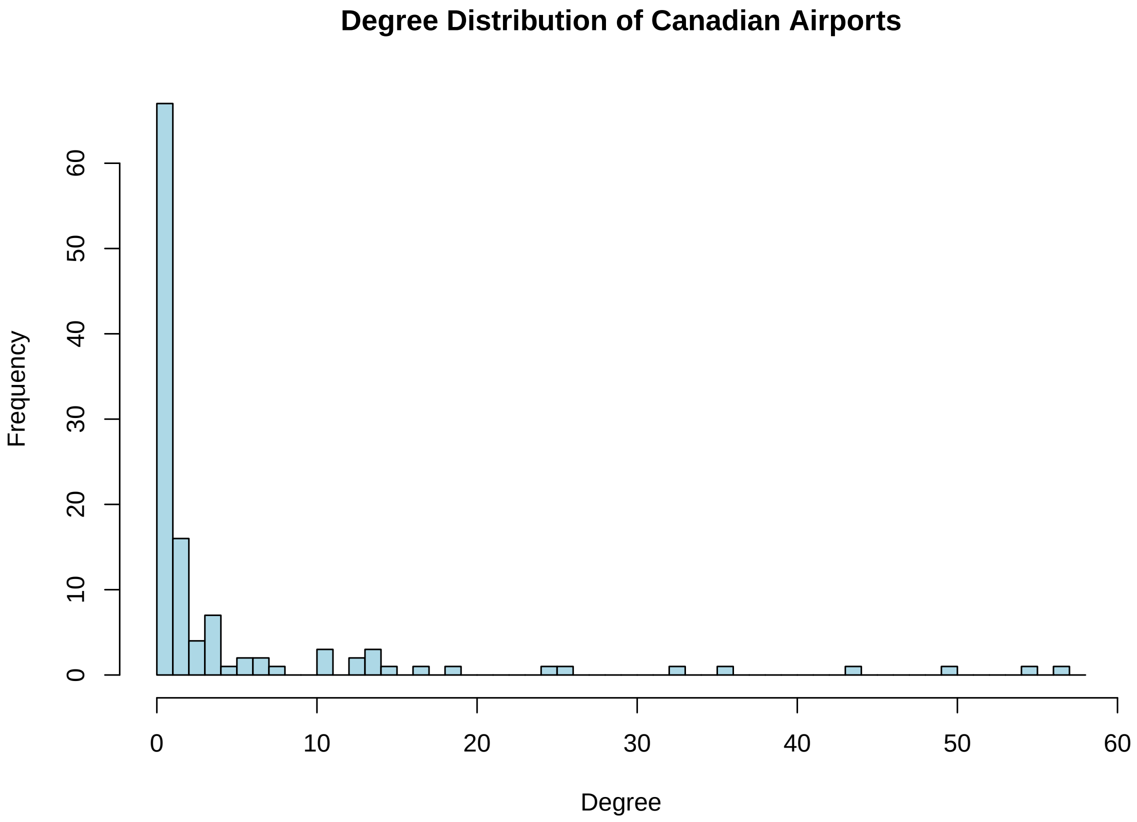
```

Two least connected airports:
```

```
for (i in 1:length(least_connected)) {
  cat("  ", names(least_connected)[i], ": ", least_connected[i], "connection(s)\n")
}
```

```
   CAB4 :  1 connection(s)
   CAC8 :  1 connection(s)
```

Q4. Histogram of the degree distribution

```
# Create degree histogram
hist(deg,
     main = "Degree Distribution of Canadian Airports",
     xlab = "Degree",
     ylab = "Frequency",
     col = "lightblue",
     border = "black",
     breaks = seq(0, max(deg)+1, by=1))
```



```
# Display degree sequence summary
cat("Degree sequence summary:\n")
```

```
Degree sequence summary:
```

```
cat("Min:", min(deg), "Max:", max(deg), "Mean:", round(mean(deg), 2), "\n")
```

```
Min: 1 Max: 57 Mean: 5.51
```

```r
cat("Degree sequence (first 10 values):", head(sort(deg, decreasing = TRUE), 10), "\n")
```

```
Degree sequence (first 10 values): 57 55 50 44 36 33 26 25 19 17
```

Q5. Adjacency matrix

```r
# Create adjacency matrix
A <- as.matrix(as_adjacency_matrix(airports))
is_symmetric <- isSymmetric(A)

cat("Adjacency matrix dimensions:", dim(A), "\n")
```

```
Adjacency matrix dimensions: 119 119
```

```r
cat("Is the adjacency matrix symmetric?", is_symmetric, "\n\n")
```

```
Is the adjacency matrix symmetric? TRUE
```

```r
# Show a small subset
if (nrow(A) >= 5) {
  cat("First 5x5 subset of adjacency matrix:\n")
  print(A[1:5, 1:5])
} else {
  cat("Full adjacency matrix:\n")
  print(A)
}
```

```
First 5x5 subset of adjacency matrix:
     CAB4 CAC8 CAD4 CBBC CCR3
CAB4    0    0    0    0    0
CAC8    0    0    0    0    0
CAD4    0    0    0    0    0
CBBC    0    0    0    0    0
CCR3    0    0    0    0    0
```

```r
cat("Network density:", round(graph.density(airports), 4), "\n")
```

```
Warning: `graph.density()` was deprecated in igraph 2.0.0.
i Please use `edge_density()` instead.
```

```
Network density: 0.0467
```

```r
cat("Is the graph connected?", is.connected(airports), "\n")
```

```
Warning: `is.connected()` was deprecated in igraph 2.0.0.
i Please use `is_connected()` instead.
```

```
Is the graph connected? TRUE
```

```r
# Components analysis
comp <- components(airports)
cat("Number of connected components:", comp$no, "\n")
```

```
Number of connected components: 1
```

```r
cat("Size of largest component:", max(comp$csize), "\n")
```

```
Size of largest component: 119
```

```r
# Diameter analysis
if (is.connected(airports)) {
  cat("Network diameter:", diameter(airports), "\n")
} else {
  giant_component <- induced_subgraph(airports, which(comp$membership == which.max(comp$csize
  cat("Diameter of largest component:", diameter(giant_component), "\n")
}
```

```
Network diameter: 5
```