# DS210 Project WriteUp

**Introduction:**

This project focuses on analyzing ego networks using graph theory and Rust's petgraph library. Ego networks are a type of social network where nodes represent individuals, and edges represent connections (e.g., friendships). The goal of this project is to process and analyze a network dataset to gain insights into its structure, connectivity, and underlying patterns.
The program begins by reading raw data from .edges files, which contain lists of connections between nodes. It then constructs an undirected graph and performs a series of analyses to compute key metrics such as:

- **Degree Distribution**: Here, it'll identify highly connected nodes (hubs) and the overall connectivity profile of the graph.
- **Separation Metrics**: Then, calculate mean, median, and standard deviation of path lengths to measure how tightly connected the network is.
- **Preliminary File Analysis**: This summarizes the dataset by counting connections, groups (circles), and node features.

The project is modular and includes:

- **main.rs:** Handles file input/output, graph construction, and includes test functions that test all main functionality
- **stats.rs:** Contains statistical functions to compute graph metrics

**Project Breakdown:**

### main.rs

Handles the main workflow of the project, including:

- **File Analysis**:
  - Analyzes .edges, .circles, and .feat files to count lines and summarize the dataset
- **Graph Loading**:
  - Constructs an undirected graph using the load_edges function, based on edge list files (went to office hours to get this working)
- **Graph Statistics**:
  - Computes degree distributions
  - Uses functions from stats.rs to calculate mean separation, standard deviation, and median separation (went to office hours to get this working)

### stats.rs

Implements statistical functions for analyzing graph connectivity:

1. **calculate_mean_separation:**
    - Computes the average shortest path length between all node pairs in the graph
    - Useful for understanding the overall connectivity of the graph (went to office hours to get this working)
2. **calculate_standard_deviation_separation:**
    - Calculates the variability of shortest path lengths
    - Identifies whether the graph has a uniform or widely varying node separation
3. **calculate_median_separation:**
    - Finds the middle value of shortest path lengths
    - This statistic is less sensitive to outliers compared to the mean

**Test Suite**

- Located in main.rs under the #[cfg(test)] module.
- Validates key functions, including:
    - Graph construction from .edges files
    - Correctness of degree distribution
    - Accuracy of mean, standard deviation, and median calculations

**Output:**

```
Running /Users/sathvikagade/Documents/DS210Project2/target/debug/DS210Project2
Preliminary Analysis:
Graph loaded with 333 nodes and 5038 edges
Degree distribution: {66: 2, 10: 16, 28: 8, 46: 5, 36: 5, 4: 23, 136: 1, 108: 1, 84: 3, 58: 2, 26: 12, 134: 1, 124: 1, 54: 4, 150: 1, 112: 3, 20: 12, 122: 1, 52: 4, 70:
2, 76: 3, 74: 2, 38: 7, 6: 18, 94: 1, 24: 12, 142: 1, 86: 2, 90: 1, 92: 2, 110: 1, 42: 4, 22: 7, 128: 3, 144: 1, 40: 6, 8: 11, 34: 8, 78: 1, 18: 13, 116: 1, 16: 15, 88:
1, 12: 18, 30: 13, 50: 4, 80: 1, 44: 3, 154: 1, 48: 4, 2: 29, 72: 3, 96: 1, 32: 9, 60: 2, 62: 1, 14: 16}
Degrees sorted from lowest to highest:
Degree: 2, Count: 29
Degree: 4, Count: 23
Degree: 6, Count: 18
Degree: 8, Count: 11
Degree: 10, Count: 16
Degree: 12, Count: 18
Degree: 14, Count: 16
Degree: 16, Count: 15
Degree: 18, Count: 13
Degree: 20, Count: 12
Degree: 22, Count: 7
Degree: 24, Count: 12
Degree: 26, Count: 12
Degree: 28, Count: 8
Degree: 30, Count: 13
Degree: 32, Count: 9
Degree: 34, Count: 8
Degree: 36, Count: 5
Degree: 38, Count: 7
Degree: 40, Count: 6
Degree: 42, Count: 4
Degree: 44, Count: 3
Degree: 46, Count: 5
Degree: 48, Count: 4
Degree: 50, Count: 4
Degree: 52, Count: 4
Degree: 54, Count: 4
Degree: 58, Count: 2
Degree: 60, Count: 2
Degree: 62, Count: 1
Degree: 66, Count: 2
Degree: 70, Count: 2
Degree: 72, Count: 3
Degree: 74, Count: 2
Degree: 76, Count: 3
Degree: 78, Count: 1
Degree: 80, Count: 1
Degree: 84, Count: 3
Degree: 86, Count: 2
Degree: 88, Count: 1
Degree: 90, Count: 1
```

```
Degree: 90, Count: 1
Degree: 92, Count: 2
Degree: 94, Count: 1
Degree: 96, Count: 1
Degree: 108, Count: 1
Degree: 110, Count: 1
Degree: 112, Count: 3
Degree: 116, Count: 1
Degree: 122, Count: 1
Degree: 124, Count: 1
Degree: 128, Count: 3
Degree: 134, Count: 1
Degree: 136, Count: 1
Degree: 142, Count: 1
Degree: 144, Count: 1
Degree: 150, Count: 1
Degree: 154, Count: 1
Mean separation: 3.75
Standard deviation of separation: 1.69
Median separation: 3.00
```

- **Preliminary Analysis**:
    - Validates that the input files are processed correctly
    - The program processes the 0.edges file, which contains a list of 5,038 edges (connections) in the network
    - The preliminary analysis verifies that the .edges file is readable and confirms the size of the network dataset
- **Graph Properties**:

    - Confirms the graph's structure, giving a sense of the graph's density and connectivity
- **Degree Distribution**:
    - Highlights the presence of hubs and less-connected nodes, typical of real-world networks like social media
        - **Skewed distribution**: Indicates a small number of hubs (high-degree nodes) and many less-connected nodes, typical in social networks
        - **Balanced distribution**: Suggests more uniform connectivity across nodes
    - The degree distribution maps the number of connections (degree) to the count of nodes with that degree
        - For ex: Degree 2, Count 29: 29 nodes have exactly 2 connections
- **Graph Statistics**:
    - **Mean Separation**: This computes the average connectivity across the graph. The mean shortest path length between all pairs of nodes is 3.75.
        - A smaller mean separation suggests a highly connected graph, while a larger value indicates more sparsely connected components
    - **Standard Deviation**: The standard deviation measures the variability in shortest path lengths.
        - A low standard deviation suggests that most nodes are separated by similar distances, therefore proving that it's a more uniform network
        - A high standard deviation implies that some nodes are very far apart while others are close, potentially highlighting clusters or outliers
    - **Median Separation**: The median shortest path length is 3.00, meaning half of all shortest paths are 3 steps or fewer.
        - The median is less sensitive to outliers than the mean, providing a more robust measure of typical separation

**<u>Analysis and Conclusion:</u>**

- **Degree Distribution**:
    - Highlights the graph's hubs (nodes with high degrees) and its sparsely connected nodes.
- **Mean Separation**:
    - Indicates the overall connectivity of the graph. Smaller values suggest a more tightly connected network
- **Standard Deviation**:
    - Captures the variability in shortest path lengths, which can reveal structural irregularities or clusters
- **Median Separation**:
    - Offers a robust measure of typical separation, less affected by outliers compared to the mean

This project provides a framework for analyzing ego networks. It focuses on degree distribution, separation metrics, and graph structure. Key takeaways include:

- **Network Insights**: The degree distribution highlights the presence of hub nodes, indicating a scale-free structure typical of social networks. Separation metrics reveal the network's connectivity, offering insights into clustering and bottlenecks
- **Practical Applications**: The analysis methods are applicable to not only social networks, but for other real-world problems