Source Code Documentation

Arham Shariff	EB21102022
Hamza Alam Hashmi	EB21102031
Riaz Akram	EB21102077

To execute the code for the Operational Research Projects in Xcode, follow these steps

Prerequisites:

Xcode installed on your macOS device

1. **Navigation To The Project Folder and open OperationalResearchProject.xcodeproj**
2. **Build and Run**
    - Select a target (e.g., a simulator or a physical device) from the target device menu in the Xcode toolbar. - Press `Cmd + R` or select `Product` -> `Run` to build and run the project.
3. **Explore the Simulations**
    - - Use the interface to input parameters for the simulations.
    - - Click the appropriate buttons (e.g., "Show Results") to trigger the simulations and display the results.
4. **View Results**
    - Explore the results displayed in the application interface, which may include tables, charts, or other visualizations depending on the simulation.
5. **Interact with the Application**
    - Experiment with different input parameters to see how the simulations behave.
    - Use any provided functionalities (e.g., fitness tests, data analysis) to further analyze the results.

By following these steps, you can execute and explore the Operational Research Projects in Xcode, gaining insights into queuing models, random number generation, and other simulations relevant to operational research.

Here's a documentation outline for your Xcode project files and folders:

5th Semester

1. **Queuing Model**
    a. **Model:**
        i. **Result**: Represents the result of a queuing simulation.
        ii. **FitTest**: Performs a fitness test on the queuing model.
    b. **QueuingView:** Manages the interface for the queuing model simulation inputs and results display.
    c. **QueuingViewModel:** Handles the logic for the queuing model simulation.
2. **Random Number Generator**
    a. **RandomNumberView:** Manages the interface for generating random numbers. -
    b. **RandomViewModel:** Handles the logic for generating random numbers.

6th Semester

1. **MM1 Priority**
    a. **Model**
        i. **Customer**`: Represents a customer in the M/M/1 priority queuing model.
        ii. **GrantChartData**`: Stores data for generating Gantt Chart.
    b. **MM1PriorityView**: Manages the interface for the M/M/1 priority queuing model simulation inputs and results display.
    c. **MM1PriorityViewModel**: Handles the logic for the M/M/1 priority queuing model simulation.

2. **LCG (Linear Congruential Generator)**
    a. **Model**
        i. **LCGRow**: Represents a row in the LCG table.
        ii. **LCGInput**`: Represents input parameters for the LCG calculation
    b. **LCGView**: Manages the interface for the LCG simulation inputs and results display.
    c. **LCGViewModel**: Handles the logic for the LCG simulation.

This structure outlines the key components of your project, organizing them by semester and functionality.