1. Yes, the application builds and runs correctly under a command-line environment that includes 'javac' and 'java'.   
   We ran the algorithm through five different network settings, taking graphic screenshots of the robot's greedy and approximation approaches for each test case.
2. **Compile the Program**1. **Compilation** :

    - Use the following command to compile all `.java` files:

javac --module-path "lib" --add-modules javafx.controls,javafx.fxml \*.java

2. **Execution**:

    - Run the program with:

      java --module-path "lib" --add-modules javafx.controls,javafx.fxml Main

1. We examined the robot's energy consumption across several network configurations using the Greedy and Approximation algorithms.  
     
   Greedy algorithm.  
   This algorithm always selects the nearest unvisited node, therefore it works well in dense or uniform layouts. However, because it does not take into account the global structure, it can result in inefficient pathways in irregular or dispersed networks, causing extra diversions.  
     
   Approximation Algorithm This method initially creates a Minimum Spanning Tree (MST), which connects all rendezvous places with the shortest overall distance. A Depth-First Search (DFS) is then used to construct a visiting path. It produces more balanced, global routes while avoiding the local traps of the greedy strategy.

**Observation:**

The Approximation algorithm using MST Preorder Traversal typically consumes less energy than the Greedy algorithm, as it considers the overall structure of the network. By following a minimum spanning tree, it guarantees a tour with energy cost no more than twice the optimal, offering a strong approximation.

However, in certain configurations — such as clustered or linear arrangements of nodes — the Greedy algorithm may perform similarly well,for example check the test case 4 since its locally optimal choices can align closely with the global optimum.

Dependency:

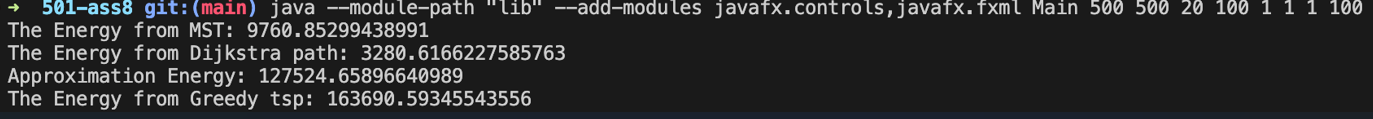
The relative performance depends on the topology of the network:

* In dense and irregularly spread networks, the MST-based approach usually leads to more efficient paths.
* In highly clustered layouts, the Greedy algorithm can sometimes be more or equally efficient, due to short intra-cluster distances.

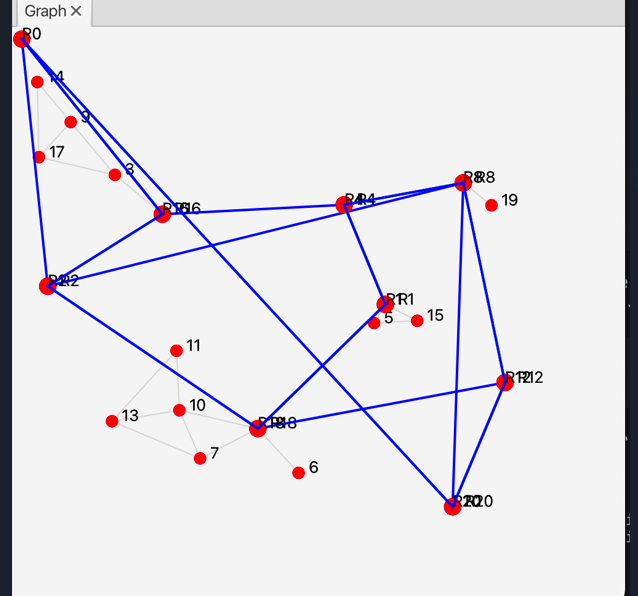
1. Typically, the robot uses more energy than the sensor nodes. This is because:  
   The robot's energy consumption increases as it goes, and the cost per unit distance is rather significant.  
   Sensor nodes send data using the shortest path algorithm, which often results in decreased energy consumption, even when numerous hops are involved.  
   Dependency: The output may vary depending on the network layout.  
   In dense networks, sensors can efficiently relay data through neighboring nodes, lowering energy costs.  
   When rendezvous spots are far apart or the robot's trip is long as in sparse networks the robot consumes significantly more energy.  
   In rare cases when sensors require significant multi-hop communication, their energy consumption may become comparable to the robot's, albeit this is exceptional.
2. Yes I did the extra credit
3. Gopisetty Gurudevi Lavanya

**Test case 1:**A screenshot of a graph

AI-generated content may be incorrect.



Test Case 2



A screen shot of a computer

AI-generated content may be incorrect.

**Test case 3:**

**A screen shot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Test case 4:**

**Test Case 5:**