324885417 aseelzatmy3@gmail.com

void Graph::loadGraph(const std::vector<std::vector<int>> &matrix)
The function iterates through the adjacency matrix and formats it into a readable string, where each row is enclosed in square brackets, and elements within the row are separated by commas.

Graph Graph::operator+(const Graph& other) const
The function enables the addition of two graphs by adding their adjacency matrices element-wise. It checks for size compatibility, initializes a result graph, performs the addition using nested loops, and returns the result. This implementation ensures that the adjacency matrices are of the same size and combines their edge information in a new graph.

Graph& Graph::operator+=(const Graph& other)
function enables the addition of another graph's adjacency matrix to the current graph's adjacency matrix in place. It checks for size compatibility, performs the addition using nested loops, and updates the current graph. This operator returns a reference to the updated graph

Graph Graph::operator+() const
The function provides a way to create a copy of a Graph object using the unary plus operator. It achieves this by invoking the copy constructor of the Graph class and returning the resulting copy. This operator does not modify the original graph and simply returns an identical new graph. This can be useful in scenarios where a non-modifying reference or temporary copy of a graph is needed.

Graph Graph::operator-(const Graph& other) const
The function enables the subtraction of another graph's adjacency matrix from the current graph's adjacency matrix, resulting in a new graph. It checks for size compatibility, initializes a result graph, performs the subtraction using nested loops, and returns the result.

The Graph::operator-= function overloads the subtraction assignment operator (-=) for the Graph class. This allows for subtracting the adjacency matrix of another graph directly from the current graph's adjacency matrix and updating the current graph in place.

Graph Graph::operator-() const
The implementation of the unary negation operator for the Graph class creates a new graph object with the same number of vertices and an adjacency matrix where each element is the negation of the corresponding element in the original graph's adjacency matrix. This operator overload allows you to use the unary - on a Graph object to get a new Graph with all edges' weights negated.

bool Graph::operator==(const Graph& other) const
The implementation of the equality operator for the Graph class compares two graphs by checking if they have the same number of vertices and if their adjacency matrices are identical. If both conditions are met, the graphs are considered equal; otherwise, they are not. This operator overload allows you to use the equality operator (==) to compare two Graph objects directly.

bool Graph::operator!=(const Graph& other) const
The implementation of the operator!= for the Graph class relies on the equality operator to check if

two graphs are unequal. By negating the result of the equality operator, it effectively determines if the graphs are not the same.

bool Graph::isContainedIn(const Graph& other) const
The isContainedIn method for the Graph class checks whether the current graph is contained within another graph. It first ensures that the current graph does not have more vertices than the other graph. Then, it verifies that all edges in the current graph are present and identical in the other graph. If both conditions are met, it returns true; otherwise, it returns false. This method effectively determines if the current graph can be considered a subgraph of the other graph based on vertex count and edge presence.

Graph& Graph::operator++()
The prefix increment operator for the Graph class increments every element in the adjacency matrix by 1.

Graph Graph::operator++(int)
The postfix increment operator for the Graph class increments every element in the adjacency matrix by 1 and returns a copy of the graph as it was before the increment operation. This is achieved by first creating a temporary copy of the current graph, applying the prefix increment operation to the current object, and then returning the previously saved state as a copy.

Graph& Graph::operator—()
The prefix decrement operator for the Graph class decrements every element in the adjacency matrix by 1. The method then returns a reference to the modified graph object.

Graph Graph::operator--(int)
The postfix decrement operator for the Graph class decrements every element in the adjacency matrix by 1 and returns a copy of the graph as it was before the decrement operation. This is achieved by first creating a temporary copy of the current graph, applying the prefix decrement operation to the current object, and then returning the previously saved state as a copy.

Graph Graph::operator*(int scalar) const
to multiply each edge weight in the graph by a scalar value and return a new graph with the scaled edge weights.

Graph Graph::operator*(const Graph& other) const
The multiplication operator (*) for the Graph class allows for performing matrix multiplication between two graphs representing adjacency matrices. It checks for compatibility between the matrices, initializes a result graph, performs matrix multiplication, and returns the resulting graph.

std::ostream& operator<<(std::ostream& os, Graph& graph)
The output operator (<<) for the Graph class enables printing the graph to an output stream by calling the printGraph() method and sending the generated string representation to the output stream.

Graph& Graph::operator/=(int scalar)
this function divides each element of the adjacency matrix of the graph by the given scalar value