PROJECT 3 EVALUATION RUBRIC

Program	**Description**	**Points**
Documentation	Main Comment Block contains: (author (1)and program description (1)).	2
	Comments have been added to each group of logically related statements **Comments are written above each user defined function to describe what the function does**	3
Style	Variable: • Meaningful variable names are used unless specified by the program description (1) • No global variable is used (1) Function • Meaningful function names are used (1) • In the client program, function prototypes declared above the main function and function definitions written after the main function (2)	5
	Indentation and white spaces are used to make the program easier to read. • All the decision statements are indented properly. • All the repetition statements (loops) are indented properly • Body of the functions are indented properly • Blank lines are used in front of each block of logically related statements	4
	Array and vector size should be declared as a constant.	1
Correctness	Program solves the assigned problem using data structure and methods described in project description.	35
	Program compiles without errors.	5
	Program executes without crashing.	5
	Program produces the correct output in table format as shown in the example program output. • Origin cities sorted, destination cities for each origin city sorted	40
TOTAL		100

Program requirements:

1. Define the flight record as a struct type. Put the definition in the header file type.h
    - Overload the operators ==, <, =, and << operators for this struct type.
    - Put the implementation of these operators, and any other methods you want, in type.cpp

2. Implement a FlightMap class, which has the following data and the following methods:
    - Data
        1. Number of cities served by the company
        2. list of cities served by the company
            - The STL vector is to be used for the list of cities served by the company**.**
        3. flight map implemented in the form of an adjacency list, e.g., array of lists.
            - The STL list needs to be used to implement each list
            - The array needs to be created dynamically. The actual size of the array is based on the number of cities served by the company. Therefore, the array needs to be defined as a pointer to the list of flight records.
            - It should be noted that this array is parallel to the array of cities, e.g., data item 2 above
    - Methods:
        - constructor(s) and destructor
            - default constructor
            - copy constructor
                - make sure to use new operator to allocate space for the flight map before copying the lists
            - destructor – releases memory space dynamically allocated
        - operations
            - read cities (cities.dat)
                - This method takes one parameter: the input file stream opened for the data file: "cities.dat"
                - The input file stream should be opened in the main function and passed in to this method as parameter. Do not open this specific file in the method itself
            - read flight information and build the adjacency list (flights.dat)
                - This is the code that builds the adjacency list with information from the flights.dat file.
                - Dynamically allocate space for the flight map pointer before start reading the flight records and build the adjacency list
            - Overloaded << operator that displays the flight information as shown above.