Project 1 – Big-Oh

The data structures that Team EagleEye have used for Project 1 are database, vectors, queues, and maps. Firstly, the database “Restaurants.db” stores the index, name, and distance of restaurants.

The vectors are used to store the names(QString), distances(float), purchases(struct), restaurant choices, menu items and its prices, of the restaurants from the database(“Restaurants.db).

Queues are used to store the restaurants in a specific order when “Initial List”, “Custom List”, “Choice+Number” options are chosen.

The maps were used in the STL in order to sort the distances.

void GetAllRestaurants(QVector<QString>& names, QVector<float> &distances) stores names and distances into the string vector and float vector that are passed in by reference. The method has a **Big-Oh of O(N).**

QVector<Restaurant> AllRestaurantInfo() stores and returns a vector of type Restaurant that contains all restaurants in the database(“Restaurants.db”).The method has a **Big-Oh of O(N^2)** because the algorithm goes through all the indexes and loops through all the indexes at each index.

bool database::SearchForName(QString key) searches for the restaurant with the name passed in. If restaurant is found, a Boolean of true is returned. If the restaurant is not found, a Boolean of false is returned. The method has a **Big-Oh of O(N)** as it goes through the indexes once to find the key**.**

bool database::SearchForNumber(int key) searches for the restaurant with the number passed in. If restaurant is found, a Boolean of true is returned. If the restaurant is not found, a Boolean of false is returned. The method has a **Big-Oh of O(N)** as it goes through the indexes once to find the key**.**

Restaurant database::GetRestaurant(QString name) gets restaurant number and distance from the name passed in. The method has a **Big-Oh of O(N)** because the algorithm goes through the index to find the specific restaurant.

bool AddMenuItem(QString item, float price, QString name) adds a menu item with the item, price, and name that were passed in. The method has a **Big-Oh of O(N)** as it loops through the query to find the number of the restaurant name that was passed in.

bool database::DelteMenuItem(QString item, QString name) removes a menu item from a restaurant name that was passed in. The method has a **Big-Oh of O(N)** as it loops through the query to find the number of the restaurant name that was passed in.

bool CheckMenuItem(QString name) checks if the menu item passed in is already in the restaurant menu. It is **O(N)** as it loops through the query to find the number of the restaurant name that was passed in.

bool database::ChangePrice(QString itemName, float newPrice) changes the price of a menu item to the price that was passed in. It has a **Big-Oh of O(1)** as it only has a simple if else statement to check the query.

bool database::AddRestaurant(Restaurant add) adds a new restaurant to the database. The method executes the query and then sets both restaurant columns to the same restaurant so the distance between them is zero. It then enters previously added restaurants distance from the new restaurants to the distances table and adds the restaurants menu items and prices to the menu table. The method has a **Big-Oh of O(N)** as it loops through the query once through, on 2 occasions.