**A:** Nearest Neighbor Algorithm

**B1:**

# view current location

# for(each package in packageList)

# for(each string in addressList

# return index of match for destination and current address

# if (hashTableDistance[matchIndex][matchIndex] < minimum

# load minimum package

**B2:** I used visual studio code. I used the Github integration for version control. I used the csv library to parse the csv files. I organized the program into different classes for organization. I used the Python debugger to help identify and fix bugs. I also used the Python interpreter.

**B3:** For the truck loading component the space-time complexity is *O(N^2)* because there are 2 nested “for” loops. For the parsing component, the space-time complexity is *O(N)* because it increases in processing power as the csv file gets larger.

**B4:** At the core, the program is very adaptable of sorting and loading packages. However, regarding the exceptions (some packages needing to be delivered with other packages, or needing to be on a specific truck, or needing to arrive by a certain deadline, etc.) the program would struggle to scale. Those exceptions were handled very specifically and not necessarily scalable. If the program were to be scaled, the logic would need to be upgraded to include these exceptions in a way that would apply in various ways.

**B5:** The software is easy to maintain because it is sorted into different classes. Each class primarily manages itself and is called in a easy-to-read way in the Main file. Even the exception handling in the truck loading method, is well documented and written in a non-clunky way. Most everything that could be stowed away in a helper method was to improve readability.

**B6:**  The hash table is good because it can store the key value and the pair. This allowed easy access to pull out a specific package without further iteration, increasing runtime complexity. However, a weakness was that in order to compare different packages, they need to be pulled out separately and analyzed which can make comparisons a bit more clunky than a list.

**C:** Complete

**C1:** Complete.

**C2:** Complete.

**D:** An object with instances stored in a hash table was used to store the package details.

**D1:** The object instance allows the data to be stored concisely in a way that makes sure the data stays together and cannot be mixed up. The hash table allows the package instances to be stored together in an organized manner. Using the package ID for the key value for each package, the programmer can pull out a specific package very easily without further iteration.