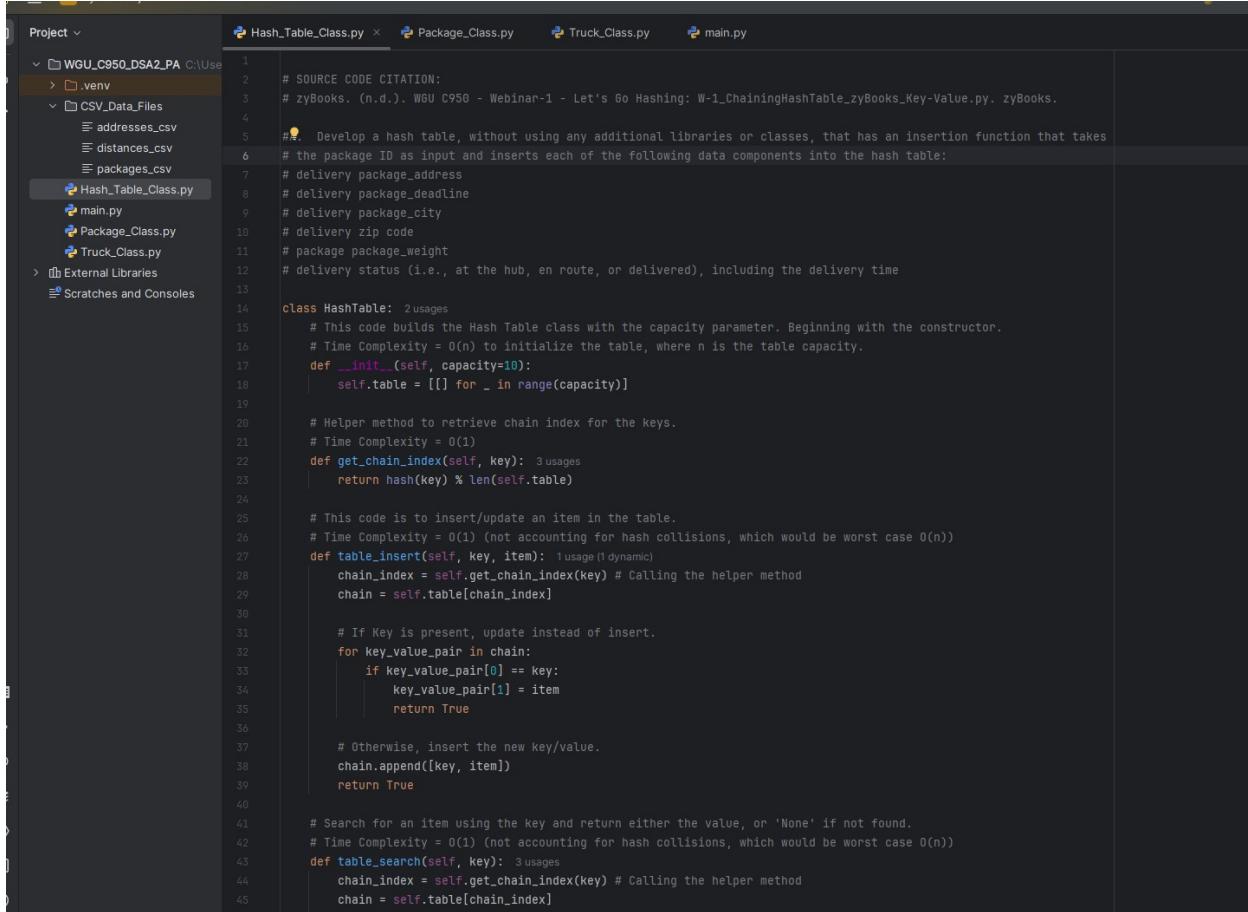


## WGUPS Write-Up

John McGinnes

02/24/25

## A. Hash Table



The screenshot shows a code editor interface with a project structure on the left and a code editor window on the right.

**Project Structure:**

- WGU\_C950\_DSA2\_PA
- .venv
- CSV\_Data\_Files
  - addresses.csv
  - distances.csv
  - packages.csv
- Hash\_Table\_Class.py
- main.py
- Package\_Class.py
- Truck\_Class.py
- External Libraries
- Scratches and Consoles

**Code Editor Content:**

```
1 # SOURCE CODE CITATION:
2 # zyBooks. (n.d.). WGU C950 - Webinar-1 - Let's Go Hashing: W-1_ChainingHashTable_zyBooks_Key-Value.py. zyBooks.
3
4 #💡 Develop a hash table, without using any additional libraries or classes, that has an insertion function that takes
5 # the package ID as input and inserts each of the following data components into the hash table:
6 # delivery package_address
7 # delivery package_deadline
8 # delivery package_city
9 # delivery zip code
10 # package package_weight
11 # delivery status (i.e., at the hub, en route, or delivered), including the delivery time
12
13 class HashTable: 2 usages
14     # This code builds the Hash Table class with the capacity parameter. Beginning with the constructor.
15     # Time Complexity = O(n) to initialize the table, where n is the table capacity.
16     def __init__(self, capacity=10):
17         self.table = [[] for _ in range(capacity)]
18
19     # Helper method to retrieve chain index for the keys.
20     # Time Complexity = O(1)
21     def get_chain_index(self, key): 3 usages
22         return hash(key) % len(self.table)
23
24     # This code is to insert/update an item in the table.
25     # Time Complexity = O(1) (not accounting for hash collisions, which would be worst case O(n))
26     def table_insert(self, key, item): 1 usage (dynamic)
27         chain_index = self.get_chain_index(key) # Calling the helper method
28         chain = self.table[chain_index]
29
30         # If Key is present, update instead of insert.
31         for key_value_pair in chain:
32             if key_value_pair[0] == key:
33                 key_value_pair[1] = item
34                 return True
35
36         # Otherwise, insert the new key/value.
37         chain.append([key, item])
38         return True
39
40     # Search for an item using the key and return either the value, or 'None' if not found.
41     # Time Complexity = O(1) (not accounting for hash collisions, which would be worst case O(n))
42     def table_search(self, key): 3 usages
43         chain_index = self.get_chain_index(key) # Calling the helper method
44         chain = self.table[chain_index]
```

The screenshot shows a code editor interface with a project navigation pane on the left and a code editor pane on the right.

**Project Navigation:**

- WGU\_C950\_DSA2\_PA
- CSV\_Data\_Files
  - addresses.csv
  - distances.csv
  - packages.csv
- Hash\_Table\_Class.py
- main.py
- Package\_Class.py
- Truck\_Class.py
- External Libraries
- Scratches and Consoles

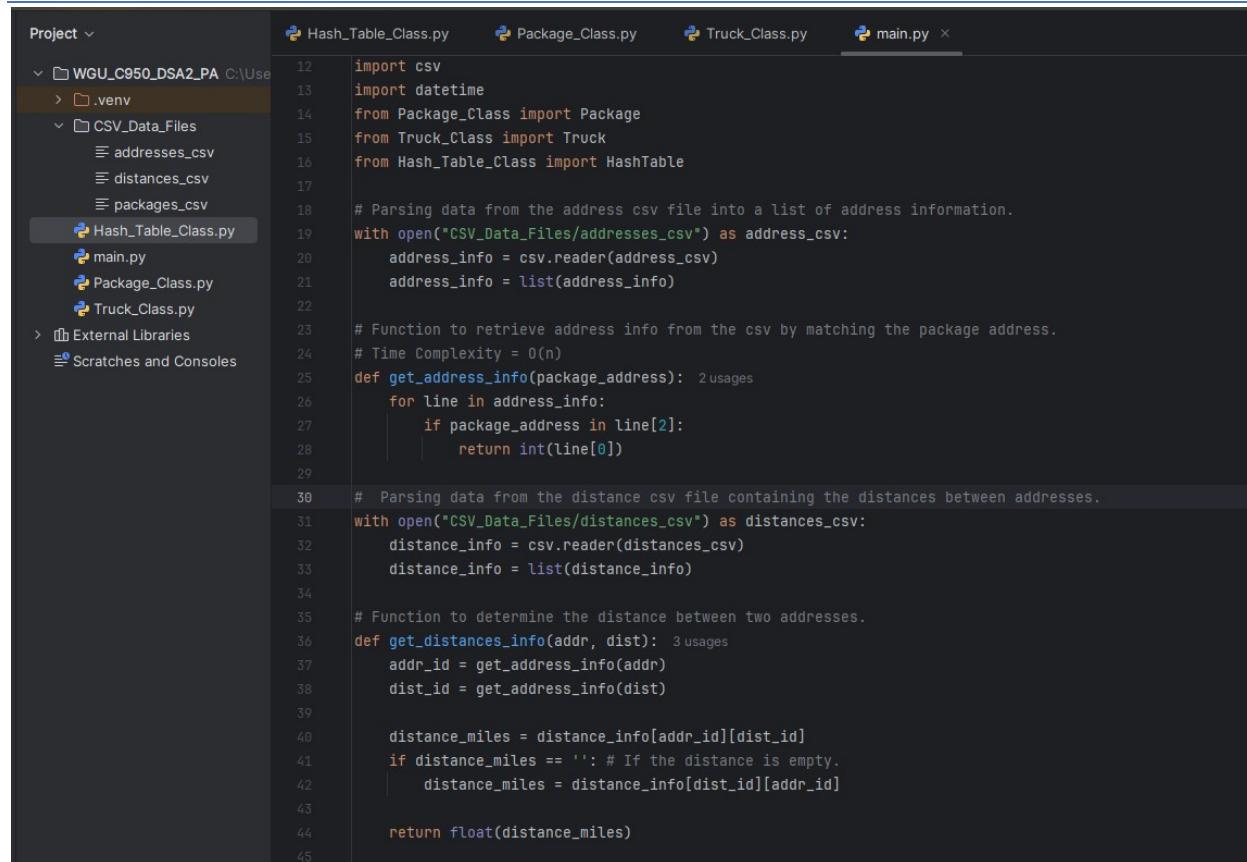
**Code Editor (Hash\_Table\_Class.py):**

```
14     class HashTable: 2 usages
15
16     # This code is to insert/update an item in the table.
17     # Time Complexity = O(1) (not accounting for hash collisions, which would be worst case O(n))
18     def table_insert(self, key, item): 1 usage (dynamic)
19         chain_index = self.get_chain_index(key) # Calling the helper method
20         chain = self.table[chain_index]
21
22         # If Key is present, update instead of insert.
23         for key_value_pair in chain:
24             if key_value_pair[0] == key:
25                 key_value_pair[1] = item
26
27             return True
28
29         # Otherwise, insert the new key/value.
30         chain.append([key, item])
31
32         return True
33
34
35
36
37     # Search for an item using the key and return either the value, or 'None' if not found.
38     # Time Complexity = O(1) (not accounting for hash collisions, which would be worst case O(n))
39     def table_search(self, key): 3 usages
40         chain_index = self.get_chain_index(key) # Calling the helper method
41         chain = self.table[chain_index]
42
43         # Look a key in the chain
44         for key_value_pair in chain:
45             if key_value_pair[0] == key:
46                 return key_value_pair[1]
47
48             return None
49
50
51
52
53     # Remove an item from the hash table using the key
54     # Time Complexity = O(1) (not accounting for hash collisions, which would be worst case O(n))
55     def table_remove(self, key):
56         chain_index = self.get_chain_index(key) # Calling the helper method
57         chain = self.table[chain_index]
58
59         # Search for the key to remove.
60         for key_value_pair in chain:
61             if key_value_pair[0] == key:
62                 chain.remove(key_value_pair)
63
64             return True # Indicates removal success.
65
66         return False # Returns false if no key was found.
```

## B. Look-Up Functions

```
186
187 # Objective B:
188 # Develop a look-up function that takes the following components...
189
190 # Function to find a package by the ID number and print the current status.
191 def package_search(package_id, current_time=datetime.timedelta(hours=17)): 7 usages
192     package = package_hash_table.table_search(package_id) # Time Complexity = O(1)
193     if package is None: # Time Complexity = O(1)
194         print(f"ERROR PACKAGE ID {package_id} INVALID.")
195         return
196
197     # For package #9, check the time and set the correct address
198     if package.pkg_id == 9:
199         # Time comparison: Package address is incorrect before 10:20
200         time_threshold = datetime.timedelta(hours=10, minutes=20)
201         if current_time < time_threshold:
202             package.pkg_address = "300 State St" # Incorrect address before 10:20
203             package.pkg_zip = "84103" # Incorrect ZIP code
204         else:
205             package.pkg_address = "410 S State St" # Correct address after 10:20
206             package.pkg_zip = "84111" # Correct ZIP code
207
208     package.set_pkg_status(current_time) # Time Complexity = O(1)
209     print(str(package)) # Time Complexity = O(1)
210
```

## C. Original Code



The screenshot shows a Python development environment with the following details:

- Project:** WGU\_C950\_DSA2\_PA
- Files:** Hash\_Table\_Class.py, Package\_Class.py, Truck\_Class.py, main.py
- Code Editor:** The main.py file is open, showing the following code:

```
12 import csv
13 import datetime
14 from Package_Class import Package
15 from Truck_Class import Truck
16 from Hash_Table_Class import HashTable
17
18 # Parsing data from the address csv file into a list of address information.
19 with open("CSV_Data_Files/addresses_csv") as address_csv:
20     address_info = csv.reader(address_csv)
21     address_info = list(address_info)
22
23 # Function to retrieve address info from the csv by matching the package address.
24 # Time Complexity = O(n)
25 def get_address_info(package_address): 2 usages
26     for line in address_info:
27         if package_address in line[2]:
28             return int(line[0])
29
30 # Parsing data from the distance csv file containing the distances between addresses.
31 with open("CSV_Data_Files/distances_csv") as distances_csv:
32     distance_info = csv.reader(distances_csv)
33     distance_info = list(distance_info)
34
35 # Function to determine the distance between two addresses.
36 def get_distances_info(addr, dist): 3 usages
37     addr_id = get_address_info(addr)
38     dist_id = get_address_info(dist)
39
40     distance_miles = distance_info[addr_id][dist_id]
41     if distance_miles == '': # If the distance is empty.
42         distance_miles = distance_info[dist_id][addr_id]
43
44     return float(distance_miles)
45
```

```

46 # Objective A:
47 # Develop a hash table, without using any additional libraries or classes...
48
49 # Function to add packages to the hash table using the packages csv data.
50 def add_packages(pkg_hash_table): 1 usage
51     with open("CSV_Data_Files/packages_csv") as package_csv:
52         packages_info = csv.reader(package_csv)
53
54         # Looping through each package and adding to the hash table.
55         # Time Complexity = O(n) (n = number of packages)
56         for package_row in packages_info:
57             package_id = int(package_row[0])
58             package_address = package_row[1]
59             package_city = package_row[2]
60             package_state = package_row[3]
61             package_zip = package_row[4]
62             package_deadline = package_row[5]
63             package_weight = package_row[6]
64             package_notes = package_row[7]
65             package_status = "At The WGUPS Hub"
66             package_departure_time = ""
67             package_arrival_time = ""
68
69             package = Package(package_id, package_address, package_city, package_state, package_zip, package_deadline,
70                               package_weight, package_notes, package_status, package_departure_time, package_arrival_time)
71
72             pkg_hash_table.table_insert(package_id, package) # Time Complexity = O(1)
73
74     # Initialize the hash table and add packages.
75     package_hash_table = HashTable()
76     add_packages(package_hash_table)
77

```

```

78 # Function to add packages to trucks based on the package notes.
79 def add_packages_to_trucks(): 1 usage
80     truck1_pkg_list = []
81     truck2_pkg_list = []
82     truck3_pkg_list = []
83     # Retrieving packages
84     packages = [package_hash_table.table_search(package_id) for package_id in range(1, 41)]
85     low_priority_packages = []
86
87     # Sorting the packages based on the specifications.
88     # Time Complexity = O(n) (n = number of packages)
89     for package_tmp in packages:
90
91         if package_tmp.pkg_notes == 'TRUCK2_DELAYED': # Truck 2 will handle all packages delayed until 9:05
92             truck2_pkg_list.append(package_tmp)
93         elif package_tmp.pkg_notes == 'TRUCK3_DELAYED': # Truck 3 will handle all packages delayed until 10:20
94             truck3_pkg_list.append(package_tmp)
95         elif package_tmp.pkg_notes == 'PART_OF_GROUP': # Any packages that are part of a group are delivered together
96             truck1_pkg_list.append(package_tmp)
97         elif package_tmp.pkg_deadline != 'EARLY_DEADLINE': # All packages with an early deadline go to Truck 1
98             truck1_pkg_list.append(package_tmp)
99         elif package_tmp.pkg_notes == 'TRUCK2_ONLY': # Packages required to shp via Truck 2
100             truck2_pkg_list.append(package_tmp)
101         else:
102             low_priority_packages.append(package_tmp)
103
104     # Adding standard (low) priority packages to Truck 1 based on the distance.
105     # Time Complexity = O(n^2) (due to nested loops)
106     for package_tmp in low_priority_packages:
107         for pkgs_assigned_to_truck in truck1_pkg_list: # Time Complexity = O(n)
108             if get_distances_info(pkgs_assigned_to_truck.pkg_address, package_tmp.pkg_address) < 2.0 and len(
109                 truck1_pkg_list) < 16: # Time Complexity = O(1)
110                 truck1_pkg_list.append(package_tmp)
111                 low_priority_packages.remove(package_tmp) # Time Complexity = O(n)
112                 break
113
114     # Adding standard (low) priority packages to Truck 2 based on the distance.
115     # Time Complexity = O(n^2) (due to nested loops)
116     for package_tmp in low_priority_packages:
117         for pkgs_assigned_to_truck in truck2_pkg_list: # Time Complexity = O(n)
118             if get_distances_info(pkgs_assigned_to_truck.pkg_address, package_tmp.pkg_address) < 2.0 and len(
119                 truck2_pkg_list) < 16: # Time Complexity = O(1)
120                 truck2_pkg_list.append(package_tmp)
121                 low_priority_packages.remove(package_tmp) # Time Complexity = O(n)
122                 break

```

```

133 # Time Complexity = O(N^2) due to the nested loop required to search for the nearest package.
134 def deliver_packages(truck): 3 usages
135     tmp_pkg_queue = []
136
137     # Initializing the package queue.
138     # Time Complexity = O(n) (n = number of packages)
139     for package in truck.packages:
140         tmp_pkg_queue.append(package)
141
142     truck.packages.clear() # Time Complexity = O(1)
143
144     # Nearest-Neighbor algorithm.
145     # Time Complexity = O(n^2) (n = number of packages)
146     while len(tmp_pkg_queue) > 0: # Time Complexity = O(n)
147         nearest_address = 999
148         nearest_package = None
149
150         for package in tmp_pkg_queue: # Time Complexity = O(n) (n = number of packages)
151             package_distance = get_distances_info(truck.current_address, package.pkg_address) # Time Complexity = O(n) (n = number of packages)
152
153             if package_distance <= nearest_address: # Time Complexity = O(1)
154                 nearest_address = package_distance
155                 nearest_package = package
156
157         truck.packages.append(nearest_package.pkg_id) # Time Complexity = O(1)
158         tmp_pkg_queue.remove(nearest_package) # Time Complexity = O(n) (n = number of packages)
159
160         nearest_package.depart = truck.last_departure
161
162         truck.current_address = nearest_package.pkg_address
163         truck.total_distance += nearest_address # Time Complexity = O(1)
164
165         truck.last_departure += datetime.timedelta(hours=nearest_address / 18) # Time Complexity = O(1)
166         nearest_package.arrive = truck.last_departure
167

```

## C1. Identification Information

The screenshot shows a code editor interface with a project navigation pane on the left and a code preview pane on the right.

**Project Navigation:**

- Project: WGU\_C950\_DSA2\_PA
- Subfolders: .venv, CSV\_Data\_Files (containing addresses\_csv, distances\_csv, packages\_csv), Hash\_Table\_Class.py, main.py, Package\_Class.py, Truck\_Class.py.

**Code Preview (main.py):**

```

1  # WGU C950 Data Structures and Algorithms 2
2  # Performance Assessment
3  # John McGinnes
4  # Student ID: 011700043
5
6  # Total Time Complexity Worst Case = O(n^2) with 'n' as the number of packages. This is due to the nested loops in the
7  # add_packages_to_trucks() and deliver_packages() methods. This was the lowest complexity I could achieve without
8  # importing external libraries for additional functionality.
9  # The Total Time Complexity Best Case = O(n) if all packages are sorted already and everything is easily assigned.
10 # The Total Time Complexity Average Case = O(n^2)
11

```

## C2. Process and Flow Comments

The screenshot shows a code editor with a dark theme. On the left is a file tree for a project named 'WGU\_C950\_DSA2\_PA' located at 'C:\Use'. The tree includes a '.venv' folder, 'CSV\_Data\_Files' containing 'addresses.csv', 'distances.csv', and 'packages.csv', and three Python files: 'Hash\_Table\_Class.py', 'main.py', and 'Package\_Class.py'. Below the tree is a 'External Libraries' section and a 'Scratches and Consoles' section. The main area contains the code for 'Hash\_Table\_Class.py' with line numbers 79 to 123. The code defines a function 'add\_packages\_to\_trucks()' which handles package assignments to three trucks based on their notes and delivery requirements. It uses lists for each truck's packages and iterates through a list of packages to assign them. The code includes several comments explaining the logic and time complexity of different parts of the algorithm.

```
def add_packages_to_trucks(): 1 usage
    truck1_pkg_list = []
    truck2_pkg_list = []
    truck3_pkg_list = []
    # Retrieving packages
    packages = [package_hash_table.table_search(package_id) for package_id in range(1, 41)]
    low_priority_packages = []

    # Sorting the packages based on the specifications.
    # Time Complexity = O(n) (n = number of packages)
    for package_tmp in packages:

        if package_tmp.pkg_notes == 'TRUCK2_DELAYED': # Truck 2 will handle all packages delayed until 9:05
            truck2_pkg_list.append(package_tmp)
        elif package_tmp.pkg_notes == 'TRUCK3_DELAYED': # Truck 3 will handle all packages delayed until 10:20
            truck3_pkg_list.append(package_tmp)
        elif package_tmp.pkg_notes == 'PART_OF_GROUP': # Any packages that are part of a group are delivered together
            truck1_pkg_list.append(package_tmp)
        elif package_tmp.pkg_deadline != 'EARLY_DEADLINE': # All packages with an early deadline go to Truck 1
            truck1_pkg_list.append(package_tmp)
        elif package_tmp.pkg_notes == 'TRUCK2_ONLY': # Packages required to ship via Truck 2
            truck2_pkg_list.append(package_tmp)
        else:
            low_priority_packages.append(package_tmp)

    # Adding standard (low) priority packages to Truck 1 based on the distance.
    # Time Complexity = O(n^2) (due to nested loops)
    for package_tmp in low_priority_packages:
        for pkgs_assigned_to_truck in truck1_pkg_list: # Time Complexity = O(n)
            if get_distances_info(pkgs_assigned_to_truck.pkg_address, package_tmp.pkg_address) < 2.0 and len(truck1_pkg_list) < 10: # Time Complexity = O(1)
                truck1_pkg_list.append(package_tmp)
                low_priority_packages.remove(package_tmp) # Time Complexity = O(n)
                break

    # Adding standard (low) priority packages to Truck 2 based on the distance.
    # Time Complexity = O(n^2) (due to nested loops)
    for package_tmp in low_priority_packages:
        for pkgs_assigned_to_truck in truck2_pkg_list: # Time Complexity = O(n)
            if get_distances_info(pkgs_assigned_to_truck.pkg_address, package_tmp.pkg_address) < 2.0 and len(truck2_pkg_list) < 10: # Time Complexity = O(1)
                truck2_pkg_list.append(package_tmp)
                low_priority_packages.remove(package_tmp) # Time Complexity = O(n)
                break
```

```

13     class HashTable: 2 usages
14         # This code builds the Hash Table class with the capacity parameter. Beginning with the constructor.
15         # Time Complexity = O(n) to initialize the table, where n is the table capacity.
16         def __init__(self, capacity=10):
17             self.table = [[ ] for _ in range(capacity)]
18
19             # Helper method to retrieve chain index for the keys.
20             # Time Complexity = O(1)
21             def get_chain_index(self, key): 3 usages
22                 return hash(key) % len(self.table)
23
24             # This code is to insert/update an item in the table.
25             # Time Complexity = O(1) (not accounting for hash collisions, which would be worst case O(n))
26             def table_insert(self, key, item): 1 usage (dynamic)
27                 chain_index = self.get_chain_index(key) # Calling the helper method
28                 chain = self.table[chain_index]
29
30                 # If Key is present, update instead of insert.
31                 for key_value_pair in chain:
32                     if key_value_pair[0] == key:
33                         key_value_pair[1] = item
34                         return True
35
36                 # Otherwise, insert the new key/value.
37                 chain.append([key, item])
38
39                 return True
40
41             # Search for an item using the key and return either the value, or 'None' if not found.
42             # Time Complexity = O(1) (not accounting for hash collisions, which would be worst case O(n))
43             def table_search(self, key): 3 usages
44                 chain_index = self.get_chain_index(key) # Calling the helper method
45                 chain = self.table[chain_index]
46
47                 # Look a key in the chain
48                 for key_value_pair in chain:
49                     if key_value_pair[0] == key:
50                         return key_value_pair[1]
51
52                 return None
53
54             # Remove an item from the hash table using the key
55             # Time Complexity = O(1) (not accounting for hash collisions, which would be worst case O(n))
56             def table_remove(self, key):
57                 chain_index = self.get_chain_index(key) # Calling the helper method

```

## D. Interface

```

166     nearest_package.arrive = truck.last_departure
167
168     # Objective 6:
169     #   Provide an interface for the user to view the status and info ...
170     def user_interface(): 1 usage
171         print()
172         print('WGUPS PARCEL TRACKING SERVICE')
173         print('=' * 60)
174         print('1. VIEW PACKAGE STATUSES FOR ALL TRUCKS')
175         print('2. VIEW PACKAGE STATUS BY TIME')
176         print('3. VIEW ALL PACKAGE STATUSES AT A SPECIFIC TIME')
177         print('4. VIEW TOTAL MILEAGE FOR ALL TRUCKS')
178         print('5. CLOSE PROGRAM')
179         print('=' * 60)
180
181         # Printing package display header.
182         def pkg_disp_header(): 7 usages
183             print('PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME')
184             print('=' * 115)
185

```

The screenshot shows a code editor interface with a dark theme. On the left is a project navigation pane titled "Project" showing a folder structure for "WGU\_C950\_DSA2\_PA". Inside are ".venv", "CSV\_Data\_Files" (containing "addresses\_csv", "distances\_csv", "packages\_csv"), "Hash\_Table\_Class.py" (selected), "main.py", "Package\_Class.py", and "Truck\_Class.py". Below these are "External Libraries" and "Scratches and Consoles". The main pane displays the "main.py" file with line numbers from 200 to 254. The code implements a menu system for managing package statuses across three trucks using Hash Tables.

```
200  class Main:
201      while True:
202          user_interface()
203
204          selection = input('SELECT OPTION 1-5: ')
205
206          if selection == '1': # View all package statuses after deliveries have been completed.
207              print()
208              print('ALL PACKAGE STATUSES: ')
209              print('TRUCK 1: ' + str(truck1.total_distance) + ' TOTAL MILES. ')
210              pkg_disp_header()
211
212              # Display package statuses for Truck 1.
213              for package_id in truck1.packages:
214                  package_search(package_id, datetime.timedelta(hours=17))
215
216              print()
217              print('TRUCK 2: ' + str(truck2.total_distance) + ' TOTAL MILES. ')
218              pkg_disp_header()
219
220              # Display package statuses for Truck 2.
221              for package_id in truck2.packages:
222                  package_search(package_id, datetime.timedelta(hours=17))
223
224              print()
225              print('TRUCK 3: ' + str(truck3.total_distance) + ' TOTAL MILES. ')
226              pkg_disp_header()
227
228              # Display package statuses for Truck 3.
229              for package_id in truck3.packages:
230                  package_search(package_id, datetime.timedelta(hours=17))
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246      elif selection == '2': # View the status of one package at a designated time.
247          try:
248              print()
249              package_id = int(input('ENTER PACKAGE ID: '))
250              package = package_hash_table.table_search(selection) # Time Complexity = O(1)
251          except ValueError:
252              print("INVALID PACKAGE ID.")
253              continue
254
```

Project WGU\_C950\_DSA2\_PA C:\Users\mcgin\PycharmProjects\WGU\_C950\_DSA2\_PA

```

200   class Main:
252     print("INVALID PACKAGE ID.")
253     continue
254
255     # Retrieve and display package status for a given time.
256     selection = input('ENTER TIME (HH:MM:SS) : ')
257     try:
258       (h, m, s) = selection.split(':') # Time Complexity = O(1)
259       print()
260       print(f'PACKAGE STATUS OF ID {package_id} AT ({h}:{m}:{s})')
261       pkg_disp_header()
262       package_search(package_id, datetime.timedelta(hours=int(h), minutes=int(m), seconds=int(s))) # Time Complexity = O(1)
263     except ValueError:
264       print("INVALID TIME FORMAT.")

265
266   elif selection == '3': # View the status for all packages at a designated time.
267     try:
268       print()
269       selection = input('ENTER TIME (HH:MM:SS) : ')
270       (h, m, s) = selection.split(':') # Time Complexity = O(1)
271       print()
272       print(f'DISPLAYING ALL PACKAGE STATUSES AT ({h}:{m}:{s})')
273       print()
274       print('TRUCK 1: ' + str(truck1.total_distance) + ' TOTAL MILES. ')
275       pkg_disp_header()

276       # Display all package statuses for Truck 1 at the specified time.
277       for package_id in truck1.packages:
278         package_search(package_id, datetime.timedelta(hours=int(h), minutes=int(m), seconds=int(s)))

279       print()
280       print('TRUCK 2: ' + str(truck2.total_distance) + ' TOTAL MILES. ')
281       pkg_disp_header()

282       # Display all package statuses for Truck 2 at the specified time.
283       for package_id in truck2.packages:
284         package_search(package_id, datetime.timedelta(hours=int(h), minutes=int(m), seconds=int(s)))

285       print()
286       print('TRUCK 3: ' + str(truck3.total_distance) + ' TOTAL MILES. ')
287       pkg_disp_header()

288       # Display all package statuses for Truck 3 at the specified time.
289       for package_id in truck3.packages:
290         package_search(package_id, datetime.timedelta(hours=int(h), minutes=int(m), seconds=int(s)))
291
292
293
294
295
296   except ValueError:
297     print("INVALID TIME FORMAT.")

298
299   elif selection == '4': # View total mileage for all the trucks.
300     miles_combined = truck1.total_distance + truck2.total_distance + truck3.total_distance
301     print()
302     print('TOTAL MILEAGE OF ALL TRUCK COMBINED: ' + str(miles_combined) + ' TOTAL MILES. ')
303
304
305   elif selection == '5': # Quit program and end loop.
306     print("EXITING PROGRAM...")
307     exit()

```

```

296
297
298
299
300
301
302
303
304
305
306
307

```

Run main (1) ×

```

C:\Users\mcgin\PycharmProjects\wgu_c950_data_structures_and_algorithms_ii\.venv\Scripts\python.exe C:\Users\mcgin\PycharmProjects\WGU_C950_DSA2_PA\main.py

WGUPS PARCEL TRACKING SERVICE
=====
1. VIEW PACKAGE STATUSES FOR ALL TRUCKS
2. VIEW PACKAGE STATUS BY TIME
3. VIEW ALL PACKAGE STATUSES AT A SPECIFIC TIME
4. VIEW TOTAL MILEAGE FOR ALL TRUCKS
5. CLOSE PROGRAM
=====
SELECT OPTION 1-5:

```

## D1. First Status Check

---

```
Run main (1) x
C:\Users\mcgin\PycharmProjects\wgu_c950_data_structures_and_algorithms_ii\.venv\Scripts\python.exe C:/Users\mcgin\PycharmProjects\WGU_C950_DSA2_PA\main.py

WGUPS PARCEL TRACKING SERVICE
=====
1. VIEW PACKAGE STATUSES FOR ALL TRUCKS
2. VIEW PACKAGE STATUS BY TIME
3. VIEW ALL PACKAGE STATUSES AT A SPECIFIC TIME
4. VIEW TOTAL MILEAGE FOR ALL TRUCKS
5. CLOSE PROGRAM
=====
SELECT OPTION 1-5: 3

ENTER TIME (HH:MM:SS) : 9:00:00

DISPLAYING ALL PACKAGE STATUSES AT (9:00:00)

TRUCK 1: 27.9 TOTAL MILES.
PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
14, 4300 S 1300 E, Millcreek, UT, 84117, 10:30 AM, 88, PART_OF_GROUP, Delivered to Location, 8:00:00, 8:06:20
34, 4580 S 2300 E, Holladay, UT, 84117, 10:30 AM, 2, , Delivered to Location, 8:06:20, 8:13:00
16, 4580 S 2300 E, Holladay, UT, 84117, 10:30 AM, 88, PART_OF_GROUP, Delivered to Location, 8:13:00, 8:13:00
15, 4580 S 2300 E, Holladay, UT, 84117, 9:00 AM, 4, PART_OF_GROUP, Delivered to Location, 8:13:00, 8:13:00
29, 1330 2100 S, Salt Lake City, UT, 84106, 10:30 AM, 2, , Delivered to Location, 8:13:00, 8:29:40
2, 2530 S 500 E, Salt Lake City, UT, 84106, EARLY_DEADLINE, 44, , Delivered to Location, 8:29:40, 8:35:00
1, 195 W Oakland Ave, Salt Lake City, UT, 84115, 10:30 AM, 21, , Delivered to Location, 8:35:00, 8:40:00
40, 380 W 2880 S, Salt Lake City, UT, 84115, 10:30 AM, 45, , Delivered to Location, 8:40:00, 8:43:40
20, 3595 Main St, Salt Lake City, UT, 84115, 10:30 AM, 37, PART_OF_GROUP, Delivered to Location, 8:43:40, 8:49:00
19, 177 W Price Ave, Salt Lake City, UT, 84115, EARLY_DEADLINE, 37, PART_OF_GROUP, Delivered to Location, 8:49:00, 8:50:40
31, 3365 S 900 W, Salt Lake City, UT, 84119, 10:30 AM, 1, , Delivered to Location, 8:50:40, 8:59:40
13, 2010 W 500 S, Salt Lake City, UT, 84104, 10:30 AM, 2, PART_OF_GROUP, En Route to Location, 8:59:40, 9:19:00
5, 410 S State St, Salt Lake City, UT, 84111, EARLY_DEADLINE, 5, , At the WGUPS Hub, 9:19:00, 9:29:40
37, 410 S State St, Salt Lake City, UT, 84111, 10:30 AM, 2, , At the WGUPS Hub, 9:29:40, 9:29:40
8, 300 State St, Salt Lake City, UT, 84103, EARLY_DEADLINE, 9, , At the WGUPS Hub, 9:29:40, 9:33:00
30, 300 State St, Salt Lake City, UT, 84103, 10:30 AM, 1, , At the WGUPS Hub, 9:33:00, 9:33:00
```

```

Project <--> Version Control
Run main (1) x
Project Hash_Table_Class.py Package_Class.py Truck_Class.py main.py addresses_csv packages_csv
=====
8, 300 State St, Salt Lake City, UT, 84103, EARLY_DEADLINE, 9, , At the WGUPS Hub, 9:29:40, 9:33:00
30, 300 State St, Salt Lake City, UT, 84103, 10:30 AM, 1, , At the WGUPS Hub, 9:33:00, 9:33:00

=====
TRUCK 2: 35.599999999999994 TOTAL MILES.
PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
24, 5025 State St, Murray, UT, 84107, EARLY_DEADLINE, 7, , At the WGUPS Hub, 9:05:00, 9:13:00
25, 5383 S 900 East #104, Salt Lake City, UT, 84117, 10:30 AM, 7, TRUCK2_DELAYED, At the WGUPS Hub, 9:13:00, 9:18:40
22, 6351 South 900 East, Murray, UT, 84121, EARLY_DEADLINE, 2, , At the WGUPS Hub, 9:18:40, 9:23:00
33, 2530 S 500 E, Salt Lake City, UT, 84106, EARLY_DEADLINE, 1, , At the WGUPS Hub, 9:23:00, 9:43:00
28, 2835 Main St, Salt Lake City, UT, 84115, EARLY_DEADLINE, 7, TRUCK2_DELAYED, At the WGUPS Hub, 9:43:00, 9:46:40
4, 380 W 2880 S, Salt Lake City, UT, 84115, EARLY_DEADLINE, 4, , At the WGUPS Hub, 9:46:40, 9:50:00
32, 3365 S 900 W, Salt Lake City, UT, 84119, EARLY_DEADLINE, 1, TRUCK2_DELAYED, At the WGUPS Hub, 9:50:00, 9:55:40
17, 3148 S 1100 W, Salt Lake City, UT, 84119, EARLY_DEADLINE, 2, , At the WGUPS Hub, 9:55:40, 9:57:40
6, 3060 Lester St, West Valley City, UT, 84119, 10:30 AM, 88, TRUCK2_DELAYED, At the WGUPS Hub, 9:57:40, 10:02:00
36, 2300 Parkway Blvd, West Valley City, UT, 84119, EARLY_DEADLINE, 88, TRUCK2_ONLY, At the WGUPS Hub, 10:02:00, 10:07:20
18, 1488 4800 S, Salt Lake City, UT, 84123, EARLY_DEADLINE, 6, TRUCK2_ONLY, At the WGUPS Hub, 10:07:20, 10:20:40
10, 600 E 900 South, Salt Lake City, UT, 84105, EARLY_DEADLINE, 1, , At the WGUPS Hub, 10:20:40, 10:54:20
38, 410 S State St, Salt Lake City, UT, 84111, EARLY_DEADLINE, 9, TRUCK2_ONLY, At the WGUPS Hub, 10:54:20, 11:00:20
3, 233 Canyon Rd, Salt Lake City, UT, 84103, EARLY_DEADLINE, 2, TRUCK2_ONLY, At the WGUPS Hub, 11:00:20, 11:03:40

=====
TRUCK 3: 35.00000000000001 TOTAL MILES.
PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
21, 3595 Main St, Salt Lake City, UT, 84115, EARLY_DEADLINE, 3, , At the WGUPS Hub, 10:30:00, 10:36:40
26, 5383 S 900 East #104, Salt Lake City, UT, 84117, EARLY_DEADLINE, 25, , At the WGUPS Hub, 10:36:40, 10:50:00
11, 2600 Taylorsville Blvd, Salt Lake City, UT, 84118, EARLY_DEADLINE, 1, , At the WGUPS Hub, 10:50:00, 11:06:20
23, 5100 South 2700 West, Salt Lake City, UT, 84118, EARLY_DEADLINE, 5, , At the WGUPS Hub, 11:06:20, 11:07:40
35, 1060 Dalton Ave S, Salt Lake City, UT, 84104, EARLY_DEADLINE, 88, , At the WGUPS Hub, 11:07:40, 11:30:40
27, 1060 Dalton Ave S, Salt Lake City, UT, 84104, EARLY_DEADLINE, 5, , At the WGUPS Hub, 11:30:40, 11:30:40
39, 2010 W 500 S, Salt Lake City, UT, 84104, EARLY_DEADLINE, 9, , At the WGUPS Hub, 11:30:40, 11:36:00
9, 300 State St, Salt Lake City, UT, 84103, EARLY_DEADLINE, 2, TRUCK3_DELAYED, At the WGUPS Hub, 11:36:00, 11:50:00
7, 1330 2100 S, Salt Lake City, UT, 84106, EARLY_DEADLINE, 8, , At the WGUPS Hub, 11:50:00, 12:07:40
12, 3575 W Valley Central Station bus Loop, West Valley City, UT, 84119, EARLY_DEADLINE, 1, , At the WGUPS Hub, 12:07:40, 12:26:40

=====
WGUPS PARCEL TRACKING SERVICE
=====
1. VIEW PACKAGE STATUSES FOR ALL TRUCKS
2. VIEW PACKAGE STATUS BY TIME
3. VIEW ALL PACKAGE STATUSES AT A SPECIFIC TIME
4. VIEW TOTAL MILEAGE FOR ALL TRUCKS
5. CLOSE PROGRAM
=====
```

## D2. Second Status Check

---

```
Project ▾ Run main (1) ▾
WGUPS PARCEL TRACKING SERVICE
=====
1. VIEW PACKAGE STATUSES FOR ALL TRUCKS
2. VIEW PACKAGE STATUS BY TIME
3. VIEW ALL PACKAGE STATUSES AT A SPECIFIC TIME
4. VIEW TOTAL MILEAGE FOR ALL TRUCKS
5. CLOSE PROGRAM
=====
SELECT OPTION 1-5: 3

ENTER TIME (HH:MM:SS) : 10:00:00

DISPLAYING ALL PACKAGE STATUSES AT (10:00:00)

TRUCK 1: 27.9 TOTAL MILES.
PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
14, 4300 S 1300 E, Millcreek, UT, 84117, 10:30 AM, 88, PART_OF_GROUP, Delivered to Location, 8:00:00, 8:00:20
34, 4580 S 2300 E, Holladay, UT, 84117, 10:30 AM, 2, , Delivered to Location, 8:06:20, 8:13:00
16, 4580 S 2300 E, Holladay, UT, 84117, 10:30 AM, 88, PART_OF_GROUP, Delivered to Location, 8:13:00, 8:13:00
15, 4580 S 2300 E, Holladay, UT, 84117, 9:00 AM, 4, PART_OF_GROUP, Delivered to location, 8:13:00, 8:13:00
29, 1550 2100 S, Salt Lake City, UT, 84106, 10:30 AM, 2, , Delivered to Location, 8:13:00, 8:29:40
2, 2530 S 500 E, Salt Lake City, UT, 84106, EARLY_DEADLINE, 44, , Delivered to Location, 8:29:40, 8:35:00
1, 195 W Oakland Ave, Salt Lake City, UT, 84115, 10:30 AM, 21, , Delivered to Location, 8:35:00, 8:40:00
40, 380 W 2800 S, Salt Lake City, UT, 84115, 10:30 AM, 45, , Delivered to Location, 8:40:00, 8:43:40
20, 3595 Main St, Salt Lake City, UT, 84115, 10:30 AM, 37, PART_OF_GROUP, Delivered to Location, 8:43:40, 8:49:00
19, 177 W Price Ave, Salt Lake City, UT, 84115, EARLY_DEADLINE, 37, PART_OF_GROUP, Delivered to Location, 8:49:00, 8:50:40
31, 3365 S 900 W, Salt Lake City, UT, 84119, 10:30 AM, 1, , Delivered to Location, 8:50:40, 8:59:40
13, 2010 W 500 S, Salt Lake City, UT, 84104, 10:30 AM, 2, PART_OF_GROUP, Delivered to Location, 8:59:40, 9:19:00
5, 410 S State St, Salt Lake City, UT, 84111, EARLY_DEADLINE, 5, , Delivered to Location, 9:19:00, 9:29:40
37, 410 S State St, Salt Lake City, UT, 84111, 10:30 AM, 2, , Delivered to Location, 9:29:40, 9:29:40
8, 300 State St, Salt Lake City, UT, 84103, EARLY_DEADLINE, 9, , Delivered to Location, 9:29:40, 9:33:00
30, 300 State St, Salt Lake City, UT, 84103, 10:30 AM, 1, , Delivered to Location, 9:33:00, 9:33:00

TRUCK 2: 35.599999999999994 TOTAL MILES.
PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
24, 5025 State St, Murray, UT, 84107, EARLY_DEADLINE, 7, , Delivered to Location, 9:05:00, 9:13:00
25, 5383 S 900 East #104, Salt Lake City, UT, 84117, 10:30 AM, 7, TRUCK2_DELAYED, Delivered to Location, 9:13:00, 9:18:40
```

```

Project  Version control
Run  main (1) x
File  Project  Hash_Table_Class.py  Package_Class.py  Truck_Class.py  main.py x  addresses_csv  packages_csv
Run  main (1) x
... 3, 410 S State St, Salt Lake City, UT, 84111, EARLY_DEADLINE, 0, , Delivered to Location, 9:17:00, 9:27:40
37, 410 S State St, Salt Lake City, UT, 84111, 10:30 AM, 2, , Delivered to Location, 9:29:40, 9:29:40
8, 300 State St, Salt Lake City, UT, 84103, EARLY_DEADLINE, 9, , Delivered to Location, 9:29:40, 9:33:00
30, 300 State St, Salt Lake City, UT, 84103, 10:30 AM, 1, , Delivered to Location, 9:33:00, 9:33:00

TRUCK 2: 35.599999999999994 TOTAL MILES.
PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
24, 5025 State St, Murray, UT, 84107, EARLY_DEADLINE, 7, , Delivered to Location, 9:05:00, 9:13:00
25, 5383 S 900 East #104, Salt Lake City, UT, 84117, 10:30 AM, 7, TRUCK2_DELAYED, Delivered to Location, 9:13:00, 9:18:40
22, 6351 South 900 East, Murray, UT, 84121, EARLY_DEADLINE, 2, , Delivered to Location, 9:18:40, 9:23:00
33, 2530 S 500 E, Salt Lake City, UT, 84106, EARLY_DEADLINE, 3, , Delivered to Location, 9:23:00, 9:43:00
28, 2855 Main St, Salt Lake City, UT, 84115, EARLY_DEADLINE, 7, TRUCK2_DELAYED, Delivered to Location, 9:43:00, 9:46:40
4, 380 W 2880 S, Salt Lake City, UT, 84115, EARLY_DEADLINE, 4, , Delivered to Location, 9:46:40, 9:50:00
32, 3365 S 900 W, Salt Lake City, UT, 84119, EARLY_DEADLINE, 1, TRUCK2_DELAYED, Delivered to Location, 9:50:00, 9:55:40
17, 3148 S 1100 W, Salt Lake City, UT, 84119, EARLY_DEADLINE, 2, , Delivered to Location, 9:55:40, 9:57:40
6, 3600 Lester St, West Valley City, UT, 84119, 10:30 AM, 88, TRUCK2_DELAYED, En Route to Location, 9:57:40, 10:02:00
36, 2300 Parkway Blvd, West Valley City, UT, 84119, EARLY_DEADLINE, 88, TRUCK2_ONLY, At the WGUPS Hub, 10:02:00, 10:07:20
18, 1488 4800 S, Salt Lake City, UT, 84123, EARLY_DEADLINE, 6, TRUCK2_ONLY, At the WGUPS Hub, 10:07:20, 10:20:40
10, 600 E 900 South, Salt Lake City, UT, 84105, EARLY_DEADLINE, 1, , At the WGUPS Hub, 10:20:40, 10:54:20
38, 410 S State St, Salt Lake City, UT, 84111, EARLY_DEADLINE, 9, TRUCK2_ONLY, At the WGUPS Hub, 10:54:20, 11:00:20
3, 233 Canyon Rd, Salt Lake City, UT, 84103, EARLY_DEADLINE, 2, TRUCK2_ONLY, At the WGUPS Hub, 11:00:20, 11:03:40

TRUCK 3: 35.000000000000001 TOTAL MILES.
PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
21, 3595 Main St, Salt Lake City, UT, 84115, EARLY_DEADLINE, 3, , At the WGUPS Hub, 10:30:00, 10:36:40
20, 5383 S 900 East #104, Salt Lake City, UT, 84117, EARLY_DEADLINE, 25, , At the WGUPS Hub, 10:36:40, 10:50:00
11, 2600 Taylorsville Blvd, Salt Lake City, UT, 84118, EARLY_DEADLINE, 1, , At the WGUPS Hub, 10:50:00, 11:06:20
23, 5100 South 2700 West, Salt Lake City, UT, 84118, EARLY_DEADLINE, 5, , At the WGUPS Hub, 11:06:20, 11:07:40
35, 1060 Dalton Ave S, Salt Lake City, UT, 84104, EARLY_DEADLINE, 88, , At the WGUPS Hub, 11:07:40, 11:30:40
27, 1060 Dalton Ave S, Salt Lake City, UT, 84104, EARLY_DEADLINE, 5, , At the WGUPS Hub, 11:30:40, 11:30:40
39, 2010 W 500 S, Salt Lake City, UT, 84104, EARLY_DEADLINE, 9, , At the WGUPS Hub, 11:30:40, 11:56:00
9, 300 State St, Salt Lake City, UT, 84103, EARLY_DEADLINE, 2, TRUCK3_DELAYED, At the WGUPS Hub, 11:36:00, 11:50:00
7, 1330 2100 S, Salt Lake City, UT, 84106, EARLY_DEADLINE, 8, , At the WGUPS Hub, 11:50:00, 12:07:40
12, 3575 W Valley Central Station bus Loop, West Valley City, UT, 84119, EARLY_DEADLINE, 1, , At the WGUPS Hub, 12:07:40, 12:26:40

WGUPS PARCEL TRACKING SERVICE
=====
1. VIEW PACKAGE STATUSES FOR ALL TRUCKS
2. VIEW PACKAGE STATUS BY TIME
3. VIEW ALL PACKAGE STATUSES AT A SPECIFIC TIME
4. VIEW TOTAL MILEAGE FOR ALL TRUCKS
5. CLOSE PROGRAM

```

## D3. Third Status Check

---

Project ▾ Hash\_Table\_Class.py Package\_Class.py Truck\_Class.py main.py × addresses\_csv packages\_csv

Run main (1) ×

```

ENTER TIME (HH:MM:SS) : 12:30:00

DISPLAYING ALL PACKAGE STATUSES AT (12:30:00)

TRUCK 1: 27.9 TOTAL MILES.

PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
14, 4300 S 1300 E, Millcreek, UT, 84117, 10:30 AM, 88, PART_OF_GROUP, Delivered to Location, 8:00:00, 8:06:20
34, 4580 S 2300 E, Holladay, UT, 84117, 10:30 AM, 2, , Delivered to Location, 8:06:20, 8:13:00
16, 4580 S 2300 E, Holladay, UT, 84117, 10:30 AM, 88, PART_OF_GROUP, Delivered to Location, 8:13:00, 8:13:00
15, 4580 S 2300 E, Holladay, UT, 84117, 9:00 AM, 4, PART_OF_GROUP, Delivered to Location, 8:13:00, 8:13:00
29, 1330 2100 S, Salt Lake City, UT, 84106, 10:30 AM, 2, , Delivered to Location, 8:13:00, 8:29:40
2, 2530 S 500 E, Salt Lake City, UT, 84106, EARLY_DEADLINE, 44, , Delivered to Location, 8:29:40, 8:35:00
1, 195 W Oakland Ave, Salt Lake City, UT, 84115, 10:30 AM, 21, , Delivered to Location, 8:35:00, 8:40:00
40, 380 W 2880 S, Salt Lake City, UT, 84115, 10:30 AM, 45, , Delivered to Location, 8:40:00, 8:43:40
20, 3595 Main St, Salt Lake City, UT, 84115, 10:30 AM, 37, PART_OF_GROUP, Delivered to Location, 8:43:40, 8:49:00
19, 177 W Price Ave, Salt Lake City, UT, 84115, EARLY_DEADLINE, 37, PART_OF_GROUP, Delivered to Location, 8:49:00, 8:50:40
31, 3365 S 900 W, Salt Lake City, UT, 84119, 10:30 AM, 1, , Delivered to Location, 8:50:40, 8:59:40
13, 2010 W 500 S, Salt Lake City, UT, 84104, 10:30 AM, 2, PART_OF_GROUP, Delivered to Location, 8:59:40, 9:19:00
5, 410 S State St, Salt Lake City, UT, 84111, EARLY_DEADLINE, 5, , Delivered to Location, 9:19:00, 9:29:40
37, 410 S State St, Salt Lake City, UT, 84111, 10:30 AM, 2, , Delivered to Location, 9:29:40, 9:29:40
8, 300 State St, Salt Lake City, UT, 84103, EARLY_DEADLINE, 9, , Delivered to Location, 9:29:40, 9:33:00
30, 300 State St, Salt Lake City, UT, 84103, 10:30 AM, 1, , Delivered to Location, 9:33:00, 9:33:00

TRUCK 2: 35.59999999999994 TOTAL MILES.

PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
24, 5025 State St, Murray, UT, 84107, EARLY_DEADLINE, 7, , Delivered to Location, 9:05:00, 9:13:00
25, 5383 S 900 East #104, Salt Lake City, UT, 84117, 10:30 AM, 7, TRUCK2_DELAYED, Delivered to Location, 9:13:00, 9:18:40
22, 6351 South 900 East, Murray, UT, 84121, EARLY_DEADLINE, 2, , Delivered to Location, 9:18:40, 9:23:00
33, 2530 S 500 E, Salt Lake City, UT, 84106, EARLY_DEADLINE, 1, , Delivered to Location, 9:23:00, 9:43:00
28, 2835 Main St, Salt Lake City, UT, 84115, EARLY_DEADLINE, 7, TRUCK2_DELAYED, Delivered to Location, 9:43:00, 9:46:40
4, 380 W 2880 S, Salt Lake City, UT, 84115, EARLY_DEADLINE, 4, , Delivered to Location, 9:46:40, 9:50:00
32, 3365 S 900 W, Salt Lake City, UT, 84119, EARLY_DEADLINE, 1, TRUCK2_DELAYED, Delivered to Location, 9:50:00, 9:55:40
17, 3148 S 1100 W, Salt Lake City, UT, 84119, EARLY_DEADLINE, 2, , Delivered to Location, 9:55:40, 9:57:40
6, 3060 Lester St, West Valley City, UT, 84119, 10:30 AM, 88, TRUCK2_DELAYED, Delivered to Location, 9:57:40, 10:02:00
36, 2300 Parkway Blvd, West Valley City, UT, 84119, EARLY_DEADLINE, 88, TRUCK2_ONLY, Delivered to Location, 10:02:00, 10:07:20
18, 1488 4800 S, Salt Lake City, UT, 84123, EARLY_DEADLINE, 6, TRUCK2_ONLY, Delivered to Location, 10:07:20, 10:20:40
10, 600 E 900 South, Salt Lake City, UT, 84105, EARLY_DEADLINE, 1, , Delivered to Location, 10:20:40, 10:54:20
38, 410 S State St, Salt Lake City, UT, 84111, EARLY_DEADLINE, 9, TRUCK2_ONLY, Delivered to Location, 10:54:20, 11:00:20
3, 233 Canyon Rd, Salt Lake City, UT, 84103, EARLY_DEADLINE, 2, TRUCK2_ONLY, Delivered to Location, 11:00:20, 11:03:40

TRUCK 3: 35.00000000000001 TOTAL MILES.

```

```
Project ▾ Hash_Table_Class.py Package_Class.py Truck_Class.py main.py × addresses_csv packages_csv
Run main (1) ×

8, 300 State St, Salt Lake City, UT, 84103, EARLY_DEADLINE, 9, , Delivered to Location, 9:29:40, 9:33:00
30, 300 State St, Salt Lake City, UT, 84103, 10:30 AM, 1, , Delivered to Location, 9:33:00, 9:33:00

TRUCK 2: 35.599999999999994 TOTAL MILES.
PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
24, 5025 State St, Murray, UT, 84107, EARLY_DEADLINE, 7, , Delivered to Location, 9:05:00, 9:13:00
25, 5383 S 900 East #104, Salt Lake City, UT, 84117, 10:30 AM, 7, TRUCK2_DELAYED, Delivered to Location, 9:13:00, 9:18:40
22, 6551 South 900 East, Murray, UT, 84121, EARLY_DEADLINE, 2, , Delivered to Location, 9:18:40, 9:23:00
33, 2530 S 500 E, Salt Lake City, UT, 84106, EARLY_DEADLINE, 1, , Delivered to Location, 9:23:00, 9:43:00
28, 2835 Main St, Salt Lake City, UT, 84115, EARLY_DEADLINE, 7, TRUCK2_DELAYED, Delivered to Location, 9:43:00, 9:46:40
4, 380 W 2880 S, Salt Lake City, UT, 84115, EARLY_DEADLINE, 4, , Delivered to Location, 9:46:40, 9:50:00
32, 3365 S 900 W, Salt Lake City, UT, 84119, EARLY_DEADLINE, 1, TRUCK2_DELAYED, Delivered to Location, 9:58:00, 9:55:40
17, 3148 S 1100 W, Salt Lake City, UT, 84119, EARLY_DEADLINE, 2, , Delivered to Location, 9:55:40, 9:57:40
6, 3060 Lester St, West Valley City, UT, 84119, 10:30 AM, 88, TRUCK2_DELAYED, Delivered to Location, 9:57:40, 10:02:00
36, 2300 Parkway Blvd, West Valley City, UT, 84119, EARLY_DEADLINE, 88, TRUCK2_ONLY, Delivered to Location, 10:02:00, 10:07:20
18, 1488 4800 S, Salt Lake City, UT, 84123, EARLY_DEADLINE, 6, TRUCK2_ONLY, Delivered to Location, 10:07:20, 10:20:40
10, 600 E 900 South, Salt Lake City, UT, 84105, EARLY_DEADLINE, 1, , Delivered to Location, 10:20:40, 10:54:20
38, 410 S State St, Salt Lake City, UT, 84111, EARLY_DEADLINE, 9, TRUCK2_ONLY, Delivered to Location, 10:54:20, 11:00:20
3, 233 Canyon Rd, Salt Lake City, UT, 84103, EARLY_DEADLINE, 2, TRUCK2_ONLY, Delivered to Location, 11:00:20, 11:03:40

TRUCK 3: 35.000000000000001 TOTAL MILES.
PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
21, 3595 Main St, Salt Lake City, UT, 84115, EARLY_DEADLINE, 3, , Delivered to Location, 10:30:00, 10:30:40
26, 5383 S 900 East #104, Salt Lake City, UT, 84117, EARLY_DEADLINE, 25, , Delivered to Location, 10:36:40, 10:50:00
11, 2600 Taylorsville Blvd, Salt Lake City, UT, 84118, EARLY_DEADLINE, 1, , Delivered to Location, 10:50:00, 11:06:20
23, 5100 South 2700 West, Salt Lake City, UT, 84118, EARLY_DEADLINE, 5, , Delivered to Location, 11:06:20, 11:07:40
35, 1060 Dalton Ave S, Salt Lake City, UT, 84104, EARLY_DEADLINE, 88, , Delivered to Location, 11:07:40, 11:30:40
27, 1060 Dalton Ave S, Salt Lake City, UT, 84104, EARLY_DEADLINE, 5, , Delivered to Location, 11:30:40, 11:30:40
39, 2010 W 500 S, Salt Lake City, UT, 84104, EARLY_DEADLINE, 9, , Delivered to Location, 11:30:40, 11:36:00
9, 410 S State St, Salt Lake City, UT, 84111, EARLY_DEADLINE, 2, TRUCK3_DELAYED, Delivered to Location, 11:36:00, 11:50:00
7, 1330 2100 S, Salt Lake City, UT, 84106, EARLY_DEADLINE, 8, , Delivered to Location, 11:50:00, 12:07:40
12, 3575 W Valley Central Station bus Loop, West Valley City, UT, 84119, EARLY_DEADLINE, 1, , Delivered to Location, 12:07:40, 12:26:40

W6UPS PARCEL TRACKING SERVICE
=====
1. VIEW PACKAGE STATUSES FOR ALL TRUCKS
2. VIEW PACKAGE STATUS BY TIME
3. VIEW ALL PACKAGE STATUSES AT A SPECIFIC TIME
4. VIEW TOTAL MILEAGE FOR ALL TRUCKS
5. CLOSE PROGRAM
```

## E. Screenshot of Code Execution

```

Project: PythonProject2 Version control: main (1) 
File: Hash_Table_Class.py Package_Class.py Truck_Class.py main.py

200     class Main:
201         for package_id in truck3.packages:
202             package_search(package_id, datetime.timedelta(hours=int(h), minutes=int(m), seconds=int(s)))
203     except ValueError:
204         print("INVALID TIME FORMAT.")
205
206     elif selection == '4': # View total mileage for all the trucks.
207         miles_combined = truck1.total_distance + truck2.total_distance + truck3.total_distance
208         print()
209         print('TOTAL MILEAGE OF ALL TRUCK COMBINED: ' + str(miles_combined) + ' TOTAL MILES.')
210
211     elif selection == '5': # Quit program and end loop.
212         print('EXITING PROGRAM...')
213         exit()
214
215
216 Run: main (1) 
C:\Users\mcgin\PycharmProjects\wgu_c950_data_structures_and_algorithms_ii\.venv\Scripts\python.exe C:\Users\mcgin\PycharmProjects\WGU_C950_DSA2_PA\main.py
WGUPS PARCEL TRACKING SERVICE
=====
1. VIEW PACKAGE STATUSES FOR ALL TRUCKS
2. VIEW PACKAGE STATUS BY TIME
3. VIEW ALL PACKAGE STATUSES AT A SPECIFIC TIME
4. VIEW TOTAL MILEAGE FOR ALL TRUCKS
5. CLOSE PROGRAM
=====
SELECT OPTION 1-5: 4
TOTAL MILEAGE OF ALL TRUCK COMBINED: 96.5 TOTAL MILES.

WGUPS PARCEL TRACKING SERVICE
=====
1. VIEW PACKAGE STATUSES FOR ALL TRUCKS
2. VIEW PACKAGE STATUS BY TIME
3. VIEW ALL PACKAGE STATUSES AT A SPECIFIC TIME
4. VIEW TOTAL MILEAGE FOR ALL TRUCKS
WGU C950 DSA2 PA > main.py
307:19 CRLF UTF-8 4 spaces Python 3.13 (wgu_c950_da_structures_and_algorithms_ii)

Project: PythonProject2 Version control: main (1) 
File: Hash_Table_Class.py Package_Class.py Truck_Class.py main.py

miles_combined = truck1.total_distance + truck2.total_distance + truck3.total_distance

Run: main (1) 
=====
SELECT OPTION 1-5: 1
=====
ALL PACKAGE STATUSES:
TRUCK 1: 27.9 TOTAL MILES.
PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
14, 4300 S 1300 E, Millcreek, UT, 84117, 10:30 AM, 88, PART_OF_GROUP, Delivered to Location, 8:00:00, 8:06:20
34, 4580 S 2300 E, Holladay, UT, 84117, 10:30 AM, 2, , Delivered to Location, 8:06:20, 8:13:00
16, 4580 S 2300 E, Holladay, UT, 84117, 10:30 AM, 88, PART_OF_GROUP, Delivered to Location, 8:13:00, 8:13:00
15, 4580 S 2300 E, Holladay, UT, 84117, 9:00 AM, 4, PART_OF_GROUP, Delivered to Location, 8:13:00, 8:13:00
29, 1330 2100 S, Salt Lake City, UT, 84106, 10:30 AM, 2, , Delivered to Location, 8:13:00, 8:29:40
2, 2530 S 500 E, Salt Lake City, UT, 84106, EARLY_DEADLINE, 44, , Delivered to Location, 8:29:40, 8:35:00
1, 195 W Oakland Ave, Salt Lake City, UT, 84115, 10:30 AM, 21, , Delivered to Location, 8:35:00, 8:40:00
40, 380 W 2800 S, Salt Lake City, UT, 84115, 10:30 AM, 45, , Delivered to Location, 8:40:00, 8:43:40
20, 3595 Main St, Salt Lake City, UT, 84115, 10:30 AM, 37, PART_OF_GROUP, Delivered to Location, 8:43:40, 8:49:00
19, 177 W Price Ave, Salt Lake City, UT, 84115, EARLY_DEADLINE, 37, PART_OF_GROUP, Delivered to Location, 8:49:00, 8:56:40
31, 3365 S 900 W, Salt Lake City, UT, 84119, 10:30 AM, 1, , Delivered to Location, 8:50:40, 8:59:40
13, 2010 W 500 S, Salt Lake City, UT, 84104, 10:30 AM, 2, PART_OF_GROUP, Delivered to Location, 8:59:40, 9:19:00
5, 410 S State St, Salt Lake City, UT, 84111, EARLY_DEADLINE, 5, , Delivered to Location, 9:19:00, 9:29:40
37, 410 S State St, Salt Lake City, UT, 84111, 10:30 AM, 2, , Delivered to Location, 9:29:40, 9:29:40
8, 300 State St, Salt Lake City, UT, 84103, EARLY_DEADLINE, 9, , Delivered to Location, 9:29:40, 9:33:00
30, 380 State St, Salt Lake City, UT, 84103, 10:30 AM, 1, , Delivered to Location, 9:33:00, 9:33:00
=====
TRUCK 2: 35.59999999999994 TOTAL MILES.
PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
24, 5625 State St, Murray, UT, 84107, EARLY_DEADLINE, 7, , Delivered to Location, 9:05:00, 9:13:00
25, 5383 S 900 East #104, Salt Lake City, UT, 84117, 10:30 AM, 7, TRUCK2_DELAYED, Delivered to Location, 9:13:00, 9:18:40
22, 6351 South 900 East, Murray, UT, 84121, EARLY_DEADLINE, 2, , Delivered to Location, 9:18:40, 9:23:00
33, 2530 S 500 E, Salt Lake City, UT, 84106, EARLY_DEADLINE, 1, , Delivered to Location, 9:23:00, 9:43:00
28, 2835 Main St, Salt Lake City, UT, 84115, EARLY_DEADLINE, 7, TRUCK2_DELAYED, Delivered to Location, 9:43:00, 9:46:40
4, 380 W 2800 S, Salt Lake City, UT, 84115, EARLY_DEADLINE, 4, , Delivered to Location, 9:46:40, 9:50:00
32, 3365 S 900 W, Salt Lake City, UT, 84119, EARLY_DEADLINE, 1, TRUCK2_DELAYED, Delivered to Location, 9:50:00, 9:55:40
17, 3148 S 1100 W, Salt Lake City, UT, 84119, EARLY_DEADLINE, 2, , Delivered to Location, 9:55:40, 9:57:40
1, 3040 Laramie St, Salt Lake City, UT, 84119, 10:30 AM, 60, TRUCK2_DELAYED, Delivered to Location, 9:57:40, 10:00:00

```

```

Project  Hash_Table_Class.py Package_Class.py Truck_Class.py main.py
Run main()  miles_combined = truck1.total_distance + truck2.total_distance + truck3.total_distance

30, 300 State St, Salt Lake City, UT, 84103, 10:30 AM, 1, , Delivered to Location, 9:33:00, 9:33:00
TRUCK 2: 35.599999999999994 TOTAL MILES.
PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
24, 5025 State St, Murray, UT, 84107, EARLY_DEADLINE, 7, , Delivered to Location, 9:05:00, 9:13:00
25, 5383 S 900 East #104, Salt Lake City, UT, 84117, 10:30 AM, 7, TRUCK2_DELAYED, Delivered to Location, 9:13:00, 9:18:40
22, 6351 South 900 East, Murray, UT, 84121, EARLY_DEADLINE, 2, , Delivered to Location, 9:18:40, 9:23:00
33, 2530 S 500 E, Salt Lake City, UT, 84106, EARLY_DEADLINE, 1, , Delivered to Location, 9:23:00, 9:43:00
28, 2835 Main St, Salt Lake City, UT, 84115, EARLY_DEADLINE, 7, TRUCK2_DELAYED, Delivered to Location, 9:43:00, 9:46:40
4, 300 W 2800 S, Salt Lake City, UT, 84115, EARLY_DEADLINE, 4, , Delivered to Location, 9:46:40, 9:50:00
32, 3365 S 900 W, Salt Lake City, UT, 84119, EARLY_DEADLINE, 1, TRUCK2_DELAYED, Delivered to Location, 9:50:00, 9:55:00
17, 3148 S 1100 W, Salt Lake City, UT, 84119, EARLY_DEADLINE, 2, , Delivered to Location, 9:55:40, 9:57:40
6, 3660 Lester St, West Valley City, UT, 84119, 10:30 AM, 88, TRUCK2_DELAYED, Delivered to Location, 9:57:40, 10:02:00
36, 2300 Parkway Blvd, West Valley City, UT, 84123, EARLY_DEADLINE, 88, TRUCK2_ONLY, Delivered to Location, 10:02:00, 10:07:20
18, 1488 4800 S, Salt Lake City, UT, 84123, EARLY_DEADLINE, 6, TRUCK2_ONLY, Delivered to Location, 10:07:20, 10:20:40
10, 600 E 900 South, Salt Lake City, UT, 84105, EARLY_DEADLINE, 1, , Delivered to Location, 10:20:40, 10:54:20
38, 410 S State St, Salt Lake City, UT, 84111, EARLY_DEADLINE, 9, TRUCK2_ONLY, Delivered to Location, 10:54:20, 11:00:20
3, 233 Canyon Rd, Salt Lake City, UT, 84103, EARLY_DEADLINE, 2, TRUCK2_ONLY, Delivered to Location, 11:00:20, 11:03:40

TRUCK 3: 33.000000000000001 TOTAL MILES.
PACKAGE ID, ADDRESS, CITY, STATE, ZIP CODE, DEADLINE, WEIGHT, PACKAGE NOTES, STATUS, DEPARTURE TIME, ARRIVAL TIME
=====
21, 3595 Main St, Salt Lake City, UT, 84115, EARLY_DEADLINE, 3, , Delivered to Location, 10:30:00, 10:36:40
26, 5383 S 900 East #104, Salt Lake City, UT, 84117, EARLY_DEADLINE, 25, , Delivered to Location, 10:36:40, 10:50:00
11, 2600 Taylorsville Blvd, Salt Lake City, UT, 84118, EARLY_DEADLINE, 1, , Delivered to Location, 10:50:00, 11:06:20
23, 5100 South 2700 West, Salt Lake City, UT, 84118, EARLY_DEADLINE, 5, , Delivered to Location, 11:06:20, 11:07:40
35, 1060 Dalton Ave S, Salt Lake City, UT, 84104, EARLY_DEADLINE, 88, , Delivered to Location, 11:07:40, 11:30:40
27, 1060 Dalton Ave S, Salt Lake City, UT, 84104, EARLY_DEADLINE, 5, , Delivered to Location, 11:30:40, 11:39:40
39, 2010 W 500 S, Salt Lake City, UT, 84104, EARLY_DEADLINE, 9, , Delivered to Location, 11:30:40, 11:36:00
9, 410 S State St, Salt Lake City, UT, 84111, EARLY_DEADLINE, 2, TRUCK3_DELAYED, Delivered to Location, 11:36:00, 11:46:40
7, 1330 2100 S, Salt Lake City, UT, 84106, EARLY_DEADLINE, 8, , Delivered to Location, 11:46:40, 12:01:00
12, 3575 W Valley Central Station bus Loop, West Valley City, UT, 84119, EARLY_DEADLINE, 1, , Delivered to Location, 12:01:00, 12:20:00

WGUPS PARCEL TRACKING SERVICE
=====
1  UPNP PACKAGE STATUSES FOR ALL TRUCKS

```

## F1. Strengths of the Chosen Algorithm

One of the main strengths of the nearest neighbor algorithm is the ease of implementation. It is less complex than similar algorithms while still efficiently choosing the closest location. It also is a greedy algorithm, which uses the immediate location to find the next path. This algorithm works well for small datasets, though it may have time complexity issues for larger datasets.

## F2. Verification of Algorithm

---

The Nearest Neighbor algorithm does meet all the requirements for this scenario by determining the closest location throughout the route of each truck. It is used to efficiently deliver packages with minimal issues, as this is a relatively small set of data.

## F3. Other Possible Algorithms

---

Other possible algorithms would include Dijkstra's algorithm and the A\* (A Star) algorithm.

### F3a. Algorithm Differences

Dijkstra's Algorithm would be able to determine all possible routes to find the shortest path, where the Nearest Neighbor only uses the current location. Though it is more complex to implement, it would improve the time complexity to  $O(n \log n)$ , which would be beneficial for larger datasets.

A\* algorithm is similar to Dijkstra's algorithm with the addition of heuristics to prioritize a certain path, such as priority. This would also improve performance with large datasets.

## G. Different Approach

---

If I were to take an alternate approach to the project, I would have explored the option of implementing one of the algorithms listed above to optimize time complexity. Also, if the constraints were removed for using external libraries, I would be able to remove the nested loops used for sorting and distance calculation and handle those using the NumPy or pandas libraries.

I would also consider breaking down some of the logic into additional helper methods, particularly the truck routing or distance calculations, to make the code more modular.

## H. Verification of Data Structure

---

The hash table data structure easily meets the requirements of the program. It provides  $O(1)$  time complexity for inserting, updating, and searching, which efficiently stores the data using the package ID as a key.

This data structure will also properly scale to accommodate larger sets of data while maintaining stability and speed of access.

## H1. Other Data Structures

---

A Binary Search Tree or an Array would also be reasonable data structures to use in this scenario.

### H1a. Data Structure Differences

---

A Binary Search Tree would use a tree-based structure and organize elements hierarchically, which would be  $O(\log n)$  time complexity if the tree is balanced. This is less efficient than the hash table, however, which maps the keys to a specific memory location.

An array could be used in other circumstances. This would store the information sequentially in a fixed-sized structure, but it would not allow access based on keys as the hash table does. The time complexity for an array would be  $O(n)$ , which would be less efficient than the hash table.

## I. Sources

---

Lysecky, R., & Vahid, F. (2018, June). C950: Data Structures and Algorithms II. zyBooks. Retrieved February 24, 2025, from

<https://learn.zybooks.com/zybook/WGUC950Template2023>