

WGUPS Routing Program

Dalton J Riley

WGU

C950 Data Structures and Algorithms 2

Sidney Ruby

7/7/2022

WGUPS Routing System

Section A

Algorithm Identification

For this program, I chose the nearest neighbor algorithm to complete the packages' delivery.

Section B

Logic Comments

Set current address variable with current address

Set shortest distance variable to a value larger than every distance

Initiliaze closest address variable

Initialize closest_id variable

Loop through all packages on a truck

Set next address variable containing the current package's address

Set a distance check variable containing the distance from the current address to next address

If the value of distance check is less than or equal to the shortest distance variable

Set shortest distance with value of distance check

Set closest address variable with the value from next address

Set closest id variable with the id of the current package

Return shortest distance, closest address, and closest id

Development Environment

The software used was Python 3.9.2 PyCharm 2021.2.3 and the hardware used was my local machine, an Asus ROG Strix Scar 15 running Windows 10.

Space-Time and Big-O

This can be found within the comments of the program.

Scalability and Adaptability

The two scalable elements in my project are the self-adjusting algorithm (`deliver_packages()`) and the hashtable. Both can take N amount of variables and I believe both will scale reasonably well. Since the hashtable runs in polynomial time and the algorithm runs in exponential time, the hash table is more scalable. The self-adjusting algorithm would be much more scalable if the complexity were taken down to $O(N)$.

Software Efficiency and Maintainability

My program's space-time complexity is $O(N^2)$. Since it runs in exponential time it is not efficient. The program is somewhat maintainable. The functions and variables are clearly named and I have written comments explaining the purpose of all major sections. In the future, I could make it more maintainable by compartmentalizing the functions into classes with their own files. This would make the structure of the program much more understandable and it would clean up the `main.py` file.

Self-Adjusting Data Structures

The self-adjusting data structure in this program is the hash table. It can take n amount of elements and organize them into respective buckets. It can also search for a specific element and return that element efficiently. The fact that we can have an unlimited amount of items in the hashtable and that we are able to search for them efficiently are some of the strengths. The following are weaknesses I would like to improve in the future: creating an update function to manipulate the data once inside. It could also be improved if a function would allow seeing the entire hashtable without having to call each element, one at a time.

Section C

These requirements can be found within the program.

Section D

Explanation of Data Structure

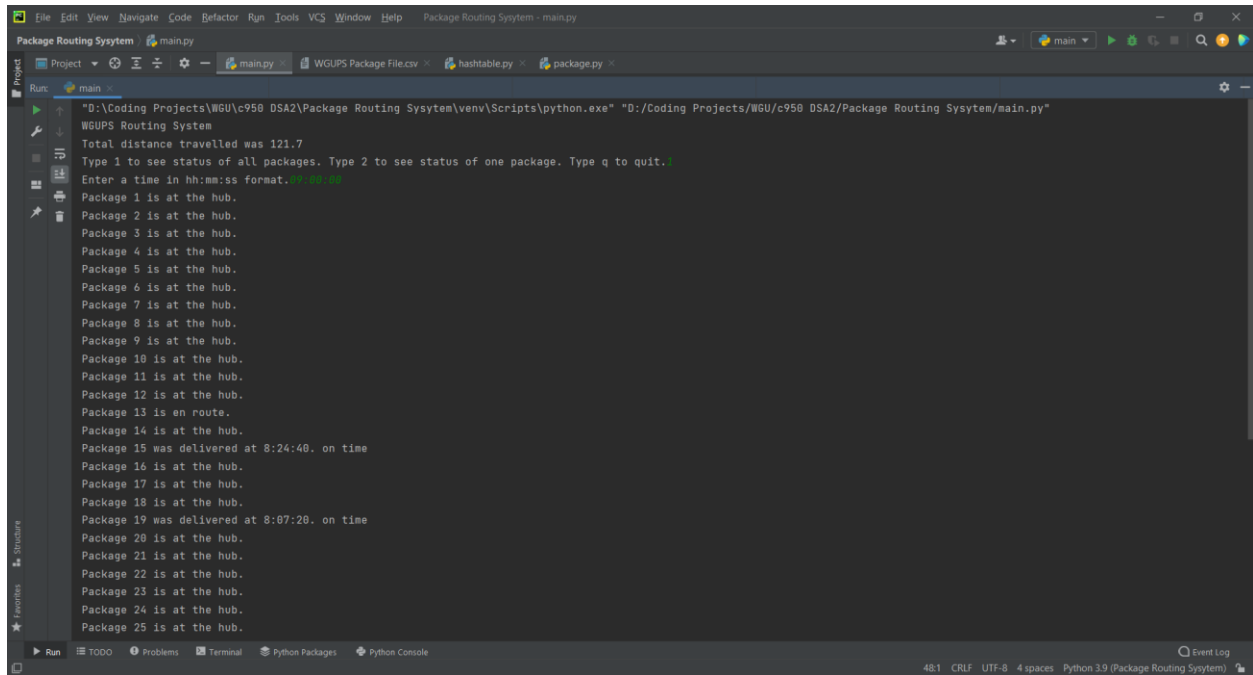
The data structure that I used was a hash table. It creates a table with a specified amount of buckets. If an amount isn't specified, 10 buckets are created. It places item-key (package object is item, package id is key) pairs into the buckets using the modulo function. The result of key modulo 10 is the bucket it is placed in, appending the list. It can then use the key to search or remove the desired item.

Sections E and F

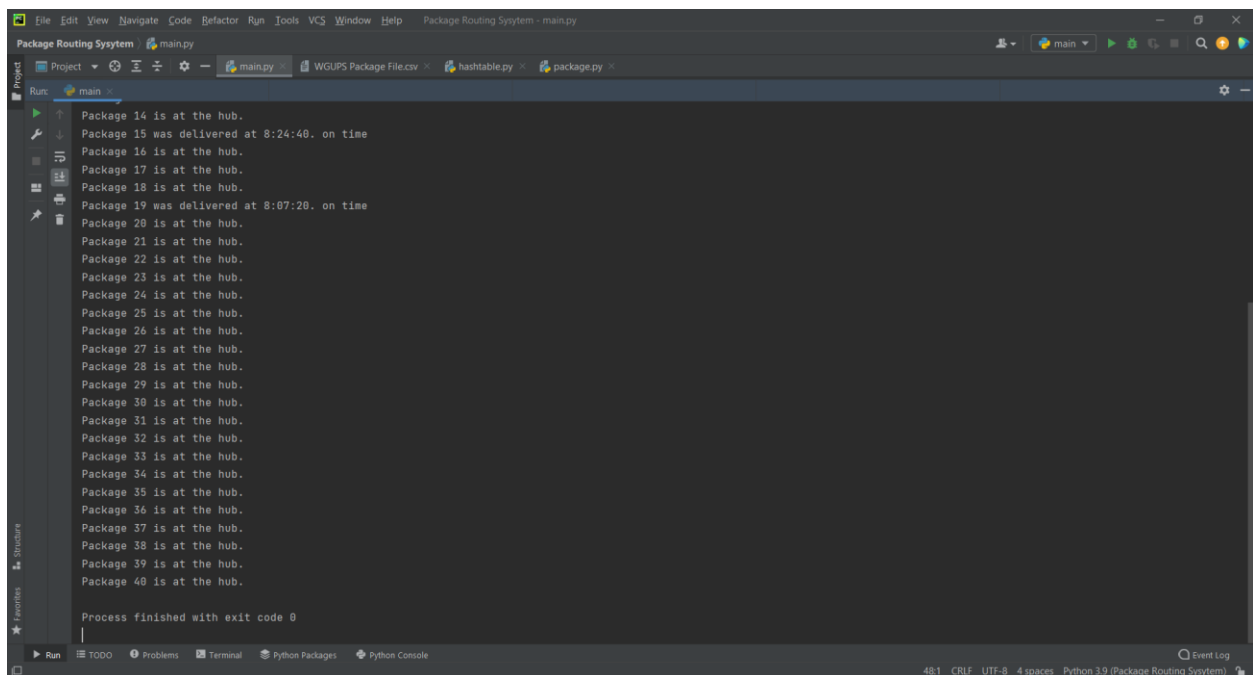
These requirements can be found within the program.

Section G

First Status Check



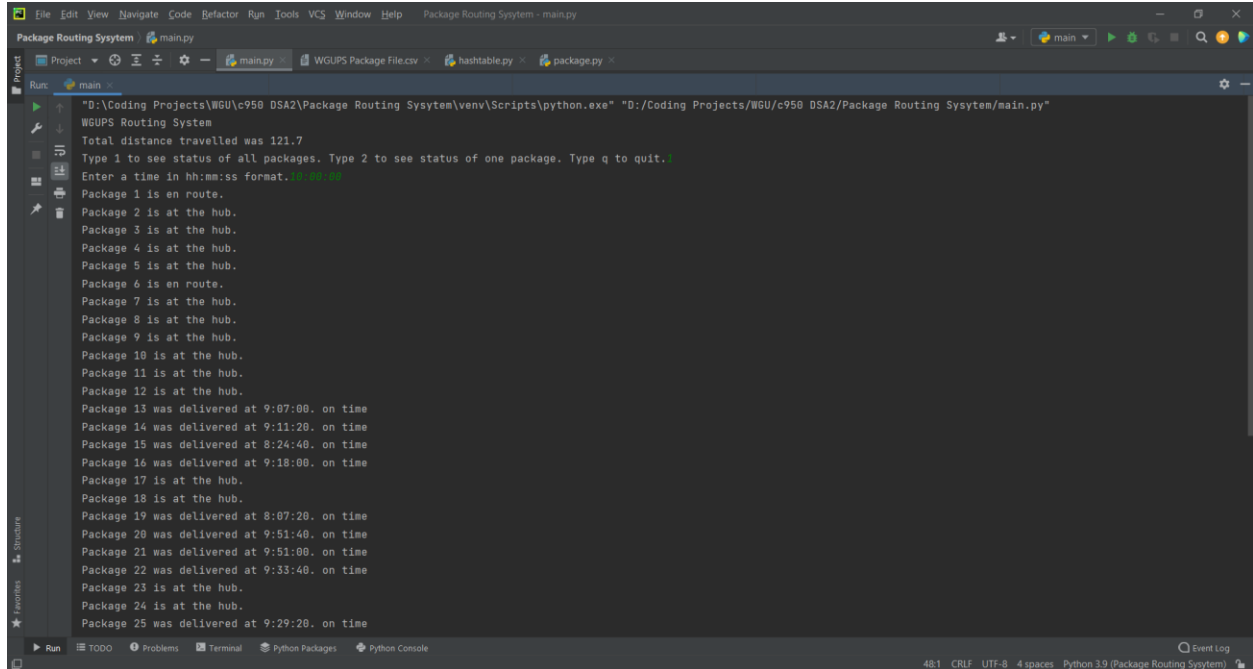
```
Package Routing System - main.py
Project | main.py | WGUPS Package File.csv | hashtable.py | package.py
Run | main
"D:\Coding Projects\WGU\c958 DSA2\Package Routing Sysyem\venv\Scripts\python.exe" "D:\Coding Projects\WGU\c958 DSA2\Package Routing Sysyem\main.py"
WGUPS Routing System
Total distance travelled was 121.7
Type 1 to see status of all packages. Type 2 to see status of one package. Type q to quit.
Enter a time in hh:mm:ss format. 08:00:00
Package 1 is at the hub.
Package 2 is at the hub.
Package 3 is at the hub.
Package 4 is at the hub.
Package 5 is at the hub.
Package 6 is at the hub.
Package 7 is at the hub.
Package 8 is at the hub.
Package 9 is at the hub.
Package 10 is at the hub.
Package 11 is at the hub.
Package 12 is at the hub.
Package 13 is en route.
Package 14 is at the hub.
Package 15 was delivered at 8:24:40. on time
Package 16 is at the hub.
Package 17 is at the hub.
Package 18 is at the hub.
Package 19 was delivered at 8:07:20. on time
Package 20 is at the hub.
Package 21 is at the hub.
Package 22 is at the hub.
Package 23 is at the hub.
Package 24 is at the hub.
Package 25 is at the hub.
```



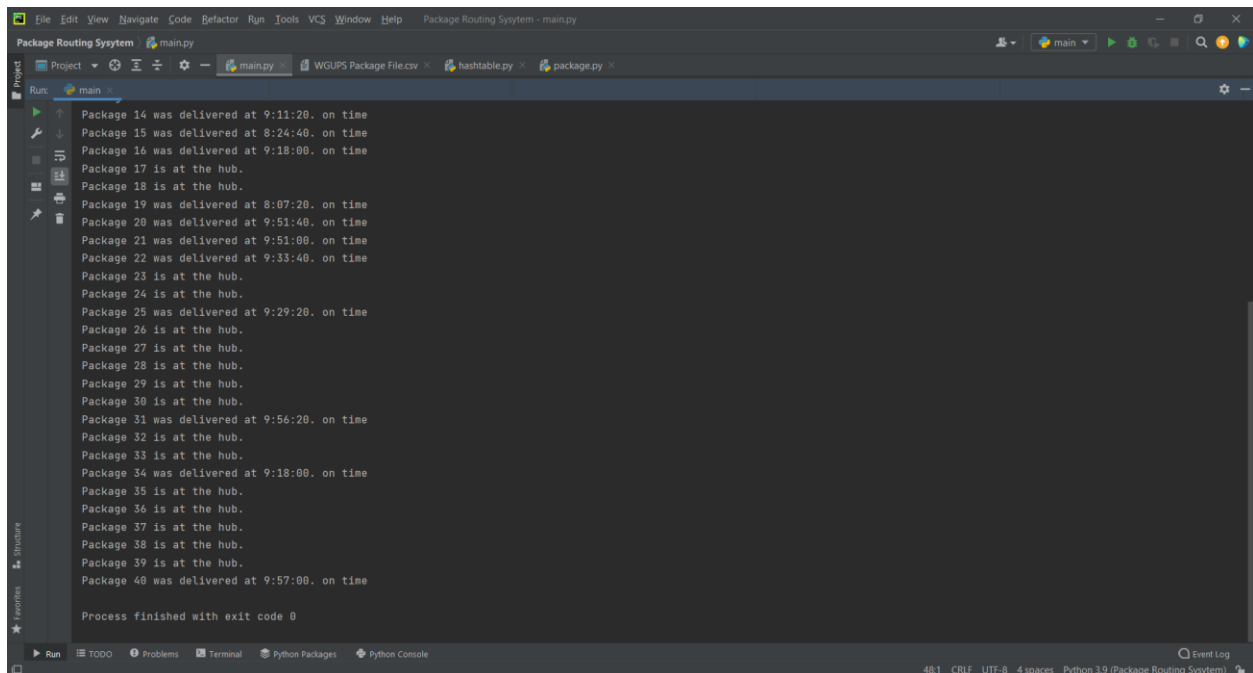
```
Package Routing System - main.py
Project | main.py | WGUPS Package File.csv | hashtable.py | package.py
Run | main
Package 14 is at the hub.
Package 15 was delivered at 8:24:40. on time
Package 16 is at the hub.
Package 17 is at the hub.
Package 18 is at the hub.
Package 19 was delivered at 8:07:20. on time
Package 20 is at the hub.
Package 21 is at the hub.
Package 22 is at the hub.
Package 23 is at the hub.
Package 24 is at the hub.
Package 25 is at the hub.
Package 26 is at the hub.
Package 27 is at the hub.
Package 28 is at the hub.
Package 29 is at the hub.
Package 30 is at the hub.
Package 31 is at the hub.
Package 32 is at the hub.
Package 33 is at the hub.
Package 34 is at the hub.
Package 35 is at the hub.
Package 36 is at the hub.
Package 37 is at the hub.
Package 38 is at the hub.
Package 39 is at the hub.
Package 40 is at the hub.

Process finished with exit code 0
```

Second Status Check



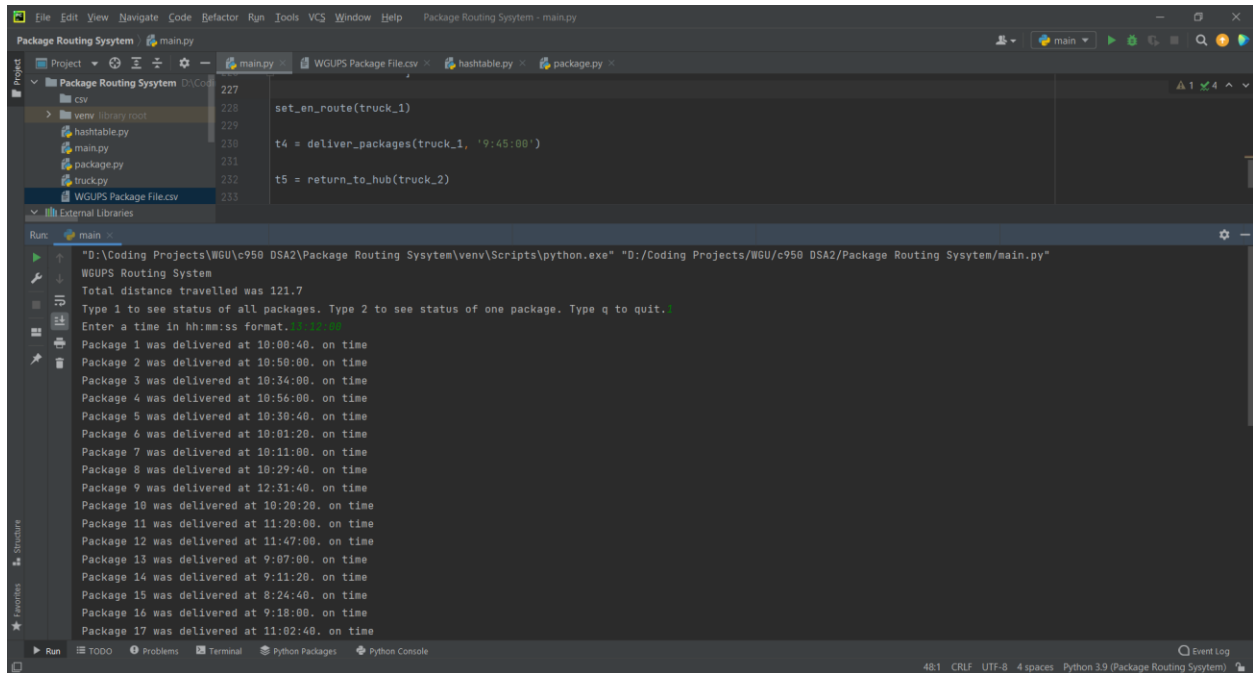
```
"D:\Coding Projects\WGU\c950 DSA2\Package Routing Sysytem\venv\Scripts\python.exe" "D:/Coding Projects/WGU/c950 DSA2/Package Routing Sysytem/main.py"
WGUPS Routing System
Total distance travelled was 121.7
Type 1 to see status of all packages. Type 2 to see status of one package. Type q to quit.
Enter a time in hh:mm:ss format. 8:00:00
Package 1 is en route.
Package 2 is at the hub.
Package 3 is at the hub.
Package 4 is at the hub.
Package 5 is at the hub.
Package 6 is en route.
Package 7 is at the hub.
Package 8 is at the hub.
Package 9 is at the hub.
Package 10 is at the hub.
Package 11 is at the hub.
Package 12 is at the hub.
Package 13 was delivered at 9:07:00. on time
Package 14 was delivered at 9:11:20. on time
Package 15 was delivered at 8:24:40. on time
Package 16 was delivered at 9:18:00. on time
Package 17 is at the hub.
Package 18 is at the hub.
Package 19 was delivered at 8:07:20. on time
Package 20 was delivered at 9:51:40. on time
Package 21 was delivered at 9:51:00. on time
Package 22 was delivered at 9:33:40. on time
Package 23 is at the hub.
Package 24 is at the hub.
Package 25 was delivered at 9:29:20. on time
```



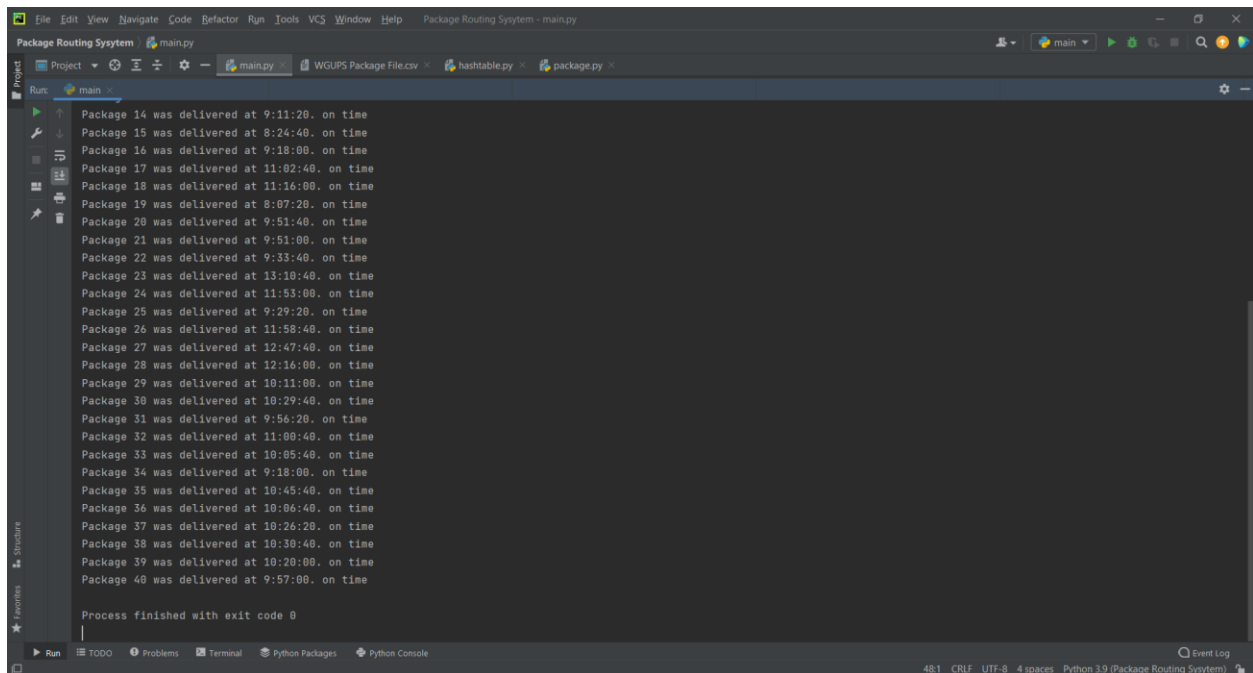
```
Package 14 was delivered at 9:11:20. on time
Package 15 was delivered at 8:24:40. on time
Package 16 was delivered at 9:18:00. on time
Package 17 is at the hub.
Package 18 is at the hub.
Package 19 was delivered at 8:07:20. on time
Package 20 was delivered at 9:51:40. on time
Package 21 was delivered at 9:51:00. on time
Package 22 was delivered at 9:33:40. on time
Package 23 is at the hub.
Package 24 is at the hub.
Package 25 was delivered at 9:29:20. on time
Package 26 is at the hub.
Package 27 is at the hub.
Package 28 is at the hub.
Package 29 is at the hub.
Package 30 is at the hub.
Package 31 was delivered at 9:56:20. on time
Package 32 is at the hub.
Package 33 is at the hub.
Package 34 was delivered at 9:18:00. on time
Package 35 is at the hub.
Package 36 is at the hub.
Package 37 is at the hub.
Package 38 is at the hub.
Package 39 is at the hub.
Package 40 was delivered at 9:57:00. on time

Process finished with exit code 0
```

Third Status Check



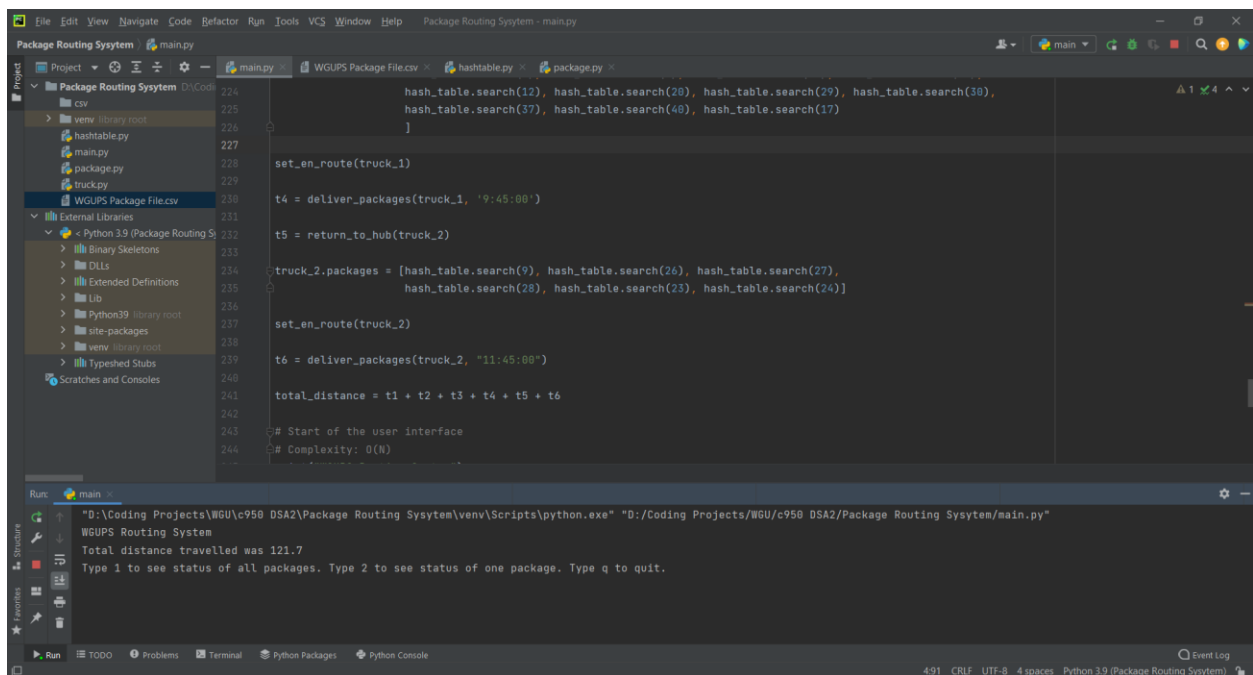
```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Package Routing System - main.py
Package Routing System main.py
Project Package Routing System D:\Coding Projects\WGU\c950 DSA2\Package Routing Sysytem
  csv
  venv\library root
  hashtable.py
  main.py
  package.py
  truck.py
  WGUPS Package File.csv
External Libraries
Run main
"D:\Coding Projects\WGU\c950 DSA2\Package Routing Sysytem\venv\Scripts\python.exe" "D:\Coding Projects\WGU\c950 DSA2\Package Routing Sysytem/main.py"
WGUPS Routing System
Total distance travelled was 121.7
Type 1 to see status of all packages. Type 2 to see status of one package. Type q to quit.
Enter a time in hh:mm:ss format. 8:45:00
Package 1 was delivered at 10:00:40. on time
Package 2 was delivered at 10:00:00. on time
Package 3 was delivered at 10:34:00. on time
Package 4 was delivered at 10:56:00. on time
Package 5 was delivered at 10:30:40. on time
Package 6 was delivered at 10:01:20. on time
Package 7 was delivered at 10:11:00. on time
Package 8 was delivered at 10:29:40. on time
Package 9 was delivered at 12:31:40. on time
Package 10 was delivered at 10:20:20. on time
Package 11 was delivered at 11:20:00. on time
Package 12 was delivered at 11:47:00. on time
Package 13 was delivered at 9:07:00. on time
Package 14 was delivered at 9:11:20. on time
Package 15 was delivered at 8:24:40. on time
Package 16 was delivered at 9:18:00. on time
Package 17 was delivered at 11:02:40. on time
```



```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Package Routing System - main.py
Package Routing System main.py
Project Package Routing System D:\Coding Projects\WGU\c950 DSA2\Package Routing Sysytem
  csv
  venv\library root
  hashtable.py
  main.py
  package.py
  truck.py
  WGUPS Package File.csv
External Libraries
Run main
Package 14 was delivered at 9:11:20. on time
Package 15 was delivered at 8:24:40. on time
Package 16 was delivered at 9:18:00. on time
Package 17 was delivered at 11:02:40. on time
Package 18 was delivered at 11:16:00. on time
Package 19 was delivered at 8:07:20. on time
Package 20 was delivered at 9:51:40. on time
Package 21 was delivered at 9:51:00. on time
Package 22 was delivered at 9:33:40. on time
Package 23 was delivered at 13:10:40. on time
Package 24 was delivered at 11:53:00. on time
Package 25 was delivered at 9:29:20. on time
Package 26 was delivered at 11:58:40. on time
Package 27 was delivered at 12:47:40. on time
Package 28 was delivered at 12:16:00. on time
Package 29 was delivered at 10:11:00. on time
Package 30 was delivered at 10:29:40. on time
Package 31 was delivered at 9:56:20. on time
Package 32 was delivered at 11:00:40. on time
Package 33 was delivered at 10:05:40. on time
Package 34 was delivered at 9:18:00. on time
Package 35 was delivered at 10:45:40. on time
Package 36 was delivered at 10:06:40. on time
Package 37 was delivered at 10:26:20. on time
Package 38 was delivered at 10:30:40. on time
Package 39 was delivered at 10:20:00. on time
Package 40 was delivered at 9:57:00. on time
Process finished with exit code 0
```

Section H

Code Execution



```

224         hash_table.search(12), hash_table.search(20), hash_table.search(29), hash_table.search(38),
225         hash_table.search(37), hash_table.search(40), hash_table.search(17)
226     ]
227
228     set_en_route(truck_1)
229
230     t4 = deliver_packages(truck_1, '9:45:00')
231
232     t5 = return_to_hub(truck_2)
233
234     truck_2.packages = [hash_table.search(9), hash_table.search(26), hash_table.search(27),
235                        hash_table.search(28), hash_table.search(23), hash_table.search(24)]
236
237     set_en_route(truck_2)
238
239     t6 = deliver_packages(truck_2, "11:45:00")
240
241     total_distance = t1 + t2 + t3 + t4 + t5 + t6
242
243     # Start of the user interface
244     # Complexity: O(N)

```

Run: main

```

"D:\Coding Projects\WGU\c950 DSA2\Package Routing Sysytem\venv\Scripts\python.exe" "D:/Coding Projects/WGU/c950 DSA2/Package Routing Sysytem/main.py"
WGUPS Routing System
Total distance travelled was 121.7
Type 1 to see status of all packages. Type 2 to see status of one package. Type q to quit.

```

Section I

Strengths of Chosen Algorithm

One strength of the algorithm I chose is that it is adaptable. It can n amount of packages and deliver them effectively. Another strength is that it keeps track of the time packages were loaded and delivered, as well as the distance traveled for each package.

Verification of Algorithm

The total miles traveled by all trucks using this algorithm was 121.7. All packages were delivered on time and within their specifications. This can be found by using the user interface or by looking at the screenshots above.

Other Possible Algorithms

Two other algorithms that would be possible are nearest insertion and random insertion.

Algorithm Differences

The nearest insertion algorithm starts with two locations and adds the closest location next to them making a route. It then continues finding the closest location and adding them to the route so that the route remains as short as it can be until no locations are left. This algorithm would find a more optimal solution and the complexity is still $O(N^2)$. The random insertion algorithm starts with two locations and randomly adds other locations into the route until none are left. This algorithm might produce a more optimal solution or might not because of its random nature. The time complexity is also $O(N^2)$.

Section J**Different Approach**

I would change the way I loaded the distance and address files, and the way I loaded the trucks. For ease, I loaded everything manually, but this is not sustainable if it were to scale larger. I would create functions to read the CSV files and automatically load the contents of them

into lists. I would also make a function that takes a truck as a parameter and automatically loads packages until the truck is full, checking package constraints along the way.

Section K

Verification of Data Structure

The total combined miles for all trucks was 121.7. All packages were delivered on time and according to package specifications. An efficient hashtable that can look up items is present and the package status and info can be found through the user interface or the screenshots above.

Efficiency

The look-up function for the hash table has a complexity of $O(N)$. Adding more items will affect the time linearly. If it takes one second with one item, it will take 10 seconds with 10 items.

Overhead

The same is true of space complexity. The look-up function runs in $O(N)$ so the space required is directly affected by the number of items being stored.

Implications

In this program adding additional trucks and cities would have no impact on the hashtable. The only thing stored in the hashtable is the packages.

Other Data Structures

Another possible data structure could be a hashtable with an update function. Another could be a python dictionary.

Data Structure Differences

The difference for the hashtable with the update function would be that we could make the hashtable private, preventing manipulation without a method being used. This would also make the code cleaner, having a method call instead of directly changing the package objects. The difference with using a dictionary would be that the key-item pairs would be stored in one large list instead of different buckets. This would decrease scalability.

References

WGU. (2020, November 17). *Lets Go Hashing* [Video]. Panopto.

<https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=f08d7871-d57a-496e-a6a1-ac7601308c71>

WGU. *C950 WGUPS Project Implementation Steps - Example*

<https://srm--c.na127.visual.force.com/apex/coursearticle>