A. Identify a Self-Adjusting Algorithm:

We will use the nearest neighbor algorithm to create the program for delivering the packages. This algorithm starts at the hub and repeatedly selects the nearest package based on the given criteria to be delivered next.

B. Identify a Self-Adjusting Data Structure:

We will use a hash table to store the package data efficiently. Each package can be accessed quickly using a unique package ID as the key. This data structure allows for fast retrieval, insertion, and modification of package information.

1. Explain Data Structure Relationship:

The hash table allows us to associate each unique package ID with its corresponding package details. This relationship ensures that we can quickly retrieve package information by referencing the unique package ID.

C. Overview of the Program:

1. Algorithm Logic Using Pseudocode:

PROCEDURE deliverAllPackages():

Initialize the hash table to store package information.

Initialize trucks and drivers.

Start at the hub with all trucks loaded at 8:00 a.m.

WHILE there are undelivered packages:

FOR each truck:

distanceIDs EQUALS []

distances EQUALS []

FOR package in trucksPackages:

IF package is not None AND packageStatus is not 'Delivered':

APPEND packageDistanceID to distanceIDs

APPEND distance\_data AT INDEX OF packageDistanceID TO distances

END IF

END FOR

closestDistance EQUALS smallest value in distances

closestDistance EQUALS min(distances)

packageKey EQUALS INDEX OF closestDistance in distances

minutesToClosestNeighbor EQUALS closestDistance DIVIDED BY truckSpeed AND MULTIPLIED BY 60

UPDATE previousLocation TO currentLocation

UPDATE currentLocation TO destination of packageToUpdate

UPDATE currentTime TO currentTime PLUS minutesToClosestNeighbor

UPDATE truckMileage TO truckMileage PLUS closestDistance

deliveredPackage EQUALS package with matching packageKey in truckPackages

UPDATE deliveredPackage TO 'Delivered'

INSERT deliveredPackage INTO truckPackages

END FOR

END WHILE

WHEN all packages are delivered:

End the day.

END PROCEDURE

2. Programming Environment:

We will create the Python application using the following environment:

- Software: Python 3.10.4, VSCode 1.83

- Hardware: HP ENVY x360 (Intel i7 12th Gen, 16GB RAM) Running Windows 11

3. Space-Time Complexity:

- The algorithm's time complexity is O(n^2) since it iterates over packages for each truck.

- The space complexity is O(n), where n is the number of packages.

4. Scalability:

The solution can adapt to a growing number of packages by continuing to use the nearest neighbor algorithm. It can handle additional packages within the defined constraints and does not require significant changes to accommodate more packages.

5. Efficiency and Maintainability:

The software design is efficient because it minimizes the total distance traveled and delivers packages within specified constraints. It is easy to maintain due to clear pseudocode, modular design, and the use of a self-adjusting data structure for quick access to package information.

6. Strengths and Weaknesses of Hash Table:

- Strengths: Fast access and modification of package data, constant-time complexity for typical operations.

- Weaknesses: Hash collisions may occur if not managed properly, leading to slower performance.

7. Choice of Key for Delivery Management:

We will use the package ID as the key for efficient delivery management. This is the most straightforward and efficient way to access package information because each package ID is unique and can directly map to the corresponding package details. Other attributes, such as delivery deadline or status, can change over time, while the package ID remains constant.