NHP3 — NHP3 TASK 2: WGUPS ROUTING PROGRAM IMPLEMENTATION

DATA STRUCTURES AND ALGORITHMS II — C950
PRFA — NHP3

COMPETENCIES

4048.5.1 : Non-Linear Data Structures
The graduate creates software applications that incorporate non-linear data structures for efficient and maintainable software.

4048.5.2 : Hashing Algorithms and Structures
The graduate writes code using hashing techniques within an application to perform searching operations.

4048.5.5 : Self-Adjusting Heuristics
The graduate writes code using self-adjusting heuristics to improve the performance of applications.

INTRODUCTION

For Tasks 1 and 2, you will apply the algorithms and data structures you studied in this course to solve a real programming problem. You will also implement an algorithm to route delivery trucks that will allow you to meet all delivery constraints while traveling under 140 miles. You will then describe and justify the decisions you made while creating this program.

The skills you showcase in your completed project may be useful in responding to technical interview questions for future employment. This project may also be added to your portfolio to show to future employers.

SCENARIO

This task is the implementation phase of the WGUPS Routing Program.

The Western Governors University Parcel Service (WGUPS) needs to determine an efficient route and delivery distribution for their daily local deliveries (DLD) because packages are not currently being consistently delivered by their promised deadline. The Salt Lake City DLD route has three trucks, two drivers, and an average of 40 packages to deliver each day. Each package has specific criteria and delivery requirements that are listed in the attached "WGUPS Package File."

Your task is to determine an algorithm, write code, and present a solution where all 40 packages will be delivered on time while meeting each package's requirements and keeping the combined total distance traveled under 140 miles for all trucks. The specific delivery locations are shown on the

attached "Salt Lake City Downtown Map," and distances to each location are given in the attached "WGUPS Distance Table." The intent is to use the program for this specific location and also for many other cities in each state where WGU has a presence. As such, you will need to include detailed comments to make your code easy to follow and to justify the decisions you made while writing your scripts.

The supervisor should be able to see, at assigned points, the progress of each truck and its packages by any of the variables listed in the "WGUPS Package File," including what has been delivered and at what time the delivery occurred.

ASSUMPTIONS

- Each truck can carry a maximum of 16 packages, and the ID number of each package is unique.
- The trucks travel at an average speed of 18 miles per hour and have an infinite amount of gas with no need to stop.
- There are no collisions.
- Three trucks and two drivers are available for deliveries. Each driver stays with the same truck as long as that truck is in service.
- Drivers leave the hub no earlier than 8:00 a.m., with the truck loaded, and can return to the hub for packages if needed.
- The delivery and loading times are instantaneous (i.e., no time passes while at a delivery or when moving packages to a truck at the hub). This time is factored into the calculation of the average speed of the trucks.
- There is up to one special note associated with a package.
- The delivery address for package #9, Third District Juvenile Court, is wrong and will be corrected at 10:20 a.m. WGUPS is aware that the address is incorrect and will be updated at 10:20 a.m. However, WGUPS does not know the correct address (410 S. State St., Salt Lake City, UT 84111) until 10:20 a.m.
- The distances provided in the "WGUPS Distance Table" are equal regardless of the direction traveled.
- The day ends when all 40 packages have been delivered.

REQUIREMENTS

Your submission must be your original work. No more than a combined total of 30% of the submission and no more than a 10% match to any one individual source can be directly quoted or closely paraphrased from sources, even if cited correctly. The similarity report that is provided when you submit your task can be used as a guide.

You must use the rubric to direct the creation of your submission because it provides detailed criteria that will be used to evaluate your work. Each requirement below may be evaluated by more than one rubric aspect. The rubric aspect titles may contain hyperlinks to relevant portions of the course.

*Tasks may **not** be submitted as cloud links, such as links to Google Docs, Google Slides, OneDrive, etc., unless specified in the task requirements. All other submissions must be file types that are uploaded and submitted as attachments (e.g., .docx, .pdf, .ppt).*

Note: Use only appropriate built-in data structures, except dictionaries. You must design, write, implement, and debug all code that you turn in for this assessment. Code downloaded from the internet or acquired from

another student or any other source may not be submitted and will result in automatic failure of this assessment.

A. Develop a hash table, without using any additional libraries or classes, that has an insertion function that takes the package ID as input and inserts each of the following data components into the hash table:
   - delivery address
   - delivery deadline
   - delivery city
   - delivery zip code
   - package weight
   - delivery status (i.e., at the hub, en route, or delivered), including the delivery time

B. Develop a look-up function that takes the package ID as input and returns *each* of the following corresponding data components:
   - delivery address
   - delivery deadline
   - delivery city
   - delivery zip code
   - package weight
   - delivery status (i.e., at the hub, en route, or delivered), including the delivery time

C. Write an original program that will deliver *all* packages and meet all requirements using the attached supporting documents "Salt Lake City Downtown Map," "WGUPS Distance Table," and "WGUPS Package File."
   1. Create an identifying comment within the first line of a file named "main.py" that includes your student ID.
   2. Include comments in your code to explain both the process and the flow of the program.

D. Provide an intuitive interface for the user to view the delivery status (including the delivery time) of any package at any time and the total mileage traveled by all trucks. (The delivery status should report the package as at the hub, en route, or delivered. Delivery status must include the time.)
   1. Provide screenshots to show the status of *all* packages loaded onto *each* truck at a time between 8:35 a.m. and 9:25 a.m.
   2. Provide screenshots to show the status of *all* packages loaded onto *each* truck at a time between 9:35 a.m. and 10:25 a.m.
   3. Provide screenshots to show the status of *all* packages loaded onto *each* truck at a time between 12:03 p.m. and 1:12 p.m.

E. Provide screenshots showing successful completion of the code that includes the total mileage traveled by *all* trucks.

F. Justify the package delivery algorithm used in the solution as written in the original program by doing the following:
   1. Describe **two or more** strengths of the algorithm used in the solution.
   2. Verify that the algorithm used in the solution meets *all* requirements in the scenario.
   3. Identify **two** other named algorithms that are different from the algorithm implemented in the solution and would meet *all* requirements in the scenario.
      a. Describe how *both* algorithms identified in part F3 are different from the algorithm used in the solution.

G. Describe what you would do differently, other than the two algorithms identified in part F3, if you did this project again, including details of the modifications that would be made.

H. Verify that the data structure used in the solution meets *all* requirements in the scenario.
   1. Identify **two** other data structures that could meet the same requirements in the scenario.
      a. Describe how *each* data structure identified in H1 is different from the data structure used in the solution.

I. Acknowledge sources, using in-text citations and references, for content that is quoted, paraphrased, or summarized.

J. Demonstrate professional communication in the content and presentation of your submission.

File Restrictions

File name may contain only letters, numbers, spaces, and these symbols: ! - _ . * ' ( )
File size limit: 200 MB
File types allowed: doc, docx, rtf, xls, xlsx, ppt, pptx, odt, pdf, csv, txt, qt, mov, mpg, avi, mp3, wav, mp4, wma, flv, asf, mpeg, wmv, m4v, svg, tif, tiff, jpeg, jpg, gif, png, zip, rar, tar, 7z

RUBRIC

A. :HASH TABLE

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A hash table is not provided.	The hash table contains errors or has an insertion function with additional libraries or classes, or it does not take the package ID as input. Or the insertion function does not insert 1 or more of the given data components.	The hash table is free from errors and has an insertion function, without using any additional libraries or classes, that takes the package ID as input and inserts *each* of the given data components.

B. :LOOK-UP FUNCTION

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A look-up function is not provided.	The look-up function does not complete, or it completes with runtime errors. Or it does not take the package ID as input, or it does not return 1 or more of the given data components.	The look-up function completes without runtime errors and takes the package ID as input and returns *each* of the given data components.

C. :ORIGINAL CODE

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
An original program is not provided.	The program and code are original, but they run with errors or warnings. Or they do not deliver 1 or more packages or do not meet 1 or more requirements.	The program and code are original. They run without errors or warnings, delivers *all* packages, and meets *all* requirements.

C1. :IDENTIFICATION INFORMATION

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
An identifying comment is not provided.	The identifying comment is missing from the first line of a file named “main.py,” the file is not named “main.py,” or the comment is missing the student ID, or 1 or more of these elements is incorrect.	The identifying comment is located within the first line of a file named “main.py” that includes the student ID.

C2. :PROCESS AND FLOW COMMENTS

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
The code does not include comments.	The code does not include detailed comments, or the comments do not accurately explain *either* the process or the flow of the program. Or comments are missing for *either* the process or the flow of the program.	The code includes detailed comments that accurately explain *both* the process and the flow of the program.

D. :INTERFACE

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
An interface is not provided.	The interface does not provide an intuitive means for the user to *either* view the delivery status or for the user to determine the total mileage traveled by 1 or more trucks. Or the delivery	The interface provides an intuitive means for the user to *both* view the delivery status and for the user to determine the total mileage traveled by *all* trucks. The delivery status includes the delivery time.

	status is missing the delivery time.	

D1. :FIRST STATUS CHECK

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A screenshot of the first status check is not provided.	The screenshots provided do not capture 1 or more packages loaded onto *each* truck, or 1 or more trucks is missing. Or they do not capture the status of 1 or more packages at a time between 8:35 a.m. and 9:25 a.m.	The screenshots provided capture *all* packages loaded onto *each* truck and they capture the status of *each* package at a time between 8:35 a.m. and 9:25 a.m.

D2. :SECOND STATUS CHECK

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A screenshot of the second status check is not provided.	The screenshots provided do not capture 1 or more packages loaded onto *each* truck, or 1 or more trucks is missing. Or they do not capture the status of 1 or more packages at a time between 9:35 a.m. and 10:25 a.m.	The screenshots provided capture *all* packages loaded onto *each* truck and they capture the status of *each* package at a time between 9:35 a.m. and 10:25 a.m.

D3. :THIRD STATUS CHECK

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A screenshot of the third status check is not provided.	The screenshots provided do not capture 1 or more packages loaded onto *each* truck, or 1 or more trucks is missing. Or they do not capture the status of 1 or more packages at a time between 12:03 p.m. and 1:12 p.m.	The screenshots provided capture *all* packages loaded onto *each* truck and they capture the status of *each* package at a time between 12:03 p.m. and 1:12 p.m.

E. :SCREENSHOTS OF CODE EXECUTION

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A screenshot that shows a complete execution of the code is not provided.	The screenshots capture an incomplete execution of the code,	The screenshots capture a complete execution of the code that is free from runtime errors or

	or they capture runtime errors or warnings. Or the screenshots do not capture the total mileage traveled by 1 or more trucks.	warnings and include the total mileage traveled by *all* trucks.

F1. :STRENGTHS OF THE CHOSEN ALGORITHM

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A description of the strengths of the algorithm used in the solution is not provided.	The description does not accurately explain 1 or more strengths of the algorithm used in the solution.	The description accurately explains 2 or more strengths of the algorithm used in the solution.

F2. :VERIFICATION OF ALGORITHM

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
The submission does not verify the algorithm used in the solution.	The submission does not verify that the algorithm used in the solution meets 1 or more requirements in the scenario.	The submission verifies that the algorithm used in the solution meets *all* requirements in the scenario.

F3. :OTHER POSSIBLE ALGORITHMS

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
The submission does not identify 2 other algorithms.	The submission identifies 2 algorithms different from the one used in the solution, but 1 or both algorithms do not meet *all* requirements in the scenario.	The submission identifies 2 algorithms different from the one used in the solution, and *both* algorithms meet *all* requirements in the scenario.

F3A. :ALGORITHM DIFFERENCES

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A description of how the algorithms are different is not provided.	The description is missing details, or it does not accurately compare how 1 or both algorithms identified in part F3 are different from the algorithm used in the solution.	The description thoroughly and accurately compares how *both* algorithms identified in part F3 are different from the algorithm used in the solution.

G. :DIFFERENT APPROACH

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A description of what would be done differently is not provided.	The description does not appropriately explain what would be done differently, or it does not include details of the modifications that would be made.	The description appropriately explains what would be done differently and includes details of the modifications that would be made.

H. :VERIFICATION OF DATA STRUCTURE

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
The submission does not verify the data structure used in the solution.	The submission does not verify the data structure used in the solution meets 1 or more requirements in the scenario.	The submission verifies the data structure used in the solution meets *all* requirements in the scenario.

H1. :OTHER DATA STRUCTURES

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
The submission does not identify 2 other data structures.	The submission identifies 2 data structures that are different from the one used in the solution, but 1 or *both* data structures do not meet *all* requirements in the scenario.	The submission identifies 2 data structures that are different from the one used in the solution, and *both* data structures meet *all* requirements in the scenario.

H1A. :DATA STRUCTURE DIFFERENCES

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A description that compares how the data structures are different is not provided.	The description is missing details, or it does not accurately compare how 1 or more data structures identified in H1 is different from the data structure used in the solution.	The description thoroughly and accurately compares how *each* data structure identified in H1 is different from the data structure used in the solution.

I. :SOURCES

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT

The submission does not include both in-text citations and a reference list for sources that are quoted, paraphrased, or summarized.	The submission includes in-text citations for sources that are quoted, paraphrased, or summarized and a reference list; however, the citations or reference list is incomplete or inaccurate.	The submission includes in-text citations for sources that are properly quoted, paraphrased, or summarized and a reference list that accurately identifies the author, date, title, and source location as available.

J. :PROFESSIONAL COMMUNICATION

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
Content is unstructured, is disjointed, or contains pervasive errors in mechanics, usage, or grammar. Vocabulary or tone is unprofessional or distracts from the topic.	Content is poorly organized, is difficult to follow, or contains errors in mechanics, usage, or grammar that cause confusion. Terminology is misused or ineffective.	Content reflects attention to detail, is organized, and focuses on the main ideas as prescribed in the task or chosen by the candidate. Terminology is pertinent, is used correctly, and effectively conveys the intended meaning. Mechanics, usage, and grammar promote accurate interpretation and understanding.

SUPPORTING DOCUMENTS

- Sample Core Algorithm Overview.docx
- SLC downtown map.docx
- WGUPS Distance Table.xlsx
- WGUPS Package File.xlsx