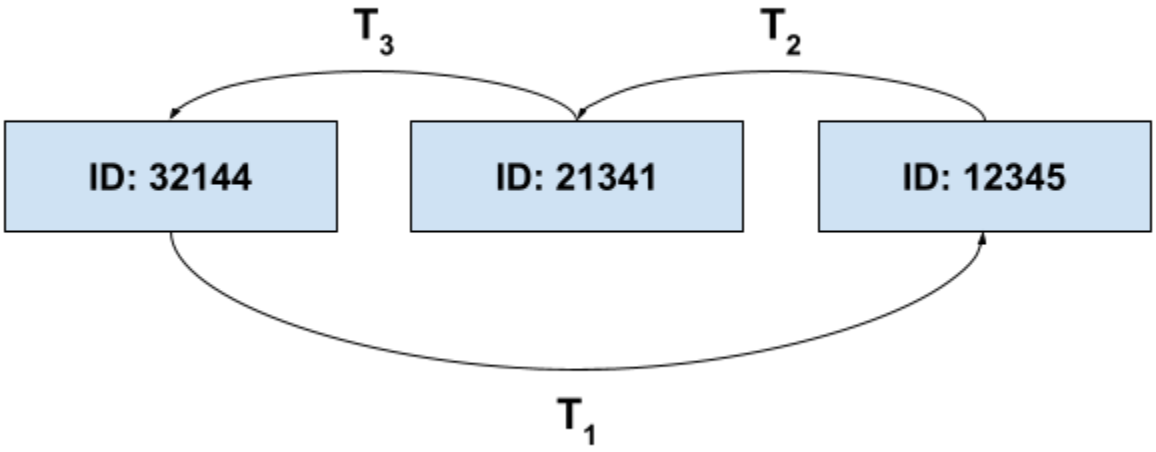
Lab 1: Linked Lists with AlgoZon

You are employed by a new company called "AlgoZon," with the task of enhancing the package delivery process. The company aims to identify whether a driver has taken a circular route during package deliveries. Your responsibility is to determine if, at any point in the trip, the driver returned to a location they had previously visited.

The monitoring system currently in use at AlgoZon tracks routes by drivers as a Linked List, where each node has an **ID** (integer) and a **timestamp** (in Unix format).

Each node represents a package pickup zone where a driver gets packages.



To check for inefficiency, AlgoZon has assigned you the task of writing a function that will detect whether a driver has returned to the same pickup zone. Additionally, you must also audit whether the linked list of destinations is in chronological order; in other words, we want to detect an error if a driver reaches a destination before the previous one in the linked list.

Write a function that takes in the first node in the monitoring linked list and **returns** the total time for the cycle, or **null** if there isn't one. The function should **throw an InvalidRouteError** if any destination in the linked list is reached before its predecessor (even after a cycle).

You will be provided starter code that will include the code for a node class and provide the InvalidRouteError class.

Deliverable Explanation

You must efficiently solve this problem in either Python or Java, using one of the attached starter code files. Several unit tests are provided to help you test your code.

Having the optimal runtime will grant you extra credit. On the other hand, you may also lose points if your code is substantially inefficient.

Your code will be evaluated against a set of visible and hidden test cases (the visible ones are also provided in the starter code for your convenience) to test correctness. The hidden test cases are intended to verify that you have a robust solution that considers possible edge cases and various inputs.

If you are using Java, your file's package must be set to "student".

Submission Instructions

Submit the following file on Gradescope:
AlgoZon.java to Lab1 (Java)
OR
algo_zon.py to Lab1 (Python)

Grading Methodology

The lab is worth 100 points and is broken down into the following categories:

- Test Cases Passing (80 points)
- Overall code performance and style (10 points)
- Credit for submission (10 points)
- Having Optimal Running Time (5 points of Extra Credit)