Problem Statement

Question 1

Use Case: Shipping Company Optimization

Background

A shipping company needs to efficiently allocate various shipments into available containers. Each shipment consists of a specific number of units, and each container has a maximum capacity of units it can hold. The goal is to determine the minimum number of containers required to accommodate all shipments, even if units from the same shipment are split across multiple containers.

Scenario

Company: SwiftLogistics

Shipments: The company has `n` shipments to be delivered. Each shipment consists of a different number of units. This is represented by the array `shipments` where `shipments[i]` indicates the number of units in the `i-th` shipment.

Containers: The company also has `m` containers available for use, each with a specific capacity. This is represented by the array `container_limits` where `container_limits[j]` indicates the maximum number of units that the `j-th` container can hold.

Problem

The operations team at SwiftLogistics needs to figure out the minimum number of containers required to redistribute the units from all shipments into the available containers.

Example

Inputs:

- `shipments = [10, 20, 30]`
- `container_limits = [15, 15, 20, 10]`

Output:

- Minimum number of containers needed: 4
- If there is an insufficient container_limit the function should return -1.

Explanation

1. The first container (15 units) can take 10 units from the first shipment and 5 units from the second shipment.
2. The second container (15 units) can take the remaining 15 units from the second shipment.
3. The third container (20 units) can take the entire third shipment of 30 units, but since it exceeds its capacity, the fourth container (10 units) can be used to accommodate the remaining 10 units.

Thus, the minimum number of containers needed is 4.

Note: You just need to write the required function for the problem, and you must take two arrays as input parameters to the function.

Question 2

Question-Scenario:

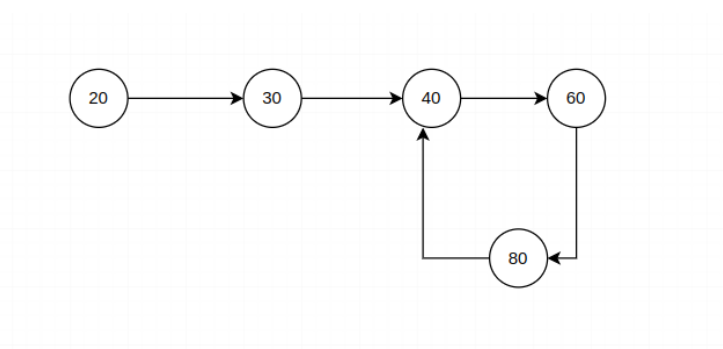
You are a software developer at a logistics company. Your company maintains a system to track packages as they move through different stages of delivery. Each package's journey is represented as a linked list, where each node contains information about a stage in the delivery process.

Recently, some packages have been reported to be stuck in an infinite loop in the tracking system, causing delays and confusion. To prevent this, you need to implement a function to detect any structural issues in the package tracking system.

Problem Statement:

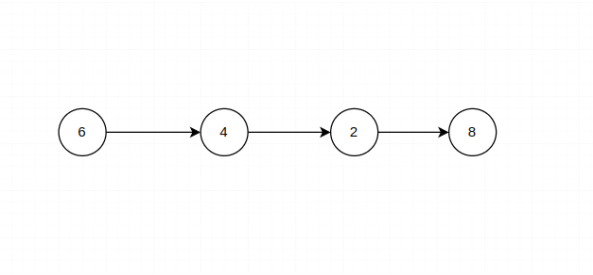
Given the head of a linked list representing the stages of a package's journey, you need to determine if there is a structural issue causing an infinite loop.

Example 1:
Input :
head = [20,30,40,60,80]
Output :
true
Explanation: There is a cycle in the linked list where the tail connects to the 3rd node.



Example 2:
Input:
head = [6,4,2,8]
Output:
false

Explanation: The linked do not have a cycle.



Note: You need to write the required function for this problem, and you must take the head of the linked list as input parameter to the function.

Full Stack Expense Tracker

The task is to create an **expense tracker** application.
The project should include a user authentication system and functionality to manage expenses. The application must feature a user-friendly interface for adding, viewing, editing, and deleting expenses. Additionally, the application should provide visual representation through a pie chart of category-wise expense distribution.

Requirements

1. Authentication

Sign Up & Login Pages:

Sign Up: Allows users to create an account.

Login: Allows users to log in using either Basic Authentication or JWT (JSON Web Token) as per your choice.

2. Expense Management

Add Expense:
Users can add a new expense.
Input fields: Category, Amount, Comments (optional).

View Expenses:
Display expenses in a table format.
Columns: Category, Amount, Created At, Updated At, Comments.
The table should be sorted by the latest added record.

Edit Expense:
Users can edit existing expenses.

Delete Expense:
Users can delete expenses.

Data Visualization (Optional Feature)
Category-wise Expense Distribution:
A page displaying a pie chart of expenses distributed by category.

Note: Data Visualization feature is optional. You can add these features on a single page or multiple pages as per your choice.