1) **Given an array of integers `nums` and an integer `target`, return *indices of the two numbers such that they add up to `target`*.**

You may assume that each input would have *exactly* **one solution**, and you may not use the *same* element twice.

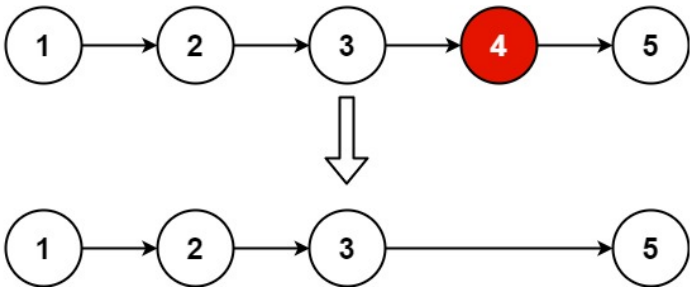
You can return the answer in any order.

Example 1**:**

```
Input: nums = [2,7,11,15], target = 9

Output: [0,1]

Explanation: Because nums[0] + nums[1] == 9, we return [0, 1].

Input: nums = [2,7,11,15], target = 9

Output: [0,1]

Explanation: Because nums[0] + nums[1] == 9, we return [0, 1].
```

2) **Given the `head` of a linked list, remove the `n`th node from the end of the list and return its head.**

Example 1:




```
Input: head = [1,2,3,4,5], n = 2

Output: [1,2,3,5]
```

Constraints:

- The number of nodes in the list is `sz`.
- `1 <= sz <= 30`
- `0 <= Node.val <= 100`
- `1 <= n <= sz`

3) SQL Schema:

A company's executives are interested in seeing who earns the most money in each of the company's departments. A **high earner** in a department is an employee who has a salary in the **top three unique** salaries for that department.

Write an SQL query to find the employees who are **high earners** in each of the departments.

Return the result table **in any order**.

Input:

Employee table:

```
+----+-------+--------+--------------+
| id | name  | salary | departmentId |
+----+-------+--------+--------------+
| 1  | Joe   | 85000  | 1            |
| 2  | Henry | 80000  | 2            |
| 3  | Sam   | 60000  | 2            |
| 4  | Max   | 90000  | 1            |
| 5  | Janet | 69000  | 1            |
| 6  | Randy | 85000  | 1            |
| 7  | Will  | 70000  | 1            |
+----+-------+--------+--------------+
```

Department table:

```
+----+-------+
| id | name  |
+----+-------+
| 1  | IT    |
| 2  | Sales |
+----+-------+
```

Output:

```
+------------+----------+--------+
| Department | Employee | Salary |
+------------+----------+--------+
| IT         | Max      | 90000  |
| IT         | Joe      | 85000  |
| IT         | Randy    | 85000  |
| IT         | Will     | 70000  |
| Sales      | Henry    | 80000  |
| Sales      | Sam      | 60000  |
+------------+----------+--------+
```

4) **Create a WPF project (with appropriate nice GUI) to take birthday date from the user "Day/Month/Year" and then give him his actual age depending on the current date.**
   Example 1:
   Input: 21/4/1997
   Output: "Your current age is 25 years, 5 months and 25 days".

Tasks must be submitted on GitHub. Every developer has to create his own branch and push his code. "https://github.com/MahmoudHassan7/Onspec-Tasks.git"
