1.Implement a Custom JSON Parser

Write a function parse_json(json_string) that:

- Converts a JSON-formatted string into a Python dictionary.
- Handles **nested structures, lists, and basic types.**

Hint: Use recursion and eval() **(safely).**

2.Find All Anagrams of a String in Another String

Given a string s and a pattern p, find all **anagram positions** of p in s.

Example:

find_anagrams("cbaebabacd", "abc") → [0, 6]

Hint: Use **sliding window + frequency map.**

3.Implement a Custom Iterable Class

Write a class RangeLike(start, stop, step) that:

- Behaves like Python's built-in range().
- Supports iteration using __iter__() and __next__().

Hint: Implement __iter__() **and** __next__().

4.Find the Longest Palindromic Substring

Given a string s, find the **longest palindromic substring.**

Example:

longest_palindrome("babad") → "bab" (or "aba")

Hint: Use **expand-around-center** for $O(n^2)$ time complexity.

5.Implement a Function Decorator with Arguments

Write a decorator @repeat(n) that:

- Repeats the decorated function **n times** before returning its result.

Example:

```
@repeat(3)
def greet():
    print("Hello!")

greet()
```

Output:

```
Hello!
Hello!
Hello!
```

Hint: Use **closures and** *args, **kwargs **handling.**

6.Implement a Real-Time Stock Price Tracker

Write a class that:

- Supports **real-time stock price updates.**
- Returns the **highest and lowest price** efficiently.

Hint: Use a **sorted dictionary (collections.SortedDict).**

Design a Logger Rate Limiter

Implement a **logger system** that:

- Limits log messages so that the same message is printed **only once every 10 seconds.**

Hint: Use a **dictionary with timestamps.**

7.Design a Simple Cache (FIFO Cache)

Implement a **First-In-First-Out (FIFO) Cache** that:

- Stores N elements.
- Evicts the oldest item when full.

Hint: Use **OrderedDict** or a **deque.**

8.Find the Intersection of Two Linked Lists

Given two **linked lists**, find the **node where they intersect.**

Hint: Use **two-pointer technique** to meet at the intersection.